AI-Based Cost Estimation for Automotive Suspension and Steering Parts

A thesis submitted to the
Institute for Data Science and Artificial Intelligence

by

Ahmet Bilal ARIKAN

in partial fulfillment for the
degree of Master of Science

in
Data Science and Artificial Intelligence

Declaration of Authorship

I, Ahmet Bilal Arikan, declare that this thesis titled, 'AI-Based Cost Estimation for Automotive Suspension and Steering Parts' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: 28 August 2025

AI-Based Cost Estimation for Automotive Suspension and Steering Parts

Ahmet Bilal Arikan

Abstract

This thesis presents a machine learning approach that predicts unit manufacturing cost directly from two dimensional engineering drawings of suspension and steering parts. The dataset contains 13,684 drawings grouped into twenty four product categories. Drawings are parsed to extract geometric and dimensional entities, and about two hundred features are derived for each case. These features include descriptive statistics for lines, arcs, circles, and dimension values, histogram summaries with twelve bins, and distances between drawing level histograms and product group references using the Euclidean distance and the Kullback–Leibler divergence. Cost is modeled with methods that use gradient boosting with decision trees. Three machine learning algorithms are evaluated, namely XGBoost, CatBoost, and LightGBM. Models are trained per product group under a common validation and tuning protocol. Across the twenty four groups, the average mean absolute percentage error (MAPE) for XGBoost, CatBoost, and LightGBM is about 11% on the test dataset. Interpretation through feature importance and SHAP shows that maxima of rotated dimensions, statistics that describe arc geometry, and divergence measures from the histogram comparisons are consistent drivers of cost. The proposed end-to-end pipeline has the potential to operate without detailed process plans, shorten quotation lead times, improve consistency within part families, and be linked to design or enterprise systems to provide near real-time cost feedback.

Keywords: cost estimation, manufacturing cost, engineering drawing, gradient boosting machines, geometric parsing

Otomotiv Süspansiyon ve Yönlendirme Parçaları için Yapay Zeka Tabanlı Maliyet Tahmini

Ahmet Bilal Arıkan

Öz

Bu tez, süspansiyon ve yönlendirme parçalarına ait iki boyutlu teknik resimlerden birim üretim maliyetini doğrudan tahmin eden bir makine öğrenmesi yaklaşımı sunar. Veri kümesi, yirmi dört ürün kategorisinde gruplanmış 13,684 çizimden oluşur. Çizimler, geometrik ve boyutsal objeleri çıkarmak için çözünlenmiş ve her çizim için yaklaşık iki yüz özellik türetilmiştir. Bu özellikler; çizgi, yay ve çember gibi öğeler ile boyut değerleri için tanımlayıcı istatistikleri; on iki bölmeli histogram özetlerini; ve çizim düzeyindeki histogramların ürün grup referanslarıyla karşılaştırılmasından elde edilen uzaklıkları (Öklid uzaklığı ile Kullback–Leibler ayrışımı) içerir. Maliyet, karar ağaçlarıyla gradyan artırma (gradient boosting) kullanan yöntemlerle modellenmiştir. XGBoost, CatBoost ve LightGBM olmak üzere üç makine öğrenmesi algoritması değerlendirilmiştir. Modeller, ortak bir doğrulama ve optimizasyon protokolü altında ürün grubu bazında eğitilmiştir. Yirmi dört grup genelinde, XGBoost, CatBoost ve LightGBM için ortalama MAPE performansı yaklaşık %11 olarak elde edilmiştir. Özellik önemi ve SHAP ile yapılan yorumlama, boyut uzunluklarının maksimumlarının, yay geometrisine ilişkin istatistiklerin ve histogram karşılaştırmalarından gelen uzaklık ölçülerinin maliyetin çoğunluklu belirleyicileri olduğunu göstermektedir. Önerilen uçtan uca metot, ayrıntılı süreç planlarına gerek kalmadan çalışma, teklif hazırlama sürelerini kısaltma, parça aileleri içinde tutarlılığı artırma ve tasarım ya da kurumsal sistemlere bağlanarak gerçek zamana yakın maliyet geribildirimi sağlama potansiyeline sahiptir.

Anahtar Sözcükler: maliyet tahmini, üretim maliyeti, teknik resim, gradyan artırma algoritmaları, geometrik çözümleme

Acknowledgments

I would like to express my sincere gratitude to my advisor, Assoc. Prof. Şener Özönder, for his invaluable guidance, insightful suggestions, and continuous support throughout the preparation of this thesis.

This work was supported by the Scientific and Technological Research Council of Turkey (TÜBİTAK) under the 1711 Artificial Intelligence Ecosystem Call, grant number 3237003. I also gratefully acknowledge the financial support of TÜBİTAK through the BİDEB 2211 graduate scholarship.

Contents

List of Figures

List of Tables

Abbreviations

AI	**A**rtificial **I**ntelligence
CAD	**C**omputer-**A**ided **D**esign
CatBoost	**Cat**egorical **Boost**ing
CNC	**C**omputer **N**umerical **C**ontrol
DWG	Dra**wing** file format
DXF	**D**rawing E**x**change **F**ormat
ERP	**E**nterprise **R**esource **P**lanning
GBM	**G**radient **B**oosting **M**achine
LightGBM	**Light G**radient **B**oosting **M**achine
MAE	**M**ean **A**bsolute **E**rror
MAPE	**M**ean **A**bsolute **P**ercentage **E**rror
ML	**M**achine **L**earning
OEM	**O**riginal **E**quipment **M**anufacturer
RMSE	**R**oot **M**ean **S**quared **E**rror
SHAP	**SH**apley **A**dditive ex**P**lanations
XGBoost	E**x**treme **G**radient **Boost**ing

Symbols

Symbol	Name
a, b	Smoothing constants in CatBoost target statistics
$b_i^{(d)}$	Proportion in bin i for a given drawing
$b_i^{(m)}$	Mean proportion in bin i for the product group
d	Maximum tree depth
D	Training dataset consisting of input and target pairs (x_i, y_i)
D_{Euc}	Euclidean distance between histograms
D_{KL}	Kullback–Leibler divergence between histograms
$F^{(t)}(x)$	Model prediction at iteration t
g_i	Gradient of the loss for sample i
h_i	Hessian (second derivative) of the loss for sample i
i	Bin index $(1, \ldots, 12)$
n	Number of training samples
T	Number of boosting iterations
$\text{Tree}_t(x)$	Decision tree fitted at iteration t
α	L1 regularization parameter for leaf weights
γ	Regularization parameter for split penalty
η	Learning rate
λ	L2 regularization parameter for leaf weights
π	Permutation of training indices
ϵ	Small positive constant for numerical stability

Chapter 1

Introduction

In today's competitive manufacturing environment, the ability to predict production costs with high precision is essential for ensuring profitability, sustaining competitiveness, and guiding strategic decisions that shape the long horizon of company policies [1, 2].

Among the key inputs to this process are two-dimensional (2D) engineering drawings, most commonly prepared in DWG format, which serve as foundational documents in industrial production pipelines. These technical drawings are indispensable across diverse industries such as automotive, aerospace, electronics, and heavy machinery, since they convey critical geometric and material specifications including dimensions, tolerances, and material properties [3]. Acting as the initial stage for production planning and machining operations, these documents also provide the baseline for reliable cost estimation [4].

Cost estimation itself requires a systematic consideration of both direct and indirect expenditures. Direct costs typically involve raw materials, labor, and machine utilization, while indirect costs cover overheads, energy consumption, depreciation, and equipment maintenance [5]. Traditional approaches to cost estimation vary widely, ranging from empirical rules and heuristic calculations to detailed analytical models, and are frequently dependent on the experience and judgment of domain experts or historical company data [6]. In machining-intensive industries, cost determination is often carried out by calculating shop rates, which integrate labor, machine, and overhead expenses, and multiplying these by estimated machining times [7].

Nevertheless, estimating machining time accurately is far from trivial. It demands careful process planning, including the selection of appropriate cutting parameters such as feed rate, spindle speed, and depth of cut across various operations like milling, turning, drilling, or grinding. Furthermore, unproductive activities such as setup and handling times must also be accounted for [7].

This complexity makes traditional methods vulnerable to human error, inconsistencies, and time delays, often resulting in lengthy quotation periods. Moreover, conventional regression models frequently fail to capture the nonlinear patterns and interdependencies inherent in production data, which significantly reduces their predictive effectiveness [8].

Given these limitations, advanced data-driven techniques, particularly machine learning (ML) methods, have attracted growing attention in recent years. Among these, regression tree-based models such as XGBoost have emerged as powerful tools for tackling the cost estimation problem [8, 9]. By learning from historical data that inherently embed both direct and indirect costs, such models are capable of uncovering complex relationships that remain inaccessible to linear approaches or approaches based on explicit rules.

Importantly, periodic retraining allows ML models to dynamically adapt to evolving production processes and fluctuating market conditions, thereby maintaining predictive reliability over time.

Manual cost estimation is still the most widely used approach in many industrial settings, although it requires considerable resources. Teams of experts may require weeks to analyze a single drawing, significantly increasing both quotation lead times and operational expenses. This approach not only diminishes competitiveness but also introduces variability and bias due to subjective human assessments. In line with the principles of Industry 4.0 and cybermanufacturing, there is a growing industrial demand for automated cost estimation systems capable of providing rapid, accurate, and consistent outputs [10–12]. In manufacturing scenarios that operate on demand, where responsiveness and agility are critical, the ability to generate reliable cost predictions directly from technical drawings without requiring detailed process routes offers a significant advantage.

The automotive supply chain represents one of the most pressing contexts for such innovation. Suppliers working with both domestic and international original equipment manufacturers (OEMs) are often required to deliver quotations within 48 hours. However, the geometric diversity and complexity of automotive components make it difficult to adhere to this timeframe, with many quotations extending beyond a week. As manufacturing ecosystems become increasingly digitalized, the ability to generate cost estimates almost instantly has shifted from being a competitive advantage to becoming a necessity. Artificial intelligence (AI) based estimation tools, capable of producing reliable results within seconds, offer a clear edge by facilitating faster contract negotiations, supporting design optimization, and enhancing procurement strategies.

This research proposes a machine learning-based framework for automated cost estimation that directly leverages 2D DWG engineering drawings. In the proposed approach, drawings are parsed to extract a wide range of geometric and material features, such as line lengths, circle radii, angles, arc dimensions, and material information. These features, comprising approximately 200 descriptors per product group, are then used to train gradient boosting decision tree models.

The methodology presents several distinct advantages. First, decision tree-based models such as XGBoost inherently provide interpretability, offering insights into feature importance. This allows designers to identify specific geometric factors that disproportionately increase manufacturing costs, thereby encouraging design decisions that are more cost aware. Second, the framework drastically accelerates the quotation process for digital manufacturing platforms by enabling instant, accurate cost predictions from uploaded engineering drawings, facilitating real-time customer interaction [13]. Third, it ensures consistency and reliability of cost estimations throughout a wide range of components, enhancing supplier competitiveness. Finally, by highlighting discrepancies between predicted and historical costs, the system can reveal inaccuracies in previous manual estimations, encouraging organizations to reassess their costing strategies and further advance their digital transformation initiatives.

Chapter 2

Related Work

The literature is reviewed along two connected themes. The first theme concerns cost estimation methods used in manufacturing, with attention to studies that predict cost from product descriptions and design information. The second theme concerns the parsing of engineering drawings, since any method that relies on two dimensional drawings must first extract geometry, dimensions, and related annotations in a reliable way.

2.1 Cost Estimation in Manufacturing

Although three-dimensional (3D) Computer-Aided Design (CAD) models are increasingly common in industry, 2D engineering drawings remain an essential part of many manufacturing workflows, especially in processes such as Computer Numerical Control (CNC) machining. The academic literature on automated cost estimation derived directly from 2D drawings is still limited. Many studies that use artificial intelligence for cost estimation concentrate on 3D CAD models [14–18]. In industrial practice, cost estimation from 2D drawings is often carried out by experts who interpret geometric features, dimensions, and tolerances by hand. Costs are then inferred using personal experience or lookup tables [19]. This process requires significant effort and can lead to inconsistent outcomes [20, 21]. In parallel with these practices, research on sheet metal parts shows that multiple regression and neural networks can predict cost, with reported gains for neural networks when sufficient data are available [22].

A smaller body of work extracts features that drive cost directly from 2D drawings [3, 23, 24]. Serrat et al. focused on custom industrial hoses and reconstructed a 3D representation either from an STL file or from orthographic views in a 2D drawing; they then estimated time and cost from the recovered geometry [25]. Other studies use representations based on graphs and frameworks based on ontologies to extract knowledge that is relevant to cost from 2D engineering drawings [26–29]. Work on electronic assemblies reports gains from models that detect object patterns on printed circuit boards and map them to cost, which improves the accuracy and efficiency of predictions [30].

In additive manufacturing, much of the literature predicts build duration rather than cost. Analytical models incorporate process variables such as laser power, scanning speed, toolpath layout, and layer thickness [20, 31–35]. Attempts to predict machining time using only analytical models often face limits, because accurate prediction demands detailed knowledge of machine dynamics, control settings, and the specific configuration of the controller, which is hard to obtain in practice [36].

Model interpretability remains an important concern. Engineers and designers want to understand which aspects of a design contribute most to cost. Interpretable models such as decision trees or rule-based systems have long been used for this reason. Decision tree algorithms produce rules of the form "if ... then ..." that make the reasoning explicit [37]. Early expert systems encoded rules such as: if the material is steel, the number of holes is greater than ten, and the tolerance is less than 0.1 mm, then increase the base cost by twenty percent. Such systems can be effective in stable domains, but they require substantial effort to build and maintain. Decision tree models are still valued for clarity in cost modelling [37]. Even when more complex models are used, a simplified tree can be extracted afterward to aid explanation. These interpretable approaches appear in settings such as process selection and supplier quotation where clear justification matters.

Beyond individual decision trees, ensembles are used to balance accuracy with some level of interpretability. Gradient boosting implementations such as XGBoost provide feature importance scores and partial dependence plots that help explain model behaviour. Mandolini et al. emphasised the need for explainable models for products that are engineered to order [1]. Their framework used an automated estimation tool to generate training data and then trained learning models on this data. The result was a set of models that

predicted cost and also identified which input features increased cost [17]. This information allows design teams to adjust features, for example by increasing a small fillet radius that would make machining difficult, in order to reduce expected cost [1].

2.2 Data Parsing from Engineering Drawings

A crucial prerequisite for AI-based cost estimation from two-dimensional engineering drawings is the ability to parse and interpret the information they contain. These drawings typically include geometric entities, dimensional annotations, and material specifications. Automated extraction is difficult because drawing standards vary, annotations and symbols are diverse, and non-geometric elements such as title blocks and notes can interfere with recognition [38–41].

Several studies explain how to convert engineering drawings into structured data that software can use. Early work often relied on manual or semi-automated recognition, where an operator guided the software to identify geometry. More recent research introduces representations that use graphs, frameworks that rely on ontologies, and methods that learn from data in order to infer higher level design features directly from raw geometry, with the goal of producing machine readable output for cost estimation or process planning [23, 26, 27].

Despite this progress, robust parsing remains a central difficulty. Reliable extraction of lines, arcs, circles, and dimensions is essential for building feature sets that remain stable under variation in drawing style and content. Errors at the parsing stage tend to propagate through later steps, which motivates continued emphasis on methods that generalize well over heterogeneous drawing collections and industrial contexts [39, 42, 43].

Some studies work directly with digital drawings that preserve vector content. These methods read geometric and textual objects from the file and then cluster leaders, arrowheads, and numerals into consistent dimension sets. Reported systems describe how the recovered requirements integrate with quality assurance and production once dimensions are available as structured entities [43].

Other studies start from images that come from scans or from rendered pages and first isolate text so that geometric analysis proceeds without interference from captions and

notes. Classical pipelines combine connected component analysis with line detection that suppresses long strokes, then group characters into strings to form a dependable text layer for later stages [38, 39].

A related stream focuses on recovering dimensions when only raster input is available. Methods convert strokes to vectors, detect arrowheads and extension lines, and link these geometric cues with nearby numeral strings so that each dimension becomes a coherent statement with a value and a spatial anchor for computation [42].

Industrial systems often process the whole drawing in a staged workflow. They remove borders, title blocks, and tables, segment the remaining content into regions, detect symbols with template matching, trace line networks, and apply optical character recognition so that textual items are associated with the correct graphical entities in a structured output [40, 41].

To handle variation in drawing styles without extensive hand-written rules, some work combines visual detectors with explicit models of layout and relations. The goal is to infer tables, leaders, and meaningful groupings rather than returning isolated tokens, which preserves document structure for search and reasoning over drawing content [23].

While these studies demonstrate progress in cost estimation using 3D CAD models and, to a lesser extent, 2D drawings, important limitations remain. Existing methods often rely on manual feature selection or lack interpretability when neural networks are applied. Furthermore, there is limited evidence on the feasibility of parsing raw 2D engineering drawings at scale for automated cost prediction. This gap motivates the present thesis, which introduces a framework that combines large-scale geometric parsing with gradient boosting models to provide accurate and interpretable cost estimates directly from 2D drawings. In this framework, DXF entities are read directly from the source file, which provides precise coordinates, layers, and orientations for geometry and dimensions without the need for optical character recognition, aligning the parsing step with the cost estimation process described in the following chapters.

Chapter 3

Dataset

The dataset utilized in this study was constructed through a systematic preprocessing pipeline designed to transform raw engineering drawings into a structured and analyzable format. The pipeline begins with the collection of DWG files, which are subsequently converted into DXF format to facilitate parsing. Following this step, anomalous files are filtered and discarded, ensuring that only valid technical drawings progress through the workflow. For each valid file, entities are parsed and stored in an intermediate structured representation. From these parsed entities, both statistical summaries and distance-based features are computed, allowing for the generation of a rich tabular dataset that integrates geometric and descriptive attributes.

The resulting dataset incorporates several categories of features, including shape-based descriptors extracted from the drawings, encoded information related to material specifications, and numerical distance metrics derived from geometric relationships. These features collectively capture the complexity and variability of the components represented in the drawings. The target variable for prediction is the manufacturing cost, which is obtained from historical production data and normalized to reflect present-day price levels by adjusting for inflation.

3.1 Dataset Overview

The dataset employed in this study comprises a total of 13,684 2D engineering drawings in DWG format, representing a diverse set of suspension and steering components

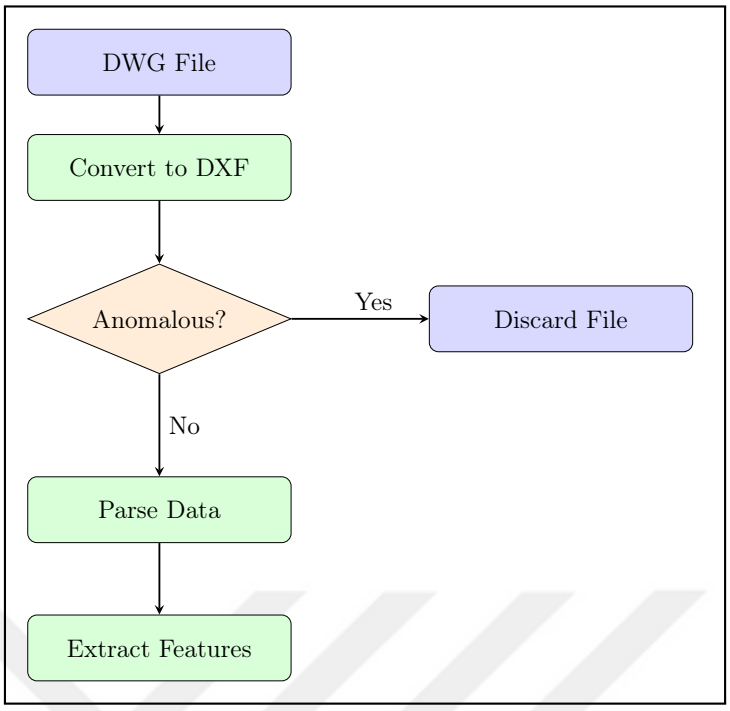


FIGURE 3.1: Flowchart of the processing pipeline for a single DWG file.

sourced from 30 different automobile manufacturers. These drawings encompass a broad spectrum of mechanical parts, which were systematically categorized into 24 product groups. Each DWG file corresponds to either an individual component or a complete assembly, and is linked to a historical unit manufacturing cost recorded in Euros, with values ranging from 0.50 to 50.00. An illustrative example of an engineering drawing is provided in Figure 3.2, highlighting the typical layout and annotation conventions used in the raw DWG files.

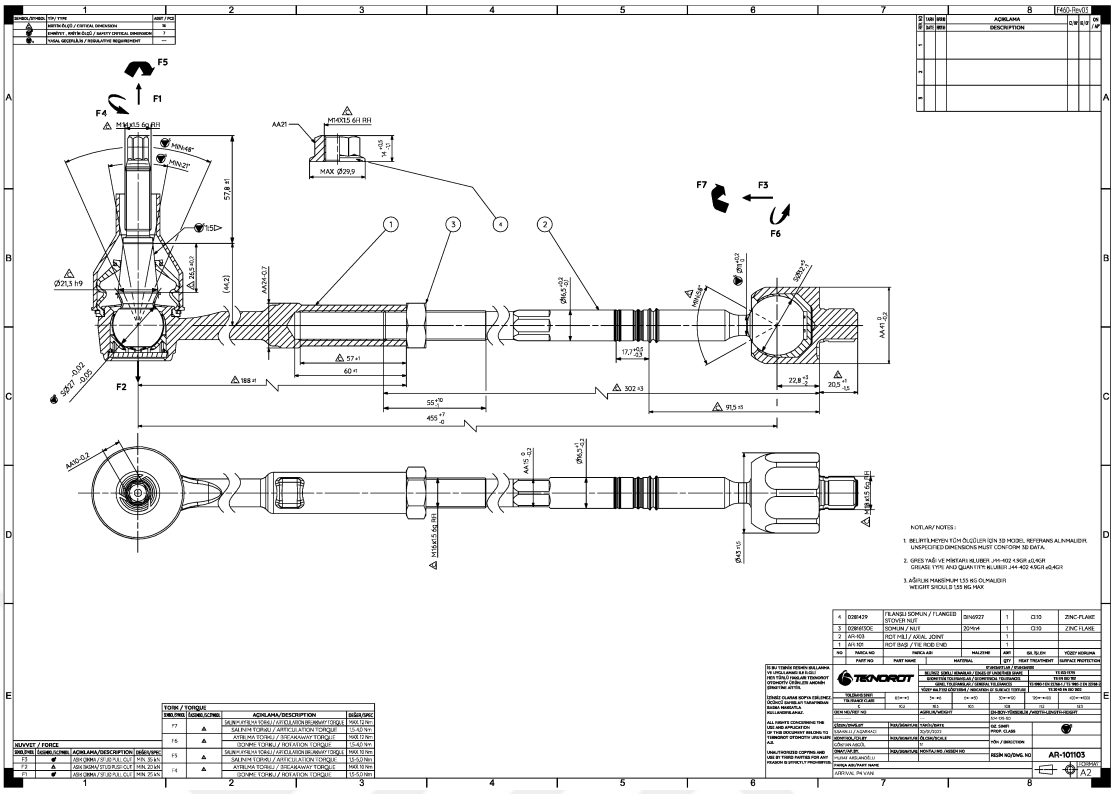
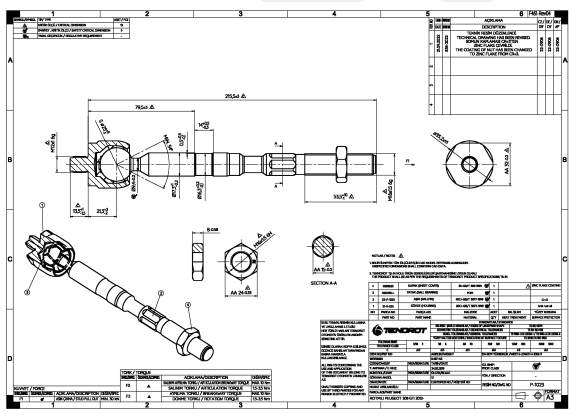


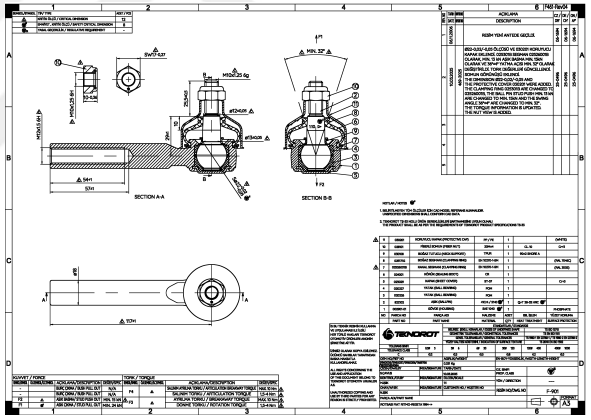
FIGURE 3.2: Representative engineering drawing from the dataset.

To enable reliable parsing, all DWG files were batch-converted into the DXF format, a text-based representation that allows consistent and standardized extraction of geometric entities. This conversion step was critical, as it preserves the exact geometric and semantic details contained in the drawings. In contrast to vision-based approaches, which may suffer from rendering inaccuracies, resolution limitations, or feature detection errors, this method ensures that the original engineering information is retained in full fidelity.

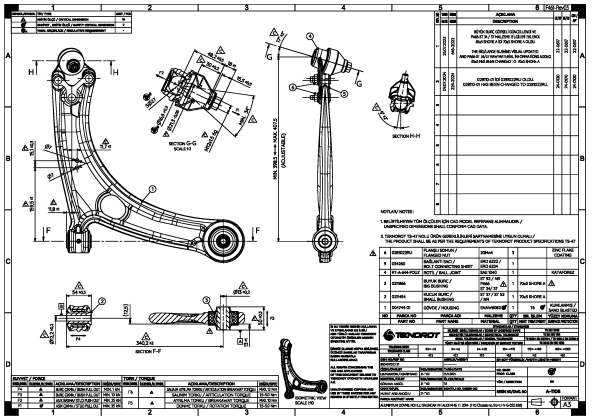
The 24 product groups in the dataset exhibit significant diversity in terms of geometric complexity, material characteristics, and cost ranges. Figure 3.3 presents representative samples from selected categories, where each drawing is displayed alongside its product group label and the number of available DWG files. This visualization highlights both the variability of part geometries and the distribution of drawings among groups.



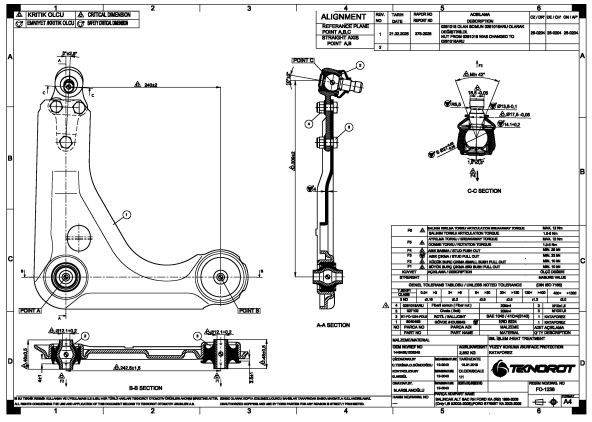
(A) Inner Tie Rod (960)



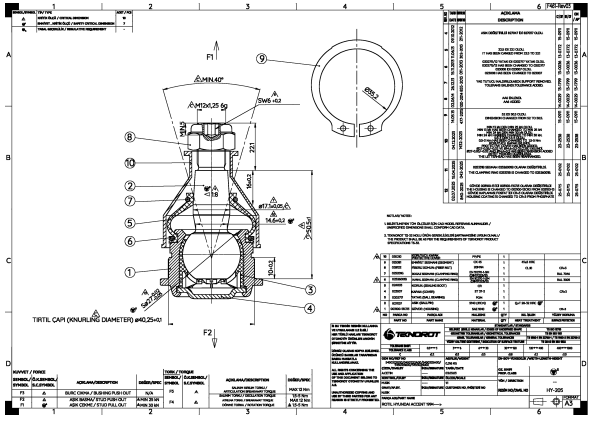
(B) Tie Rod End (1157)



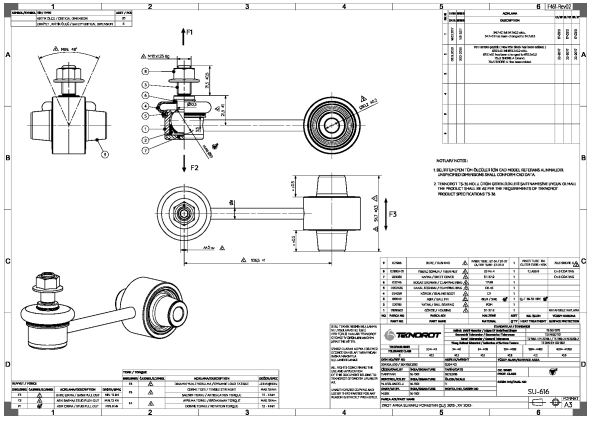
(C) Control Arms (651)



(D) Wishbones (1604)



(E) Ball Joint (811)



(F) Link Stabilizer (1553)

FIGURE 3.3: Examples of drawings from different product groups, accompanied by the drawing counts per group.

TABLE 3.1: Summary of the properties parsed for different geometric entities.

Entity Type	**Properties**
Line	Start point, end point, length
Circle	Center point, radius
Arc	Center point, radius, start angle, end angle
Spline	Control points, fit points, degree
Ellipse	Center point, major axis, minor axis

3.1.1 Dimensions

In addition to geometric primitives, dimension entities were also parsed. These included Rotated, Angular, Diametric, and Radial Dimensions, each of which conveys measurement-related information embedded in the drawings. For every dimension entity, the type of dimension, the displayed measurement text, the actual numerical value, and any associated tolerances were extracted. Furthermore, a global scale factor was derived from the Rotated Dimension, which was subsequently applied to translate geometric lengths and sizes into real-world units. This step ensured consistency between the drawing scale and actual component dimensions.

3.1.2 Textual Content

The third category included textual entities, namely `TEXT` and `MTEXT`, which frequently contained material specifications. These extracted material labels were treated as categorical variables and incorporated into the dataset as additional features. Such information provided essential context for manufacturing cost estimation, as material type is a critical determinant of overall production cost.

The outcome of the parsing stage is therefore a structured set of features for each drawing, encompassing geometric entities, dimension-based measurements, and material annotations. Together, these features form the basis for the subsequent feature engineering process.

3.2 Feature Engineering

This section describes the final step that converts the parsed entities into tabular data that the models can use. All measurements that depend on length or size were first rescaled with the drawing's unit factor so that values are expressed in consistent physical units.

3.2.1 Statistical Summaries

For each drawing, we computed descriptive summaries from the parsed geometric and dimensional quantities. The quantities include line lengths, arc lengths, arc angles, circle radii, and dimension values. From each of these, we extracted the following statistics: count, minimum, maximum, range, mean, median, mode, standard deviation, skewness, and kurtosis. To represent the full distribution, we also formed histograms with twelve bins and recorded both the bin counts and the corresponding proportions. These features provide a clear description of size and shape variation within a drawing and serve as stable inputs for the models in later chapters.

3.2.2 Distance Measures

We next derived measures that quantify how the distributions of a drawing differ from what is typical for its product group. We used two measures: Euclidean distance and Kullback–Leibler divergence (KL). Euclidean distance treats the two histograms as points with twelve coordinates and measures the distance between them as a straight line; larger values indicate broad differences in the proportions across bins. The KL divergence is a measure from information theory that increases when the drawing assigns little probability to bins where the product group assigns much more, which highlights mismatches in where probability mass is placed. For each product group and for each quantity listed above, we computed a reference histogram by averaging the bin proportions over all drawings in that group. These proportions are normalized so that they sum to one, which allows us to compare the drawing with the group reference using the two measures introduced here. Let $b_i^{(d)}$ denote the proportion in bin i for a given drawing and let $b_i^{(m)}$ denote the mean proportion in bin i for its product group, with $i = 1, \ldots, 12$.

We report the Euclidean distance between the two histograms with twelve bins as

$$D_{\mathrm{Euc}}\big(b^{(d)}, b^{(m)}\big) = \sqrt{\sum_{i=1}^{12}\Big(b_i^{(d)} - b_i^{(m)}\Big)^2}, \tag{3.1}$$

where a larger value indicates a stronger deviation from the reference distribution in terms of overall shape. We also report the Kullback–Leibler divergence of the group reference relative to the drawing as

$$D_{\mathrm{KL}}\big(b^{(m)} \,\|\, b^{(d)}\big) = \sum_{i=1}^{12} b_i^{(m)} \, \log\left(\frac{b_i^{(m)} + \epsilon}{b_i^{(d)} + \epsilon}\right), \tag{3.2}$$

where $\epsilon > 0$ is a small constant added to avoid undefined logarithms in empty bins, and where larger values indicate that the drawing departs from the group's typical distribution in a way that the mean histogram would emphasize.

Figure 3.4 compares the histogram with twelve bins for line lengths in one drawing with the mean histogram of its product group. The drawing exceeds the group mean in some bins and falls short in others. With this pattern, the Euclidean distance $D_{\mathrm{Euc}} = 0.06$ summarizes the overall spread of differences across bins, while the KL divergence $D_{\mathrm{KL}} = 0.04$ indicates limited mismatch in bins that carry most of the group probability. When small differences are spread across many bins, D_{Euc} tends to increase more than D_{KL}. When the drawing is close to zero in bins where the group has noticeable mass, D_{KL} rises quickly. When the drawing is high in bins that the group uses rarely, D_{Euc} still increases but D_{KL} changes less because those bins have little weight under the chosen orientation $D_{\mathrm{KL}}(b^{(m)} \| b^{(d)})$. Taken together, the reported values suggest a modest departure from the group pattern.

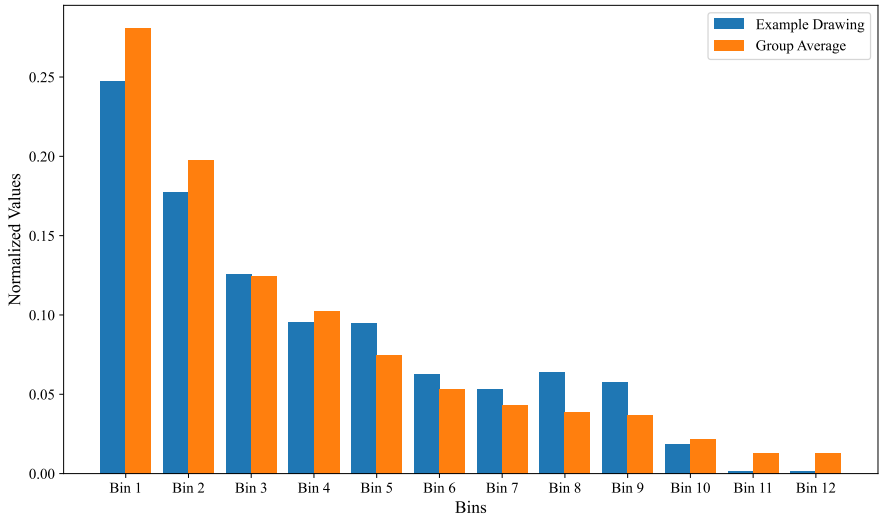


FIGURE 3.4: Histogram with twelve bins for line lengths in one drawing (blue bars) and for the mean of its product group (orange bars).

The two sets of features turn each drawing into a vector with the same number of entries for every case. The statistical summaries describe geometry and scale within the drawing, and the distance measures compare the drawing with the pattern that is typical for its product group. The resulting tabular data serve as inputs to the models in the next chapter.

Chapter 4

Implementation

This section sets out how the cost prediction models were trained and evaluated using the tabular features defined earlier. We built a separate model for each of the 24 product groups and used gradient boosting with decision trees. The common data preparation, tuning procedure, and evaluation settings are described in Section 4.1. Section 4.2 then introduces the three machine learning algorithms that we use and describes settings that are specific to each model.

4.1 Hyperparameter Tuning Approach

We tuned all models with a Bayesian optimization strategy [44]. The method builds a probabilistic model that links hyperparameter settings to the validation loss and then proposes the next trial where an improvement is likely, balancing exploration with exploitation. We used Optuna to run this procedure [45]. Optuna implements Bayesian optimization with a tree structured Parzen estimator [46] and selects new trials by maximizing an improvement criterion.

Some of the other methods commonly used for hyperparameter tuning are grid search and random search. Grid search quickly becomes impractical in our setting because the search space includes many parameters, each defined over wide ranges, which leads to a combinatorial explosion in the number of candidate configurations. Random search reduces this burden by sampling points at random, and it can sometimes identify good regions of the space more efficiently than grid search. However, it does not exploit

information from earlier trials, so many evaluations are often required before a strong solution is reached. In contrast, Bayesian optimization, as implemented in Optuna, adapts the search process by updating a probabilistic model of the objective function and directing future trials toward promising regions. This property makes it more suitable for tuning multiple models across 24 product groups with varying dataset sizes and feature complexities.

For each product group, we split the data into training, validation, and test sets in the ratio 70, 15, and 15 percent. The objective that Optuna minimized was the mean absolute percentage error (MAPE) on the validation set. Each study ran for 150 trials. Within every trial, the model was trained with early stopping and a patience of twenty rounds so that training stopped when the validation loss did not improve; this limits overfitting and avoids unnecessary computation [47]. In addition, we used cross validation with five folds on the training data to obtain stable estimates of out of sample error and to verify that performance was not driven by a single favorable split [48]. After the study finished, we refit a final model with the best settings found by Optuna and evaluated it on the held out test set.

4.2 Gradient Boosting Machines (GBMs)

This section outlines the learning approach used for cost prediction and motivates the choice of methods that apply gradient boosting with decision trees. These methods fit trees in sequence to reduce the loss and can learn relations that are not linear and interactions among predictors without manual construction [49, 50].

The properties of the data favor this class of methods. Missing values occur because some drawings do not contain every entity or measurement. Many predictors are strongly correlated since the statistical summaries describe the same underlying quantities and adjacent histogram bins move together. Most features are numeric and arise from distributions of lengths, radii, and dimensions, with additional categorical labels for material. Gradient boosting handles these conditions effectively because trees can route missing values during splitting without external imputation, and the sequential fitting tends to stabilize learning when predictors are correlated [51]. In contrast, parametric models

assume a specified functional form and additivity, require explicit imputation for missing values, and produce unstable coefficients under multicollinearity. Bagging methods such as random forests reduce variance but do not correct residual bias, which limits performance when thresholds and feature interactions are important. Distance-based methods such as k-nearest neighbors degrade in high dimension, are sensitive to scaling and correlation, and are computationally expensive at prediction time.

Within this class we compare three widely used algorithms: XGBoost [52], CatBoost [53], and LightGBM [54]. They share the same boosting principle but differ in how candidate splits are searched, how trees are grown, how missing and categorical values are handled, and which regularization and efficiency mechanisms are used. The subsections that follow describe these differences and the settings adopted in our experiments.

4.2.1 CatBoost

CatBoost is a gradient boosting algorithm that is designed to work well on tabular data with mixed feature types and missing values [53]. Figure 4.1 presents a short pseudocode that highlights the elements that distinguish CatBoost in our experiments. Categorical features are encoded with target statistics that use only earlier items in a permutation, which reduces leakage. Boosting follows the same order so that residuals for an item are computed from models that do not include that item. Trees are grown in a symmetric way with one split applied across each level, which gives a stable structure and fast inference. Missing values are routed by a learned default branch at each split, so no external imputation is required.

Input: data $D = \{(x_i, y_i)\}_{i=1}^{n}$; iterations T; depth d; learning rate η
Permute indices $\pi \leftarrow \mathrm{perm}(\{1, \ldots, n\})$
Encode categorical features with ordered target statistics (CTS; smoothing a, b)
Initialize model $F^{(0)}(x) \leftarrow 0$
for $t = 1$ to T **do**
    Compute ordered residuals (use only items preceding each π_i)
    Grow symmetric depth-d tree; learn default branch for missing values
    Update $F^{(t)}(x) \leftarrow F^{(t-1)}(x) + \eta\, \mathrm{Tree}_t(x)$
end for
Output: $F^{(T)}$

Figure 4.1 Pseudocode for CatBoost training algorithm. Residuals are negative gradients; CTS uses smoothing constants a, b; the default branch is the learned direction for missing values.

Table 4.1 summarizes the CatBoost hyperparameters used in this study, with brief descriptions and the ranges or fixed settings applied during tuning. The learning rate, tree depth, L2 penalty on leaf values, and the two randomness controls (bagging temperature and random strength) were varied within the listed intervals, while the number of estimators and the early stopping rule were fixed as shown. These choices were explored by Bayesian optimization with Optuna, as described in Section 4.1.

TABLE 4.1: Hyperparameters tuned for CatBoost with corresponding descriptions.

Hyperparameter	Description	Range
`learning_rate`	Step size of each boosting update	0.01 – 0.30
`max_depth`	Maximum tree depth, controls base learner complexity	3 – 10
`n_estimators`	Number of boosting iterations	5000
`early_stopping_rounds`	Stop if validation score does not improve for the given rounds	20
`l2_leaf_reg`	L2 regularization on leaf values to limit overfitting	1 – 10
`bagging_temperature`	Amount of randomization in sampling when building trees	0 – 1
`random_strength`	Noise added when choosing among candidate splits with similar gain	0 – 1

4.2.2 LightGBM

LightGBM is a gradient boosting algorithm for decision trees that speeds up training by building histograms of feature values and searching splits on these histograms [54]. Trees are grown by adding the single leaf that gives the largest reduction in loss, with a depth limit to control complexity. The library also includes efficiency options such as sampling that keeps large gradients and bundling of features that rarely take non-zero values. Missing values are handled within the split logic by learning a default

direction. Figure 4.2 shows a short pseudocode that highlights these elements as used in our experiments.

Input: data $D = \{(x_i, y_i)\}_{i=1}^{n}$; iterations T; max depth d; learning rate η
(Optional) Bundle mutually exclusive sparse features; set sampling that keeps large gradients
for $t = 1$ to T **do**
    Initialize a root node; build feature histograms at the node
    for current leaves **do**
        From histograms, compute the best split and its gain; set default branch for missing values
    end for
    Choose the leaf with the largest gain and split it (respect the depth limit); repeat until no gain or depth limit reached
    Update $F^{(t)}(x) \leftarrow F^{(t-1)}(x) + \eta \operatorname{Tree}_t(x)$
end for
Output: $F^{(T)}$

Figure 4.2 Pseudocode for LightGBM training algorithm. Histograms are used for split search; growth adds the leaf with the largest loss reduction subject to a depth limit; a default direction is learned for missing values; optional steps include gradient-based sampling and feature bundling.

Table 4.2 summarizes the LightGBM hyperparameters used in this study, with brief descriptions and the ranges or fixed settings applied during tuning. The learning rate, tree depth, maximum number of leaves, minimum samples per leaf, and the row and column sampling fractions were varied within the listed intervals, while the number of estimators and the early stopping rule were fixed as shown.

Table 4.2: Hyperparameters tuned for LightGBM with corresponding descriptions.

Hyperparameter	**Description**	**Range**
`learning_rate`	Step size for each boosting update	0.03 – 0.20
`max_depth`	Maximum depth of a tree to limit complexity	4 – 10
`n_estimators`	Number of boosting iterations	5000
`early_stopping_rounds`	Stop if validation score does not improve for the given rounds	20
`num_leaves`	Maximum number of leaves in a tree	15 – 64
`min_child_samples`	Minimum number of samples in a leaf	5 – 30
`subsample`	Fraction of rows used per iteration	0.70 – 1.00
`colsample_bytree`	Fraction of features used to build each tree	0.70 – 1.00

4.2.3 XGBoost

XGBoost is a gradient boosting algorithm for decision trees that combines fast split search with explicit regularization [52]. Trees are grown level by level and candidate splits are evaluated with a gain that includes penalties for model complexity. Split search can use exact counts or approximate quantile sketches.

Input: data $D = \{(x_i, y_i)\}_{i=1}^{n}$; iterations T; max depth d; learning rate η; α, λ, γ
(Optional) Build feature quantile sketches for approximate split search
Initialize model $F^{(0)}(x) \leftarrow 0$
for $t = 1$ to T **do**
    Compute gradients g_i and Hessians h_i from the loss at $F^{(t-1)}$
    Grow a tree level by level up to depth d:
  For each node at the current level, evaluate candidate splits using gain with α, λ, γ; set default branch for missing values; choose the best split if gain is positive
    Fit leaf weights by regularized least squares using (g_i, h_i)
    Update $F^{(t)}(x) \leftarrow F^{(t-1)}(x) + \eta\,\mathrm{Tree}_t(x)$
end for
Output: $F^{(T)}$

Figure 4.3 Pseudocode for XGBoost training algorithm. Residuals are first and second order gradients of the loss; gain includes regularization with α and λ for leaf weights and γ as a split penalty; a default branch is learned for missing values; split search may use quantile sketches.

Missing values are handled inside the split logic by learning a default direction, so no external imputation is required. Figure 4.3 gives a short pseudocode that highlights these elements as used in our experiments.

XGBoost exposes several hyperparameters that control the learning dynamics and the strength of regularization. The learning rate sets the update size at each boosting step. The maximum depth limits how complex a single tree can become. The row and column sampling fractions introduce randomness across iterations. The parameter γ sets the minimum loss reduction required to make a split. The parameters α and λ apply L_1 and L_2 penalties to leaf weights. An early stopping rule halts training when validation performance no longer improves. Table 4.3 lists the ranges and fixed settings used in this study.

TABLE 4.3: Hyperparameters tuned for XGBoost with corresponding descriptions.

Hyperparameter	**Description**	**Range**
`learning_rate`	Step size for each boosting update	0.01 – 0.30
`max_depth`	Maximum depth of a tree	3 – 10
`n_estimators`	Number of boosting iterations	1000
`subsample`	Fraction of rows used per iteration	0.50 – 1.00
`colsample_bytree`	Fraction of features used to build each tree	0.30 – 1.00
`gamma`	Minimum loss reduction to allow a split	0 – 1
`reg_alpha`	L_1 penalty on leaf weights	0 – 5
`reg_lambda`	L_2 penalty on leaf weights	0 – 5
`early_stopping_rounds`	Stop if validation score does not improve for the given rounds	20

To examine how XGBoost responds to changes in key settings, we conducted a *grid search* over `max_depth` and `learning_rate` for each product group. Figure 4.4 shows the result for the Link Stabilizer group, reporting mean absolute error averaged over cross validation with five folds for every pair of values. The lowest error occurs at `max_depth` = 7 and `learning_rate` = 0.05 (MAE = 0.285). Error tends to fall as depth increases from 3 to 7, with limited benefit and signs of overfitting beyond that range. Learning rates of 0.05 and 0.10 perform better than both 0.01 and 0.20, which indicates that values in the middle balance progress per step with stability. The heatmap pattern is consistent with this view, showing that performance improves away from the edges and approaches combinations near depth 7 with a learning rate around 0.05-0.10.

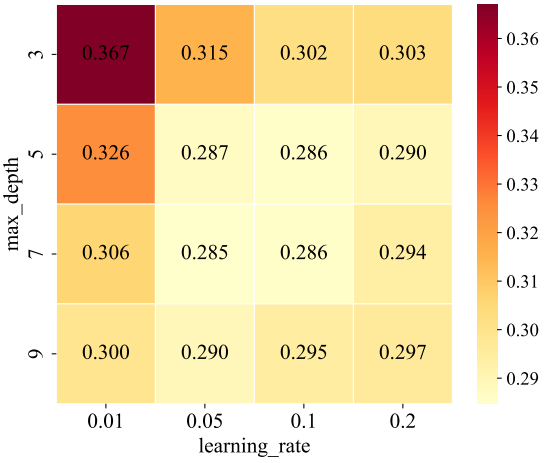


FIGURE 4.4: Mean absolute error across cross validation with five folds for combinations of `max_depth` and `learning_rate` in the XGBoost model trained on the Link Stabilizer group.

Chapter 5

Results and Discussion

This chapter provides a comprehensive evaluation of the machine learning framework developed for predicting manufacturing costs directly from two-dimensional engineering drawings. The analysis focuses on the performance of the models on the test sets, the role of feature importance, and the broader practical implications of the proposed approach. In this context, we evaluate three gradient boosting algorithms, namely XGBoost, CatBoost and LightGBM, across a dataset comprising 24 product groups. All models are trained using the same set of geometric and dimensional features extracted from the technical drawings. Furthermore, particular attention is given to identifying which geometric and dimensional characteristics have the greatest influence on cost prediction outcomes.

Table 5.1 presents the comparative performance of the XGBoost, CatBoost and LightGBM models across 24 product categories, evaluated using two metrics: MAE and MAPE. Among these, MAPE serves as the principal measure of evaluation because it provides more intuitive insights for cost estimation [55]. Across all 24 categories, MAPE values range from 3.91% to 18.51%, with 10 product groups achieving results below 10%. These 10 groups are not necessarily the ones with the largest sample sizes, which is consistent with the understanding that dataset size alone does not determine predictive performance [56]. Some of these groups include fewer than 300 samples, while others contain more than 1000. This observation indicates that the accuracy of the models is not solely linked to the number of drawings within each group. Instead, aspects such as the uniformity of geometric features, the presence of noise, and the inherent variability of manufacturing costs among similar components could potentially play a more significant

role. For example, standardized parts that share consistent geometric characteristics can lead to higher predictive accuracy even with limited data, whereas more diverse or irregular parts may require substantially larger datasets to achieve similar levels of performance.

When comparing the best performance across individual product groups, XGBoost obtained the lowest MAPE values in 11 categories, while CatBoost led in 9 and LightGBM in 4. Within the six largest product groups, each consisting of more than 1000 samples, CatBoost and XGBoost both demonstrated the strongest results in three groups each. This indicates that, in the context of larger datasets, these two algorithms tended to perform better than LightGBM. Regarding performance differences, several cases revealed a margin of at least 1% in MAPE compared to the second-best model. Examples include XGBoost for Inner Tie Rod – Female Inner Tie Rod, Wishbones – Wishbones, and Control Arms – Forged Control Arm with Conical Hole; CatBoost for Components – Control Arm Housing with Ball Joint and Components – Machined Male Tie Rod Ends; and LightGBM for Control Arms – Forged Control Arm.

The findings suggest that although all three gradient boosting algorithms provide competitive results, XGBoost tends to deliver more stable performance, particularly in product groups characterized by distinct geometric patterns or moderate dataset sizes. CatBoost, in contrast, frequently performs well in groups with higher complexity or noise, which may be attributed to its capability to manage categorical features and reduce overfitting. LightGBM, while slightly less effective overall, still achieves the best outcomes in certain cases, especially where deeper trees with relatively few leaves are more effective at capturing geometric variations. Taken together, the results support the applicability of gradient boosting methods for modeling the complex and nonlinear relationship between geometric features extracted from engineering drawings and manufacturing costs in the automotive context.

TABLE 5.1: MAE and MAPE results on test sets for all product types and models. The "Size" column shows the total number of drawings in each group, with 15% used for testing. The lowest MAPE values in each row are shown in bold.

Product group	**Size**	**XGBoost**		**CatBoost**		**LightGBM**	
		MAE	**MAPE**	**MAE**	**MAPE**	**MAE**	**MAPE**
Components - Machined Ball Joints with Bolt	591	0.28	14.04	0.25	**12.26**	0.26	13.00
Components - Machined Female Tie Rod Ends	1153	0.13	**7.06**	0.13	7.19	0.13	7.07
Components - Machined Male Tie Rod Ends	329	0.21	11.56	0.18	**9.89**	0.21	11.11
Components - Machined Arms with Ball Joint	434	1.30	14.60	1.28	14.44	1.43	**14.09**
Components - Machined Tie Rod Drag Links	870	0.10	9.18	0.10	9.20	0.10	**9.14**
Components - Machined Housing for Link Stabilizer	1707	0.10	13.62	0.09	**12.02**	0.09	12.91
Components - Tie Rod End Housing	1146	0.16	12.92	0.16	**12.50**	0.17	13.17
Components - Control Arm Housing with Ball Joint	483	1.07	18.51	0.88	**13.91**	0.91	15.15
Tie Rod Assembly - Tie Rod Assembly	235	0.41	**5.27**	0.50	6.32	0.42	5.35
Tie Rod End - Female Tie Rod Ends	1157	0.17	**5.01**	0.18	5.30	0.18	5.24
Ball Joint - Ball Joints with Bolt	442	0.52	14.57	0.43	**12.10**	0.47	12.92
Ball Joint - Round Ball Joints	369	0.55	17.10	0.50	**15.30**	0.50	15.62
Inner Tie Rod - Female Inner Tie Rod	58	0.52	**13.28**	0.72	17.84	0.67	17.82
Inner Tie Rod - Male Inner Tie Rod	902	0.33	**11.55**	0.34	11.76	0.36	12.33
Wishbones - Wishbones	1151	1.70	**11.17**	1.86	12.50	1.77	12.20
Wishbones - Wishbones with Conical Hole	91	1.02	**3.91**	1.00	4.00	1.23	5.72
Wishbones - Wishbones without Ball Joint	362	1.34	10.81	1.29	**10.68**	1.39	11.28
Control Arms - Stamped Lateral Arm	64	1.07	**12.66**	1.38	16.17	1.15	13.14
Control Arms - Aluminum Arm with Ball Joint	270	1.42	**9.47**	1.51	10.01	1.71	11.68
Control Arms - Forged Arm with Ball Joint	138	2.27	12.52	2.34	13.07	2.08	**11.78**
Control Arms - Forged Control Arm	108	2.15	12.30	1.97	11.51	1.83	**9.74**
Control Arms - Forged Control Arm with Conical Hole	34	0.75	**3.93**	0.82	5.58	0.89	4.99
Control Arms - Forged Control Arm without Ball Joint	37	1.23	**4.73**	1.97	6.86	1.47	5.39
Link Stabilizer	1553	0.29	10.79	0.28	**10.12**	0.30	11.37

To evaluate the overall predictive accuracy across all product groups, actual unit costs were compared with the corresponding predictions generated by the XGBoost model. Figure 5.1 presents a scatter plot in which each point corresponds to an engineering drawing. The black dashed line illustrates the ideal case in which predicted and actual values are identical. The results generally align with this diagonal, indicating satisfactory predictive performance across thousands of drawings, although some outliers are visible. These outliers appear more prominently in components with higher actual costs, where

the model tends to underestimate. Such a pattern may be related to the greater variability and structural complexity of high-cost components or to their relatively limited presence in the training dataset.

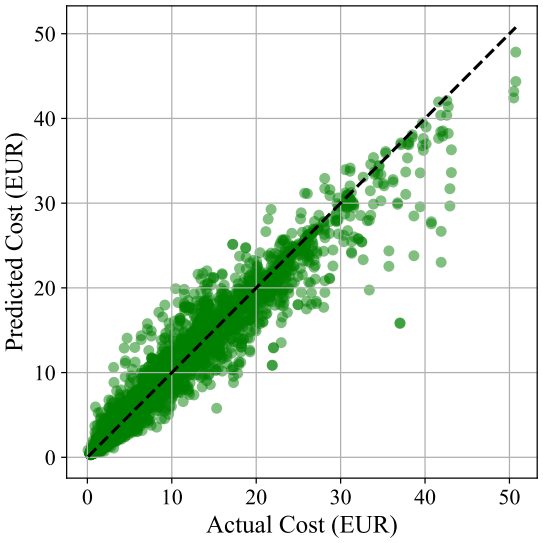


FIGURE 5.1: Comparison of actual and predicted unit costs obtained by the XGBoost model on 13,684 drawings from all product groups

A closer inspection of prediction errors across different cost ranges reveals meaningful patterns. For drawings with actual costs ≤ 1 EUR, the models achieved a mean absolute percentage error of 15.36 % over 2,062 drawings. This relatively high error is expected since even small absolute deviations translate into large percentage errors at such low cost levels. The majority of drawings, which fall between 1 EUR and 10 EUR with 9,545 samples, showed improved performance with an average MAPE of 10.66 %. In the intermediate range of 10 EUR to 20 EUR with 1,590 drawings, accuracy further improved with a MAPE of 8.81 %, while the highest cost range above 20 EUR with 487 drawings reached a similar level at 8.67 %. Although high-cost drawings yield lower relative errors on average, their absolute deviations are naturally larger in monetary terms when compared to low-cost cases. In addition, drawings that deviate strongly from typical group characteristics tend to exhibit disproportionately high errors, which suggests that unusual geometries or cost structures remain challenging for the models.

Feature importance was examined across all 24 trained models for each algorithm. For every gradient boosting method, average normalized feature importance values were computed based on split counts of the product-specific regressors. The top 20 features for

each model are presented in Figure 5.2. When considering the models collectively, several features consistently appeared among the most influential. For instance, the feature `rotated_max`, which captures the maximum measured length from rotated dimensions, was assigned the highest score in CatBoost (0.08) and was also ranked within the top five in both LightGBM and XGBoost. In addition, features associated with arc geometry, including `arc_angle_mean`, `arc_angle_min`, `arc_mean`, and `arc_total`, were frequently ranked highly across all three models, which may indicate that curvature-related information plays an important role in cost prediction for many components. Likewise, histogram-based descriptors such as `norm_line_bin8`, `norm_line_bin9`, and `arc_angle_bin12` appeared in multiple models, suggesting that the distributional characteristics of geometric features could also contribute to predictive accuracy.

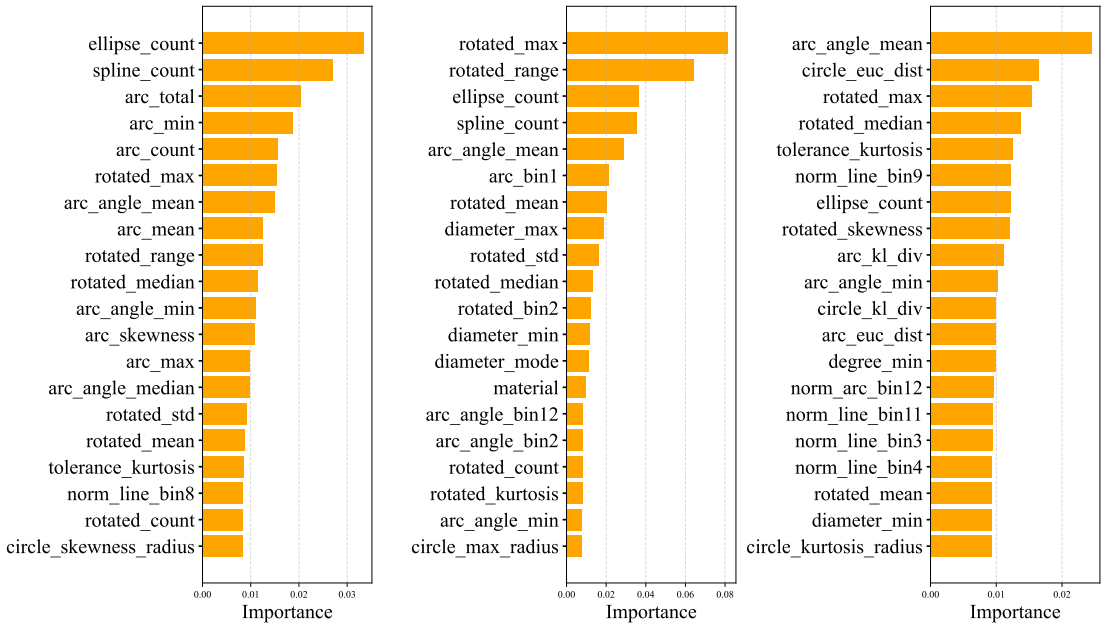


FIGURE 5.2: Feature importance rankings of the top 20 variables for XGBoost, CatBoost, and LightGBM, respectively, where importance is calculated using the weight criterion.

XGBoost and CatBoost tended to assign relatively greater importance to dimensional summary statistics, such as `rotated_std`, `rotated_range`, and `diameter_max`. In contrast, LightGBM highlighted additional metrics, including divergence-based measures like `arc_kl_div` and `circle_kl_div`, as well as distance-based descriptors such as `circle_euc_dist` and `arc_euc_dist`. The presence of these KL divergence and Euclidean distance features suggests that departures from typical geometric distributions could be indicative of complexity factors that influence cost.

Overall, the importance rankings confirm that geometric characteristics, particularly those linked to arcs and dimensional attributes, are fundamental determinants of manufacturing cost. The recurring prominence of features such as `ellipse_count`, `spline_count`, and `rotated_count` further demonstrates that both shape complexity and the presence of specific curve types exert a significant influence on cost variation.

To enhance interpretability of how features affect manufacturing cost, a standalone decision tree regressor [57] was trained specifically for the Link Stabilizer group. Although gradient boosting models rely on ensembles of many such trees, a single representative tree offers a transparent illustration of the decision rules learned from the data. Figure 5.3 displays this tree, showing how geometric and dimensional attributes contribute to cost estimation. Each internal node specifies the splitting feature (such as `diameter_median`, `ellipse_count`, or `tolerance_skewness`) together with a threshold value; samples with feature values less than or equal to the threshold proceed to the left branch, while those above follow the right. Thresholds are expressed in scaled physical units derived during training from the 2D engineering drawings. The second line of each node reports the number of training samples satisfying that condition, and the final line indicates the predicted unit cost in Euros, corresponding to the average of all samples reaching the leaf. Key split features including `diameter_median`, `ellipse_count`, and `tolerance_skewness` confirm earlier findings that dimensional statistics, curve counts, and distributional asymmetries are major cost determinants. Further splits involving `circle_std_radius` and `arc_min` highlight the role of curvature and spatial variation. This visualization complements the feature importance analysis by explicitly mapping the sequential decision logic that differentiates costs within a product group, thereby providing engineers with concrete guidance for cost-conscious design.

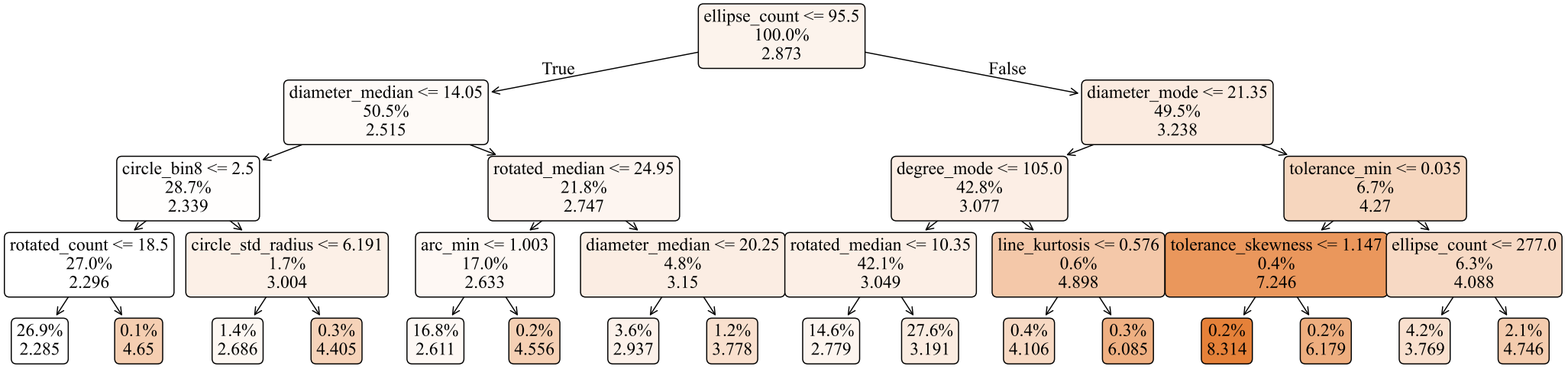


FIGURE 5.3: Visualization of a decision tree for the Link Stabilizer group, highlighting the role of features in cost prediction. Node details include, respectively, the split criterion, the sample ratio, and the predicted cost.

Shapley Additive Explanations (SHAP) were applied to the XGBoost model trained on the Link Stabilizer group in order to quantify how each feature shifts the predicted cost. The summary in Figure 5.4 ranks features by mean absolute SHAP value. The feature `ellipse_count` shows the largest average effect, and higher values are associated with positive shifts in the estimate. The material indicator `41CR4` appears next in the ranking, and its presence tends to increase the prediction. Measures that reflect overall size, including `diameter_max` and `rotated_median`, are also highly influential, and larger values generally push the output upward. The feature `spline_count` ranks near the top, which indicates that curved geometry contributes meaningfully to cost formation. Additional effects are observed for `tolerance_std`, `diameter_min`, `rotated_max`, `circle_count`, `rotated_range`, and `diameter_total`. Material indicators `ST372` and `20MN4` display smaller but noticeable impacts when compared with `41CR4`. A distribution feature, `norm_line_bin1`, also enters the ranking, which shows that the shape of the length histogram provides information beyond simple summaries. Taken together, the results indicate that geometry and scale, variation in tolerances, and material choice are central drivers of the model's predictions for this product group.

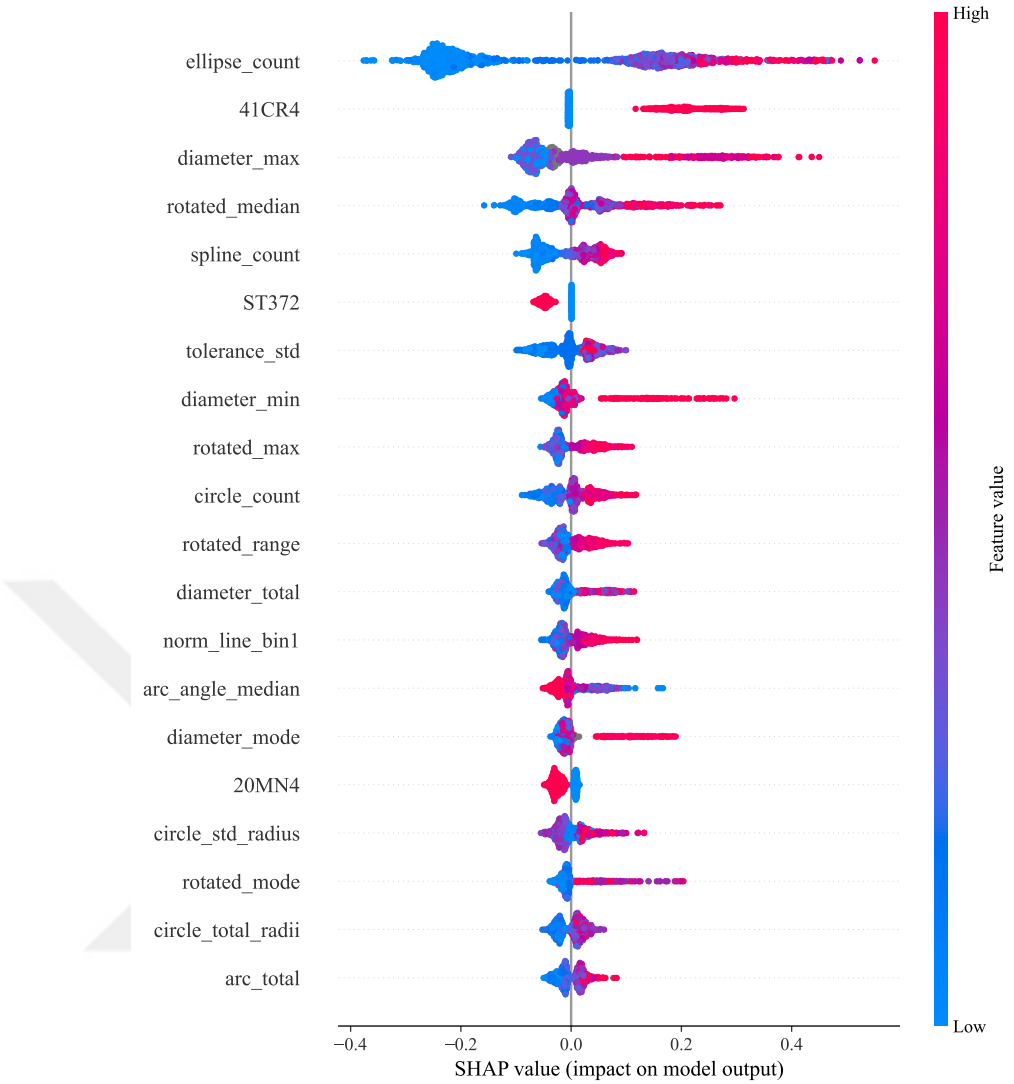


FIGURE 5.4: Summary of SHAP values for the XGBoost model applied to the Link Stabilizer product group.

As noted above, some material codes are influential for the Link Stabilizer group. They do not appear prominently in the feature importance table in Figure 5.2 because those scores were averaged across the twenty four product groups. A material can be important in a few groups and unimportant in others, so the averaging reduces its overall score. This does not mean that materials are ineffective. Within a single group they can shift the cost prediction, which is why the SHAP summary for the Link Stabilizer model shows clear material effects.

Chapter 6

Conclusion

This thesis has introduced a fully automated framework for manufacturing cost prediction, developed to operate directly on 2D DWG engineering drawings. By extracting a wide range of geometric and statistical features and applying gradient-boosted decision tree regressors, the proposed system demonstrates that it is possible to achieve high predictive accuracy without the need for explicit process planning or manual feature annotation. Across 24 product groups and 13,684 individual components, the models achieved an average MAPE of about 11% on the test datasets, with several product groups obtaining results below 10%. A comparative evaluation of XGBoost, CatBoost, and LightGBM confirmed that gradient boosting methods are particularly effective in capturing the nonlinear dependencies between geometric complexity, dimensional variation, material selection, and production cost.

Beyond technical accuracy, these results also carry important financial implications. Prior research has shown that inaccuracies in cost estimation under cost-plus pricing can directly erode profit margins and even threaten firm viability when errors are systematic [58]. With an average MAPE of around 11%, the models developed in this thesis demonstrate competitive performance, yet such deviations are far from negligible in a commercial context. At low unit costs, small absolute errors inflate percentage deviations and may distort margins, while at higher unit costs the absolute deviations implied by an 11% error can be significant in monetary terms, reducing competitiveness in bidding. These considerations underline that improving predictive accuracy is not only a statistical objective but also a direct contributor to sustaining profitability.

In addition to predictive accuracy and financial relevance, the framework places emphasis on interpretability. The integration of feature importance analysis, decision tree visualization, and SHAP attribution methods revealed that the dominant cost drivers identified by the model are in strong alignment with established engineering knowledge. Variables such as the count of elliptical entities, arc geometry statistics, and dimensional extrema repeatedly emerged as significant contributors to manufacturing cost. These insights extend the utility of the system beyond estimation, as they provide practitioners with actionable guidelines for cost-aware design optimization. In this way, the framework bridges the gap between purely data-driven estimation and practical engineering decision-making.

The potential applications of this machine learning framework are broad and significant. Within the automotive industry, the ability to automatically extract geometric information from engineering drawings and predict unit production costs offers early-stage insights that can support design iterations and quotation preparation. This reduces reliance on expert-driven manual evaluations and historical costing records, which are often time-consuming and inconsistent. The approach is particularly advantageous in scenarios involving new product designs or additive manufacturing, where reference data may be limited or unavailable. Furthermore, the proposed methodology is not restricted to automotive manufacturing alone; by tailoring the feature extraction stage, the workflow could be readily extended to other domains such as civil engineering, architecture, aerospace, or shipbuilding, where technical drawings serve as a central input to production processes.

The developed system can also be envisioned as a decision-support tool within digital production pipelines. Its integration into CAD environments or Enterprise Resource Planning (ERP) systems would allow real-time cost feedback during the design phase, thereby supporting procurement, planning, and strategic decision-making. Such integration would not only accelerate cost evaluation but also contribute to the broader goals of Industry 4.0 by enabling smarter, data-driven production ecosystems.

This study has certain limitations that should be noted. The dataset consists of 13,684 drawings obtained from one industrial source in the automotive sector. The product groups are diverse, but the findings may not generalize to other industries or drawing standards. The framework relies only on 2D engineering drawings, which are common in

practice but do not contain volumetric or process information that can influence cost. As a result, features related to 3D geometry, machining strategies, or production planning are not represented. In addition, the evaluation is restricted to gradient boosting decision tree models. These methods provide strong predictive accuracy and interpretability, but other approaches such as deep learning or physics-based methods were not considered and might reveal different patterns, although they are often less transparent.

Looking forward, several avenues for improvement can be identified. One potential direction involves expanding the training dataset by generating synthetic technical drawings, thereby addressing class imbalance and enhancing model robustness. Another promising extension is to enrich the feature set by incorporating spatial relationships between geometric entities, such as adjacency, symmetry, and relative positioning. Capturing these spatial dependencies through graph-based models or advanced feature engineering techniques could further enhance the model's ability to represent and predict cost structures. These future developments would strengthen the adaptability and generalization of the framework, consolidating its role as a reliable and scalable tool for automated cost estimation in manufacturing.

Bibliography

[1] Marco Mandolini, Luca Manuguerra, Mikhailo Sartini, Giulio Marcello Lo Presti, and Francesco Pescatori. A cost modelling methodology based on machine learning for engineered-to-order products. *Engineering Applications of Artificial Intelligence*, 136(A):108957, October 2024.

[2] Mohammad Mahdi Rounaghi, Hajer Jarrar, and Leo-Paul Dana. Implementation of strategic cost management in manufacturing companies: overcoming costs stickiness and increasing corporate sustainability. *Future Business Journal*, 7(1), September 2021.

[3] Christoph Haar, Hangbeom Kim, and Lukas Koberg. Ai-based engineering and production drawing information extraction. In Kyoung-Yun Kim, Leslie Monplaisir, and Jeremy Rickli, editors, *Flexible Automation and Intelligent Manufacturing: The Human-Data-Technology Nexus*, pages 374–382, Cham, 2023. Springer International Publishing.

[4] Yi-Hsin Lin, Yu-Hung Ting, Yi-Cyun Huang, Kai-Lun Cheng, and Wen-Ren Jong. Integration of deep learning for automatic recognition of 2d engineering drawings. *Machines*, 11(8):802, August 2023.

[5] Fang Li. Automated cost estimation for 3-axis cnc milling and stereolithography rapid prototyping. Master's thesis, University of Manitoba, Winnipeg, MB, Canada, 2005.

[6] Adnan Niazi, Jian S Dai, Stavroula Balabani, and Lakmal Seneviratne. Product cost estimation: Technique classification and methodology review. *Journal of Manufacturing Science and Engineering*, 128(2):563–575, 2006.

[7] Antonio Armillotta. On the role of complexity in machining time estimation. *Journal of Intelligent Manufacturing*, 32(8):2281–2299, February 2021.

[8] Md. Mahfuzul Islam Shamim, Abu Bakar bin Abdul Hamid, Tadiwa Elisha Nyamasvisva, and Najmus Saqib Bin Rafi. Advancement of artificial intelligence in cost estimation for project management success: A systematic review of machine learning, deep learning, regression, and hybrid models. *Modelling*, 6(2), 2025.

[9] Raden Achmad Chairdino Leuveano Anas Maâruf, Ali Akbar Ramadani Nasution. Machine learning approach for early assembly design cost estimation: A case from make-to-order manufacturing industry. *International Journal of Technology*, 15(4):1037–1047, July 2024.

[10] M. Imran Khan, Tabassam Yasmeen, Mushtaq Khan, Noor Ul Hadi, Muhammad Asif, Muhammad Farooq, and Sami G. Al-Ghamdi. Integrating industry 4.0 for enhanced sustainability: Pathways and prospects. *Sustainable Production and Consumption*, 54:149–189, 2025.

[11] Andrew Kusiak. Smart manufacturing. *International Journal of Production Research*, 56(1-2):508 – 517, 2018.

[12] Heiner Lasi, Peter Fettke, Hans-Georg Kemper, Thomas Feld, and Michael Hoffmann. Industry 4.0. *Business & Information Systems Engineering*, 6(4):239–242, June 2014.

[13] Dazhong Wu, David W. Rosen, Lihui Wang, and Dirk Schaefer. Cloud-based design and manufacturing: A new paradigm in digital manufacturing and design innovation. *Computer-Aided Design*, 59:1–14, 2015.

[14] Jinwon Lee, Hyunoh Lee, and Duhwan Mun. 3d convolutional neural network for machining feature recognition with gradient-based visual explanations from 3d cad models. *Scientific Reports*, 12(1):14864, 2022.

[15] Zhibo Zhang, Prakhar Jaiswal, and Rahul Rai. Featurenet: Machining feature recognition based on 3d convolution neural network. *Computer-Aided Design*, 101:12–22, August 2018.

[16] Fangwei Ning, Yan Shi, Maolin Cai, Weiqing Xu, and Xianzhi Zhang. Manufacturing cost estimation based on a deep-learning method. *Journal of Manufacturing Systems*, 54:186–195, January 2020.

[17] Soyoung Yoo and Namwoo Kang. Explainable artificial intelligence for manufacturing cost estimation and machining feature visualization. *Expert Systems with Applications*, 183:115430, November 2021.

[18] Siu L. Chan, Yanglong Lu, and Yan Wang. Data-driven cost estimation for additive manufacturing in cybermanufacturing. *Journal of Manufacturing Systems*, 46:115–126, 2018.

[19] Narges Sajadfar and Yongsheng Ma. A hybrid cost estimation framework based on feature-oriented data mining approach. *Advanced Engineering Informatics*, 29(3):633–647, 2015.

[20] Changqing Liu, Yingguang Li, Wei Wang, and Weiming Shen. A feature-based method for nc machining time estimation. *Robotics and Computer-Integrated Manufacturing*, 29(4):8–14, 2013.

[21] Sergio Cavalieri, Paolo Maccarrone, and Roberto Pinto. Parametric vs. neural network models for the estimation of production costs: A case study in the automotive industry. *International Journal of Production Economics*, 91(2):165–177, September 2004.

[22] B. Verlinden, J.R. Duflou, P. Collin, and D. Cattrysse. Cost estimation for sheet metal parts using multiple regression and artificial neural networks: A case study. *International Journal of Production Economics*, 111(2):484–492, February 2008.

[23] Dries Van Daele, Nicholas Decleyre, Herman Dubois, and Wannes Meert. An automated engineering assistant: Learning parsers for technical drawings. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 15195–15203, Virtual Conference, February 2021.

[24] Rujie Liu, Yuehong Wang, Takayuki Baba, and Daiki Masumoto. Shape detection from line drawings with local neighborhood structure. *Pattern Recognition*, 43(5):1907–1916, 2010.

[25] Joan Serrat, Felipe Lumbreras, and Antonio M. López. Cost estimation of custom hoses from stl files and cad drawings. *Computers in Industry*, 64(3):299–309, April 2013.

[26] Liuyue Xie, Yao Lu, Tomotake Furuhata, Soji Yamakawa, Wentai Zhang, Amit Regmi, Levent Kara, and Kenji Shimada. Graph neural network-enabled manufacturing method classification from engineering drawings. *Computers in Industry*, 142:103697, 2022.

[27] Wentai Zhang, Joe Joseph, Yue Yin, Liuyue Xie, Tomotake Furuhata, Soji Yamakawa, Kenji Shimada, and Levent Burak Kara. Component segmentation of engineering drawings using graph convolutional networks. *Computers in Industry*, 147:103885, 2023.

[28] Denis R. Kasimov, Aleksandr V. Kuchuganov, and Valeriy N. Kuchuganov. Individual strategies in the tasks of graphical retrieval of technical drawings. *Journal of Visual Languages & Computing*, 28:134–146, 2015.

[29] Qingmai Wang and Xinghuo Yu. Ontology based automatic feature recognition framework. *Computers in Industry*, 65(7):1041–1052, 2014.

[30] Frank Bodendorf, Stefan Merbele, and Jörg Franke. Deep learning based cost estimation of circuit boards: a case study in the automotive industry. *International Journal of Production Research*, 60(23):6945–6966, December 2022.

[31] Zicheng Zhu, Vimal Dhokia, and Stephen T Newman. A new algorithm for build time estimation for fused filament fabrication technologies. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, 230(12):2214–2228, 2016.

[32] Isamu Nishida and Keiichi Shirase. Machine tool assignment realized by automated nc program generation and machining time prediction. *International Journal of Automation Technology*, 13(5):700–707, 2019.

[33] Elamir S Gadelmawla, Fahad A Al-Mufadi, and Abdualaziz S Al-Aboodi. Calculation of the machining time of cutting tools from captured images of machined parts using image texture features. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, 228(2):203–214, November 2013.

[34] Turan Gurgenc, Ferhat Ucar, Deniz Korkmaz, Cihan Ozel, and Yunus Ortac. A study on the extreme learning machine based prediction of machining times of the cycloidal gears in cnc milling machines. *Production Engineering*, 13(6):635–647, October 2019.

[35] Eun-Young Heo, Dong-Won Kim, Bo-Hyun Kim, and F. Frank Chen. Estimation of nc machining time using nc block distribution for sculptured surface machining. *Robotics and Computer-Integrated Manufacturing*, 22(5):437–446, 2006.

[36] Reginaldo Teixeira Coelho, Adriano Fagali de Souza, Alessandro Rodrigues Roger, Aldo Marcel Yoshida Rigatti, and Alexandre Alves de Lima Ribeiro. Mechanistic approach to predict real machining time for milling free-form geometries applying high feed rate. *Int. J. Adv. Manuf. Technol.*, 46(9-12):1103–1111, February 2010.

[37] Karen Mourikas, Joe King, and Denise Nelson. Machine learning approach to cost analysis. In *Proceedings of the 2017 ICEAA Professional Development & Training Workshop*, Portland, Oregon, USA, June 2017.

[38] Lloyd A. Fletcher and Rangachar Kasturi. A robust algorithm for text string separation from mixed text/graphics images. *IEEE transactions on pattern analysis and machine intelligence*, 10(6):910–918, 2002.

[39] Karl Tombre, Salvatore Tabbone, Loïc Pélissier, Bart Lamiroy, and Philippe Dosch. Text/graphics separation revisited. In Daniel Lopresti, Jianying Hu, and Ramanujan Kashi, editors, *Document Analysis Systems V*, pages 200–211, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg.

[40] Sung-O Kang, Eul-Bum Lee, and Hum-Kyung Baek. A digitization and conversion tool for imaged drawings to intelligent piping and instrumentation diagrams (pid). *Energies*, 12(13):2593, 2019.

[41] Guanqun Su, Shuai Zhao, Tao Li, Shengyong Liu, Yaqi Li, Guanglong Zhao, and Zhongtao Li. Image format pipeline and instrument diagram recognition method based on deep learning. *Biomimetic Intelligence and Robotics*, 4(1):100142, 2024.

[42] Dov Dori and Yelena Velkovitch. Segmentation and recognition of dimensioning text from engineering drawings. *Computer Vision and Image Understanding*, 69(2):196–201, 1998.

[43] Beate Scheibel, Juergen Mangler, and Stefanie Rinderle-Ma. Extraction of dimension requirements from engineering drawings for supporting quality control in production processes. *Computers in Industry*, 129:103442, 2021.

[44] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[45] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2623–2631, 2019.

[46] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011.

[47] Lutz Prechelt. *Early Stopping — But When?*, pages 53–67. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[48] Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'95, page 1137â1143, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.

[49] JH Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29(5):1189–1232, October 2001.

[50] Ravid Shwartz-Ziv and Amitai Armon. Tabular data: Deep learning is not all you need. *Information Fusion*, 81:84–90, 2022.

[51] Prativa Pokhrel and Alina Lazar. Towards machine learning interpretability for tabular data with mixed data types. In *Proceedings of the 35th International FLAIRS Conference*, Florida, USA, May 2022.

[52] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge*

Discovery and Data Mining, KDD '16, pages 785–794, New York, NY, USA, 2016. Association for Computing Machinery.

[53] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[54] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[55] Arnaud de Myttenaere, Boris Golden, BÃ©nÃ©dicte Le Grand, and Fabrice Rossi. Mean absolute percentage error for regression models. *Neurocomputing*, 192:38–48, June 2016.

[56] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer, New York, NY, 2 edition, 2009.

[57] Leo Breiman, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone. *Classification and Regression Trees.* Wadsworth International Group, Belmont, California, 1984.

[58] Graziano Coller and Paolo Collini. The value of cost accuracy in fullâcost pricing decisions. In *9th Conference on New Directions in Management Accounting*, Brussels, December 2014.