Politecnico di Torino

Politecnico di Torino

Master's Degree Course in
A.y. 2023/2024
October 2024

Optimizing Construction Cost Predictions

A Comparative Study of Machine Learning Algorithms

Supervisor:

Dr. Timur Narbaev

Candidate:

Seyedmohammadamin Aziminasrabad

Table of Contents

1. Abstract

Accurate cost forecasting is vital in construction project management, where budget overruns and delays can have significant impacts. Traditional methods like Earned Value Management (EVM) are widely used, but they rely on static, linear assumptions that often fail to capture the complexities of real-world projects. This thesis explores the potential of machine learning (ML) algorithms to improve cost forecasting by addressing the limitations of EVM and offering more dynamic, data-driven predictions during project execution.

A comparative analysis of six machine learning models—XGBoost, Extremely Randomized Trees, Random Forest, Support Vector Machine (SVM), Light Gradient Boosting Machine (LightGBM), and K-Nearest Neighbors (KNN)—was conducted using a dataset of 90 real-world construction projects, selected from 181 initial projects. Key project performance metrics, such as Actual Cost (AC), Earned Value (EV), and the Cost Performance Index (CPI), were used as inputs, along with newly introduced features: Project Regularity (RI) and Project Seriality (SP). These static features were introduced to account for non-linear project growth patterns and task structures.

The machine learning models were trained on 75% of the data and tested on the remaining 25%, with performance evaluated using Mean Absolute Percentage Error (MAPE) and Normalized Root Mean Squared Error (NRMSE). Results indicated that all ML models significantly outperformed traditional EVM methods, with XGBoost achieving the lowest error rates. The inclusion of RI and SP further enhanced model accuracy, particularly in projects with non-linear progress.

Project Regularity (RI) and Project Seriality (SP) were found to be valuable features for improving the predictive power of ML models. RI captured deviations from linear project progression, while SP reflected the structure of tasks, whether serial or parallel. These additional features enabled the models to better account for the dynamic and complex nature of construction projects, leading to more accurate forecasts at various stages of project execution.

In conclusion, the study demonstrates that machine learning models offer a superior alternative to traditional cost forecasting methods like EVM. By incorporating dynamic and static project features, ML models provide more precise, adaptive, and reliable cost predictions, helping project managers mitigate risks and make more informed decisions. These findings suggest that further integration of ML in project management practices could lead to improved project outcomes, especially as ML techniques continue to evolve.

2. Introduction

In the mid-20th century, project management began to take shape as a distinct professional discipline. This period saw the introduction of key methodologies such as the Critical Path Method (CPM) and the Program Evaluation and Review Technique (PERT), which provided structured approaches to planning and scheduling project tasks. These techniques, grounded in operations research, aimed to identify the most efficient sequence of activities, ensuring that projects could be completed on time and within budget while effectively utilizing available resources (Vanhoucke, 2012).

In the mid-20th century, project management began to take shape as a distinct professional discipline. This period saw the introduction of key methodologies such as the Critical Path Method (CPM) and the Program Evaluation and Review Technique (PERT), which provided structured approaches to planning and scheduling project tasks. These techniques, grounded in operations research, aimed to identify the most efficient sequence of activities, ensuring that projects could be completed on time and within budget while effectively utilizing available resources (Vanhoucke, 2012).

2.1. Basics of Project Management

According to the Project Management Institute (PMI), a project is defined as “a temporary endeavour undertaken to create a unique product, service, or result.” The temporary nature of a project implies that it has a clear start and end date, distinguishing it from ongoing operations. A project's uniqueness refers to its output's distinctiveness, which could be a product, service, or result that is different from other similar deliverables in key aspects.(PMI, 2019)

A project's success is typically measured by its ability to deliver the intended output with the features and functions initially defined. However, achieving this success is subject to various constraints, commonly illustrated by the Project Management Triangle of Constraints, also known as the Iron Triangle. This triangle highlights three fundamental dimensions: time, cost, and quality.(Atkinson, 1999)

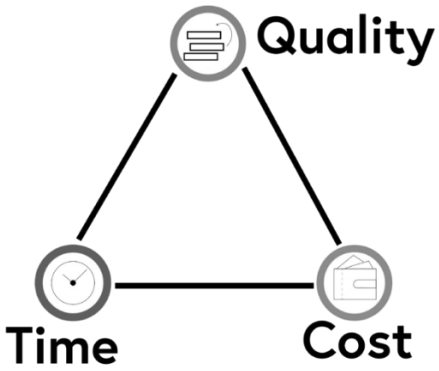

Figure 1-Project Management Triangle of Constraints

Central to project management, the Iron Triangle emphasizes the delicate balance required between these dimensions. Time refers to the schedule and the timely completion of project tasks within the set deadlines. Cost encompasses the financial resources allocated to the project, with a focus on adhering to the budget to prevent overruns. Quality pertains to the standards of the deliverables, ensuring that the final product meets the specified requirements and satisfies stakeholder expectations.

Balancing cost, quality, and time is crucial to achieving the project's scope and objectives. The project manager must carefully prioritize one constraint, often adjusting the others to meet the project's specific goals. For instance, emphasizing quality might require extending the timeline or increasing costs.

Given that projects are unique and resource-limited, they inherently involve uncertainty. This makes monitoring and control essential. By regularly reviewing progress and making adjustments as needed, project managers can effectively navigate challenges, ensuring the project stays on track and meets its intended outcomes.

2.2. Project Monitoring and Control

Project Monitoring and Control, often collectively referred to as Project Control, are integral components of a feedback system designed to ensure that a project stays aligned with its planned objectives. monitoring is the process that follows the planning phase and extends throughout project execution. Its primary purpose is to track the actual progress of the project by employing various methods and practices to gather real-time performance data. This involves measuring current progress in terms of cost, time, and scope against the project baseline. The Work Breakdown Structure (WBS) and Cost Breakdown Structure (CBS) are essential tools in this process, as they help to organize and define the scope of work and budget allocations respectively. By comparing actual performance metrics with these structured plans, project managers can identify any discrepancies, assess the project's status, and determine whether the project is on track or if corrective measures are needed (De Marco, 2018).

Control, on the other hand, is the process that follows monitoring, aimed at addressing any deviations from the project plan. The primary goal of control is to analyze the causes of these variances and implement corrective actions to realign the project with its intended course. This may involve adjusting schedules, reassigning resources, or even redefining certain project elements to better meet the objectives. Effective control not only helps bring the project back on track but also plays a crucial role in preventive management, allowing project managers to anticipate potential issues before they become significant problems. Additionally, by continuously monitoring and controlling project activities, stakeholders are kept informed about the project's status, any concerns that arise, and the ongoing improvement efforts, thereby facilitating better communication and alignment with stakeholder expectations.

2.3. An Earned Value Management System

Earned Value Management (EVM) was developed in the late 1960s as a direct response to the increasing complexity of large-scale projects within the U.S. Department of Defense (DoD). Traditional project management approaches, which focused separately on cost or schedule, were proving inadequate for managing the intricate and interdependent activities characteristic of defense projects. Recognizing the need for a more integrated system, the DoD introduced EVM to combine scope, schedule, and cost into a single performance measurement framework. This effort was formalized in 1967 with the issuance of Instruction 7000.2, which laid down 35 criteria that contractors had to meet, forming the basis for what would eventually become the ANSI/EIA-748 standard for EVM. This initiative was first implemented by the U.S. Air Force under the Cost/Schedule Planning Control System (C/SPCS), which set the groundwork for EVM's development and broader application in various industries.(W. Fleming & M. Koppelman, 1998)

The practical necessity of EVM became particularly evident during the troubled A-12 "Avenger" aircraft program in the 1980s, where significant cost overruns and delays highlighted the limitations of existing project management tools. The failure of the A-12 program underscored the importance of integrating cost and schedule performance metrics, which EVM provided, offering a more reliable method of forecasting and managing project outcomes. In response to these challenges, EVM was increasingly adopted as a key project management tool within defense and beyond. Its integration into the Project Management Institute's (PMI) standards in 1999 marked its recognition as a best practice, establishing EVM as an essential methodology for project control across various sectors, including construction, manufacturing, and IT.

2.3.1. Fundamentals of Earned Value Management (EVM)

At the core of Earned Value Management (EVM) is the establishment of a Performance Measurement Baseline (PMB), which serves as a reference point for measuring project performance. The PMB integrates the project's scope, schedule, and cost, providing a comprehensive framework against which actual performance can be assessed. Monitoring the project's performance relative to the PMB is essential for identifying deviations early and making informed decisions to correct course when necessary.

Over time, several key metrics and formulas have been developed to quantify project performance and predict future outcomes using EVM. These metrics are essential for the measurement, forecasting, and analysis of a project's cost and schedule performance. Regardless of the type of project, EVM relies on four fundamental values: Planned Value (PV), Budget at Completion (BAC), Actual Cost (AC), and Earned Value (EV).(PMI, 2019)

- Planned Value (PV) also known as Budgeted Cost of Work Scheduled (BCWS)

also known as Budgeted Cost of Work Scheduled (BCWS), represents the budgeted cost for the work planned to be completed within a specific time frame. It essentially reflects the value of work that should have been completed by a particular date according to the project schedule. PV is crucial for establishing the PMB and serves as a benchmark against which actual performance is compared to highlight deviations.

$$PV = Planned\ Percentage\ of\ Completed\ Work \times Budget\ at\ Completion\ (BAC)$$

- Budget at Completion (BAC)

is the total budget allocated for the project, representing the sum of all the planned values. BAC is a critical metric as it defines the total financial commitment to the project and serves as a target for the project manager to achieve.

- Actual Cost (AC) also known as Actual Cost of Work Performed (ACWP)

measures the actual expenditure incurred for the work completed at any given point in time. This metric provides insight into the project's financial performance, allowing comparison between the planned budget and actual spending.

$$AC = Sum\ of\ Actual\ Costs\ for\ Work\ Completed$$

- Earned Value (EV) also known as Budgeted Cost of Work Performed (BCWP)

is the value of work actually completed in terms of the budget assigned to that work. EV is a vital metric as it quantifies the progress made and allows for the calculation of key performance indicators such as Cost Variance (CV) and Schedule Variance (SV).

$$BCWP = Actual\ Percentage\ of\ Work\ Performed \times Budget\ at\ Completion\ (BAC)$$

2.3.2. S-Curves for Cost Monitoring and Control

In the context of Earned Value Management (EVM), S-Curves serve as a powerful visual tool for representing the three key metrics: Planned Value (PV), Earned Value (EV), and Actual Cost (AC). These curves are plotted on a chart to facilitate a clear understanding of how these values evolve over time, providing a snapshot of the project's status. Typically, S-Curves are updated and analyzed both periodically (such as weekly or monthly) and cumulatively, with time represented on the x-axis and cost on the y-axis.

The reason these graphs are referred to as S-Curves lies in their distinctive shape, which resembles an "S". In the early stages of a project, the curves start off slowly, reflecting the gradual initiation of activities, often limited to preliminary tasks like market research or initial planning. As the project progresses, there is a period of accelerated growth, characterized by rapid completion of tasks and significant resource expenditure, which

forms the steep middle portion of the S-curve. This phase, often referred to as the "point of inflection," is critical as it represents the peak period of activity where most of the project's resources and budget are utilized. Following this peak, the rate of progress typically begins to slow, marking the project's maturity phase. During this time, the remaining tasks are usually of lower intensity, focusing on final reviews, quality checks, and project closeout activities, leading to the upper asymptote of the S-curve.(Narbaev & De Marco, 2017)

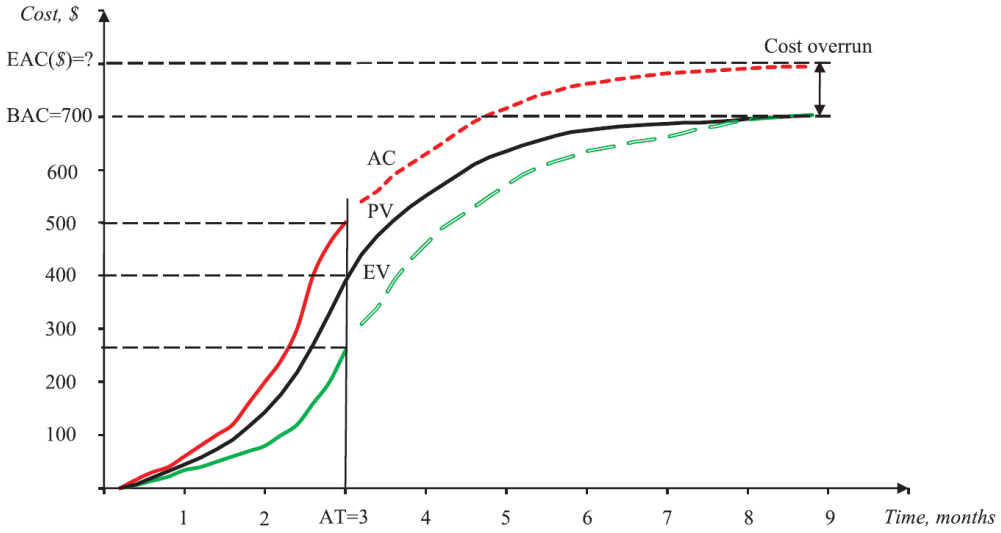

Figure 2-Typical S-Curve of a Project (Narbaev et al., 2024)

Each S-Curve represents a different aspect of project performance: PV remains fixed as it represents the budgeted plan, while EV and AC are updated as the project progresses. The comparison between EV and AC helps identify cost variances, while the comparison between EV and PV allows project managers to assess any schedule deviations. In the typical S-Curve graph shown above, you can observe how these values evolve over time. The graph highlights the potential for cost overruns when the Actual Cost (AC) exceeds the Earned Value (EV), as well as any delays if the Earned Value (EV) lags behind the Planned Value (PV). Analyzing these curves is crucial for identifying and addressing potential issues early, ensuring that the project stays on track.

2.3.3. EVM Performance Metrics

In Earned Value Management (EVM), performance metrics are critical tools that help project managers assess how well a project is adhering to its budget and schedule. These metrics provide a quantitative measure of the project's efficiency in utilizing its resources and adhering to its planned timeline. The key performance metrics include the Cost Performance Index (CPI), Schedule Performance Index (SPI), and Cost Ratio (CR).

- **Cost Performance Index (CPI)**

The Cost Performance Index (CPI) is a measure of the cost efficiency of a project. It compares the budgeted cost of work performed (BCWP) with the actual cost of work performed (ACWP). The CPI is calculated using the following formula:

$$CPI = \frac{EV}{AC} = \frac{BCWP}{ACWP}$$

A CPI value greater than 1 indicates that the project is under budget, meaning it is spending less than planned for the work completed. Conversely, a CPI value less than 1 indicates a budget overrun, suggesting that the project is spending more than anticipated.

- **Schedule Performance Index (SPI)**

The Schedule Performance Index (SPI) measures the efficiency of time utilization in the project. It compares the earned value (EV) with the planned value (PV) and is calculated as follows:

$$SPI = \frac{EV}{PV} = \frac{BCWP}{BCWS}$$

An SPI value greater than 1 indicates that the project is ahead of schedule, meaning more work has been completed than was planned for that time period. An SPI value less than 1 indicates a delay, meaning less work has been completed than planned.

- **Cost Ratio (CR)**

The Critical Ratio (CR) is a comprehensive metric used to assess both the cost and schedule performance of a project. It combines the Cost Performance Index (CPI) and the Schedule Performance Index (SPI) to provide a single indicator of overall project health. The formula for CR is:

$$CR = CPI \times SPI = \frac{EV}{AC} \times \frac{EV}{PV}$$

A CR value greater than 1 indicates that the project is spending less than planned, while a CR value less than 1 indicates that the project is spending more than anticipated.

- **Cost Variance (CV)**

The Cost Variance (CV) metric provides a monetary measure of cost performance. It shows the difference between the earned value and the actual cost incurred, and it is calculated using the following formula:

$$CV = EV - AC = BCWP - ACWP$$

- **Schedule Variance (SV)**

The Schedule Variance (SV) metric measures the difference between the earned value and the planned value. It reflects the degree to which the project is ahead or behind schedule and is calculated as:

$$SV = EV - PV = BCWP - BCWS$$

A positive SV indicates that the project is ahead of schedule, while a negative SV indicates that it is behind schedule.

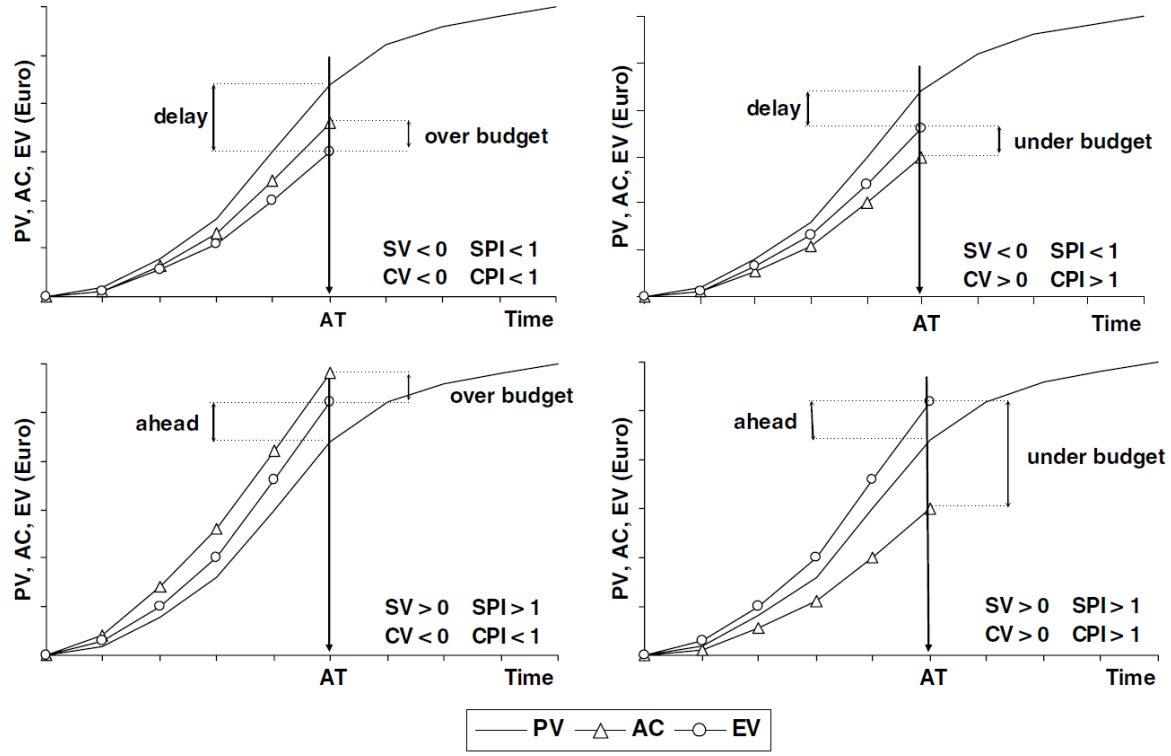

Figure 3-The EVM key parameters PV, AC, and EV for a project under four scenarios. Scenario 1: late project, over budget; Scenario (Vanhoucke & Vandevoorde, 2007)

2.3.4. Estimation At Compilation (EAC)

Estimate at Completion (EAC) is a critical metric in Earned Value Management (EVM) that provides a forecast of the total cost or time required to complete a project, based on its current performance. EAC helps project managers predict whether a project will be completed within the original budget and schedule, or if adjustments will be necessary.

- **Cost Estimate at Completion (CEAC)**

The EAC for cost forecasts the total cost of the project upon completion. It is particularly useful when the original assumptions about cost performance are no longer valid, requiring an updated estimate based on current progress and spending patterns.

There are several methods to calculate EAC for cost, depending on the assumptions made about future cost performance:

EAC using the Cost Performance Index (CPI), Assumes that the project's current cost performance will continue. The formula is:

$$EAC = AC + \frac{BAC}{CPI}$$

EAC using Critical Ratio (CR), Accounts for both cost and schedule performance. The formula is:

$$EAC = AC + \frac{BAC - EV}{CPI \times SPI}$$

These formulas provide forecasts based on the project's ongoing performance, helping to predict the final cost and allowing for necessary adjustments if cost or schedule variances are detected.

- **Time Estimation At Compilation(TEAC)**

The EAC for time forecasts the total duration required to complete the project, taking into account the current schedule performance. Similar to EAC for cost, it helps predict whether the project will meet its original deadline or if the timeline needs to be adjusted. The formula for TEAC is:

$$EAC(Time) = \frac{Total\ Planned\ Duration}{SPI}$$

If the SPI is greater than 1, the project is expected to be completed ahead of schedule, whereas an SPI less than 1 indicates a longer completion time than originally planned.

The EAC metrics for both cost and time are valuable tools for forecasting project outcomes, they are inherently dynamic and should be recalculated regularly as the project progresses. These metrics offer project managers crucial insights, enabling them to decide whether corrective actions are necessary to align the project with its objectives. EAC helps in managing stakeholder expectations and allows for

adjustments in resource allocation and project strategies to address any emerging variances.

However, despite its utility, EAC has limitations, particularly in its reliance on historical performance and linear assumptions about future trends. This approach may not always capture the complexities and uncertainties of real-world projects, especially in environments where project variables are highly volatile or interdependent. As such, there is a growing need for more advanced forecasting methods. This is where the introduction of Artificial Intelligence (AI) into cost forecasting comes into play, offering the potential to enhance accuracy and adaptability beyond what traditional EAC calculations can achieve.

2.4. Problem Statement

While Earned Value Management (EVM) is a well-established methodology in project management, it faces several critical limitations that impact its effectiveness in cost forecasting. These limitations often result in inaccurate predictions, particularly in complex and dynamic project environments. Below are the key challenges associated with traditional EVM methods:

- **Assumption of Linear Cost Growth**

One of the fundamental assumptions of EVM is that project costs grow in a linear fashion over time. However, this assumption is frequently contradicted by the actual cost behavior observed in real-world projects, which often follows a non-linear, S-shaped curve. This discrepancy is particularly evident in large-scale projects, where initial spending is slow, accelerates during peak periods of activity, and tapers off as the project nears completion. The failure of EVM to account for this non-linear growth can lead to significant inaccuracies in cost forecasting, especially in the early stages of a project when data is limited (Narbaev & De Marco, 2017; Pellerin & Perrier, 2019).

- **Inadequate Early-Stage Forecasting**

EVM's reliance on historical performance data can be particularly problematic during the early stages of a project. At this point, the data available is often insufficient to provide reliable forecasts, leading to estimates that may be overly optimistic or pessimistic. This limitation is exacerbated by the fact that EVM does not adequately adjust for the limited data points available early on, resulting in forecasts that may not accurately reflect the project's likely cost trajectory (Kim & Reinschmidt, 2011).

- **Overlooking Performance Trends and Future Risks**

Traditional EVM models typically assume a static labor profile and cost performance, which do not account for the dynamic nature of many projects. This assumption

overlooks critical performance trends and potential future risks that could significantly alter the project's cost and timeline. As a result, EVM can produce forecasts that give a false sense of certainty, leading project managers to underestimate the potential for cost overruns and delays (Ottaviani & Marco, 2021).

- **Inflexibility in Dynamic Environments**

EVM is inherently backward-looking, relying heavily on past performance to predict future outcomes. This approach can be problematic in dynamic project environments where conditions are constantly changing. EVM's static formulas may fail to capture these changes, leading to forecasts that do not accurately reflect the project's evolving realities. This inflexibility can result in flawed decision-making and missed opportunities to adjust course early enough to avoid project failure (Narbaev et al., 2024b).

Given these limitations, there is a pressing need for more advanced forecasting methodologies that can adapt to the complexities of modern project environments. Artificial Intelligence (AI) offers a promising solution to this challenge. By leveraging AI's ability to analyze large datasets and detect patterns that traditional methods might overlook, project managers can achieve more accurate and reliable cost forecasts. This research aims to explore the integration of AI into cost forecasting models, with the goal of enhancing the precision and adaptability of project management practices, leading to better decision-making and improved project outcomes.

2.5. Introduction to Machine Learning (ML) in Cost Forecasting

2.5.1. Machine Learning Basics

Machine Learning (ML) is a branch of artificial intelligence that focuses on developing algorithms and statistical models that enable computers to perform specific tasks without using explicit instructions. Instead of relying on hard-coded rules, ML systems learn from data by identifying patterns, making inferences, and improving their performance over time as they are exposed to more data.

The core idea behind ML is to allow computers to learn from experience, much like humans do. By processing large datasets, ML algorithms can make predictions, detect patterns, and even make decisions based on the information they have been trained on. This ability to learn and adapt makes ML particularly powerful in areas where traditional programming falls short, such as recognizing complex patterns, adapting to new situations, or handling vast amounts of data that would be impractical for humans to analyze manually.

Machine Learning is increasingly being used across various industries to automate tasks, improve decision-making, and drive innovations. Whether it's in healthcare for

diagnosing diseases, in finance for detecting fraudulent transactions, or in project management for forecasting costs and schedules, ML has become a critical tool for enhancing efficiency and accuracy in complex systems.

2.5.2. Types of Machine Learning Models

Machine Learning (ML) encompasses a wide variety of models and algorithms, each designed to address different types of tasks and data structures. These models can be broadly categorized into several types, depending on how they learn from data and the kinds of problems they are intended to solve. The main types of ML models include supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning.(Mahesh, 2020)

- **Supervised Learning**

Supervised learning is one of the most common types of ML, where the model is trained on a labeled dataset. Each example in the dataset consists of an input and a corresponding output label. The model learns to map inputs to outputs by minimizing the error between the predicted and actual outcomes. This type of learning is widely used for tasks such as classification and regression. For example, in project management, supervised learning models can predict project costs or completion times by learning from historical data .

- **Unsupervised Learning**

In unsupervised learning, the model is given a dataset without labeled outputs and must identify patterns or structures within the data. This type of learning is often used for clustering, where the model groups similar data points together, or for dimensionality reduction, which simplifies the data while retaining essential information. Unsupervised learning is useful in situations where the underlying structure of the data is unknown and needs to be explored. For instance, clustering can help segment projects based on similar characteristics, which can then inform resource allocation strategies .

- **Semi-Supervised Learning**

Semi-supervised learning is a hybrid approach that falls between supervised and unsupervised learning. In this method, the model is trained on a small amount of labeled data along with a larger set of unlabeled data. This approach is particularly useful when labeling data is expensive or time-consuming, as it allows the model to learn effectively with less labeled data. Semi-supervised learning can improve the accuracy of predictions when fully labeled datasets are not available, which is often the case in complex project management scenarios .

- **Reinforcement Learning**

Reinforcement learning involves training a model through interactions with an environment, where the model receives feedback in the form of rewards or penalties based on its actions. The goal is to learn a strategy that maximizes cumulative rewards over time. Reinforcement learning is especially useful in situations where decisions must be made sequentially, such as in robotics or autonomous systems. Although less common in project management, reinforcement learning can be applied to optimize decision-making processes, such as adjusting schedules or resources in response to changing project conditions.

2.5.3. How Machine Learning Models Work

Machine Learning (ML) operates through a systematic process that transforms raw data into predictive models capable of making informed decisions. The process can be broken down into several key steps, each essential to developing a robust and accurate ML model.

- Data Collection: The journey begins with gathering a dataset that is representative of the problem you wish to solve. This dataset serves as the foundation for the entire ML process, containing the features (input variables) and labels (output variables) needed to train the model.
- Data Preprocessing: Raw data often contains noise, inconsistencies, or irrelevant information. Therefore, the next step is data preprocessing, which includes:
    - Data Cleaning: Removing or correcting erroneous data points.
    - Data Transformation: Converting data into a suitable format for analysis, such as normalizing numerical values or encoding categorical variables.
    - Data Reduction: Simplifying the dataset by reducing its dimensionality or selecting relevant features to improve model efficiency and performance.
- Data Splitting: The pre-processed data is then split into three distinct sets:
    - Training Set: Used to train the model by adjusting its parameters to learn from the data.
    - Validation Set: Helps tune the model's hyperparameters and prevents overfitting by providing a separate dataset to test the model during training.
    - Test Set: Used to evaluate the final model's performance on unseen data, ensuring that it generalizes well to new inputs.

- Model Selection and Training: With the training data ready, the next step is to select an appropriate ML algorithm that fits the problem's requirements. The model is then trained using the training set, where it learns to map input features to the desired output labels by minimizing errors between predictions and actual outcomes.
- Model Evaluation: Once trained, the model is evaluated using the test set to determine its accuracy, precision, recall, and other relevant metrics. This step is crucial to assess how well the model performs on new data and to identify any potential issues, such as overfitting or underfitting.
- Hyperparameter Tuning: If necessary, the model's hyperparameters are fine-tuned to optimize its performance. This process often involves iterative testing and adjustment using the validation set to find the best configuration.
- Model Deployment: After achieving satisfactory performance, the model is deployed for practical use. It is now capable of making predictions on new data, driving decisions in real-world applications.
- Model Monitoring and Maintenance: Finally, the deployed model must be continuously monitored to ensure it maintains accuracy and relevance over time. As new data becomes available, the model may need to be retrained or updated to adapt to changing conditions.

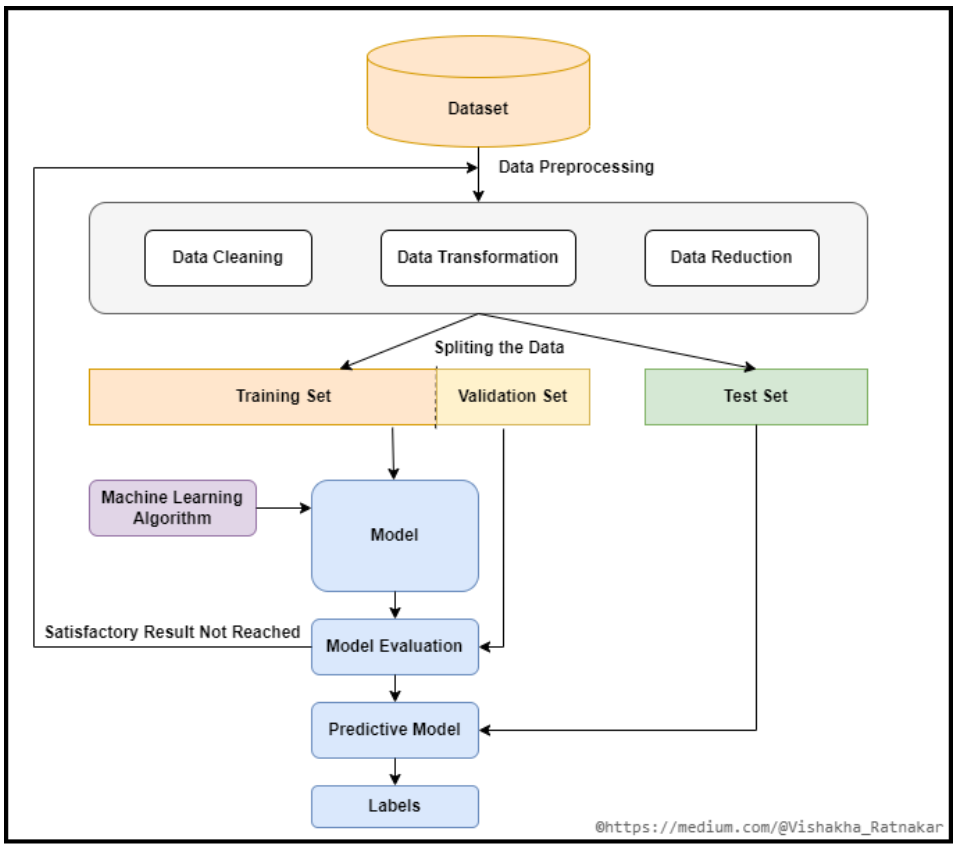

Figure 4-Supervised Learning Algorithms Workflow

2.5.4. Application of Machine Learning in Project Cost Forecasting

As discussed in the preceding sections, Machine Learning (ML) has increasingly been recognized for its potential to significantly enhance cost forecasting accuracy in project management. Building on the limitations of traditional methods like Earned Value Management (EVM) highlighted earlier, ML models offer a sophisticated alternative, particularly in managing complex and nonlinear relationships within project data. This section reviews several key studies that have employed ML for cost forecasting, focusing on the phases of the project where these models were applied, the data and inputs utilized, and the outcomes achieved.

Given that ML models in the conceptual phase primarily rely on project-based properties rather than EVM indexes, our focus here is on studies that apply ML models during the execution phase of projects. In the execution phase, ML models can directly incorporate EVM indexes along with other real-time project data, providing a more dynamic and responsive approach to cost forecasting.

Pewdum, Rujirayanyong, and Sooksatra (2009) explored the use of Artificial Neural Networks (ANN) during the execution phase of highway construction projects in Thailand. The study utilized a dataset of 1,022 valid data patterns from 51 projects,

incorporating inputs such as traffic volume, topography, weather conditions, contract duration, construction budget, and the percentage of planned versus actual completion.

The ANN models significantly outperformed traditional Earned Value Management (EVM) methods, achieving a Mean Absolute Percentage Error (MAPE) of 2.44% for budget forecasting and 2.77% for duration forecasting on specific projects. The study concluded that ANN models provide more accurate and stable forecasts than traditional methods, highlighting the potential of Machine Learning to improve cost and duration predictions during the execution phase of construction projects.

Ottaviani and De Marco (2021) explored the improvement of cost forecasting accuracy using a Multiple Linear Regression (MLR) model within the framework of Earned Value Management (EVM). The study focused on enhancing the Estimate at Completion (EAC) metric, which is crucial during the execution phase of a project. Data was sourced from 29 real-life projects, comprising a total of 805 observations. The projects spanned various industries, allowing the model to be tested across diverse contexts.

The input parameters for the MLR model included key EVM variables such as the Planned Value (PV), Earned Value (EV), Actual Cost (AC), and other relevant project-specific factors like Schedule Performance Index (SPI) and Cost Performance Index (CPI). The model was developed to address the limitations of traditional EAC calculations by incorporating these variables into a regression framework, thereby improving both accuracy and reducing error variance.

The study's findings indicated that the MLR model provided a significant improvement in forecasting accuracy over traditional EVM methods. Specifically, the model achieved a Mean Absolute Percentage Error (MAPE) of 13.91%, compared to the 15.76% MAPE obtained using traditional EAC calculations. Additionally, the standard deviation of the error was reduced by approximately 9 percentage points, further underscoring the model’s enhanced reliability. These results suggest that integrating MLR models into the EVM framework can substantially improve cost forecasting during the project execution phase.

Timur et al. (2024) conducted a study aimed at improving cost forecasting accuracy during the execution phase of projects through the application of the XGBoost machine learning model. The study utilized a robust dataset consisting of 110 real-life projects, which included 1,268 cost data points. The data was strategically segmented into early, middle, and late stages of the project lifecycle. This stage-based approach allowed for a nuanced analysis of forecasting accuracy, providing insights into how cost predictions could be refined as the project progressed.

The XGBoost model was trained using essential input parameters derived from Earned Value Management (EVM), including Actual Cost (AC), Budget at Completion (BAC),

Earned Value (EV), and the Cost Performance Index (CPI). These inputs enabled the model to effectively learn from historical project data, thereby enhancing its predictive capabilities. The model's performance was compared against traditional EVM methods and other machine learning models like Random Forest, Support Vector Regression, LightGBM, and CatBoost.

The results of the study were compelling, with the XGBoost model significantly outperforming the traditional EVM methods. The Mean Absolute Percentage Error (MAPE) achieved by the XGBoost model ranged from 6.53% to 9.70% in the early stage, 6.42% to 8.57% in the middle stage, and 6.22% to 8.28% in the late stage. This demonstrated the model's effectiveness in providing highly accurate cost forecasts, particularly in the critical early and middle stages of the project. The study highlighted the potential of advanced machine learning models like XGBoost to offer project managers more reliable tools for cost estimation, ultimately leading to better decision-making and project outcomes.

Wauters and Vanhoucke (2014)explored the application of Support Vector Regression (SVR), a machine learning technique, for improving the accuracy of cost and time forecasting in project control, particularly within the context of Earned Value Management (EVM). The study focused on the execution phase of projects and used a dataset generated through Monte Carlo simulations, which introduced variability in project activity durations and costs across a diverse set of 900 project networks.

The SVR model was trained using input parameters derived from EVM metrics, including the Schedule Performance Index (SPI), Cost Performance Index (CPI), and Earned Schedule (ES). The data were divided into training and test sets, with a robust cross-validation and grid search procedure employed to fine-tune the model's parameters. This allowed the model to learn from historical project data and improve its predictive accuracy.

The results demonstrated that the SVR model significantly outperformed traditional EVM-based forecasting methods. Specifically, the SVR model achieved a lower Mean Absolute Percentage Error (MAPE) compared to other methods, with MAPE values of 1.28% for cost forecasting under the best conditions. The study also highlighted the robustness of the SVR model, noting that it maintained superior performance even when there were discrepancies between the training and test datasets. This research underscores the potential of SVR as a valuable tool in project management for enhancing the reliability of cost and time forecasts during the execution phase.

Capone et al. (2024) conducted a study to enhance cost forecasting accuracy during the execution phase of projects by applying Machine Learning models, specifically XGBoost and Random Forest. The study utilized data from 110 completed projects globally, with key variables such as Actual Cost (AC), Planned Value (PV), Earned Value

(EV), and tracking period information. The dataset was divided into three distinct stages—early (1-29%), middle (30-69%), and late (70-100%)—to allow for a detailed, stage-based analysis of forecasting accuracy.

The ML models were trained using normalized data from Earned Value Management (EVM) metrics, including AC, Budget at Completion (BAC), EV, Cost Performance Index (CPI), and Schedule Performance Index (SPI). The study employed a rigorous approach to model training, with data split into 75% for training and 25% for testing. Hyperparameter tuning, particularly for XGBoost, played a critical role in preventing overfitting and optimizing the model's accuracy across different project stages.

The results of the study indicated that XGBoost outperformed Random Forest and traditional EVM methods, particularly in the early and middle stages of project execution. The XGBoost model achieved a Mean Absolute Percentage Error (MAPE) ranging from 6.46% to 9.26% in the early stage, 6.47% to 8.67% in the middle stage, and 6.22% to 8.32% in the late stage. These findings underscore the effectiveness of ML models like XGBoost in providing more accurate and reliable cost forecasts, which are essential for informed decision-making during the execution phase of projects.

Authors(Year)	ML Model Used	Data Source	Input Parameters
Pewdum, R., Rujirayanyong, T., & Sooksatra, V. (2009)	Artificial Neural Network	51 highway construction projects in Thailand (1,022 data points)	Project Physical Feature+PD(Planned Duration)+BAC+WP+WS
Ottaviani, R., & De Marco, A. (2021)	Multiple Linear Regression	29 real-life projects (805 observations)	CPI+WP+fEAC
Timur, O., Ong, S., Lu, H., & Matous, P. (2024)	XGBoost (primary model), Random Forest, Support Vector Regression (SVR), LightGBM, and CatBoost.	110 global projects (1,268 data points)	AC+BAC+EV+CPI
Wauters, M., & Vanhoucke, M. (2014)	Support Vector Regression (SVR)	Simulated data via Monte Carlo (900 project networks)	SPI,$SPI_{(t)}$,CPI,ES
Capone, A., Greco, G., & Palumbo, G. (2024)	XGBoost and Random Forest	110 global projects	AC, BAC, EV, CPI, SPI

Table 1-Comparison of Project Cost Forecasting Studies

The application of Machine Learning (ML) in project cost forecasting has shown promising results, particularly in comparison to traditional methods like Earned Value Management (EVM). Studies such as those by Timur et al. (2024) and Pewdum et al. (2009) have demonstrated the superior accuracy of ML models, with significant reductions in forecasting errors during project execution phases. However, despite these advancements, the accuracy of these models often hinges on the quality and comprehensiveness of the input features used.

Many of these studies primarily rely on conventional EVM metrics and other readily available project data, which, while useful, may not fully encapsulate the complexity of project networks. For instance, traditional metrics like Actual Cost (AC) and Earned Value (EV) focus on financial and schedule performance but fail to consider the structural characteristics of project tasks and their interdependencies. This omission can lead to less accurate predictions in complex or irregular projects, where the sequence and regularity of tasks can significantly impact project outcomes.

Given these limitations, there is a growing recognition in the literature of the need for additional features to better capture project structures' intricacies. This is where the introduction of the Project Regularity Index (RI) and Project Seriality (SP) becomes crucial. These Indices offer a more nuanced understanding of project networks by evaluating the regularity and sequence of tasks, thereby providing ML models with richer, more detailed inputs. Incorporating these new features could address the gaps identified in previous studies and lead to even more accurate cost forecasting in project management.

2.5.5. Project Seriality (SP)

Project seriality refers to the network structure of a project, indicating how closely the network aligns with either a completely serial or parallel configuration. This is measured using the serial/parallel indicator (SP), which can range from 0 to 1. An SP value of 0 represents a fully parallel project, while an SP value of 1 signifies a completely serial project (Batselier & Vanhoucke, 2017). Projects with SP values between these extremes have network structures that are closer to either a serial or parallel arrangement.

The formula for calculating the SP is as follows:

$$SP = \frac{s_n - 1}{t_n - 1}$$

In this formula, s_n represents the maximum number of subsequent activities in the network (also known as the maximum progressive level), and t_n is the total number of activities. It is important to note that for a project consisting of only one activity, the SP value is by definition equal to 1, indicating a completely serial project.

2.5.6. Project Regularity (RI)

The concept of project regularity is a relatively new addition to the literature. To provide context, (Jacob & Kane, 2004) compared different time forecasting approaches—ESM, EDM, and PVM—and concluded that "as long as the planned value (PV) is linear, all formulas will always yield exact results, but if the PV is non-linear, errors or discrepancies could be introduced." Thus, a project with a perfectly linear PV curve can be considered fully regular, with minimal potential for forecasting errors (a viewpoint also supported by Vandevoorde & Vanhoucke (2006). Irregularities in a project are therefore defined by deviations of the actual PV curve from this ideal linear form.

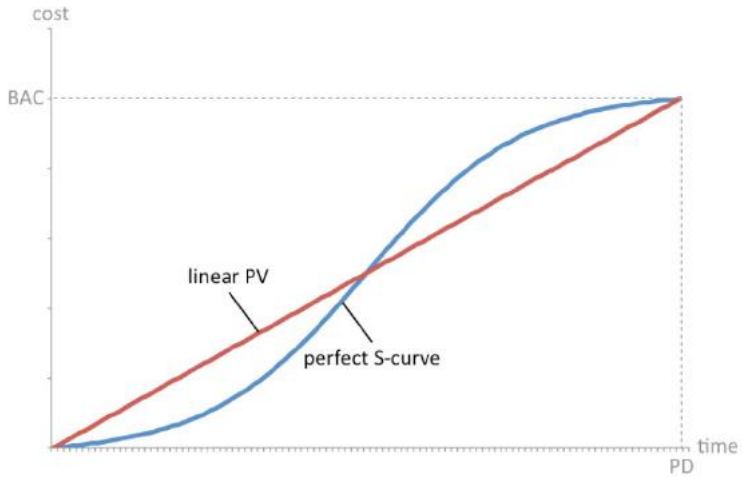

Figure 5-Linear PV and perfect S-curve comparison (Batselier & Vanhoucke, 2017)

To quantify the regularity of a project, a new metric called the regular/irregular indicator (RI) has been introduced(Batselier & Vanhoucke, 2017). This indicator is conceptually similar to the serial/parallel indicator (SP). Just as a completely serial project has an SP value of 1, a project with a perfectly linear PV curve is characterized by an RI of 1. Conversely, a project that is highly irregular—where the earned value (EV) remains zero for most of the project and only spikes to the budget at completion (BAC) near the end—would have an RI value of 0. Most projects fall somewhere between these two extremes, with their RI values calculated using the following formula:

$$RI = 1 - \frac{\sum_{i=1}^{r} m_i - \sum_{i=1}^{r} a_i}{\sum_{i=1}^{r} m_i}$$

In this formula, m_i represents the maximum possible deviation, and a_i represents the actual deviation of the project's PV curve from a perfectly linear curve at specific time points, i, across r equidistant evaluation points. It is important to note that r refers to the shape of the PV curve, rather than the number of tracking periods.

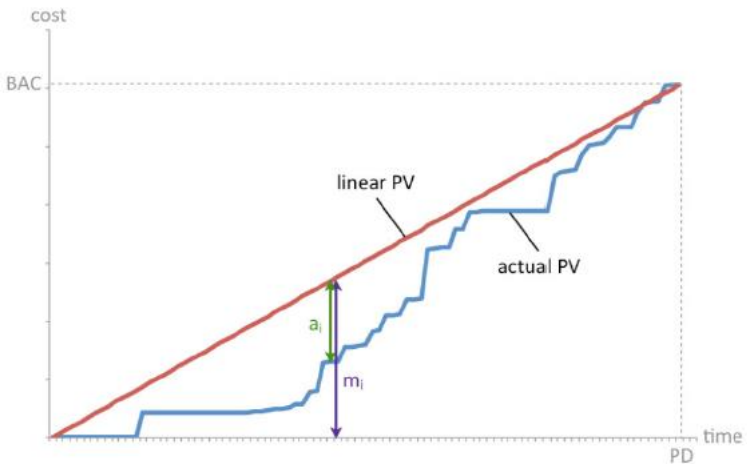

Figure 6-RI calculation (Batselier & Vanhoucke, 2017)

3. Methodology

This chapter details the methodological framework adopted in this study to assess and compare the performance of various machine learning algorithms in project cost forecasting. The central objective is to determine whether machine learning models can enhance forecasting accuracy compared to the traditional Earned Value Management (EVM) method. By doing so, the study aims to identify the most effective approach for predicting project costs, particularly in the early stages where traditional methods often fall short due to linear assumptions and limited data.

To achieve this, we implement and evaluate six distinct machine learning algorithms: XGBoost, Extremely Randomized Trees (ExtraTrees), Random Forest, Support Vector Machine (SVM), Light Gradient Boosting Machine (LightGBM), and K-Nearest Neighbors (KNN). These models are benchmarked against the traditional EVM approach to determine if they offer superior accuracy in forecasting project costs.

In addition to comparing machine learning models with the EVM method, this study also explores the impact of two project characteristics—Project Regularity (RI) and Project Seriality (SP)—as inputs in the machine learning models. The secondary objective is to assess the effectiveness of these indicators in enhancing the predictive power of the models. By systematically integrating and evaluating these inputs across different models, we aim to identify which characteristics, if any, contribute to more reliable cost forecasts.

In this section, we will provide a detailed explanation of the entire methodological process, beginning with data collection and filtering. First, we will describe how the data was gathered from real-life projects and the criteria used to filter and clean the dataset for accuracy. Following that, we will outline the steps taken to calculate the accuracy of the traditional Earned Value Management (EVM) method based on the refined data, which serves as a baseline for comparison with the machine learning models.

Next, we will introduce and describe the inputs used in the machine learning models, highlighting both the dynamic and static features that were considered. This will be followed by a discussion on the implementation of these models in Python, including the specific algorithms utilized and the coding framework employed.

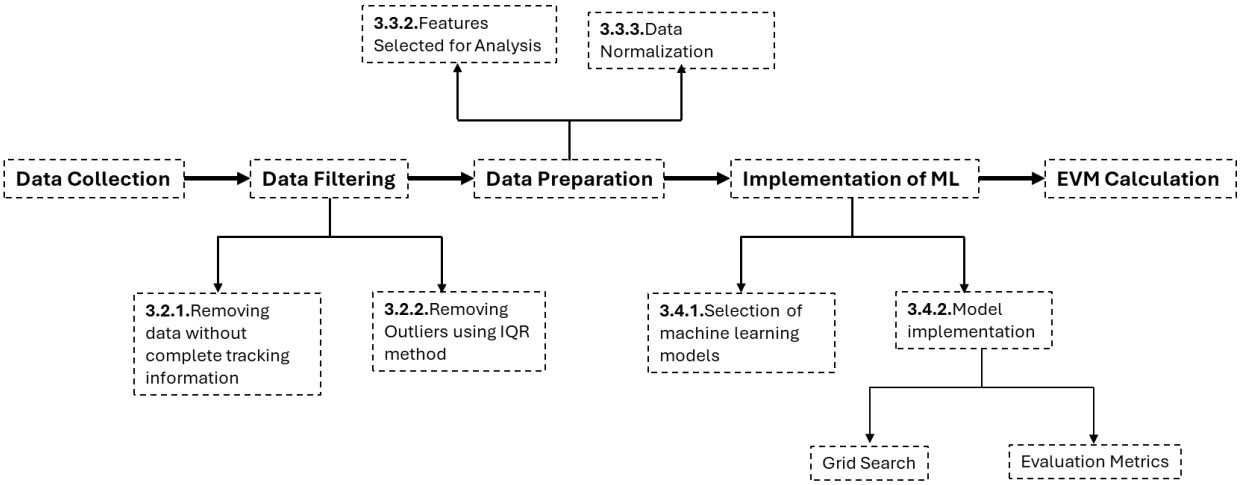

Figure 7-Methodology Overview Diagram

We will then detail the process of hyperparameter tuning, explaining how grid search was used to optimize the models for better performance. Finally, we will introduce the evaluation indicators, such as Mean Absolute Percentage Error (MAPE) and Normalized Root Mean Squared Error (NRMSE), which were employed to assess and compare the forecasting accuracy of the models.

3.1. Data collection

The data utilized in this study is sourced from the Dynamic Scheduling Library (DSLIB), a comprehensive empirical project database developed by the Operations Research and Scheduling research group at Ghent University.[] This extensive database includes detailed information on 181 real-life projects across various industries, making it a valuable resource for analyzing project management practices and outcomes.

The dataset contains data that spans across different stages of each project's lifecycle:

- **Planned Values:** These represent the data calculated before the project execution begins, including key planning documents and metrics such as:
    - **Budget at Completion (BAC):** The total budget planned for the project.
    - **Gantt Chart:** The project's schedule, detailing the planned sequence and timing of tasks.
    - **Planned Project Duration:** The expected timeframe for completing the project.

- **Tracking Values:** These are recorded at various points during the project's execution. The frequency and pattern of these tracking points vary between projects—some projects record data weekly, others monthly, and some have irregular intervals. The tracking data includes:
    - **Earned Value (EV):** The value of the work performed up to each specific tracking point.
    - **Actual Cost (AC):** The cost incurred for the work completed at each tracking point.
    - **Cost Performance Index (CPI):** A measure of cost efficiency calculated as the ratio of earned value to actual cost.
    - **Schedule Performance Index (SPI):** A measure of schedule efficiency calculated as the ratio of earned value to planned value.
- **Final Outcome Values:** These capture the actual results once the project has been fully completed, providing the basis for comparing the initial plans with the actual outcomes. These include:
    - **Real Duration:** The actual time taken to complete the project.
    - **Real Cost at Completion:** The total cost incurred by the end of the project.

In total, the dataset offers a comprehensive view of 181 projects, capturing the planned intentions, the ongoing tracking of progress, and the final outcomes. To illustrate how these stages are documented, A detailed example of a project will be provided below, including tables that display the planned values, actual values at various tracking points, and the final outcomes.

CODE	PROJECT NAME	SECTOR	PD (DAYS)	BAC	SP	RI	DURATION	COST
C2011-07	Patient Transport System	IT	389	180,759 €	70%	74%	445	191,065 €

Table 2-the planned value of the project C2011-07

NAME	START TRACKING PERIOD	STATUS DATE	PLANNED VALUE (PV)	EARNED VALUE (EV)	ACTUAL COST (AC)	COST PERFORMANCE INDEX(CPI)	SCHEDULE PERFORMANCE INDEX (SPI)
TP1	01/01/2010 8:00	28/01/2010 17:00	5,262.40€	5,262.40€	5,262.40€	1	1
TP2	28/01/2010 17:00	25/02/2010 17:00	5,262.40€	5,262.40€	5,262.40€	1	1
TP3	25/02/2010 17:00	25/03/2010 17:00	9,434.40€	9,434.40€	10,155.84€	0.92896304	1
…..	…..	…..	…..	…..	…..	…..	…..
TP21	14/07/2011 17:00	11/08/2011 17:00	180,759.44€	170,569.08€	178,592.92€	0.955071903	0.943624742
TP22	11/08/2011 17:00	08/09/2011 17:00	180,759.44€	172,251.28€	181,673.06€	0.948138816	0.952931034
TP23	08/09/2011 17:00	15/09/2011 20:00	180,759.44€	180,759.44€	191,065.06€	0.946062247	1

Table 3-the Tracking values of the project C2011-07

CODE	PROJECT NAME	REAL COST	REAL DURATION
C2011-07	Patient Transport System	191,065 €	445

Table 4-the Final Outcome of the project C2011-07

3.2. Data filtering

3.2.1. Removing data without complete tracking information

Following the initial collection of data, a filtering process was applied to ensure the quality and relevance of the dataset for subsequent analysis. The original dataset comprised 181 projects; however, not all of these projects provided the necessary completeness or consistency required for accurate and meaningful cost forecasting.

The first step in the filtering process involved selecting projects that contained complete and essential data. Projects that lacked critical information—such as the Budget at Completion (BAC), Project Schedule, or key tracking metrics like Earned Value (EV) and Actual Cost (AC)—were excluded from the analysis. This initial screening reduced the dataset from 181 projects to 103, focusing only on those projects that had comprehensive data across all stages of their lifecycle.

3.2.2. Removing outliers using IQR Method

After selecting 103 projects with complete tracking information, the next step in the data filtering process involved identifying and removing outliers to ensure the accuracy and reliability of the dataset. To accomplish this, we utilized the **Interquartile Range (IQR) Method**, a robust statistical technique widely used for detecting outliers in various datasets.

IQR method is particularly effective because it does not rely on the assumption of a normal distribution, making it well-suited for datasets that may exhibit skewed or non-normal characteristics.[] The method works by dividing the continuous range of data into quartiles—segments that represent the distribution of the dataset. The first quartile (Q1) marks the 25th percentile, while the third quartile (Q3) marks the 75th percentile, with the interquartile range (IQR) being the difference between these two values:

$$IQR = Q3 - Q1$$

The IQR represents the spread of the middle 50% of the data, also known as the "middle fifty." This range is crucial for identifying the central tendency of the data and detecting outliers—values that fall significantly outside this central range.

To determine the thresholds for outlier detection, the IQR method calculates the lower and upper bounds using the following formulas:

$$Lower\ Bound = Q1 - 1.5 \times IQR$$

$$Upper\ Bound = Q3 + 1.5 \times IQR$$

Any data point that lies below the lower bound or above the upper bound is considered an outlier. The multiplier of 1.5 is a conventional choice that balances the need to identify genuine outliers while maintaining the integrity of the dataset by not excluding too many data points.

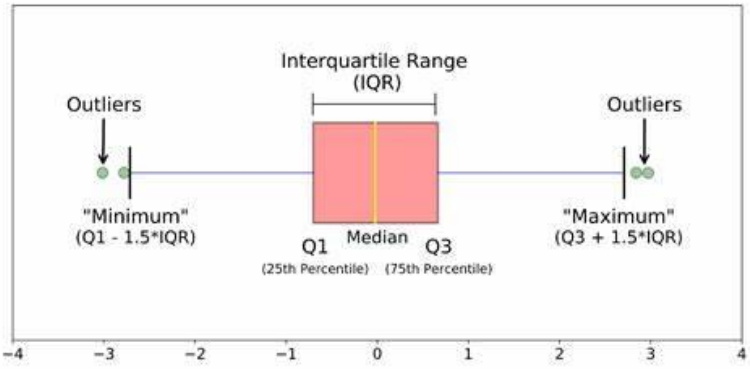

Figure 8-Interquartile Range (IQR) Method for Outlier Detection

In this study, we applied the IQR method to the "Real Cost" values of the projects in our dataset. The upper bound, calculated using the IQR method, was $3,376,316. As shown

in the plot, 11 projects had a "Real Cost" exceeding this upper bound, indicating that these projects are outliers with significantly higher costs.

Notably, there were no projects that fell below the lower bound, meaning no outliers were identified on the lower end of the cost spectrum. By removing these 11 high-cost outlier projects, the dataset was refined from 101 projects to a more consistent set of 90 projects. This careful curation of the data ensures that the subsequent analysis will be based on a reliable and representative sample, minimizing the potential for skewed results due to extreme outliers.

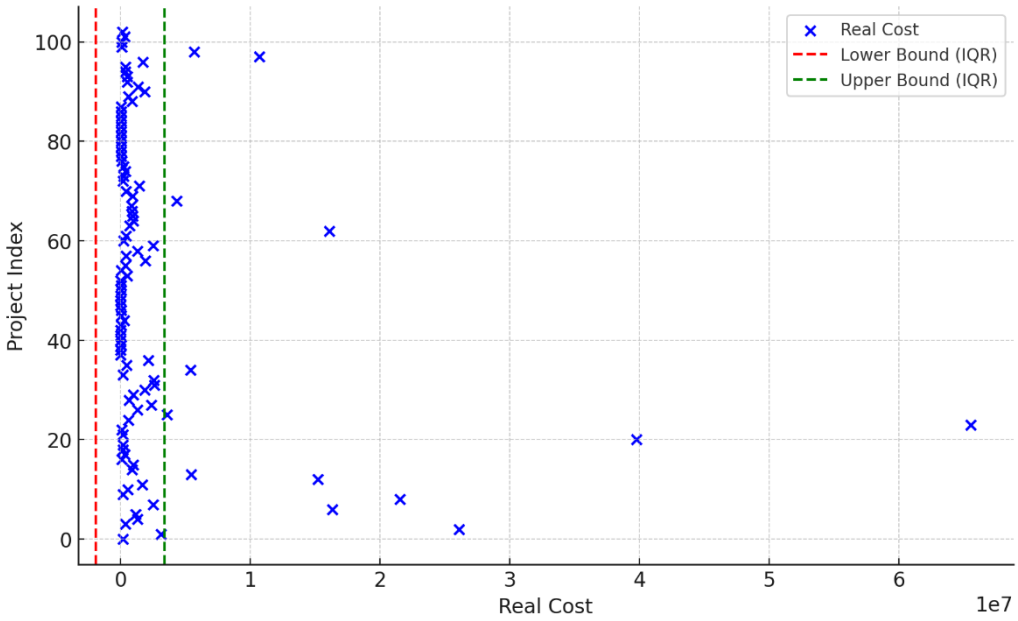

Figure 9-Real Cost of Projects with IQR-Based Outlier Boundaries.

3.3. Data Preparation

3.3.1. Uploading Data into Excel Spreadsheets

Following the data filtering process, the dataset was refined to 90 projects. To facilitate further analysis, all relevant data was organized and systematically uploaded into Excel spreadsheets. This encompassed the tracking values recorded at various points during the project's execution, along with the planned values and the final outcomes associated with each tracking period.

By organizing the data into Excel spreadsheets, we created a structured and accessible format that facilitated further analysis. This setup enabled us to apply machine learning models to each individual tracking point, using the performance metric indicators and planned values of each project. This approach ensures that the models can accurately assess project performance over time, providing deeper insights into how each project progresses through its lifecycle.

3.3.2. Features Selected for Analysis

The features retained for analysis were categorized into two main types: static data and dynamic data. This distinction allowed for a comprehensive assessment of both the planned aspects of each project and the actual performance over time. These features serve as the inputs for the machine learning models, which will be used to predict the final outcomes of the projects.

- Static Data: These are the planned values related to the project, which remain constant across all tracking points. The static data includes:
    - Budget at Completion (BAC)
    - Project Regularity (RI)
    - Project Seriality (SP)
- Dynamic Data: These are the tracking values that change over time, reflecting the actual performance of the project at each tracking point. The dynamic data includes:
    - Earned Value (EV)
    - Actual Cost (AC)
    - Cost Performance Index (CI)

These static and dynamic data points are the inputs to the machine learning models. The models will utilize these inputs to predict the final outcome, specifically the Real Cost at Completion, which serves as the target variable for the analysis.

By retaining both static and dynamic data, and clearly defining the target, we ensured that the dataset is well-prepared for applying machine learning techniques to predict project outcomes accurately. In the table below, we can see the complete set of data used as inputs and targets for the machine-learning models.

Inputs						**Target**
Dynamic Inputs			Static Inputs			Final Outcome
CI	**AC**	**EV**	**BAC**	**RI**	**SP**	**Real Cost**
1	$ 5,262.40	$ 5,262.40	$ 180,759.00	0.74	0.7	$ 191,065.06
1	$ 5,262.40	$ 5,262.40	$ 180,759.00	0.74	0.7	$ 191,065.06
0.928963	$ 10,155.84	$ 9,434.40	$ 180,759.00	0.74	0.7	$ 191,065.06
…..	…..	…..	…..	…..	…..	…..
1	$ 5,716.85	$ 5,716.85	$ 3,027,133.00	0.75	0.41	$ 3,102,395.91
0.895511	$ 283,418.65	$ 253,804.57	$ 3,027,133.00	0.75	0.41	$ 3,102,395.91
0.94356	$ 685,780.46	$ 647,075.30	$ 3,027,133.00	0.75	0.41	$ 3,102,395.91
…..	…..	…..	…..	…..	…..	…..

Table 6-Inputs and Target Variables for Machine Learning Models (Before Normalization)

3.3.3. Data Normalization

To ensure that the data was suitable for machine learning analysis and to make comparisons across projects of different scales meaningful, a data normalization process was applied. Normalization is essential to bring all the input features onto a common scale, preventing features with larger numerical ranges from disproportionately influencing the results of the machine learning models.

In this study, normalization was performed using the Budget at Completion (BAC) as the base value. Specifically, key dynamic features such as Earned Value (EV), Actual Cost (AC), Cost at Completion (CAC), and the target variable Real Cost at Completion were normalized by dividing each of these values by the corresponding BAC for each project. This method effectively scaled the data to unity, where all values are expressed as fractions or multiples of the BAC.

For example, after normalization:

- A normalized Earned Value (EV) of 0.5 indicates that 50% of the planned budget has been earned at a given tracking point.
- A normalized Actual Cost (AC) of 0.8 suggests that 80% of the planned budget has been spent.
- A normalized Real Cost at Completion of 1.1 would indicate that the project cost exceeded the planned budget by 10%.

This normalization process ensured that the machine learning models could accurately compare and analyze projects of varying sizes without bias from the absolute scale of the values. By transforming both the input features and the target variable into a consistent range, we enhanced the robustness and reliability of the predictive models.

The normalized data, including the Real Cost at Completion as the normalized target, was then used for training and evaluating the machine learning models. The table below shows the normalized data used as inputs and targets for the machine learning models.

Inputs						**Target**
Dynamic Inputs			Static Inputs			Final Outcome
CI	**AC**	**EV**	**BAC**	**RI**	**SP**	**Real Cost**
1	0.029112797	0.029112797	1	0.74	0.7	1.057015474
1	0.029112797	0.029112797	1	0.74	0.7	1.057015474
0.928963	0.056184422	0.052193252	1	0.74	0.7	1.057015474
…..	…..	…..	…..	…..	…..	…..
1	0.001888536	0.001888536	1	0.75	0.41	1.024862768
0.895511	0.093626098	0.083843217	1	0.75	0.41	1.024862768
0.94356	0.226544544	0.213758464	1	0.75	0.41	1.024862768
…..	…..	…..	…..	…..	…..	…..

Table 7-Inputs and Target Variables for Machine Learning Models (After Normalization)

3.4. Implementation of Machine Learning Models

3.4.1. Selection of machine learning models

To evaluate the effectiveness of different machine learning models in predicting project costs, we selected six widely recognized algorithms, each known for its strengths in handling structured data and regression tasks. These models have been successfully applied in various forecasting contexts, and we reference relevant studies to support their selection:

- **Extreme Gradient Boosting (XGBoost):**

One of the models we have selected is XGBoost, which has been effectively applied in project cost forecasting. For example, (Narbaev et al., 2024b) applied XGBoost to a dataset of 110 projects and demonstrated its strong predictive performance. XGBoost belongs to the family of supervised machine learning models and works by accurately predicting a target variable through an ensemble approach, combining estimates from a series of simpler, weaker models. This method is grounded in the boosting technique developed by (Friedman, 2001), where each model in the sequence aims to correct the errors of its predecessor.

XGBoost is particularly powerful as an ensemble algorithm because it efficiently implements decision trees, creating a composite model that significantly outperforms individual models when used alone. The model's effectiveness is further enhanced by its ability to robustly handle various data types and complex relationships, as well as its flexibility in hyperparameter tuning, which allows for fine-tuning to optimize performance.

Overall, XGBoost stands out for its scalability, precision, and ability to deliver superior predictive accuracy in large-scale data analysis.

The formal additive function of the XGBoost algorithm is defined by the following equation (Chen & Guestrin, 2016):

$$L(\theta) = \sum_{i=1}^{n} l(\hat{y}_i, y_i) + \sum_{k=1}^{k} \Omega(f_k)$$

where i represents a given instance, n is the total number of instances, and $\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum \omega^2$.

In this equation:

- The L component is a differentiable convex loss function that quantifies the difference between the forecasted value $\hat{y_i}$ and the actual value y_i.

- The Ω term acts as a regularization factor, preventing overfitting by smoothing the learned weights ω. This regularization term penalizes the complexity of the regression-based tree functions, where T denotes the number of leaves in the tree.
- The parameters γ and λ control the degree of regularization, with γ penalizing the number of leaves and λ penalizing the magnitude of the leaf weights.
- Each f_k corresponds to an independent tree structure and its associated leaf weights ω.

In recent years, the XGBoost model has gained significant popularity in applied machine learning for both classification and regression tasks, owing to its superior performance and speed (Jabeur et al., 2024; Uddin et al., 2022).

- **Support Vector Regression (SVR):**

One of the models we have selected is Support Vector Regression (SVR), which has been effectively applied in various predictive modelling tasks. For example, Da-Ying Li et al., 2009 applied SVR to real estate price prediction in China and demonstrated its strong predictive performance. SVR belongs to the family of supervised machine learning models and works by accurately predicting a target variable through the optimization of a hyperplane that best fits the data within a specified margin of tolerance.

SVR is particularly powerful as a regression algorithm because it efficiently handles both linear and non-linear relationships by utilizing kernel functions to map input features into high-dimensional spaces. The model's effectiveness is further enhanced by its ability to control model complexity through regularization, which allows for fine-tuning to optimize performance.

The formal objective function of the SVR algorithm is defined by the following equation (Smola et al., 2004):

$$\min_{w,b} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*)$$

subject to the constraints:

$$y_i - (W^T X_i + b) \leq \epsilon + \xi_i$$

$$(W^T X_i + b) - y_i \leq \epsilon + \xi_i^*$$

$$\xi_i \xi_i^* \geq 0$$

where i represents a given instance, n is the total number of instances, and C is the regularization parameter controlling the trade-off between margin size and prediction error.

In this equation:

- The $\|w\|$ component represents the norm of the weight vector, which SVR minimizes to ensure the flatness of the function.
- The ϵ term defines the margin of tolerance within which no penalty is given to errors, allowing some flexibility in the predictions.
- The ξ_i and ξ_i^* terms are slack variables that allow for some degree of error in the margin, providing robustness to the model.

SVR is highly regarded for its robustness and flexibility in handling complex, non-linear data relationships. The model's ability to effectively utilize kernel functions for mapping data into higher-dimensional spaces allows it to achieve high accuracy across various predictive tasks. These attributes make SVR an excellent choice for project cost forecasting, where the underlying data can be intricate and challenging for traditional regression models (Sricharan & Joshi, 2022).

- **Random Forest (RF):**

One of the models we have selected is Random Forest, a highly robust and widely utilized ensemble learning method that excels in both classification and regression tasks. Random Forest operates by constructing a multitude of decision trees during training and then aggregating their predictions to produce a more accurate and stable result. This ensemble approach enhances the model's robustness and accuracy, making it particularly suitable for complex predictive modeling tasks (Israel-Nyemeche et al., 2023; Salman et al., 2024)

Random Forest is particularly powerful as a predictive algorithm because it effectively mitigates the risk of overfitting, a common issue in individual decision trees. It achieves this through a process known as bootstrap aggregation, or bagging, where each tree in the forest is trained on a different random subset of the training data. At each split in the trees, only a random subset of features is considered, introducing further randomness and diversity into the model. This approach not only reduces the correlation between individual trees but also enhances the model's ability to generalize to new, unseen data, leading to improved accuracy and stability (Hu, 2024).

For example, in a study by (Israel-Nyemeche et al., 2023), Random Forest was applied to a large dataset in the context of predictive analytics for health insurance premiums. The model demonstrated superior performance compared to other algorithms,

effectively handling the complex relationships within the data and providing highly accurate predictions. This success highlights Random Forest's suitability for similar tasks, such as project cost forecasting, where data complexity is often a significant challenge.

The formal prediction of the Random Forest model for a regression task is defined by the following equation:

$$\hat{y} = \frac{1}{T}\sum_{t=1}^{T} f_t(x)$$

where:

- T is the total number of trees in the forest,
- $f_t(x)$ is the prediction from the t-th tree.

Each tree $f_t(x)$ is built using a bootstrapped sample from the original data, and only a subset of the features is considered for splitting at each node. This method enhances the model's robustness by ensuring that individual trees do not overfit to noise in the training data, thereby improving the ensemble's overall performance (Salman et al., 2024).

Overall, Random Forest is highly regarded for its simplicity, robustness, and ability to handle large datasets with many features. These attributes make Random Forest an excellent choice for project cost forecasting, where the data may be complex, noisy, and challenging for traditional models.

- **Extremely Randomized Trees (ET):**

One of the models we have selected is Extra Trees, or Extremely Randomized Trees, an ensemble learning method closely related to Random Forests but with key differences that introduce more randomness into the model-building process. Extra Trees has demonstrated strong performance in various complex tasks. For instance, in a study on brain tumor segmentation, Extra Trees outperformed traditional Random Forests, demonstrating its effectiveness in handling complex datasets, which is applicable to tasks such as cost forecasting (Götz et al., n.d.). Like Random Forest, Extra Trees constructs multiple decision trees and aggregates their predictions to improve accuracy and robustness.

However, Extra Trees differs from Random Forest in two significant ways:

1. **Random Split Selection:** Instead of choosing the best possible split based on criteria like Gini impurity or information gain, Extra Trees selects splits entirely at random from the range of values available for each feature. This increased

randomness helps to decorrelate the trees in the ensemble, reducing the variance of the model's predictions (Geurts et al., 2006)

2. **Use of the Entire Dataset:** Unlike Random Forests, where each tree is trained on a bootstrapped sample of the data, Extra Trees typically use the entire dataset to grow each tree. This approach reduces bias and ensures that the model fully leverages the available data to capture complex patterns (Götz et al., n.d.).

The prediction of the Extra Trees model follows the same formal structure as the Random Forest, where the prediction $\hat{y}$ is the average of the predictions from all the trees in the ensemble. However, the key differences between the two models lie in the method of tree construction, as detailed above.

Extra Trees is a powerful extension of Random Forest, particularly useful when high variance in the data can lead to overfitting. Its ability to produce diverse models while utilizing the full dataset makes it an excellent choice for tasks requiring high accuracy and robustness in cost forecasting (Geurts et al., 2006).

- **k-Nearest Neighbour (KNN):**

One of the models we have selected is K-Nearest Neighbours (KNN), a simple yet effective method widely applied in various predictive modelling tasks. Imandoust and Bolandraftar (2013) applied KNN to predict economic events, specifically in the context of credit risk assessment. By analyzing historical data on loan applicants, they demonstrated KNN's ability to provide accurate and reliable forecasts by examining the patterns of similar instances. This method is non-parametric and does not require any assumptions about the underlying data distribution, making it versatile and easy to implement.

KNN groups data into coherent clusters and classifies new inputs based on their similarity with previously labelled data (Taunk et al., 2019). The model operates by identifying the 'k' nearest data points in the training set and using these neighbours to make predictions, either by taking a majority vote in classification tasks or by averaging the target values in regression tasks. The effectiveness of KNN is enhanced by the choice of 'k' and the distance metric, such as Euclidean or Manhattan distance, which allows for fine-tuning to optimize performance.

The KNN algorithm calculates the distance between instances using a chosen metric, such as Euclidean distance, defined by the following equation:

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

where x and y represent the feature vectors of two instances, and n is the total number of features. The algorithm then selects the 'k' closest instances and predicts the target variable based on their values.

In this equation:

- The distance metric $d(x, y)$ determines how closeness is measured, impacting which neighbors are considered.
- The parameter 'k' controls the number of neighbours involved in making the prediction, balancing sensitivity to noise (smaller k) with generalization (larger k).

KNN is particularly well-suited for cost forecasting because it leverages historical project data to identify patterns and relationships that can inform predictions about new projects. By comparing a new project to similar past projects, KNN can provide accurate cost estimates based on the most relevant examples from the training data. Its simplicity, coupled with its robust performance in various applications, makes KNN a reliable choice in scenarios requiring high accuracy and interpretability (Batista & Monard, 2002; Taunk et al., 2019).

- **Light Gradient Boosting Machine (LGBM):**

One of the models we have selected is LightGBM (LGBM), a highly efficient implementation of the Gradient Boosting Decision Tree (GBDT) algorithm. LGBM is recognized for its ability to handle large-scale data efficiently while maintaining high predictive accuracy. Ke et al. (2017) developed LightGBM to address the scalability issues of traditional GBDT models, demonstrating superior performance in tasks such as multi-class classification and ranking. The model's efficiency and scalability make it particularly useful in cost forecasting, where large datasets and high-dimensional feature spaces are common.

LightGBM builds decision trees sequentially, where each tree aims to correct the errors of the previous ones. Unlike traditional GBDT implementations, LightGBM introduces two key innovations: Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB). These techniques significantly reduce training time without sacrificing accuracy Ke et al. (2017)

Gradient-based One-Side Sampling (GOSS) enhances LGBM's efficiency by focusing on instances with larger gradients, which are more informative. By selectively sampling data points based on their gradient values, GOSS allows the model to retain high accuracy while reducing the computational load Ke et al. (2017). Exclusive Feature Bundling (EFB) reduces the number of features by bundling together those that are mutually exclusive, meaning features that rarely take non-zero values simultaneously. This approach speeds up the training process without compromising the model's learning ability.

LGBM's effectiveness in handling large datasets with high-dimensional sparse features makes it well-suited for cost forecasting tasks. In particular, its ability to efficiently process large volumes of data while maintaining high predictive accuracy is a significant advantage, especially in scenarios requiring quick decision-making based on complex data Ke et al. (2017). This has been further demonstrated in various applications. For instance, a study by Linda John et al. (2022) applied LGBM to predict house prices, showing that LGBM outperformed other models such as XGBoost in terms of both speed and accuracy. The study highlighted that LGBM could handle the large and complex datasets typically associated with real estate pricing more efficiently than other boosting algorithms.

The formal objective function of LightGBM in a regression setting is defined as:

$$L(y_i, \hat{y}_i) = \sum_{i=1}^{n} l\,(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k)$$

where:

- y_i and $\hat{y}_i$ represent the actual and predicted values, respectively.
- l is a differentiable convex loss function that measures the difference between actual and predicted values.
- $\Omega(f_k)$ is a regularization term that controls the complexity of the model to prevent overfitting.

LGBM's innovations in gradient-based sampling and feature bundling provide substantial improvements over traditional GBDT methods, ensuring that it delivers robust performance even in the most demanding scenarios. Its use in cost forecasting applications is particularly notable for its ability to provide accurate predictions quickly, making it an ideal choice for large-scale projects that require both precision and efficiency (Guo et al., 2023; John & Shaikh, 2022; Ke et al., 2017).

3.4.2. Model implementation

In this study, we implemented the selected machine learning models—XGBoost, SVR, Random Forest, Extra Trees, KNN, and LightGBM—using the scikit-learn library in Python. The dataset was first divided into training and testing sets with a 75/25 split, where 75% of the data was used for model development (training and cross-validation) and 25% was reserved for evaluating the final model performance.

To optimize the performance of each model, we applied a grid search combined with 3-fold cross-validation. This method systematically explored the hyperparameter space to

ensure that the models were fine-tuned for optimal performance on the training data (Hutter et al., 2019).

For each model, a grid of potential hyperparameters was defined. For example, in the case of the Random Forest model, we varied parameters such as the number of trees (n_estimators) and the maximum depth of the trees (max_depth). Similarly, for the SVR model, we adjusted the regularization parameter (C) and the kernel type to identify the best settings.

The grid search process was conducted using the GridSearchCV class from scikit-learn. During this process, the training data was divided into three subsets (folds). The model was trained on two of these subsets and validated on the third, with the process repeated three times, each time using a different fold as the validation set. This method allowed us to evaluate each combination of hyperparameters comprehensively.

The performance of each hyperparameter combination was evaluated using the Mean Absolute Percentage Error (MAPE) as the primary performance metric. MAPE measures the average absolute percentage difference between the predicted and actual values, providing a clear indication of the model's prediction accuracy in relative terms.

After completing the grid search, the best set of hyperparameters was selected based on the average MAPE across the cross-validation folds. This careful selection was guided by the principle that a thorough exploration of the hyperparameter space can identify configurations that balance bias and variance effectively, thereby avoiding both underfitting and overfitting (Hutter et al., 2019).

Once the optimal hyperparameters were identified, each model was retrained on the entire training set using these parameters. The trained models were then evaluated on the reserved 25% of the data (the test set) to assess their generalization performance. This final evaluation was conducted using two key metrics: Mean Absolute Percentage Error (MAPE) and Normalized Root Mean Squared Error (NRMSE). These metrics provided a comprehensive assessment of the models' predictive accuracy, ensuring that they were fully prepared to deliver accurate cost forecasts.

- **Mean Absolute Percentage Error (MAPE)**

measures the average absolute percentage difference between the predicted values $\hat{y}_i$ and the actual values y_i. It is defined as:

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100$$

where n is the number of observations. MAPE is expressed as a percentage, providing an intuitive understanding of the average error relative to the actual values.

- **Normalized Root Mean Squared Error (NRMSE)**

is the square root of the average squared differences between predicted values $\hat{y}_i$ and actual values y_i, normalized by the range of the data. It is defined as:

$$NRMSE = \frac{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}}{y_{max} - y_{min}}$$

where y_{max} and y_{min} are the maximum and minimum values of the actual data. NRMSE provides a normalized measure of prediction error, allowing for comparisons across different datasets or models.

These evaluation metrics were chosen because they offer complementary insights: MAPE provides a relative error percentage that is easy to interpret, while NRMSE offers a scale-independent measure of error that accounts for the variability of the data. Together, they ensured that the models' predictions were both accurate and reliable.

After establishing the optimal hyperparameters for each model and conducting the initial training and evaluation, we proceeded with a series of runs designed to assess the impact of specific project characteristics on the performance of the machine learning models. The goal was to understand how the inclusion of Project Regularity (RI) and Project Seriality (SP) affected the predictive accuracy of the models.

We began by creating a baseline model using a core set of features: Actual Cost (AC), Earned Value (EV), and Cost Performance Index (CI). This baseline provided a reference point against which the effects of adding additional features could be measured.

Next, we conducted a series of runs where we incrementally added the project characteristics. First, we introduced Project Regularity (RI) to the baseline feature set and evaluated the model's performance. Following this, we added Project Seriality (SP) separately to see how it influenced the results. Finally, a comprehensive run was conducted with both RI and SP included alongside the baseline features.

RUNS	INPUTS
FIRST RUN	AC,EV,CI
SECOND RUN	AC,EV,CI,RI
THIRD RUN	AC,EV,CI,SP
FOURTH RUN	AC,EV,CI,RI,SP

Table 8-Sequential Runs with Different Input Features

3.5. EVM Calculation and Evaluation

To compare the effectiveness of our machine learning models with traditional methods, we calculated the Earned Value Management (EVM) metrics as a baseline. The Estimate at Completion (EAC) was the primary metric used for this comparison, allowing us to evaluate how well the machine learning models performed relative to this established approach.

For each project, the EAC was calculated at various tracking points using the following formula (Batselier & Vanhoucke, 2015):

$$EAC(t) = AC_t + \frac{(BAC - EV_t)}{CPI_t}$$

where:

- t represents the tracking period
- AC_t is the actual cost incurred up to time t
- BAC is the Budget at Completion, representing the total planned budget
- EV_t is the Earned Value at the time t
- CPI_t is the Cost Performance Index at time t, calculated as $CPI_t = \frac{EV_t}{AC_t}$

After calculating the EAC for each tracking point of each project, we averaged these EAC values to obtain the overall estimated completion cost for each project. This average EAC was then used to calculate both the Mean Absolute Percentage Error (MAPE) and the Normalized Root Mean Squared Error(NRMSE), providing a comprehensive evaluation of the accuracy of the EVM-based cost forecasts.

The MAPE was calculated using the following formula:

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{EAC_i - Real\ Cost_i}{Real\ Cost_i}\right| \times 100$$

where:

- EAC_i is the average Estimate at Completion for the project i
- $Real\ Cost_i$ is the actual final cost of the project i
- n is the total number of projects

We also calculated NRMSE for the EVM forecasts, just as we did for the machine learning models, using the following formula:

$$NRMSE = \frac{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(EAC_i - Real\ Cost_i)^2}}{Real\ Cost_{max} - Real\ Cost_{min}}$$

where:

- EAC_i is the average Estimate at Completion for the project i
- $Real\ Cost_i$ is the actual final cost of the project i
- n is the total number of projects
- $Real\ Cost_{max}$ is the maximum real cost at completion among all projects
- $Real\ Cost_{min}$ is the minimum real cost at completion among all projects

By calculating both MAPE and NRMSE, we obtained a thorough understanding of the accuracy and reliability of the EVM-based forecasts, which we then compared against the predictions generated by the machine learning models.

4. Result and Discussion

The primary objective of this study is to enhance the accuracy of project cost forecasting by comparing the performance of various machine learning (ML) algorithms and determining whether Project Regularity (RI) and Project Seriality (SP) improve model accuracy. Despite their widespread use, Traditional Earned Value Management (EVM) methods often result in inaccurate predictions due to assumptions of linear cost growth and the static nature of their inputs. In contrast, this study employs six machine learning models—XGBoost, Extremely Randomized Trees (ExtraTrees), Random Forest, Support Vector Machines (SVM), Light Gradient Boosting Machine (LightGBM), and K-Nearest Neighbors (KNN)—to predict the cost at completion. The models are benchmarked against traditional EVM to assess improvements in forecast accuracy.

The inclusion of static features, RI and SP, as input variables alongside dynamic features like Actual Cost (AC), Earned Value (EV), and Cost Performance Index (CI), is expected to enhance the models' predictive power, particularly in complex, non-linear project environments.

Data for this study was sourced from 181 real-world projects provided by the Dynamic Scheduling Library at Ghent University, of which 90 projects were selected after filtering and removing outliers. The dataset includes key variables necessary for cost forecasting such as Budget at Completion (BAC), Actual Cost (AC), Earned Value (EV), Cost Performance Index (CI), Project Regularity (RI), and Project Seriality (SP). The data was normalized using BAC as a base to allow for comparisons across projects of different scales.

The machine learning models were trained using 75% of the dataset, with the remaining 25% held out for testing. Four different input combinations were tested across all models to evaluate the impact of including RI and SP as static features:

1. **First Run:** AC, EV, CI
2. **Second Run:** AC, EV, CI, RI
3. **Third Run:** AC, EV, CI, SP
4. **Fourth Run:** AC, EV, CI, RI, SP

Model performance was evaluated using two accuracy metrics: Mean Absolute Percentage Error (MAPE) and Normalized Root Mean Squared Error (NRMSE). These metrics allow for a clear comparison between the machine learning models and the traditional EVM method.

4.1. Machine Learning Models Results

- **Extreme Gradient Boosting (XGBoost):**

RUN	FEATURES	MAPE%	NRMSE	PARAMETERS
1	AC,EV,CI	6.00191	0.054322	{'colsample_bytree': 1.0, 'gamma': 0, 'learning_rate': 0.1, 'max_depth': 7, 'min_child_weight': 5, 'n_estimators': 300, 'reg_alpha': 0.5, 'reg_lambda': 1, 'subsample': 0.6}
2	AC,EV,CI,RI	5.380611	0.046125	{'colsample_bytree': 1.0, 'gamma': 0, 'learning_rate': 0.2, 'max_depth': 10, 'min_child_weight': 1, 'n_estimators': 300, 'reg_alpha': 0.5, 'reg_lambda': 0.1, 'subsample': 0.9}
3	AC,EV,CI,SP	5.83487	0.055427	{'colsample_bytree': 1.0, 'gamma': 0, 'learning_rate': 0.2, 'max_depth': 10, 'min_child_weight': 1, 'n_estimators': 100, 'reg_alpha': 0.5, 'reg_lambda': 0.01, 'subsample': 1.0}
4	AC,EV,CI,SP,RI	4.934205	0.04373	{'colsample_bytree': 1.0, 'gamma': 0, 'learning_rate': 0.2, 'max_depth': 10, 'min_child_weight': 1, 'n_estimators': 300, 'reg_alpha': 0.5, 'reg_lambda': 0.01, 'subsample': 0.9}

Table 9-XGBoost Accuracy

The XGBoost model results demonstrate improved cost forecasting accuracy as additional features are incorporated. Using only dynamic features (AC, EV, CI), the model achieved a MAPE of 6.00%. When Project Regularity (RI) was added, accuracy significantly improved, with a MAPE of 5.38%. In comparison, adding Project Seriality (SP) alone resulted in a MAPE of 5.83%, indicating that RI had a greater impact on enhancing the model's performance than SP.

However, the combination of RI and SP yielded the best results, with a MAPE of 4.93%. This suggests that while RI contributes more substantially to accuracy, including both static features optimizes the forecasting model's predictive power. Overall, the results indicate that the integration of RI and SP leads to the most reliable cost predictions.

- **Extremely Randomized Tree (ERT)**

RUN	FEATURES	MAPE%	NRMSE	PARAMETERS
1	AC,EV,CI	10.47520738	0.061301711	{'bootstrap': False, 'max_depth': 5, 'max_features': 'sqrt', 'min_samples_leaf': 4, 'min_samples_split': 10, 'n_estimators': 100}
2	AC,EV,CI,RI	7.769232079	0.050749483	{'bootstrap': False, 'max_depth': 5, 'max_features': 'sqrt', 'min_samples_leaf': 4, 'min_samples_split': 15, 'n_estimators': 150}
3	AC,EV,CI,SP	8.139811835	0.068115636	{'bootstrap': False, 'max_depth': 5, 'max_features': 'sqrt', 'min_samples_leaf': 4, 'min_samples_split': 15, 'n_estimators': 150}
4	AC,EV,CI,SP,RI	7.908921205	0.068884452	{'bootstrap': False, 'max_depth': 5, 'max_features': 'sqrt', 'min_samples_leaf': 4, 'min_samples_split': 10, 'n_estimators': 100}

Table 10-ERT Accuracy

The Extremely Randomized Trees model results show an improvement in cost forecasting accuracy when additional features are incorporated. In the first run, using only dynamic features (AC, EV, CI), the model achieved a MAPE of 10.48% and NRMSE of 0.0613. Adding Project Regularity (RI) in the second run reduced the MAPE to 7.77% and the NRMSE to 0.0507, showing a notable improvement in performance. In comparison, adding Project Seriality (SP) alone in the third run resulted in a higher MAPE of 8.14% and NRMSE of 0.0681, indicating that RI had a greater impact on accuracy than SP.

However, the combination of both RI and SP (Run 4) did not yield the expected improvements, with a MAPE of 7.91% and NRMSE of 0.0689, performing slightly worse than using RI alone. This suggests that while RI plays a significant role in improving accuracy, adding SP did not enhance the model's performance in this case, and using RI alone provided better forecasting accuracy than combining both features.

- **Random Forest (RF)**

RUN	FEATURES	MAPE%	NRMSE	PARAMETERS
1	AC,EV,CI	7.359134	0.0643	{'bootstrap': True, 'max_depth': 5, 'max_features': 'sqrt', 'min_samples_leaf': 4, 'min_samples_split': 10, 'n_estimators': 200}
2	AC,EV,CI,RI	5.492867	0.05484	{'bootstrap': False, 'max_depth': 5, 'max_features': 'sqrt', 'min_samples_leaf': 4, 'min_samples_split': 10, 'n_estimators': 200}
3	AC,EV,CI,SP	5.617081	0.05541	{'bootstrap': False, 'max_depth': 5, 'max_features': 'sqrt', 'min_samples_leaf': 4, 'min_samples_split': 10, 'n_estimators': 200}
4	AC,EV,CI,SP,RI	5.369513	0.05121	{'bootstrap': False, 'max_depth': 5, 'max_features': 'sqrt', 'min_samples_leaf': 4, 'min_samples_split': 10, 'n_estimators': 200}

Table 11-RF Accuracy

The results of the Random Forest model show a progressive improvement in cost forecasting accuracy as additional features are incorporated. In the first run, using only dynamic features (AC, EV, CI), the model achieved a MAPE of 7.36% and an NRMSE of 0.0643. Adding Project Regularity (RI) in the second run significantly improved accuracy, reducing the MAPE to 5.49% and the NRMSE to 0.0548. Comparatively, adding Project Seriality (SP) in the third run resulted in a MAPE of 5.62% and NRMSE of 0.0554, indicating that RI had a greater positive impact on accuracy than SP.

The fourth run, which included both RI and SP, delivered the best results with a MAPE of 5.37% and NRMSE of 0.0512. This suggests that while RI alone had a stronger effect on improving accuracy, the combination of both RI and SP provided the most optimal forecast, further enhancing the Random Forest model's predictive performance

- **Support Vector Regression (SVR)**

RUN	FEATURES	MAPE%	NRMSE	PARAMETERS
1	AC,EV,CI	6.420489	0.068131	{'C': 1, 'degree': 3, 'epsilon': 0.1, 'gamma': 'scale', 'kernel': 'poly'}
2	AC,EV,CI,RI	6.659883	0.061994	{'C': 1, 'degree': 2, 'epsilon': 0.1, 'gamma': 'scale', 'kernel': 'poly'}
3	AC,EV,CI,SP	6.559749	0.061795	{'C': 10, 'degree': 2, 'epsilon': 0.1, 'gamma': 'scale', 'kernel': 'linear'}
4	AC,EV,CI,SP,RI	6.4592	0.05858	{'C': 1, 'degree': 2, 'epsilon': 0.1, 'gamma': 'auto', 'kernel': 'rbf'}

Table 12-SVR Accuracy

The Support Vector Regression (SVR) model results show marginal improvements in cost forecasting accuracy as additional features are incorporated. In the first run, using only dynamic features (AC, EV, CI), the model achieved a MAPE of 6.42% and an NRMSE of 0.0681. Adding Project Regularity (RI) in the second run slightly worsened performance, with a MAPE of 6.66% and an NRMSE of 0.0620. Similarly, adding Project Seriality (SP) in the third run resulted in a MAPE of 6.56% and NRMSE of 0.0618, indicating that neither RI nor SP had a significant positive impact on accuracy.

However, in the fourth run, when both RI and SP were included, the model achieved a slight improvement with a MAPE of 6.46% and NRMSE of 0.0586. Although the combination of RI and SP improved the results marginally compared to the earlier runs, the overall performance of SVR did not show as substantial gains as seen in other models, suggesting that SVR is less responsive to the inclusion of RI and SP for improving forecasting accuracy.

- **Light Gradient Boosting Method (LGBM)**

RUN	FETURES	MAPE%	NRMSE	PARAMETERS
1	AC,EV,CI	6.786865	0.068725	{'colsample_bytree': 0.6, 'learning_rate': 0.05, 'max_depth': 3, 'min_child_samples': 20, 'n_estimators': 200, 'num_leaves': 15, 'subsample': 0.6}
2	AC,EV,CI,RI	5.458081	0.058478	{'colsample_bytree': 0.8, 'learning_rate': 0.05, 'max_depth': 7, 'min_child_samples': 20, 'n_estimators': 100, 'num_leaves': 15, 'subsample': 0.6}
3	AC,EV,CI,SP	5.848209	0.062745	{'colsample_bytree': 0.8, 'learning_rate': 0.05, 'max_depth': 7, 'min_child_samples': 20, 'n_estimators': 100, 'num_leaves': 31, 'subsample': 0.6}
4	AC,EV,CI,SP,RI	4.87709	0.053228	{'colsample_bytree': 0.8, 'learning_rate': 0.1, 'max_depth': 7, 'min_child_samples': 20, 'n_estimators': 200, 'num_leaves': 31, 'subsample': 0.6}

Table 13-LGBM Accuracy

The Light Gradient Boosting Machine (LGBM) model shows a consistent improvement in cost forecasting accuracy as additional features are incorporated. In the first run, using

only dynamic features (AC, EV, CI), the model achieved a MAPE of 6.79% and NRMSE of 0.0687. Adding Project Regularity (RI) in the second run significantly improved the model's performance, reducing the MAPE to 5.46% and NRMSE to 0.0585. Similarly, adding Project Seriality (SP) in the third run resulted in a MAPE of 5.85% and NRMSE of 0.0627, although the performance was slightly less accurate compared to adding RI.

The fourth run, which included both RI and SP, provided the best results, with a MAPE of 4.88% and NRMSE of 0.0532. This indicates that while both RI and SP contribute to improved accuracy, their combined inclusion yields the most optimal forecast, further enhancing the LGBM model's ability to predict project costs effectively.

- **K-Nearest Neighbours (KNN)**

RUN	FETURES	MAPE%	NRMSE	PARAMETERS
1	AC,EV,CI	5.60829	0.058587	{'algorithm': 'auto', 'leaf_size': 20, 'n_neighbors': 5, 'p': 1, 'weights': 'distance'}
2	AC,EV,CI,RI	4.256217	0.049886	{'algorithm': 'auto', 'leaf_size': 20, 'n_neighbors': 5, 'p': 1, 'weights': 'distance'}
3	AC,EV,CI,SP	5.303149	0.059782	{'algorithm': 'auto', 'leaf_size': 20, 'n_neighbors': 5, 'p': 2, 'weights': 'distance'}
4	AC,EV,CI,SP,RI	4.438995	0.057373	{'algorithm': 'auto', 'leaf_size': 20, 'n_neighbors': 5, 'p': 1, 'weights': 'distance'}

Table 14-KNN Accuracy

The K-Nearest Neighbors (KNN) model results show a substantial improvement in cost forecasting accuracy with the inclusion of static features. In the first run, using only dynamic features (AC, EV, CI), the model achieved a MAPE of 5.61% and NRMSE of 0.0586. Adding Project Regularity (RI) in the second run yielded the best improvement, significantly reducing the MAPE to 4.26% and NRMSE to 0.0499, demonstrating that RI has a strong positive effect on accuracy.

In the third run, adding Project Seriality (SP) resulted in a MAPE of 5.30% and NRMSE of 0.0598, which was less effective than RI in enhancing accuracy. The fourth run, combining both RI and SP, provided a MAPE of 4.44% and NRMSE of 0.0574. While the combination of RI and SP slightly improved accuracy compared to using SP alone, RI remained the more influential feature in improving forecasting performance.

4.2. Traditional EVM Result

In this section, we will calculate the Estimate at Completion (EAC) using the Earned Value Management (EVM) method, based on the equation outlined in the methodology:

$$EAC(t) = AC_t + \frac{(BAC - EV_t)}{CPI_t}$$

Using this formula, we will compute the EAC for each project in the dataset, which provides a forecast of the total project cost upon completion. Once the EAC values have been determined for all projects using the EVM method, we will calculate the Mean Absolute Percentage Error (MAPE) to measure the accuracy of the EVM forecasts.

By comparing the MAPE values from the EVM method with those obtained from the machine learning models (XGBoost, Random Forest, LGBM, etc.), we will assess the performance of traditional EVM forecasting versus more advanced, data-driven machine learning approaches. This comparison will highlight the improvements in forecasting accuracy achieved by leveraging machine learning techniques, especially when incorporating additional features such as Project Regularity (RI) and Project Seriality (SP).

The final MAPE result for the traditional EVM method is **7.21%**. This value will be used to compare the accuracy of the EVM method against the machine learning models. As seen from this result, the EVM method yields a relatively higher MAPE compared to the machine learning models such as XGBoost (4.93%) and LGBM (4.88%). This comparison highlights the potential of machine learning algorithms in providing more accurate cost forecasts, especially when enhanced with additional features like Project Regularity (RI) and Project Seriality (SP).

In the table below, you can see the best MAPE results of the machine learning models compared with the EVM method.

METHOD	BEST MAPE RESULT	FEATURES
EVM	7.2	AC,BAC,EV,CPI
ML-XGBOOST	4.93	AC,EV,CI,SP,RI
ML-ERT	7.77	AC,EV,CI,RI
ML-RF	5.37	AC,EV,CI,SP,RI
ML-SVR	6.42	AC,EV,CI
ML-LGBM	4.88	AC,EV,CI,SP,RI
ML-KNN	4.26	AC,EV,CI,RI

Table 15-Comparison of EVM with ML models

5. Discussion

5.1. Regularity Index (RI) and Project Seriality (SP) Comparision

The inclusion of static features such as Project Regularity (RI) and Project Seriality (SP) had a significant impact on the performance of all machine learning models. In general, RI consistently provided the most substantial improvement in accuracy across all models, while SP had a more varied effect depending on the model.

- **Project Regularity (RI)**

Project Regularity (RI) proved to be the most influential static feature in improving model performance. Models like XGBoost, LGBM, and Random Forest demonstrated significant gains in accuracy when RI was included alongside dynamic features like AC, EV, and CI. This improvement was especially noticeable in KNN, where the inclusion of RI alone resulted in its best MAPE score of 4.26%. The success of RI across multiple models suggests that it captures a critical aspect of project cost forecasting, likely related to the consistency and predictability of project networks, which helps the models generate more accurate forecasts.

- **Project Seriality (SP)**

The impact of Project Seriality (SP) was more mixed compared to RI. For models like XGBoost, LGBM, and Random Forest, the inclusion of SP alongside RI provided the best results, enhancing accuracy and reducing forecasting error. However, in other models like KNN and Extremely Randomized Trees, SP did not significantly improve performance and, in some cases, slightly worsened accuracy. This suggests that SP may not always be as crucial as RI, but its inclusion in combination with RI in models like XGBoost and LGBM allows these models to capture more nuanced relationships in the data, particularly in projects with complex task sequences.

- **Combined Impact of RI and SP**

When RI and SP were combined, the best results were observed in models like XGBoost and LGBM, where the synergy of these two features resulted in optimal accuracy. The combination allowed these models to capture both the regularity and structure of project networks, providing a more holistic view of project performance and improving forecasting accuracy. In contrast, models like KNN performed better with RI alone, indicating that some models may not benefit from the added complexity introduced by SP.

5.2. Machine Learning Models Comparison

After analyzing the impact of Project Regularity (RI) and Project Seriality (SP) on model performance, it is essential to compare the machine learning models based on their overall forecasting accuracy. The models evaluated include XGBoost, Light Gradient Boosting Machine (LGBM), K-Nearest Neighbors (KNN), Random Forest, Extremely Randomized Trees, and Support Vector Regression (SVR). The comparison is based on the key metrics: Mean Absolute Percentage Error (MAPE) and Normalized Root Mean Squared Error (NRMSE).

- **XGBoost**

XGBoost emerged as one of the top performers across all feature sets. With a MAPE of 4.93% and NRMSE of 0.0437, XGBoost proved to be highly reliable, particularly when both RI and SP were included. The model demonstrated a good balance between handling outliers and maintaining overall accuracy. Its consistency across multiple runs and flexibility in tuning hyperparameters make it a powerful tool for complex project cost forecasting.

- **Light Gradient Boosting Machine (LGBM)**

LGBM closely followed XGBoost, achieving the lowest MAPE at 4.88%, but with a slightly higher NRMSE of 0.0532. Like XGBoost, LGBM performed best when both RI and SP were included, suggesting that it, too, benefits from a more comprehensive feature set. While LGBM performed similarly to XGBoost, the slightly higher NRMSE indicates that it may be more sensitive to errors or outliers, but overall, it remains one of the strongest models for project cost forecasting.

- **K-Nearest Neighbors (KNN)**

KNN delivered the lowest MAPE at 4.26%, but only when RI was included, without SP. Despite its impressive MAPE, the higher NRMSE in comparison to XGBoost and LGBM suggests that KNN struggles with more complex relationships and error handling. Additionally, the model's accuracy dropped slightly when SP was added, indicating that it is more sensitive to the feature selection. While KNN can outperform other models in certain scenarios, particularly when RI dominates, it lacks the versatility and robustness seen in XGBoost and LGBM.

- **Random Forest**

Random Forest performed well with a MAPE of 5.37% when both RI and SP were included. This model benefits from the combination of static and dynamic features, much like XGBoost and LGBM. However, it did not reach the same level of accuracy, likely due to its limitations in handling more complex interactions between the features.

Nevertheless, Random Forest remains a solid option, offering competitive performance in cost forecasting tasks.

- **Extremely Randomized Trees**

The Extremely Randomized Trees model did not perform as well as the other models, achieving a MAPE of 7.77%. Although the inclusion of RI improved its performance, adding SP did not enhance the accuracy further. The model appears less capable of leveraging the static features compared to XGBoost, LGBM, and Random Forest, making it less effective for this type of forecasting task.

- **Support Vector Regression (SVR)**

SVR had a MAPE of 6.46%, showing only marginal improvements with the inclusion of RI and SP. It was the least responsive to static features, and its overall performance lagged behind the other models. SVR's limited adaptability and higher forecasting error suggest that it may not be as well-suited for complex project cost forecasting as the other models evaluated.

overall, XGBoost and LGBM stood out as the best-performing models due to their ability to handle both static and dynamic features effectively, with XGBoost showing a slightly lower NRMSE, making it more reliable in handling outliers. KNN, although achieving the lowest MAPE, is more sensitive to feature selection and less flexible across various project datasets, limiting its overall effectiveness in comparison to XGBoost and LGBM. Random Forest also performed well but did not reach the accuracy levels of XGBoost or LGBM, while Extremely Randomized Trees and SVR lagged behind, showing less capacity to fully utilize the feature set.

Ultimately, XGBoost emerges as the most well-rounded model, providing a balance of accuracy and robustness, followed closely by LGBM, while KNN remains highly effective in specific cases where RI dominates but lacks the versatility seen in the top models

6. Conclusion

This thesis has thoroughly investigated the application of machine learning (ML) algorithms in improving the accuracy of cost forecasting in construction project management. Traditional methods, such as Earned Value Management (EVM), though widely used, are often constrained by their reliance on static, linear models that do not adequately capture the complexities and uncertainties inherent in large-scale projects. This limitation frequently leads to inaccurate cost forecasts, particularly during the dynamic phases of project execution. In response to this challenge, this research introduced six advanced ML models—XGBoost, Extremely Randomized Trees, Random Forest, Support Vector Machine (SVM), Light Gradient Boosting Machine (LightGBM), and K-Nearest Neighbors (KNN)—to predict project costs based on a more flexible, data-driven approach.

The analysis was performed on a dataset of 90 real-world construction projects, utilizing key performance metrics such as Actual Cost (AC), Earned Value (EV), and the Cost Performance Index (CPI) as dynamic inputs to the models. Additionally, two newly introduced static features, Project Regularity (RI) and Project Seriality (SP), were incorporated to account for the non-linear and topological complexities of project structures. These features allowed the ML models to capture deviations from linear progress and the mix of serial versus parallel task execution, which are often overlooked in traditional cost forecasting methods.

The findings clearly demonstrated that machine learning models outperform traditional EVM in terms of both accuracy and adaptability. XGBoost emerged as the most effective model, achieving the lowest Mean Absolute Percentage Error (MAPE) and Normalized Root Mean Squared Error (NRMSE). The inclusion of RI and SP further enhanced the performance of all ML models, particularly in projects with non-linear progression, highlighting the importance of considering structural and topological aspects of project management when developing predictive models. This suggests that ML algorithms not only provide more accurate cost forecasts but also offer the flexibility needed to account for the complex, dynamic nature of real-world construction projects.

While the results of this study are promising, there are several limitations that need to be addressed in future research. One key limitation is the lack of comprehensive real-world data, particularly from a broader range of industries and project types. The dataset used in this research was limited in scope, and while the machine learning models performed well, a larger dataset would provide more robust results. Future studies could benefit from integrating artificial or simulated data to enhance the models' training and testing, especially for early-stage projects where real data is often sparse. This would allow for more thorough testing across a wider variety of project scenarios, improving the generalizability of the findings.

Another limitation lies in the static features used for the models. While Project Regularity (RI) and Project Seriality (SP) were useful, there are many other topological and structural features, such as network complexity, task dependencies, and resource allocation patterns, that could be explored as inputs to the machine learning models. Incorporating these additional static features could further enhance the predictive power of ML algorithms, especially in highly intricate projects with diverse task interdependencies. Future research should focus on examining how these topological factors, combined with dynamic metrics, can improve the overall accuracy and applicability of ML-based cost forecasting.

In conclusion, this study has demonstrated the significant potential of machine learning algorithms to improve cost forecasting in construction project management. By integrating both dynamic and static project features, these models offer greater flexibility and accuracy compared to traditional methods like EVM. However, further research is necessary to expand the dataset, explore additional topological features, and incorporate real-time data for even more robust and adaptable cost forecasting. As machine learning techniques continue to evolve, their integration into project management holds great promise for reducing cost overruns, enhancing decision-making, and improving project outcomes across the construction industry and beyond.

7. References

Atkinson, R. (1999). Project management: cost, time and quality, two best guesses and a phenomenon, its time to accept other success criteria. *International Journal of Project Management*, *17*(6), 337–342. https://doi.org/10.1016/S0263-7863(98)00069-6

Bafandeh Imandoust, S., & Bolandraftar, M. (2013). Application of K-Nearest Neighbor (KNN) Approach for Predicting Economic Events: Theoretical Background. In *Journal of Engineering Research and Applications www.ijera.com* (Vol. 3). www.ijera.com

Batista, G. E. A. P. A., & Monard, M. C. (2002). *A Study of K-Nearest Neighbour as an Imputation Method*. www.researchgate.net/publication/220981745

Batselier, J., & Vanhoucke, M. (2015). Empirical Evaluation of Earned Value Management Forecasting Accuracy for Time and Cost. *Journal of Construction Engineering and Management*, *141*(11). https://doi.org/10.1061/(asce)co.1943-7862.0001008

Batselier, J., & Vanhoucke, M. (2017). Project regularity: Development and evaluation of a new project characteristic. *Journal of Systems Science and Systems Engineering*, *26*(1), 100–120. https://doi.org/10.1007/s11518-016-5312-6

Capone, C., Talgat, S., Hazir, O., Abdrasheva, K., & Kozhakhmetova, A. (2024). Artificial Intelligence Models for Predicting Budget Expenditures. *Eurasian Journal of Economic and Business Studies*, *68*(1), 32–43. https://doi.org/10.47703/ejebs.v68i1.331

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, *13-17-August-2016*, 785–794. https://doi.org/10.1145/2939672.2939785

Da-Ying Li, Wei Xu, Hong Zhao, & Rong-Qiu Chen. (2009). *Proceedings of the Eighth International Conference on Machine Learning and Cybernetics : Baoding, 12-15 July 2009*. IEEE.

De Marco, A. (2018). *Project Management for Facility Constructions A Guide for Engineers and Architects*. https://doi.org/https://doi.org/10.1007/978-3-319-75432-1

Friedman, J. H. (2001). 999 REITZ LECTURE GREEDY FUNCTION APPROXIMATION: A GRADIENT BOOSTING MACHINE 1. In *The Annals of Statistics* (Vol. 29, Issue 5).

Geurts, P., Ernst, D., & Wehenkel, L. (2006). Extremely randomized trees. *Machine Learning*, *63*(1), 3–42. https://doi.org/10.1007/s10994-006-6226-1

Götz, M., Weber, C., Maier-Hein, K. H., Goetz, M., Bloecher, J., Stieltjes, B., Meinzer, H.-P., & Maier-Hein, K. (n.d.). *Extremely randomized trees based brain tumor segmentation*. https://www.researchgate.net/publication/267762444

Guo, J., Yun, S., Meng, Y., He, N., Ye, D., Zhao, Z., Jia, L., & Yang, L. (2023). Prediction of heating and cooling loads based on light gradient boosting machine algorithms. *Building and Environment*, *236*. https://doi.org/10.1016/j.buildenv.2023.110252

Hu, Y. (2024). Comparison and Analysis of the Effectiveness of Linear Regression, Decision Tree, and Random Forest Models for Health Insurance Premium Forecasting. *Advances in Economics, Management and Political Sciences*, *79*(1), 347–353. https://doi.org/10.54254/2754-1169/79/20241754

Hutter, F., Kotthoff, L., & Vanschoren, J. (2019). *The Springer Series on Challenges in Machine Learning Automated Machine Learning Methods, Systems, Challenges*. http://www.springer.com/series/15602

Israel-Nyemeche, Tamunopiriye, & O.E, T. (2023). A Random Forest Regressor Model for Forecasting Air Quality Index from Particulate Matters. *International Journal of Computer Science and Mobile Computing*, *12*(10), 57–70. https://doi.org/10.47760/ijcsmc.2023.v12i10.006

Jabeur, S. Ben, Mefteh-Wali, S., & Viviani, J. L. (2024). Forecasting gold price with the XGBoost algorithm and SHAP interaction values. *Annals of Operations Research*, *334*(1–3), 679–699. https://doi.org/10.1007/s10479-021-04187-w

Jacob, D. S., & Kane, M. (2004). Forecasting schedule completion using earned value metrics revisited. *The Measurable News, 1(11), 7*.

John, A. L., & Shaikh, S. (2022). *Predicting House Prices using Machine Learning and LightGBM*. https://ssrn.com/abstract=4108744

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T.-Y. (2017). *LightGBM: A Highly Efficient Gradient Boosting Decision Tree*. https://github.com/Microsoft/LightGBM.

Kim, B.-C., & Reinschmidt, K. F. (2011). Combination of Project Cost Forecasts in Earned Value Management. *Journal of Construction Engineering and Management*, *137*(11), 958–966. https://doi.org/10.1061/(asce)co.1943-7862.0000352

Mahesh, B. (2020). Machine Learning Algorithms - A Review. *International Journal of Science and Research (IJSR)*, *9*(1), 381–386. https://doi.org/10.21275/art20203995

Narbaev, T., & De Marco, A. (2017). A FRAMEWORK MODEL FOR RISK ADJUSTED COST FORECASTING Earned Value and Cost Contingency Management. *THE JOURNAL OF MODERN PROJECT MANAGEMENT A*, *13*. https://doi.org/10.19255/JMPM01202

Narbaev, T., Hazir, Ö., Khamitova, B., & Talgat, S. (2024a). A machine learning study to improve the reliability of project cost estimates. *International Journal of Production Research*, *62*(12), 4372–4388. https://doi.org/10.1080/00207543.2023.2262051

Narbaev, T., Hazir, Ö., Khamitova, B., & Talgat, S. (2024b). A machine learning study to improve the reliability of project cost estimates. *International Journal of Production Research*, *62*(12), 4372–4388. https://doi.org/10.1080/00207543.2023.2262051

Ottaviani, F. M., & Marco, A. De. (2021). Multiple Linear Regression Model for Improved Project Cost Forecasting. *Procedia Computer Science*, *196*, 808–815. https://doi.org/10.1016/j.procs.2021.12.079

Pellerin, R., & Perrier, N. (2019). A review of methods, techniques and tools for project planning and control. In *International Journal of Production Research* (Vol. 57, Issue 7, pp. 2160–2178). Taylor and Francis Ltd. https://doi.org/10.1080/00207543.2018.1524168

Pewdum, W., Rujirayanyong, T., & Sooksatra, V. (2009). Forecasting final budget and duration of highway construction projects. *Engineering, Construction and Architectural Management*, *16*(6), 544–557. https://doi.org/10.1108/09699980911002566

PMI. (2019). *The Standard for Earned Value Management*. https://www.pmi.org/pmbok-guide-standards/foundational /earned-value-management.

Salman, H. A., Kalakech, A., & Steiti, A. (2024). Random Forest Algorithm Overview. *Babylonian Journal of Machine Learning*, *2024*, 69–79. https://doi.org/10.58496/bjml/2024/007

Smola, A. J., Sch¨olkopf, B., & Sch¨olkopf, S. (2004). A tutorial on support vector regression *. In *Statistics and Computing* (Vol. 14). Kluwer Academic Publishers.

Sricharan, S., & Joshi, V. (2022). Comparison of SVR Techniques for Stock Market Predictions. *IEEE International Conference on Data Science and Information System, ICDSIS 2022*. https://doi.org/10.1109/ICDSIS55133.2022.9915990

Taunk, K., De, S., Verma, S., & Swetapadama, A. (2019). *2019 International Conference on Intelligent Computing and Control Systems (ICCS)*. IEEE.

Uddin, S., Ong, S., & Lu, H. (2022). Machine learning in project analytics: a data-driven framework and case study. *Scientific Reports*, *12*(1). https://doi.org/10.1038/s41598-022-19728-x

Vandevoorde, S., & Vanhoucke, M. (2006). A comparison of different project duration forecasting methods using earned value metrics. *International Journal of Project Management*, *24*(4), 289–302. https://doi.org/10.1016/j.ijproman.2005.10.004

Vanhoucke, M. (2012). *Project Management with Dynamic Scheduling*.

Vanhoucke, M., & Vandevoorde, S. (2007). A simulation and evaluation of earned value metrics to forecast the project duration. In *Journal of the Operational Research Society* (Vol. 58, Issue 10, pp. 1361–1374). Palgrave Macmillan Ltd. https://doi.org/10.1057/palgrave.jors.2602296

W. Fleming, Q., & M. Koppelman, J. (1998). Earned Value Project Management. *The Journal of Defense Software Engineering*.

Wauters, M., & Vanhoucke, M. (2014). Support Vector Machine Regression for project control forecasting. *Automation in Construction*, *47*, 92–106. https://doi.org/10.1016/j.autcon.2014.07.014