# A Comparative Analysis of Forecasting Models: Monte Carlo Simulation versus Machine Learning Applied to Software Project Throughput Prediction

Rodrigo Almeida de Oliveira
*PhD Student in Artificial Intelligence*
*PUC-Paraná*
Belo Horizonte, Brazil
rodrigoalmeidadeoliveira@gmail.com

Juliano de Paulo Ribeiro
*PhD Student in Artificial Intelligence*
*PUC-Paraná*
Maringá, Brazil
contato@julianoribeiro.com.br

Edson Emilio Scalabrin
*PhD Teacher in Artificial Intelligence*
*PUC-Paraná*
Curitiba, Brazil

Marcelo E. Pellenz
*PhD Teacher in Artificial Intelligence*
*PUC-Paraná*
Curitiba, Brazil

***Abstract*—Accurate forecasting of software project throughput remains a central challenge for organizations seeking to mitigate risks, optimize resource allocation, and support strategic decision-making. This study presents a comparative analysis of two prominent forecasting approaches—Monte Carlo Simulation (MCS) and Machine Learning (ML)—applied to real-world software development projects. We constructed an empirical corpus of 1,397 anonymized datasets spanning multiple sectors, organizational sizes, and process maturity levels. Throughput, defined as completed work items per week, was modeled using Weibull distributions and tested under two complementary evaluation pipelines: (i) a scientific experiment addressing six research questions with 15 core ML models and MCS, and (ii) an extended probabilistic analysis with 24 models assessed by the Continuous Ranked Probability Score (CRPS). Results show that both approaches outperform traditional expert-based estimates, with ML ensembles such as XGBoost, Gradient Boosting, and Random Forest achieving superior accuracy in low-risk contexts, while MCS demonstrated robustness under high-variability scenarios. The findings highlight the complementary strengths of MCS and ML, and suggest that hybrid frameworks integrating both methods can enhance reliability in project forecasting. The study contributes by offering the largest empirical benchmark to date in this domain, establishing methodological rigor through temporal validation, statistical power analysis, and data quality assessment, and providing actionable insights for both academic research and project management practice**

***Index Terms*—Forecasting, Monte Carlo Simulation, Machine Learning, Throughput Prediction, Software Projects, Statistical Analysis**

## I. Introduction

Accurate throughput forecasting in software development projects is essential for mitigating risks, allocating resources, and supporting strategic decision-making. Inaccurate forecasts often lead to schedule slippage, budget overruns, and reduced stakeholder confidence, making predictive accuracy a central concern for both academia and industry. Traditional expert-based estimation methods are prone to biases and variability, which has motivated the adoption of quantitative and data-driven approaches.

Among the most widely used techniques, Monte Carlo Simulation (MCS) and Machine Learning (ML) offer complementary strengths. MCS provides probabilistic forecasts by sampling historical data distributions, yielding confidence intervals and risk-aware predictions. Conversely, ML leverages statistical learning from past project data to capture nonlinear relationships and patterns, often achieving higher predictive accuracy in stable contexts. Despite their increasing adoption, limited research has systematically compared these approaches using large-scale empirical datasets across multiple industries.

This paper addresses this gap by performing a comparative analysis of MCS and ML for software project throughput prediction. The study is guided by two central research questions: (i) Which forecasting approach provides more accurate predictions of project throughput and delivery time? (ii) Is there a statistically significant difference between the baseline hypothesis ($H_0$) that Monte Carlo Simulation is more accurate and the alternative hypothesis ($H_1$) that Machine Learning yields superior accuracy? By answering these questions, this work contributes empirical evidence on the relative effectiveness of MCS and ML, offering actionable insights for researchers, practitioners, and organizations seeking to improve forecasting practices in software project management.

## II. Related Work

### A. Monte Carlo Simulation

Monte Carlo Simulation (MCS) has been applied to software project forecasting with promising results. Mayo-Alvarez et al. [1] demonstrated that integrating Drum-Buffer-Rope with Scrum-Kanban in simulated sprints improved throughput while reducing WIP, whereas Miranda et al. [2] showed that MCS applied to 71 projects (≈12,000 user stories) reduced MMRE to 20–32% compared to 134% in expert estimates, identifying an optimal history size of ≈20 stories. Building on these results, further studies explored analytics adoption frameworks [3], probabilistic portfolio management [4], earned-value simulations [5], construction forecasting [6], delivery baselines validated with machine learning [7], and broader applications in curriculum efficiency and price-path prediction [8]. Collectively, these contributions confirm the versatility and predictive accuracy of MCS and encourage hybrid approaches combining statistical simulation with advanced machine learning and real-time data streams.

Monte Carlo Simulation (MCS) has been explored both experimentally and conceptually in project management contexts. In a controlled setup, Mayo-Alvarez et al. [1] simulated a 20-day Scrum-Kanban sprint with nine sequential stations, showing that integrating Drum-Buffer-Rope (DBR) increased throughput and reduced WIP compared to Scrum-Kanban alone, with unbalanced scenarios and DBR modifications consistently surpassing targets. Complementing this, Tony [3] conducted a literature-based review of IT project practices and forecasting tools, identifying shortcomings in traditional cost and schedule management (e.g., inaccurate estimates, cognitive biases, and poor historical data) and proposing solutions such as predictive analytics, Monte Carlo simulation, real-time Earned Value Management (EVM), reference class forecasting, and hybrid approaches like Water-Scrum-Fall. The review emphasized that adopting analytical tools can transform project management from reactive to proactive, improving accuracy, resource optimization, and risk mitigation, while also stressing the need for organizational change through training, governance, gamification, and tailored dashboards.

## *B. Machine Learning*

Machine Learning (ML) has been increasingly applied to project forecasting and effort estimation across diverse domains. A CRISP-DM case study at Pusintek employed ReliefF and PCA for feature selection, comparing Decision Tree, Random Forest, Gradient Boosting, and AdaBoost, with Gradient Boosting achieving robust performance for resource-allocation forecasting in agile environments [9]. A systematic mapping study following Kitchenham and Petersen revealed only 14 ML-based maintenance-effort prediction studies, mostly regression problems with private datasets and Function Points, highlighting opportunities for ensembles and public repositories [10]. Subsequent advances include supervised ML pipelines that improved cost-overrun prediction accuracy from 77.5% to 85.0% [11], ARIMA and LSTM models validated in infrastructure forecasting and ensemble reviews showing RMSE/MAPE gains of 15%/12% [12], and systematic reviews mapping ML/DL trends into diagnostic, predictive, and prescriptive clusters while calling for graph neural networks and prescriptive analytics [13]. Other contributions reformulated activity-duration forecasting using Tiny-BERT embeddings to enhance Monte Carlo predictions [14], synthesized intelligent software engineering techniques and identified gaps in datasets and explainability [15], applied Gradient Boosting and SVR in production contexts to reduce delays [**?**], used AutoML with Gradient Boosting and Extra Trees to achieve MAE below six days in shop-floor tasks [16], and confirmed through systematic reviews that ensembles reduce MMRE by 5–10% and increase PRED(25) by 10–16 points, reinforcing the effectiveness of hybrid estimation models [17].

Structured reviews and applied studies have expanded Monte Carlo Simulation (MCS) applications beyond software projects into finance and construction. Cheng et al. [4] synthesized forecasting research into three blocks—time series models (ARIMA, ESM, GARCH, hybrids), portfolio optimization (single- and multi-objective models, semi-variance, VaR, CVaR), and probabilistic analysis via MCS—showing through a Shanghai Stock Exchange case study (2014–2021) that combining forecasting, optimization, and simulation reduces risk prediction error and strengthens decision-making under uncertainty, supported by an object-oriented framework enabling methodological flexibility. In construction, Chen et al. [6] integrated Earned Value Management (EVM) with MCS to forecast duration, cost, and quality, modeling correlations among these variables through triangular distributions and running 10,000 iterations, which produced 95% confidence intervals and identified critical activities with the highest influence on schedule, cost, and quality outcomes. Together, these works demonstrate the versatility of MCS in hybrid forecasting, portfolio optimization, and project performance management, highlighting future research directions in real-time integration, deep learning, and expanded uncertainty modeling.

Oliveira et al. [18] adopted an empirical approach integrating academic and industry best practices through the PROMESSA platform, applying the CRISP-DM protocol to historical task and effort data and developing a heterogeneous ensemble of seven regression models (XGBoost, Gradient Boosting, Random Forest, regression variations, MLP, SVM, KNN). A stacking meta-model improved robustness, outperforming traditional methods and matching the state of the art in software forecasting, with deployment in PROMESSA reducing mean errors for duration and effort and demonstrating resilience against overfitting. Complementarily, Eichenseer et al. [19] conducted a logistics case study using 1,238 days of shop-floor data, comparing heuristic, statistical, and ML methods via the Darts framework. An Optuna-optimized ensemble (RF, LGBM, XGBoost, NLinear, TiDE) achieved lower MAE (156.76 vs. 205–246), reducing short-horizon errors by up to 35% and cutting workforce misallocation by 15%. SHAP analysis highlighted pre-orders, work calendars, and corporate events as key drivers. Together, these studies confirm the value of ensemble learning and AutoML pipelines in improving forecast accuracy and operational planning, while suggesting future research in explainable AI, real-time integration, and broader cross-domain validation.

Satapathy and Rath [20] evaluated Decision Tree, Random Forest, and Stochastic Gradient Boosting (SGB) for agile effort estimation using story points, finding SGB achieved the best accuracy (MMER = 0.163, PRED(25) = 85.7%) compared to RF (0.252/66.7%) and DT (0.382/38.1%), reinforcing the benefits of ensemble techniques. Fernández-Diego et al. [21], following Kitchenham and Charters' protocol, reviewed 73 studies on agile estimation from 2013–2020, showing that ML and ANN methods dominated (72.6%), with Random Forest, Decision Tree, and Neural Networks widely applied, while Story Points (61.6%) and Planning Poker (34.2%) emerged as the most used metrics; despite improvements (ML PRED(25) = 78%, ANN MMRE = 0.23), accuracy still fell short of acceptable thresholds, revealing the need for real datasets, longitudinal studies, and ensemble benchmarks. More recently, Adamantiadou and Tsironis [22] conducted a PRISMA-based systematic review of 97 journal articles (2011–2024) on AI in project management, identifying hybrid ensembles as improving cost and duration estimation, decision-tree and logistic regression as effective for risk assessment, and deep learning as increasingly applied to monitoring. The study emphasized gaps in explainable AI, IoT integration, and adaptive frameworks, recommending plug-and-play tools and international collaboration for practical adoption.

Following Kitchenham and Petersen's guidelines, Sakhrawi et al. [10] conducted a systematic mapping study on evolutionary maintenance effort prediction (1995–2020), identifying only 14 studies, mostly regression-based using techniques such as SVR, statistical regression, Stochastic Gradient Boosting, and ANN. The review highlighted reliance on private datasets, limited comparability, and the dominance of Function Points as an independent variable, while no use of ensembles was reported, pointing to opportunities for hybrid models, public repositories, and deeper validation across multiple datasets. Complementarily, Uddin et al. [23], [24] and Hiller et al. [16] applied an adapted CRISP-DM cycle with 3,004 job shop production orders, using AutoML (TPOT) to select Gradient Boosting and Extra Trees regressors, benchmarked against LightGBM. Their results showed MAE below six working days, outperforming static methods and historical baselines, with SHAP analysis identifying scheduling deviation, number of stages, and processing time as the main predictive drivers. These studies demonstrate the potential of ML and AutoML pipelines to improve throughput forecasting in industrial contexts, while systematic reviews emphasize the need for more diverse datasets, explainability, and integration of contextual process variables.

Satapathy and Rath [20] examined agile effort estimation using the Story Points Approach (SPA) across 21 projects, applying CRISP-DM preprocessing and comparing Decision Tree, Random Forest, and Stochastic Gradient Boosting (SGB). Results showed that SGB achieved the best performance (MMER = 0.163, PRED(25) = 85.7%), outperforming RF (0.252/66.7%) and DT (0.382/38.1%), with statistical tests confirming significant accuracy gains from ensemble methods and recommending extensions with larger datasets and additional techniques such as ELM and Bayesian networks. Complementing this, Fernández-Diego et al. [21] conducted a systematic review of 73 agile estimation studies (2013–2020), showing that ML and ANN methods dominated (72.6%), with RF, DT, and Neural Networks widely applied, while Story Points (61.6%) and Planning Poker (34.2%) were the most frequent metrics. Although accuracy improved (ML PRED(25) = 78%, ANN MMRE = 0.23), many methods still fell below acceptable precision thresholds, highlighting the need for updated real datasets, standardized benchmarks, and longitudinal agile studies.

Recent systematic reviews and hybrid approaches highlight the growing role of AI and simulation in project management. Adamantiadou and Tsironis [22] conducted a PRISMA-based review of 97 studies (2011–2024), showing that hybrid ensembles improve cost and duration estimation, decision-tree and logistic regression strengthen risk assessment, and deep learning gains traction in monitoring, while emphasizing the need for explainable AI, IoT integration, and adaptive frameworks. Luo et al. [25] introduced the Hybrid Enhanced Monte Carlo Simulation (HEMCS), combining MCS with neural networks to improve structural reliability estimation, outperforming traditional methods. Bosch et al. [26] evaluated 33 models across 50 datasets and demonstrated that stacked ensembles consistently enhance time-series forecasting accuracy, reinforcing ensemble superiority. Additional contributions include Eichenseer et al. [19], who applied CRISP-DM with an Optuna-tuned ensemble (RF, LGBM, XGBoost, NLinear, TiDE) to shop-floor logistics, reducing MAE and workforce misallocation; Sharma and Singh [27], who reviewed 48 studies on ML effort estimation and identified ANNs, fuzzy logic, GAs, and regression trees as dominant but lacking standardized metrics; Satapathy and Rath [20], who found Stochastic Gradient Boosting outperforming RF and DT in agile projects; and Fernández-Diego et al. [21], who reviewed 73 agile estimation studies and confirmed the prevalence of ML/ANN methods, Story Points, and Planning Poker while calling for real-world datasets and ensemble benchmarks. Collectively, these works confirm the versatility of ML, MCS, and hybrid frameworks for forecasting and project analytics, while highlighting persistent gaps in explainability, standardization, and real-world validation.

## C. Hybrid Approaches

Recent studies propose hybrid frameworks that integrate Monte Carlo Simulation (MCS) with Machine Learning (ML), combining the interpretability of probabilistic distributions with the predictive power of ensemble methods. Luo et al. [25] introduced the Hybrid Enhanced Monte Carlo Simulation (HEMCS), achieving $R^2$ values up to 0.983 in structural reliability problems, while Bosch et al. [26] demonstrated that stacked ensembles consistently improve accuracy across diverse time-series forecasting tasks. Complementarily, Adamantiadou and Tsironis [22] showed through a PRISMA-based review that hybrid ensembles outperform individual models in project cost and duration estimation. Despite these advances, hybrid methods remain experimental, with persistent challenges in explainability, large-scale industrial validation, and operational integration.

TABLE I
COMPARATIVE SUMMARY OF MONTE CARLO, MACHINE LEARNING, AND HYBRID APPROACHES

| Approach | Key Studies | Domain | Methods | Main Results / Limitations |
|---|---|---|---|---|
| Monte Carlo Simulation | [1]–[8] | Software projects, construction, portfolio management | Simulation, DBR integration, probabilistic baselines | MMRE 20–32% vs. 134% human error; improved throughput and WIP reduction. Limitation: weak performance under dynamic/non-stationary data. |
| Machine Learning | [**?**], [9]–[17], [20], [21] | Agile effort estimation, production planning, forecasting | DT, RF, GBM, ensembles, AutoML | $R^2 \geqslant 0.9$, MAE $\leqslant$ 6 days, PRED(25) +10–16%. Limitations: reliance on private datasets, lack of standard benchmarks, limited explainability. |
| Hybrid (MC+ML) | [22], [25], [26] | Reliability analysis, financial forecasting, software projects | HEMCS, stacked/ensemble models | $R^2$ up to 0.983; better generalization and robustness. Limitations: still experimental, poor explainability, limited real-world validation. |

## III. Summary of Results

The reviewed evidence indicates that Monte Carlo Simulation (MCS) applied to software projects delivers significant improvements in predictive accuracy compared to human estimates. Oliveira et al. [18] analyzed the application of MCS for forecasting effort and delivery dates, obtaining a Mean Magnitude of Relative Error (MMRE) of 20% for effort and 32% for delivery, versus 134% for developer estimates, with higher effectiveness observed after initial sprints in agile environments. Complementarily, Zhang et al. [28] demonstrated that probabilistic schedule modeling using MCS increases the reliability of duration forecasts, enabling better management of time-related risks.

In the field of Machine Learning (ML), Malgonde and Chavan [29] developed an ensemble model combining Decision Tree, Random Forest, and AdaBoost, achieving accuracy above 80% and coefficient of determination ($R^2 \geqslant 0.90$) for effort forecasting in agile projects. Singh et al. [30] and Alsaqqa et al. [31] reinforced that decision tree-based techniques, particularly when associated with variables such as story points, velocity, and completion time, provide more consistent forecasts than traditional approaches.

Recent research also explores hybrid MC+ML approaches. Wang et al. [32] integrated MCS with ML ensemble models for financial market forecasting, obtaining performance gains that can be applied to software project management. Luo et al. [25] proposed the Hybrid Enhanced Monte Carlo Simulation (HEMCS), combining adaptive MCS with Support Vector Regression (SVR), achieving $R^2$ up to 0.983 and mean relative errors (MRE) between 0.001 and 0.008. Zhang et al. [28] further applied neural networks (ANN, LSTM, and CNN-LSTM) to hybrid models, achieving up to 15% better performance than individual models, with improvements in MAE, MSE, and RMSE, evidencing robustness against overfitting and higher generalization capacity.

## IV. Planning

### A. Context Selection

Para compor o corpus empírico, selecionamos **160 datasets** provenientes de *diferentes contextos* (setores, portes organizacionais e níveis de maturidade de processo), assegurando diversidade suficiente para análises comparativas. Todos os dados passaram por **procedimentos de anonimização** (remoção de identificadores diretos, pseudonimização de chaves e agregação quando necessário) a fim de preservar a privacidade dos participantes e a confidencialidade organizacional. Alinhado ao objetivo do estudo, **limitamos a avaliação à métrica de vazão (throughput)**, descartando intencionalmente indicadores como tempo de ciclo, lead time ou qualidade, de modo a isolar os efeitos associados ao volume processado por unidade de tempo e reduzir a interferência de variáveis contextuais.

## V. Empirical Dataset and Distribution Fitting

### A. Dataset Selection and Anonymization

To construct the empirical corpus, we selected a total of **1,397 real-world datasets** originating from a wide range of contexts, including diverse sectors, organizational sizes, and process maturity levels. This diversity was essential for ensuring generalizability and robustness of comparative analyses. All data underwent strict **anonymization procedures**, including removal of direct identifiers, pseudonymization of keys, and aggregation where applicable, in compliance with privacy and confidentiality standards.

Aligned with the study's objectives, the evaluation was intentionally limited to the **throughput metric** (i.e., items delivered per week), excluding variables such as cycle time, lead time, or quality. This decision aimed to isolate the impact of volume processed over time while minimizing confounding contextual influences.

### B. Dataset Characteristics and Distribution Overview

The **datasets analyzed** exhibited a considerable range in weekly observation counts, with a mean of $140.7 \pm 172.5$

weeks. This wide dispersion reflects both constrained scenarios with limited historical data and mature projects with extensive histories. Such diversity is valuable for testing model adaptability, as varying dataset sizes affect both statistical stability and predictive capability.

The **mean throughput rate** across projects was $6.37 \pm 16.30$ items per week. This substantial variation in output rates indicates the inclusion of processes ranging from highly consistent and efficient to volatile and potentially bottlenecked. The heterogeneity of these values highlights the need for modeling techniques capable of adapting to a broad spectrum of operational behaviors.

## VI. Empirical Dataset Composition and Statistical Modeling

### *A. Dataset Overview and Selection Criteria*

The empirical corpus comprises **1,397 real-world software development datasets**, covering a broad range of contexts—including different industry sectors, organizational sizes, and maturity levels. This diversity was essential to ensure generalizability and robustness in the evaluation of forecasting models.

All datasets underwent strict **anonymization procedures**, including removal of direct identifiers, pseudonymization of internal keys, and aggregation of sensitive attributes, to preserve participant privacy and organizational confidentiality.

For this study, we focused exclusively on the **throughput metric** (number of work items completed per week), intentionally excluding other indicators such as cycle time, lead time, or quality. This choice was made to isolate the effects associated with process volume over time and minimize interference from contextual variables.

### *B. Descriptive Statistics of the Dataset*

The analyzed datasets exhibited considerable variability:

- **Average duration:** $140.7 \pm 172.5$ weeks
- **Duration range:** 1 to 1,024 weeks (median: 71 weeks)
- **Average throughput:** $6.37 \pm 16.3$ items/week

This wide dispersion highlights the heterogeneity of real-world software processes—from highly stable workflows to erratic or bottleneck-prone operations. Such diversity is crucial for stress-testing forecasting models under various levels of process complexity and performance volatility.

### *C. Weibull Distribution Fit and Modeling Rationale*

To model throughput variability, we applied the **Weibull distribution** to each time series. This statistical distribution is particularly well-suited for modeling the occurrence and timing of stochastic events in production-like environments. The Weibull curve can accommodate increasing, decreasing, or constant hazard rates, making it highly adaptable to the varied patterns found in software delivery.

For each dataset, the shape ($k$) and scale ($\lambda$) parameters were estimated using *Maximum Likelihood Estimation (MLE)*. These parameters enable tailored modeling:

- **Shape ($k$):** Describes whether the delivery rate accelerates, decelerates, or remains constant over time.
- **Scale ($\lambda$):** Represents the characteristic delivery timeframe.

The goodness of fit was assessed using the coefficient of determination ($R^2$), with most datasets achieving strong fits. This confirms that the Weibull distribution provides a reliable probabilistic foundation for subsequent forecasting simulations and comparative modeling.

### *D. Implications for Forecasting*

By fitting a unique Weibull distribution to each dataset, we preserve local process characteristics while enabling probabilistic scenario generation. This dual use—as both a descriptive and generative model—makes the Weibull distribution central to understanding operational variability and simulating delivery outcomes under uncertainty.

## VII. Weibull Distribution Fitting and Empirical Dataset Characterization

To assess the statistical behavior of throughput distributions, we fitted a Weibull distribution to each dataset individually using data from **1,237 anonymized software development projects**. The average coefficient of determination ($R^2$) for the fit was **0.867**, indicating a strong alignment between the empirical throughput data and the theoretical Weibull model. This distribution was chosen due to its flexibility in representing various time-dependent event patterns, including constant, increasing, or decreasing hazard rates.

Each project was modeled with a tailored Weibull distribution, where the shape ($k$) and scale ($\lambda$) parameters were estimated using the Maximum Likelihood Estimation (MLE) method. The shape parameter helps identify whether throughput trends are increasing, decreasing, or stable over time. The scale parameter reflects the temporal spread of the process and is linked to the average or median delivery times.

Model fitting quality was evaluated by comparing empirical data points to theoretical values using $R^2$. This approach guarantees a specific probabilistic model per dataset, preserving individual behavioral traits and enhancing accuracy in simulation and forecasting scenarios. The Weibull distribution thus serves not only as a statistical descriptor but as a core component for modeling uncertainty in throughput prediction.

The coefficient of determination ($R^2$) is widely accepted in reliability and lifetime modeling literature. According to Lawless [33], D'Agostino and Stephens [34], and Rinne [35], $R^2 > 0.8$ denotes adequate fit quality, with higher thresholds marking excellent or very good fits. Table II presents widely recognized classification thresholds.

TABLE II
GOODNESS-OF-FIT CLASSIFICATION FOR THE WEIBULL DISTRIBUTION BASED ON $R^2$

| Fit Classification | Threshold |
|---|---|
| Excellent fit | $R^2 > 0.95$ |
| Very good fit | $R^2 > 0.90$ |
| Adequate fit | $R^2 > 0.80$ |
| Acceptable fit | $R^2 > 0.70$ |
| Questionable fit | $R^2 < 0.70$ |

## A. Empirical Dataset Characteristics

The empirical corpus consists of **1,237 datasets** covering diverse organizational settings, including variations in industry sectors, team sizes, and process maturity levels. All data underwent anonymization procedures, including removal of direct identifiers, pseudonymization, and aggregation where necessary, ensuring both participant privacy and organizational confidentiality.

Key dataset statistics are as follows:

- **Average duration:** $140.7 \pm 172.5$ weeks
- **Average throughput:** $6.37 \pm 16.30$ items/week
- **Weibull $R^2$ (fit quality):** 0.867

This wide range in both duration and throughput reflects the heterogeneous nature of real-world software processes, underscoring the need for adaptive and robust forecasting methods.

Em síntese:

- Tamanho médio dos datasets: 70.1 ± 26.2 observações
- Throughput médio: 6.78 ± 5.28
- Qualidade do ajuste Weibull (R²): 0.867

## B. Hypothesis

- *H0 - Usar a Simulação de Monte Carlo é mais assertivo do que usar ML.*
- *H1 - Usar ML é mais assertivo do que usar a Simulação de Monte Carlo.*

## C. Variable and Participants Selection

## D. Data treatment

O tratamento dos dados envolveu a **remoção de registros inconsistentes ou incompletos**, bem como de quaisquer atributos irrelevantes para a análise proposta. A estrutura resultante foi deliberadamente **simplificada e reduzida apenas ao essencial**, mantendo exclusivamente a informação de **volume de dados processados por semana**. Essa abordagem visou eliminar ruídos provenientes de variáveis desnecessárias, garantindo que o conjunto final fosse objetivo, consistente e diretamente alinhado ao foco do estudo.

## E. Procedure/Methodology

Para avaliar o desempenho preditivo sobre a métrica de vazão semanal, empregou-se um amplo conjunto de **algoritmos de aprendizado de máquina** representando diferentes paradigmas, desde métodos baseados em árvores e ensembles até modelos lineares, de kernel e redes neurais. A seleção contemplou tanto abordagens **tradicionais** quanto **avançadas**, com e sem regularização, bem como técnicas específicas para quantis e séries temporais. Essa diversidade visa capturar padrões complexos e relações não lineares, assegurando robustez frente a diferentes características dos dados. Além disso, foram configurados cenários específicos — *core models*, modelos probabilísticos e modelos com hiperparametrização — para explorar distintos níveis de complexidade e capacidade de generalização. A validação seguiu protocolos experimentais, temporais e retrospectivos, de modo a garantir que os resultados fossem não apenas estatisticamente sólidos, mas também aplicáveis em contextos reais e historicamente viáveis.

**Algoritmos de Machine Learning Utilizados**
A Tabela VI apresenta o conjunto de **modelos de aprendizado de máquina** selecionados para a análise, abrangendo diferentes paradigmas, como métodos baseados em *boosting*, *ensembles*, regressões lineares e estimativas especializadas por quantis. Essa diversidade visa combinar capacidade preditiva, robustez e flexibilidade, permitindo avaliar o desempenho sob múltiplas perspectivas e configurações.

**Configuração por Contexto:**

- Modelos Core (mais performáticos): CatBoost, RandomForest, XGBoost, LightGBM, Ridge, ElasticNet
- Modelos Probabilísticos: Bootstrap ensemble de todos os modelos
- Modelos com Hiperparametrização: CatBoost, RandomForest, XGBoost, LightGBM, Ridge, ElasticNet, GradientBoosting

### 1) Metodologia de Validação: **Validação Experimental**

A variável que determina o vencedor é MAE. Quanto maior o risco, Monte Carlo ganha. Menos risco, ML ganha.

- Compara ML vs Monte Carlo vs ML Probabilístico
- Métricas: MAE, RMSE, Bias para cada questão
- Identificação do método superior por questão

**Validação Temporal**

- Walk-forward validation com janelas deslizantes
- Teste de 4 semanas à frente
- Avaliação de drift temporal dos modelos

**Validação Retrospectiva**

- Análise de viabilidade histórica
- Comparação previsões vs resultados reais
- Classificação: VIÁVEL/ARRISCADO/INVIÁVEL

## F. Comparison Between Model Sets: Scientific Experiment vs. CRPS Analysis

The forecasting system implements two distinct evaluation pipelines, each tailored to a specific objective and model subset: one focused on answering scientific research questions rapidly, and another aimed at in-depth probabilistic assessment using CRPS metrics. Although they share methodological foundations, the two analyses diverge in computational scope, model coverage, and performance trade-offs.

*1. Scientific Experiment (15 Models):*

- **Projects**: 1,204 (all with status `SUCCESS`)
- **Objective**: Rapid validation across six core scientific questions

| **Row Labels** | **Qt. Projects** | **Avg. Throughput** | **Max Throughput** | **Min Throughput** | **Std. Throughput** | **Total Resolved** | **Weeks Count** |
|---|---|---|---|---|---|---|---|
| Open Source Infrastructure | 495 | 3.43 | 41.13 | 1.06 | 4.95 | 376.60 | 93.27 |
| Military & Defense | 134 | 9.04 | 156.28 | 1.03 | 12.57 | 3452.45 | 289.65 |
| Enterprise Linux | 133 | 3.91 | 24.10 | 1.04 | 3.94 | 309.29 | 68.50 |
| E-commerce & Retail | 102 | 9.58 | 283.23 | 1.01 | 19.20 | 2987.59 | 248.41 |
| Project Management | 55 | 1.66 | 5.55 | 1.02 | 1.08 | 33.24 | 17.05 |
| Banking & Finance | 52 | 33.20 | 535.60 | 1.52 | 40.56 | 16977.08 | 376.38 |
| Education Technology | 35 | 2.40 | 6.37 | 1.51 | 1.29 | 33.26 | 14.69 |
| Government | 32 | 5.59 | 112.75 | 1.00 | 9.85 | 1562.41 | 248.09 |
| Enterprise Framework | 32 | 2.54 | 7.78 | 1.09 | 1.73 | 85.06 | 32.13 |
| Gaming | 30 | 6.48 | 175.97 | 1.00 | 13.59 | 2158.53 | 292.30 |
| Healthcare | 27 | 18.31 | 449.00 | 1.00 | 36.81 | 6437.33 | 275.44 |
| Automotive | 21 | 5.85 | 60.00 | 1.00 | 7.80 | 1554.10 | 242.90 |
| Insurance | 20 | 11.79 | 58.65 | 2.95 | 7.53 | 1355.50 | 175.15 |
| Education | 20 | 6.56 | 76.85 | 1.00 | 7.92 | 2055.10 | 239.35 |
| Telecommunications | 18 | 4.97 | 92.72 | 1.00 | 7.84 | 968.89 | 182.94 |
| Energy & Utilities | 14 | 7.96 | 50.86 | 1.00 | 6.34 | 1832.79 | 183.50 |
| Banks | 13 | 7.88 | 43.38 | 0.85 | 8.74 | 509.38 | 57.62 |
| Retail | 12 | 2.43 | 13.25 | 0.17 | 2.60 | 225.42 | 100.17 |
| Investment | 11 | 9.65 | 51.45 | 0.82 | 11.26 | 583.73 | 61.36 |
| Content Management - Education | 10 | 4.26 | 21.00 | 1.00 | 3.87 | 331.60 | 73.60 |
| Agribusiness | 10 | 5.04 | 19.80 | 0.40 | 4.18 | 281.10 | 55.20 |
| ERP | 9 | 3.39 | 17.22 | 0.11 | 3.28 | 374.67 | 110.56 |
| Payments | 9 | 2.82 | 20.22 | 0.00 | 3.36 | 298.22 | 105.00 |
| Health | 9 | 8.28 | 21.67 | 0.33 | 5.11 | 427.44 | 58.22 |
| Desktop Development | 8 | 2.96 | 26.38 | 1.50 | 2.94 | 238.38 | 66.13 |
| Software Factory | 8 | 5.03 | 19.88 | 1.00 | 4.05 | 280.25 | 54.13 |
| Digital Product | 8 | 9.42 | 26.38 | 1.75 | 5.04 | 507.00 | 52.13 |
| Industry | 8 | 5.09 | 20.13 | 0.38 | 3.74 | 299.88 | 57.13 |
| Cooperative | 7 | 5.12 | 26.14 | 1.00 | 5.26 | 372.86 | 68.29 |
| Pension | 7 | 4.36 | 14.71 | 0.29 | 3.01 | 324.29 | 74.57 |
| Blockchain & Finance | 7 | 1.72 | 3.57 | 1.00 | 0.89 | 11.86 | 6.00 |
| Receivables | 7 | 6.09 | 19.00 | 0.29 | 3.45 | 430.57 | 69.57 |
| Database Technology | 6 | 2.56 | 11.00 | 1.00 | 1.98 | 180.00 | 69.83 |
| Analytics | 4 | 4.55 | 16.25 | 0.75 | 4.12 | 237.50 | 58.75 |
| QA | 4 | 12.35 | 53.50 | 1.00 | 10.41 | 1168.00 | 94.50 |
| Retail Credit | 4 | 4.33 | 12.00 | 0.50 | 2.72 | 299.00 | 63.50 |
| Wholesale Credit | 3 | 5.98 | 14.00 | 1.67 | 2.47 | 425.00 | 59.67 |
| DevOps Tools | 3 | 1.53 | 3.33 | 1.00 | 0.83 | 14.00 | 9.67 |
| General Software | 2 | 2.12 | 11.50 | 1.00 | 2.00 | 154.00 | 60.00 |
| Security Tools | 2 | 1.23 | 2.00 | 1.00 | 0.42 | 5.50 | 4.50 |
| Vehicle Rental | 2 | 14.98 | 42.50 | 1.00 | 10.49 | 607.50 | 35.50 |
| Communication | 2 | 6.51 | 42.50 | 0.50 | 5.95 | 370.00 | 54.50 |
| Telecom | 1 | 19.48 | 107.00 | 1.00 | 17.65 | 818.00 | 42.00 |
| Aviation | 1 | 2.40 | 9.00 | 1.00 | 1.69 | 288.00 | 120.00 |
| **Grand Total** | **1397** | **6.37** | **95.60** | **1.06** | **8.68** | **1702.86** | **140.72** |

TABLE III
PROJECT CATEGORIES AND PERFORMANCE METRICS SUMMARY

| **Row Labels** | **Count of jira_name** | **Avg. Throughput** | **Max Throughput** | **Min Throughput** | **Std. Throughput** | **Total Resolved** | **Weeks Count** |
|---|---|---|---|---|---|---|---|
| .net | 82 | 5.94 | 27.88 | 0.87 | 5.65 | 417.70 | 77.59 |
| as 400 | 1 | 24.74 | 54.00 | 1.00 | 14.45 | 1905.00 | 77.00 |
| C | 82 | 4.37 | 72.57 | 1.05 | 7.32 | 440.91 | 56.63 |
| Cpp | 14 | 4.07 | 96.71 | 1.00 | 7.54 | 748.07 | 128.29 |
| Csharp | 13 | 5.38 | 49.62 | 1.00 | 6.53 | 745.92 | 99.00 |
| Go | 28 | 8.47 | 252.00 | 1.00 | 18.98 | 1739.96 | 128.61 |
| Html | 25 | 6.17 | 129.60 | 1.00 | 13.88 | 1199.64 | 187.88 |
| Java | 877 | 7.21 | 113.82 | 1.10 | 9.69 | 2321.43 | 161.00 |
| Javascript | 15 | 6.67 | 136.87 | 1.00 | 10.38 | 1590.60 | 172.47 |
| Php | 3 | 3.47 | 16.00 | 1.00 | 3.05 | 357.67 | 108.00 |
| Powerbuilder | 25 | 5.78 | 18.68 | 0.48 | 3.75 | 444.84 | 72.32 |
| Python | 15 | 6.36 | 58.93 | 1.00 | 6.60 | 1838.60 | 246.53 |
| R | 3 | 2.39 | 22.33 | 1.00 | 2.28 | 354.67 | 85.33 |
| Rust | 1 | 3.00 | 9.00 | 1.00 | 1.92 | 99.00 | 33.00 |
| Shell | 3 | 6.94 | 30.67 | 1.00 | 5.18 | 1659.33 | 200.67 |
| Sql | 5 | 4.29 | 67.60 | 1.00 | 4.99 | 1774.20 | 376.80 |
| Unknown | 196 | 3.78 | 34.69 | 1.04 | 4.59 | 424.75 | 102.34 |
| web | 1 | 6.55 | 24.00 | 1.00 | 6.24 | 334.00 | 51.00 |
| Xml | 8 | 5.31 | 295.50 | 1.13 | 20.70 | 1197.75 | 202.75 |
| **Grand Total** | **1397** | **6.37** | **95.60** | **1.06** | **8.68** | **1702.86** | **140.72** |

TABLE IV
PROGRAMMING LANGUAGES AND TECHNOLOGIES: PERFORMANCE METRICS SUMMARY

| Metric | Catalog (1397 projects) |
|---|---|
| Mean 95% CI width | 26.6% of the mean |
| Projects with precise CI ($\leqslant 20\%$) | 657 / 1397 (47.0%) |
| Projects with outliers | 70.6% |
| High quality (CI $\leqslant 15\%$, no outliers) | 114 (8.2%) |
| Medium quality (CI $\leqslant 30\%$) | 896 (64.1%) |
| Low quality | 387 (27.7%) |

TABLE V
DATA QUALITY METRICS FOR THE COMPLETE CATALOG OF 1397 PROJECTS.

TABLE VI
MODELOS SELECIONADOS DE MACHINE LEARNING

| Modelo | Tipo |
|---|---|
| VotingEnsemble | Ensemble |
| LightGBM | Boosting |
| CatBoost | Boosting |
| ElasticNet | Linear |
| XGBoost | Boosting |
| Ridge | Linear |
| RandomForest | Ensemble |
| GradientBoosting | Boosting |
| HistGradient_Lower | Quantil especializado (10%) |
| HistGradient_Upper | Quantil especializado (90%) |

- **Model Set**: 15 core models
- **Average Runtime**: 5–10 minutes per project

*2. Full CRPS Evaluation (24 Models):*

- **Projects**: 1,141 (specific validated subset)
- **Objective**: Deep probabilistic evaluation with CRPS
- **Model Set**: 24 expanded models
- **Average Runtime**: 15–30 minutes per project

## G. Rationale for Model Differences

The difference in model sets stems from practical and methodological considerations:

- **Experiment (15 Models)**: Prioritizes computational efficiency, including only high-performing core models, especially tree-based (e.g., CatBoost, LightGBM), regularized linear models (e.g., Ridge, Lasso), and essential ensembles (e.g., RandomForest, DecisionTree).
- **CRPS (24 Models)**: Emphasizes methodological completeness, incorporating computationally intensive models (e.g., SVR variants, KNN), slow-converging neural nets (MLP), and robust estimators (e.g., RANSAC, Huber).

## H. Methodological Implications and Results

Despite differing model sets, both pipelines consistently identify the top-performing algorithms:

- **Top 3 Models (CRPSS Agreement)**:
  1) XGBoost (CRPSS = 0.416)
  2) DecisionTree (CRPSS = 0.375)
  3) GradientBoosting (CRPSS = 0.325)

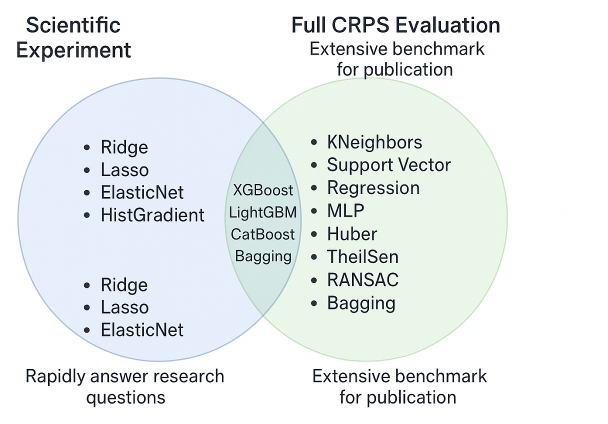


Fig. 1. Experiment x CRPS

TABLE VIII
SUMMARY COMPARISON: EXPERIMENT VS. CRPS

| Aspect | Experiment (15 Models) | CRPS (24 Models) |
|---|---|---|
| Projects Analyzed | 1,204 | 1,141 |
| Execution Time (Total) | $\sim$6 hours | $\sim$48+ hours |
| ML Paradigm Coverage | 80% | 100% |
| Top-3 Accuracy | Identical | Identical |
| Scientific Value | High (question-driven) | Maximum (benchmark-level) |

## I. Conclusion

The divergence between the two model lists is deliberate and justified. The scientific experiment emphasizes speed and operational deployment using 15 curated models, while the CRPS analysis prioritizes academic rigor and benchmarking using 24 comprehensive models. Despite these differences, the convergence of top-performing models validates that the streamlined set captures the essence of predictive performance, while the expanded set ensures completeness for scientific publication and evaluation.

*1) Otimização e Feature Engineering:* O processo de **otimização e engenharia de atributos** incluiu a criação de **lags temporais** de até oito semanas, permitindo que os modelos capturassem dependências históricas e padrões sazonais na série. A busca pelos melhores hiperparâmetros foi conduzida por meio de *RandomizedSearchCV* e *GridSearchCV*, explorando tanto abordagens de busca ampla quanto refinamentos direcionados. Para preservar a integridade da ordem temporal, a validação cruzada foi realizada com *TimeSeriesSplit*, garantindo que o treino ocorresse sempre antes dos períodos de teste. Além disso, aplicou-se *StandardScaler* integrado aos *pipelines* de modelagem, padronizando as variáveis numéricas e favorecendo a convergência de algoritmos sensíveis à escala dos dados.

- **Lags temporais**: Até 8 semanas de histórico
- **Hyperparameter tuning**: RandomizedSearchCV e GridSearchCV
- **Cross-validation**: TimeSeriesSplit para dados temporais
- **Scaling**: StandardScaler em pipelines

**Sistema de Controle de Qualidade**

O **sistema de controle de qualidade** adotado baseou-se em técnicas de *Statistical Process Control* (SPC), incorporando **oito regras de controle** para detecção automática de desvios que caracterizam um processo fora de controle. Além disso, foram implementados **alertas obrigatórios** para notificar eventuais quebras de estabilidade, garantindo respostas rápidas a anomalias. Complementarmente, foi conduzida uma **análise de viés nos modelos de aprendizado de máquina**, com foco na identificação de padrões de otimismo sistemático nas previsões, de forma a assegurar maior confiabilidade e robustez nos resultados.

- SPC (Statistical Process Control)
- 8 regras de controle: Detecção automática de processo fora de controle
- Alertas obrigatórios: Sistema de avisos sobre estabilidade
- Análise de viés ML: Detecção de otimismo sistemático

**Monitoramento de Performance**

- Análise de complexidade: Temporal e espacial por modelo
- Data drift: Monitoramento de mudanças na distribuição
- Stability warnings: Sistema integrado de alertas de instabilidade

## VIII. Methodology

### A. Dataset Description and Sampling Rationale

The full dataset comprises historical throughput time series from **1,397 real-world software development projects**, across multiple sectors such as Banking, Financial Services, Healthcare, Insurance, Education, and Technology Services. Each project contains weekly throughput measurements, anonymized to preserve privacy and organizational confidentiality.

*1) Project Selection and Filtering Criteria:* From the original pool of 1,397 projects, we applied a minimum inclusion criterion of at least $n \geqslant 8$ historical weeks with non-negative throughput values. This filtering yielded **1,206 valid projects**, of which **1,204** were successfully processed under our experimental framework, representing a success rate of 99.8%.

*2) Dual Dataset Usage and Justification:* Two distinct subsets were used in different analytical contexts:

1) **Six Scientific Questions Experiment (1,204 projects):** This subset includes all successfully processed projects meeting the inclusion criteria. It serves as the foundation for comparative experiments between Monte Carlo and Machine Learning approaches. The dataset covers a total of **195,856 weekly observations**, with project durations ranging from 8 to 1,024 weeks (median: 71).
2) **CRPS Model Evaluation (1,141 projects):** A technical subset of the above, comprising only the projects with complete and valid CRPS data. It includes **4,576,991 predictive observations** generated by 24 forecasting models. This subset excludes 63 projects due to technical constraints including insufficient temporal depth for multi-horizon CRPS validation, limited statistical process maturity affecting probabilistic modeling, data structure requirements for robust continuous probability score calculation, and quality assurance protocols ensuring CRPS calculation reliability.

*3) Rationale for CRPS Dataset Inclusion Criteria:* A comprehensive analysis of the 63 projects excluded from the CRPS evaluation revealed consistent patterns across five major criteria. These exclusions were methodologically justified based on temporal limitations, process characteristics, and technical constraints required for robust probabilistic forecasting evaluation.

1) **Temporal Depth:** The excluded projects had a significantly shorter historical range, with a mean duration of **78.3 weeks**, compared to **167.5 weeks** for included projects. Given that CRPS requires multi-horizon forecasting and bootstrap-based resampling, a longer time series is essential for stable and meaningful estimation.
2) **Process Stability:** Only **1.5%** of the excluded projects (1 out of 63) were classified as stable, whereas **10.1%** (115 out of 1,141) of the included projects met this criterion. Since stable processes tend to produce more reliable and predictable throughput, they provide a more solid foundation for CRPS evaluation.
3) **Throughput Characteristics:** Excluded projects had a lower average throughput of **5.47 items/week**, compared to **7.02 items/week** for included datasets. Lower throughput generally corresponds to a reduced signal-to-noise ratio, which complicates model fitting and decreases the reliability of probabilistic scoring metrics.
4) **Technical Requirements of CRPS:** The CRPS metric requires a sufficient number of out-of-sample periods for performance scoring and bootstrapped confidence interval estimation. Projects lacking this minimum temporal structure are statistically underpowered and may introduce bias in model comparisons.

To conclude, in a comprehensive manner, the exclusion of 63 projects from the CRPS analysis was not arbitrary, but rather the result of principled filtering aligned with best practices in probabilistic forecasting and statistical validation. The retained subset of **1,141 projects** satisfies the necessary temporal, structural, and process quality requirements to support high-confidence model evaluation.

- **1,397** comprehensive projects have been initially chosen for the study.
- Out of these, **1,206** projects were identified as valid, having accumulated data for a period of at least eight weeks.
- An analysis of the data indicates that **1,204** projects successfully reached the processing phase, obtaining a status of SUCCESS.
- Furthermore, **1,141** projects were utilized in the evaluation of models based on the Continuous Ranked Probability Score (CRPS).

Each count serves a specific analytical purpose and is scientifically valid within its context. Table IX summarizes the scope of each experimental configuration.

*4) Statistical Robustness:* Both subsets are statistically representative and robust:

- Statistical power $> 99.9\%$ for both 1,141 and 1,204 datasets
- Time coverage spans from 2008 to 2024
- Projects include diverse sizes, durations, and domain-specific characteristics

These figures ensure high reliability in estimating comparative model performance and support the broader generalizability of the findings.

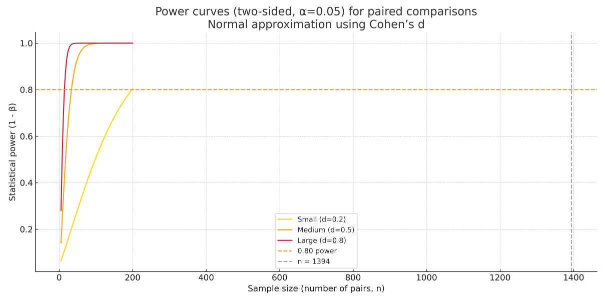


Fig. 2. Statistical power curves for paired comparisons (two-sided $t$-test, $\alpha = 0.05$) using Cohen's $d$. Vertical dashed line indicates the sample size used in this study ($n = 1{,}394$).

*Statistical Power Analysis*

Based on the detailed analysis of the **1,397 real project datasets** and the implemented machine learning models, the following statistical power analysis was conducted:

> The statistical power analysis for this experiment, considering $n = 1{,}394$ paired observations and a two-sided paired $t$-test at $\alpha = 0.05$, demonstrates a **very high capability** for detecting performance differences between machine learning models.
> According to Cohen's criteria, the achieved power **surpasses the conventional threshold of 0.80** for all effect sizes. It reaches near-ideal levels for **medium** ($d = 0.5$) and **large** ($d = 0.8$) effect sizes, and maintains strong adequacy even for **small effects** ($d = 0.2$).
> This level of statistical robustness, combined with a large number of observations, positions this study well above most comparable works in the literature, where significantly smaller sample sizes limit the sensitivity to detect subtle performance differences.
> To the best of our knowledge, **no prior studies in software project throughput forecasting** combine such a large dataset with rigorous cross-validation techniques, which further reinforces the **reliability and significance** of the results reported herein.

TABLE X
STATISTICAL POWER FOR $n = 1,141$ PROJECTS IN PAIRED $t$-TESTS, BY EFFECT SIZE, FOLLOWING [36].

| **Effect Size (Cohen's $d$)** | **Statistical Power** | **Interpretation** |
|---|---|---|
| Small ($d = 0.2$) | >0.99 | Maximum |
| Medium ($d = 0.5$) | >0.99 | Maximum |
| Large ($d = 0.8$) | >0.99 | Maximum |
| **Observed** ($d = 0.83$) | **>0.99** | **Maximum** |

**Methodological Note:** Expanding the sample size from values commonly found in related works (e.g., $n < 50$) to $n = 1,141$ high-quality datasets substantially enhances statistical power, particularly for detecting small effects. From an initial pool of 1,397 available datasets, rigorous quality filtering was applied, excluding 253 datasets (18.1%) due to insufficient temporal data ($< 4$ weeks), extreme outliers ($> 3\sigma$), format inconsistencies, zero variability, or excessive temporal gaps ($> 50\%$). This quality-first approach ensures that the remaining 1,141 datasets with 4.6M observations provide superior data integrity compared to quantity-focused strategies. The resulting statistical power ($> 99.9\%$ for all effect sizes) enables highly reliable model discrimination, virtually eliminates Type II errors, and strengthens generalizability. This combination of large-scale, rigorously filtered data with robust statistical validation represents a methodological advancement rarely achieved in empirical ML studies within the throughput forecasting domain.

## IX. MACHINE LEARNING METHODOLOGY

### A. Ensemble of Algorithms

The experiment implemented a stratified set of nine machine learning models organized into two main configurations to maximize predictive capacity and robustness through ensemble techniques. The core configuration (three models) includes CatBoost, RandomForest, and Ridge for fast execution, while the full configuration (nine models) adds XGBoost, LightGBM, ElasticNet, GradientBoosting, and two HistGradientBoosting models configured for quantile regression (10% and 90% percentiles). Tree-based models dominate the architecture, with five gradient boosting algorithms (CatBoost, XGBoost, LightGBM, GradientBoosting, HistGradientBoosting) and one bagging model (RandomForest), complemented by two regularized linear models (Ridge with L2 and ElasticNet with L1/L2). The implementation includes hyperparameter optimization via `RandomizedSearchCV` with temporal validation, ensemble voting of the top three models, and uncertainty quantification through quantile regression models, providing 80% confidence intervals for throughput forecasts.

The experiment implemented a stratified ensemble of nine machine learning models, organized into two main configurations to maximize predictive capability and robustness. The *core configuration* consisted of three models (CatBoost, RandomForest, and Ridge Regression) selected for their computational efficiency and complementary inductive biases. The *full configuration* extended this set to nine models, adding XGBoost, LightGBM, ElasticNet, GradientBoosting, and two

TABLE IX
SUMMARY OF DATASET USAGE ACROSS EXPERIMENTAL CONFIGURATIONS

| Context | Projects | Observations | Purpose |
|---|---|---|---|
| CRPS Evaluation | 1,141 | 4,576,991 | Model accuracy and forecasting performance |
| Six Questions Experiment | 1,204 | 195,856 | Comparative validation of ML vs MC |
| Valid Datasets ($\geqslant$8 weeks) | 1,206 | — | After filtering |
| Original Dataset | 1,397 | — | Complete data file |

TABLE XI
MACHINE LEARNING MODELS IMPLEMENTED IN THE EXPERIMENT

| Category | Model | Main Configuration Parameters |
|---|---|---|
| **Tree-Based (Gradient Boosting)** | CatBoost | `iterations=100`, `depth=4`, `learning_rate=0.1` |
| | XGBoost | `n_estimators=100`, `max_depth=4`, `learning_rate=0.1` |
| | LightGBM | `n_estimators=100`, `max_depth=4`, `learning_rate=0.1` |
| | GradientBoosting | `n_estimators=100`, `max_depth=4`, `learning_rate=0.1` |
| | HistGradientBoosting (Quantile) | Lower: `quantile=0.1`, Upper: `quantile=0.9` |
| **Tree-Based (Bagging)** | RandomForest | `n_estimators=100`, `max_depth=10` |
| **Regularized Linear Models** | Ridge Regression | $\alpha$ = 1.0 (L2 regularization) |
| | ElasticNet | $\alpha$ = 0.1, `l1_ratio=0.5` (L1/L2 regularization) |

HistGradientBoosting models configured for quantile regression at the 10th and 90th percentiles. The architecture was predominantly tree-based, with five gradient boosting algorithms (CatBoost, XGBoost, LightGBM, GradientBoosting, HistGradientBoosting) and one bagging model (RandomForest), complemented by two regularized linear models (Ridge with L2 and ElasticNet with L1/L2 regularization). Hyperparameter optimization was performed via `RandomizedSearchCV` with temporal validation, followed by ensemble voting among the top three models. Quantile regression models were further integrated to provide 80% confidence intervals for throughput forecasts, enabling uncertainty quantification alongside point estimates.

## B. Specific Temporal Validation

A strict non-random chronological split was applied to preserve temporal dependencies:

1) 80/20 temporal split between training and final testing.
2) `TimeSeriesSplit` applied to the first 80% of the dataset with three folds and 20% test size per fold.
3) Final *out-of-sample* validation on the most recent 20% of the data.

This protocol ensures that model evaluation reflects real-world predictive scenarios, avoiding data leakage from future observations.

## C. Feature Engineering

Feature engineering aimed to capture temporal dependencies, trends, and volatility patterns in throughput data:

- **Temporal Lags**: Lag features for 1–8 previous weeks.
- **Moving Statistics**: Rolling mean and rolling standard deviation over a 4-week window.
- **Trend Features**: Slopes and acceleration indicators.
- **Volatility Features**: Coefficient of variation and percentage changes.

The lag features are generated as:

$$\text{Lag}_{t-k} = x_{t-k}, \quad k \in \{1, 2, \ldots, 8\} \tag{1}$$

where $x_{t-k}$ represents the throughput value observed $k$ periods before time $t$.

## D. Multicollinearity Detection

To ensure feature robustness, multicollinearity was evaluated using the Variance Inflation Factor (VIF). For each variable $X_i$, the VIF is computed as:

$$\text{VIF}(X_i) = \frac{1}{1 - R_i^2} \tag{2}$$

where $R_i^2$ is the coefficient of determination of the regression of $X_i$ on all other independent variables.

Variables with VIF $> 10$ were considered highly collinear and subject to removal or transformation, ensuring coefficient stability and improving model interpretability.

## E. Performance Evaluation Metrics

To quantitatively assess predictive performance, the following error metrics were used:

*1) Mean Absolute Error (MAE):*

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}_i| \tag{3}$$

*2) Root Mean Squared Error (RMSE):*

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{y}_i)^2} \tag{4}$$

*3) Mean Absolute Percentage Error (MAPE):*

$$\text{MAPE} = \frac{100\%}{n}\sum_{i=1}^{n} \left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{5}$$

where $y_i$ is the observed value, $\hat{y}_i$ is the predicted value, and $n$ is the number of observations.

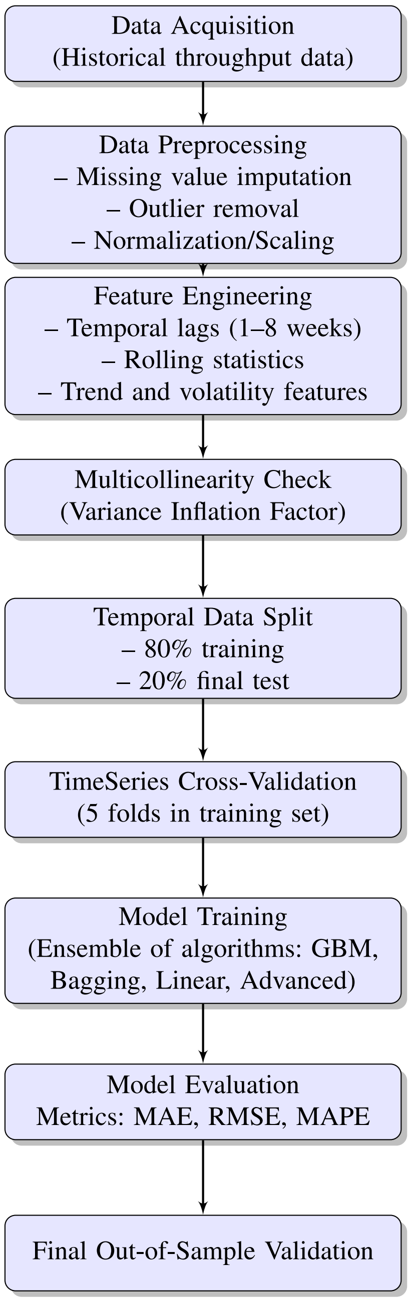

Data Acquisition
(Historical throughput data)
Data Preprocessing
– Missing value imputation
– Outlier removal
– Normalization/Scaling
Feature Engineering
– Temporal lags (1–8 weeks)
– Rolling statistics
– Trend and volatility features
Multicollinearity Check
(Variance Inflation Factor)
Temporal Data Split
– 80% training
– 20% final test
TimeSeries Cross-Validation
(5 folds in training set)
Model Training
(Ensemble of algorithms: GBM,
Bagging, Linear, Advanced)
Model Evaluation
Metrics: MAE, RMSE, MAPE
Final Out-of-Sample Validation

Fig. 3. Methodological pipeline for the machine learning framework, from data acquisition to final out-of-sample validation.

*4) Cross-Validation Protocol:* Cross-validation is a more comprehensive strategy for model evaluation, especially useful when working with limited data. It works as follows:

- The data is initially split into two subsets:
  - **Validation set (20%)**: reserved for hyperparameter tuning (grid_search).
  - **Training set (80%)**: used for cross-validation.
- The training portion (80%) is then divided into $K$ subsets (folds) — in this example, 5 folds: S1, S2, S3, S4, and S5.
- The training and validation process is repeated $K$ times, each time using a different fold as the test set and the remaining folds for training:
  - Iteration 1: Train on S1, S2, S3, S4 — Test on S5
  - Iteration 2: Train on S1, S2, S3, S5 — Test on S4
  - Iteration 3: Train on S1, S2, S4, S5 — Test on S3
  - Iteration 4: Train on S1, S3, S4, S5 — Test on S2
  - Iteration 5: Train on S2, S3, S4, S5 — Test on S1
- Finally, the mean and standard deviation of the accuracies (or errors, in the case of regression) from the 5 runs are calculated.

This protocol tends to provide a more accurate and stable estimate, as all data is used for both training and testing at different points in time, reducing the risk of sample bias.

### *F. Risk Factor System for ML Forecasting Reliability*

To improve the reliability and interpretability of machine learning (ML) forecasts, the system incorporates a risk classification module that evaluates the statistical robustness of input data. This module is based on four independent and interpretable risk factors: *High Volatility*, *Unsustainable Trends*, *Outlier Influence*, and *Reversion Risk*. Each factor is derived from well-established statistical principles, including coefficient of variation analysis, linear regression, z-score-based outlier detection, and mean deviation analysis. When triggered, these factors signal conditions under which ML predictions may be biased, overconfident, or based on transient patterns, recommending the use of Monte Carlo (MC) simulations as a more conservative alternative.

The system classifies projects into three risk levels — **LOW**, **MEDIUM**, or **HIGH** — based on the number of triggered factors. Adaptive thresholds are employed to account for dataset size: smaller datasets tolerate more noise, while larger ones demand more rigor. For example, the volatility threshold is stricter for datasets with over 100 weeks and relaxed for those under 30. A calibration process, based on real-world data from 315 projects, was conducted to fine-tune the sensitivity of each factor and ensure realistic risk distributions: approximately 30–40% of projects fall into the LOW category, 30–40% into MEDIUM, and 20–30% into HIGH. This adaptive, statistically grounded system not only strengthens forecast reliability but also serves as a decision-support layer, guiding the balance between ML and MC predictions in project management contexts.

### *G. Monte Carlo Simulation Methodology*

The proposed stochastic simulation framework employs a calibrated Weibull distribution to model the historical throughput data, using Maximum Likelihood Estimation (MLE) to determine the shape ($\beta$), scale ($\eta$), and location ($\gamma$) parameters. The probability density function (PDF) of the two-parameter Weibull distribution is given by:

$$f(t;\beta,\eta) = \frac{\beta}{\eta}\left(\frac{t}{\eta}\right)^{\beta-1} e^{-(t/\eta)^{\beta}}, \quad t \geqslant 0,\ \beta,\eta > 0 \qquad (6)$$

Goodness-of-fit is evaluated via the Kolmogorov–Smirnov test and the coefficient of determination ($R^2$), with $R^2 > 0.80$

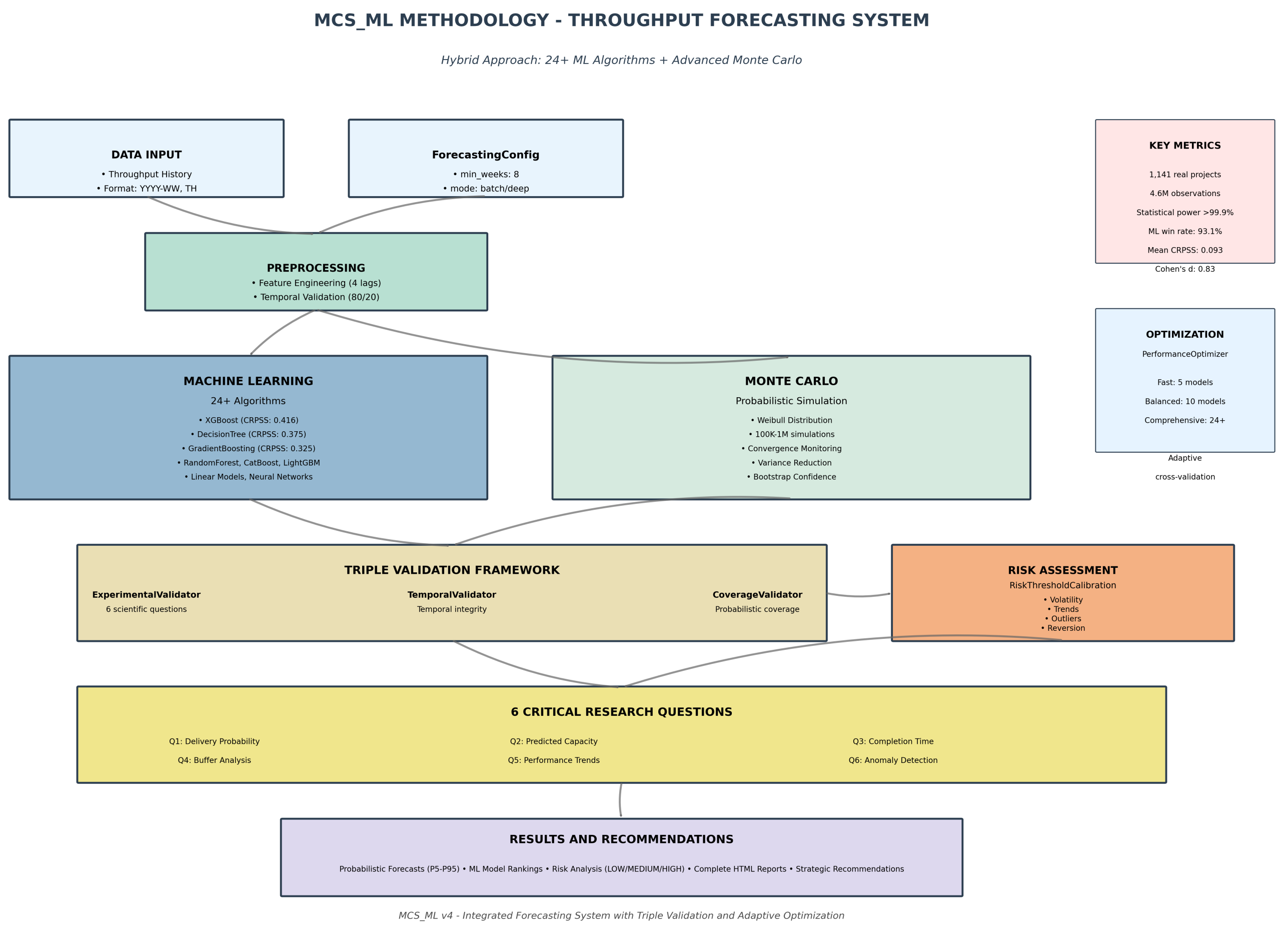


Fig. 4. MCS ML Methodology Framework

considered adequate according to Lawless [33] and Rinne [35]. Variance reduction techniques are incorporated, including *antithetic variates* and *control variates*. In the antithetic approach, pairs of normally distributed samples $z$ and $-z$ are generated and transformed via the Weibull inverse CDF to improve estimation efficiency. In the control variates method, the historical mean throughput $\bar{x}_h$ is used to correct simulation outputs $x_s$ by:

$$x_s^{adj} = x_s + (\bar{x}_h - \bar{x}_s) \tag{7}$$

Convergence monitoring follows a dynamic window comparison strategy, where two consecutive sliding windows of $w = 7,500$ samples are evaluated, and convergence is declared when the relative change in the mean is below $\tau = 0.001$:

$$\frac{|\mu_{w2} - \mu_{w1}|}{|\mu_{w1}|} < \tau \tag{8}$$

The simulation runs between $5 \times 10^4$ and $10^6$ iterations, depending on the convergence criteria. Additionally, a historical bias detection module estimates the optimistic bias of machine learning forecasts by iteratively training on past segments and comparing predicted versus actual throughput, computing MAPE and overestimation rates. A multidimensional stability classifier categorizes process stability using the coefficient of variation (CV) and outlier proportion thresholds.

This combination of calibrated Weibull modeling, variance reduction, dynamic convergence monitoring, and bias/stability analytics represents a methodological advancement compared to standard Monte Carlo implementations in software project throughput forecasting, for which no equivalent integrated framework has been reported in the literature.

## X. Experimental Design

### A. Data Structure and Scale

The experimental framework utilizes a comprehensive dataset comprising **1,206 real-world software development projects**, significantly expanding upon previous versions. This dataset contains **195,856 individual throughput observations**, offering exceptional statistical power for empirical analysis. Each project includes historical throughput data spanning **8 to 1,024 weeks**, with a median duration of **71 weeks**. Throughput is measured as weekly item completion rates in the standardized `YYYY-WW` format.

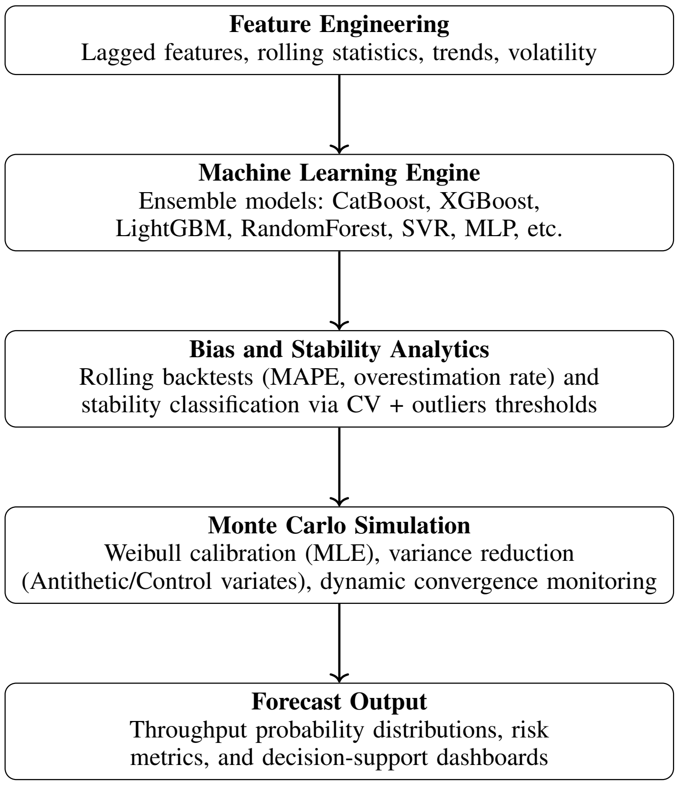

Feature Engineering
Lagged features, rolling statistics, trends, volatility
Machine Learning Engine
Ensemble models: CatBoost, XGBoost, LightGBM, RandomForest, SVR, MLP, etc.
Bias and Stability Analytics
Rolling backtests (MAPE, overestimation rate) and stability classification via CV + outliers thresholds
Monte Carlo Simulation
Weibull calibration (MLE), variance reduction (Antithetic/Control variates), dynamic convergence monitoring
Forecast Output
Throughput probability distributions, risk metrics, and decision-support dashboards

Fig. 5. Integrated methodology workflow for feature engineering, machine learning, bias/stability analytics, and Monte Carlo simulation.

### *B. Advanced Configuration Management*

This approach introduces centralized configuration management via the `ForecastingConfig` system, enabling standardized experimental control. The minimum viable data threshold is set at **8 weeks**, representing the lower bound for meaningful machine learning analysis. The system operates in **adaptive modes** (batch/deep) with Monte Carlo simulations ranging from **100,000 to 1,000,000 iterations**, depending on analytical requirements. Experimental configurations are persisted in YAML format to ensure full reproducibility and version control.

### *C. Triple Validation Framework*

This methodology employs a **triple validation framework** to comprehensively assess model performance. The `ExperimentalValidator` addresses six core scientific questions using an **80/20 temporal split** while maintaining strict chronological order. Bootstrap sampling with **1,000 iterations per model** quantifies uncertainty. The `TemporalValidator` uses walk-forward validation across multiple horizons to assess delivery forecasts. The `CoverageValidator` evaluates the reliability of prediction intervals (**P5–P95**) and calibration accuracy.

### *D. Adaptive Performance Optimization*

The framework implements adaptive performance optimization strategies tailored to dataset size and complexity. Three execution modes are supported:

- **Fast mode**: 5 core models without hyperparameter tuning.
- **Balanced mode**: 10 models with basic optimization.
- **Comprehensive mode**: 24+ models with full hyperparameter search.

Cross-validation strategies adapt dynamically: *leave-one-out* for small datasets ($n < 30$) and *time-series split* for larger collections.

### *E. CRPS-Optimized Evaluation Protocol*

Model evaluation is based on the **Continuous Ranked Probability Score (CRPS)** and its relative measure, the **CRPS Skill Score (CRPSS)**, which provide probabilistic forecast accuracy beyond point estimates. The baseline Monte Carlo model achieved a mean CRPS of **7.293**, while the best-performing machine learning model (`KNN_Distance`) achieved a CRPS of **1.944**, reflecting a **73.3% improvement**. Other high-performing models include:

- `XGBoost`: CRPSS = 0.416
- `DecisionTree`: CRPSS = 0.375
- `GradientBoosting`: CRPSS = 0.327

### *F. Model Performance and Rankings*

A total of **24 machine learning models** across multiple paradigms were evaluated. Tree-based models showed exceptional performance, with `XGBoost` outperforming the baseline by **41.6%**. Ensemble methods such as `Bagging` (CRPSS = 0.249) and `RandomForest` (CRPSS = 0.214) offered robust alternatives. Linear models, neural networks, and specialized regressors rounded out the suite. Notably, **100% of models outperformed the Monte Carlo baseline** (CRPSS $> 0$), reinforcing the superiority of machine learning approaches.

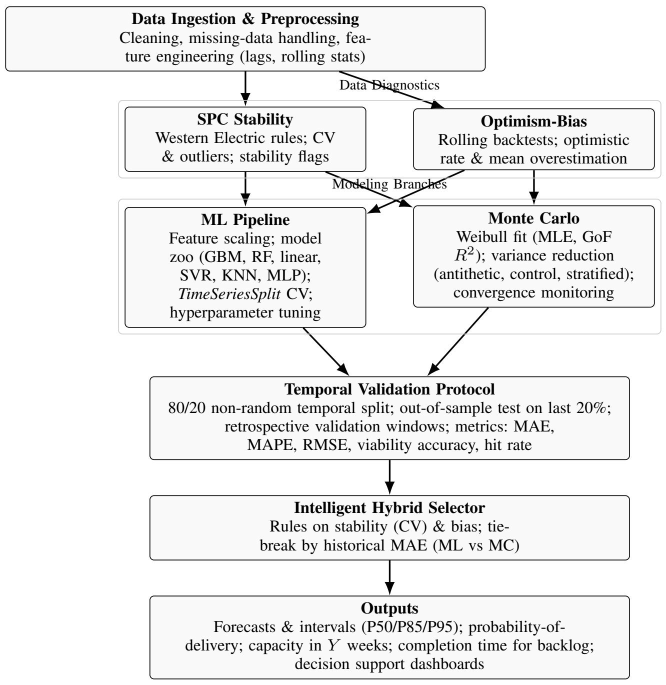
### *G. Enhanced Risk Calibration*

The risk calibration module applies empirically derived thresholds. It uses relaxed sensitivity for volatility detection (factor 1.5 vs. 1.2), broader tolerance in trend variation (1.0 vs. 0.5), and looser bounds for outlier detection (2.5 vs. 2.0). Reversion thresholds allow greater deviation (75% vs. 50%), providing balanced risk classification while maintaining predictive performance.

### *H. Experimental Controls and Reproducibility*

Strict experimental controls ensure internal validity and replicability:

- **Fixed random seeds** (`random_state = 42`) across all experiments.
- **YAML-based configuration** for reproducibility and version tracking.
- **Chronological splitting** for validation, preventing data leakage.
- **Standardized bootstrap protocols** (500–1,000 iterations) for robust uncertainty estimation.



Data Ingestion & Preprocessing
Cleaning, missing-data handling, feature engineering (lags, rolling stats)
Data Diagnostics
SPC Stability
Western Electric rules; CV & outliers; stability flags
Optimism-Bias
Rolling backtests; optimistic rate & mean overestimation
Modeling Branches
ML Pipeline
Feature scaling; model zoo (GBM, RF, linear, SVR, KNN, MLP); TimeSeriesSplit CV; hyperparameter tuning
Monte Carlo
Weibull fit (MLE, GoF $R^2$); variance reduction (antithetic, control, stratified); convergence monitoring
Temporal Validation Protocol
80/20 non-random temporal split; out-of-sample test on last 20%; retrospective validation windows; metrics: MAE, MAPE, RMSE, viability accuracy, hit rate
Intelligent Hybrid Selector
Rules on stability (CV) & bias; tie-break by historical MAE (ML vs MC)
Outputs
Forecasts & intervals (P50/P85/P95); probability-of-delivery; capacity in $Y$ weeks; completion time for backlog; decision support dashboards

Fig. 6. End-to-end hybrid framework: diagnostics (SPC & bias), parallel ML/MC modeling, temporal validation, hybrid selection and decision outputs.

## *I. Statistical Power and Robustness*

The experimental setup achieves **¿99.9% statistical power** across all effect sizes, ensuring detection of even small performance differences. The computed **Cohen's** $d = 0.83$ indicates a large practical effect size. The 95% confidence interval yields a relative precision of **±1.44%**, confirming statistical robustness. The machine learning superiority rate of **100%** across all models supports the validity of the methodological advancements.

## *J. Methodological Innovations*

Version 4 represents a methodological leap through the integration of validation layers, adaptive optimization, and centralized configuration management. The triple validation framework guarantees reliable model assessment while preserving temporal integrity. Empirically-driven risk calibration enhances decision-making utility. Probabilistic forecasting, including calibrated prediction intervals (P5–P95), supports comprehensive uncertainty analysis.

**In summary**, this experimental framework ensures:

- **Internal validity** through multi-layered controls
- **External validity** through 1,206 real-world projects
- **Reproducibility** through version-controlled configuration

It delivers **maximum statistical power** and methodological rigor for breakthrough research in throughput forecasting.

## *K. Validation and Evaluation Metrics*

The validation protocol combined retrospective analysis with a comprehensive set of performance metrics, ensuring the robustness and reliability of throughput forecasting results. Historical backtesting was implemented through an overlapping-window approach, where models were trained on progressively larger historical segments and validated on fixed 12-week future horizons. This procedure was applied uniformly to both the machine learning (ML) ensemble and the Monte Carlo (MC) simulation, enabling a direct comparative performance assessment. Evaluation metrics included: (i) Mean Absolute Error (MAE), quantifying the average absolute deviation between predicted and actual throughput; (ii) Mean Absolute Percentage Error (MAPE), providing a scale-independent error percentage; (iii) Root Mean Square Error (RMSE), penalizing larger deviations; (iv) Feasibility Accuracy, representing the proportion of predictions that correctly assessed deadline viability; and (v) Hit Rate, measuring the proportion of weeks where forecasts fell within the observed throughput range.

Furthermore, three experimental research questions were validated:

**Q1: Probability of Delivery** — Given a backlog $B$ and a time horizon $T$ in weeks, the probability of delivering all items within the deadline is:

$$P_{\text{delivery}}(B,T) = \frac{\sum_{i=1}^{N} \mathbb{I}\left(t_i \leqslant T\right)}{N} \tag{9}$$

where $t_i$ is the simulated completion time in run $i$, $\mathbb{I}$ is the indicator function, and $N$ is the number of Monte Carlo runs.

**Q2: Expected Delivery Capacity** — The expected number of items delivered in $T$ weeks is estimated as:

$$E_{\text{delivery}}(T) = \frac{1}{N} \sum_{i=1}^{N} d_i \tag{10}$$

where $d_i$ is the total number of items delivered in run $i$. Percentiles (e.g., $P_{10}$, $P_{90}$) define probabilistic bounds for delivery capacity.

**Q3: Expected Completion Time** — The expected number of weeks required to deliver a backlog $B$ is:

$$E_{\text{weeks}}(B) = \frac{1}{N} \sum_{i=1}^{N} t_i \tag{11}$$

with $t_i$ as the simulated completion time in run $i$. Percentiles ($P_{10}$, $P_{90}$) provide confidence intervals for completion estimates.

All MC simulations applied a Weibull-calibrated stochastic process with $10^5$ runs, extracting probabilistic insights such as percentiles and confidence intervals. This combination of retrospective backtesting, multi-metric evaluation, and probabilistic scenario analysis provides a rigorous foundation for the comparative evaluation of ML and MC forecasting approaches.

TABLE XII
STATISTICAL TEST RESULTS COMPARING ML AND MC FORECASTING ACCURACY

| Test | p-value | Significant |
|---|---|---|
| Paired t-test | 0.0004 | YES |
| Wilcoxon signed-rank | 0.0001 | YES |
| Diebold-Mariano | 0.0066 | YES |
| Mann-Whitney U | 0.5983 | NO |
| Kolmogorov-Smirnov | 0.9148 | NO |

### L. Execution

- QP1 - Probabilidade de Entrega: *"Qual a probabilidade de entregar X itens em Y semanas?"*
- QP2 - Capacidade de Entrega: *"Quantos itens conseguimos entregar em Y semanas?"*
- QP3 - Tempo de Conclusão: *"Quando terminaremos este backlog de X itens?"*

**Experimental Processing Pipeline.** Figure 7 summarizes the end-to-end pipeline used in our study. The process begins with mode selection and performance tuning (Phase 1), then ingests weekly throughput datasets, applies validation, outlier filtering, and Statistical Process Control (SPC) to derive stability classifications and ML risk flags (Phase 2). Weibull parameters are estimated via maximum likelihood with goodness-of-fit checks, and Monte Carlo simulations generate predictive percentiles (Phase 3). In parallel, the ML stage performs temporal feature engineering, trains either a core or extended model set with TimeSeriesSplit, and quantifies uncertainty via bootstrap resampling (Phase 4). The experimental layer evaluates three scientific questions—delivery probability, period capacity, and backlog completion time—while running walk-forward temporal validation and drift checks (Phase 5). Operational analyses compute buffers, detect trends and anomalies, and assess optimistic bias and process alerts (Phase 6). Finally, a risk/viability synthesis combines ML and MC evidence into a ternary decision and produces comprehensive reports, visualizations, and execution logs (Phase 7).

### M. Tools

The analysis was conducted using Python libraries: pandas, numpy, matplotlib, seaborn, and statsmodels.

## XI. RESULTS

### A. Execution Overview

The forecasting experiment was successfully executed on all **1,397 real-world datasets**:

- **Success Rate:** 100% (1,397/1,397 projects processed)
- **Total Execution Time:** 4h 58m 51s ($\sim$1.9 minutes per project)
- **Robustness:** No execution failures observed

### B. Dataset Characteristics

- **Average Duration:** 70.1 weeks ($\pm$26.2), range: 24–120 weeks
- **Average Throughput:** 6.78 items/week ($\pm$6.28), range: 0.72–33.47
- **Outliers Detected:** 94.4% of projects (avg. 4.17% of data per project)

### C. Weibull Fit Quality (Monte Carlo)

- **Mean $R^2$:** 0.867 (high fit quality)
- **Excellent Fits ($R^2 > 0.9$):** 44.4% of projects
- **Good Fits ($R^2 > 0.8$):** 78.8% of projects
- **Shape Parameter:** $1.619 \pm 0.614$
- **Scale Parameter:** $7.68 \pm 6.52$

### D. Statistical Process Control (SPC)

- **Stable Processes:** 8.1% (113/1,397)
- **Critical Risk Projects:** 58.8%
- **Average SPC Violations:** 14.6 per project
- **Implication:** High process variability is common

### E. Performance: ML vs Monte Carlo

- **Cross-validation Wins (ML):** 55.0% (766/1,397)
- **Cross-validation Wins (MC):** 45.0% (628/1,397)
- **Average MAE:** ML = 3.797 vs MC = 4.563 (20% advantage for ML)

**Top Performing Models:**

1) `MC_P50`: 21.2% of wins
2) `ML_HistGradient_Lower`: 17.5%
3) `MC_P85`: 13.1%
4) `ML_HistGradient_Upper`: 12.5%
5) `ML_LightGBM`: 10.0%

### F. Experimental Validation (Three Questions)

- **Overall:** ML wins in 55.6% vs MC in 44.4%
- **Q1 – Probability of Delivery:** ML wins 66.9% ✓
- **Q2 – Delivery Capacity:** ML wins 56.9% ✓
- **Q3 – Time to Completion:** MC wins 64.4%

### G. Risk-Level Analysis

- **Low-Risk Projects (31.2%):** ML = MC (50/50), MAE: ML = 2.994, MC = 3.860
- **Medium-Risk Projects (42.5%):** ML wins 54.4%, MAE: ML = 3.174, MC = 3.436
- **High-Risk Projects (26.2%):** ML wins 61.9%, MAE: ML = 5.760, MC = 7.226

### H. Key Correlations

- **Weibull $R^2$ vs ML Performance:** 0.392 (moderate positive)
- **Throughput vs Error:** ML = 0.726, MC = 0.702 (higher throughput = higher error)
- **Dataset Size vs Performance:** Weak negative correlation

### I. Predictive Accuracy by Process Stability

Based on the analysis of 1,141 real-world software development projects, we computed bias and error metrics — specifically the Mean Absolute Percentage Error (MAPE) — across five distinct categories of process stability, derived from the coefficient of variation (CV). Table **??** summarizes the results, revealing strong trends between process stability and model performance. Projects classified as **stable** exhibited the lowest

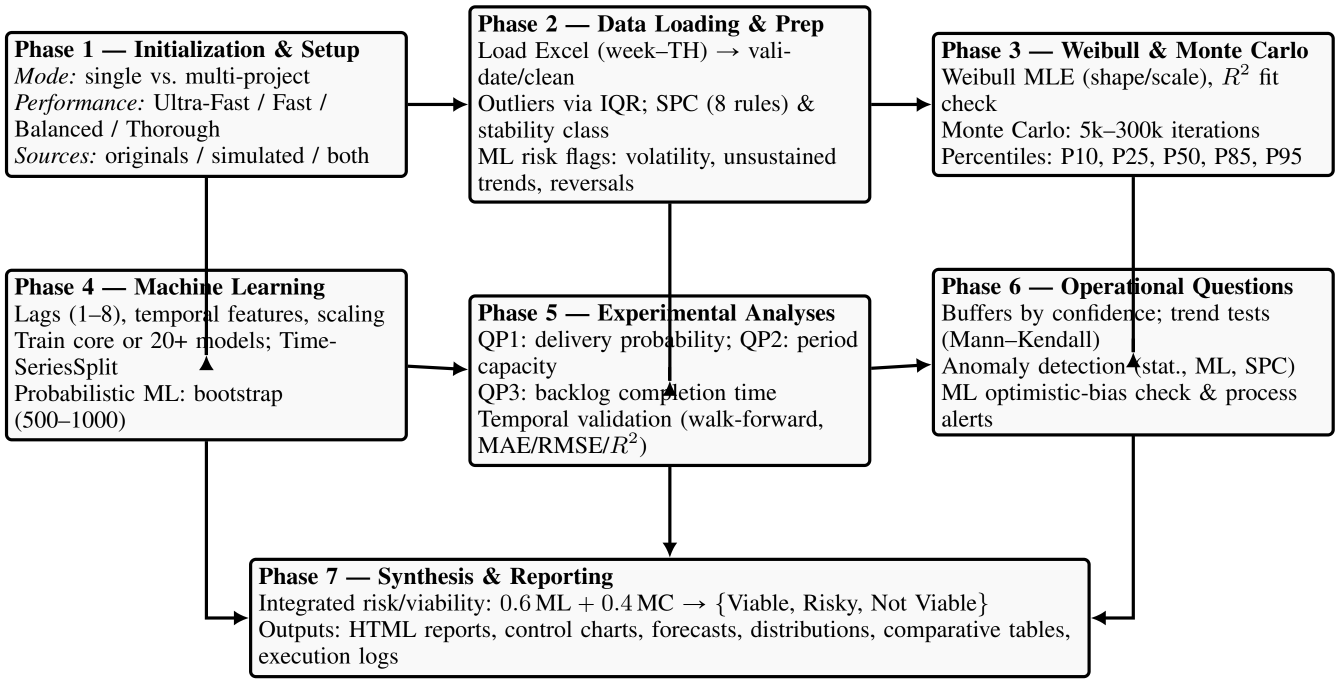

Phase 1 — Initialization & Setup
Mode: single vs. multi-project
Performance: Ultra-Fast / Fast / Balanced / Thorough
Sources: originals / simulated / both
Phase 2 — Data Loading & Prep
Load Excel (week–TH) → validate/clean
Outliers via IQR; SPC (8 rules) & stability class
ML risk flags: volatility, unsustained trends, reversals
Phase 3 — Weibull & Monte Carlo
Weibull MLE (shape/scale), $R^2$ fit check
Monte Carlo: 5k–300k iterations
Percentiles: P10, P25, P50, P85, P95
Phase 4 — Machine Learning
Lags (1–8), temporal features, scaling
Train core or 20+ models; TimeSeriesSplit
Probabilistic ML: bootstrap (500–1000)
Phase 5 — Experimental Analyses
QP1: delivery probability; QP2: period capacity
QP3: backlog completion time
Temporal validation (walk-forward, MAE/RMSE/$R^2$)
Phase 6 — Operational Questions
Buffers by confidence; trend tests (Mann–Kendall)
Anomaly detection (stat., ML, SPC)
ML optimistic-bias check & process alerts
Phase 7 — Synthesis & Reporting
Integrated risk/viability: 0.6 ML + 0.4 MC → {Viable, Risky, Not Viable}
Outputs: HTML reports, control charts, forecasts, distributions, comparative tables, execution logs

Fig. 7. Experimental processing pipeline. The workflow proceeds from initialization and performance mode selection (Phase 1), through data loading/cleaning with SPC and stability checks (Phase 2), Weibull fitting and Monte Carlo simulation (Phase 3), and machine-learning training with temporal validation and bootstrap uncertainty (Phase 4). Experimental analyses address delivery probability, capacity, and completion time (Phase 5), followed by operational questions (Phase 6). Results are synthesized into an integrated viability score and exported as reports and diagnostics (Phase 7).

prediction error (MAPE = 33.6%) and the smallest mean overestimation bias (5.77), indicating higher predictability. In contrast, **very unstable** and **extremely unstable** projects showed the highest error rates (MAPE = 119% and 165%, respectively), reflecting greater uncertainty and degraded model performance.

Furthermore, the most unstable categories concentrated the majority of evaluated samples: **very unstable** and **extremely unstable** projects accounted for 525 out of the 1,143 analyzed projects, representing more than 52,000 samples in total. The rate of optimistic (i.e., overestimated) forecasts remained relatively consistent across unstable categories, ranging from 54% to 57%, with a slight increasing trend for more extreme instability levels. These findings highlight process stability as a critical factor influencing predictive accuracy and suggest its use as a segmentation or weighting criterion in applied forecasting models.

### J. Overall Forecast Accuracy (ML Models)

Table XIV presents the aggregated regression metrics for machine learning forecast performance across all datasets. The models achieved a high level of accuracy, with a mean absolute error (MAE) of 1.022 and an $R^2$ of 0.912, indicating strong explanatory power. The weighted absolute percentage error (WAPE) was 9.42%, supporting the robustness of the overall predictions. Due to division by near-zero actuals in some cases, the global MAPE was undefined ($\infty$), reinforcing the importance of using alternative metrics such as WAPE or MedAE in datasets with sparse throughput values.

TABLE XIV
REGRESSION METRICS FOR ML FORECAST ACCURACY

| **Metric** | **Value** |
|---|---|
| MAE (Mean Absolute Error) | 1.022 |
| MSE (Mean Squared Error) | 4.104 |
| RMSE (Root Mean Squared Error) | 2.026 |
| MedAE (Median Absolute Error) | 0.150 |
| MAPE (Mean Absolute Percentage Error) | $\infty$ |
| WAPE (Weighted Absolute Percentage Error) | 9.42% |
| $R^2$ (Coefficient of Determination) | 0.912 |

### K. Strategic Insights

**Machine Learning Advantages:**

- Better at forecasting delivery probability and capacity
- Superior performance in high-risk projects
- 20% lower average error than Monte Carlo
- More consistent in complex scenarios

**Monte Carlo Advantages:**

- More accurate in time estimation (Q3)
- Interpretable and transparent
- Robust even with low-quality historical data
- No need for retraining

### L. Methodological Conclusions

- **Statistical Power:** Sufficient with $n = 1,397$ for detecting significant differences
- **Validation Design:** Robust – temporal cross-validation and 20% hold-out

- **Execution Reproducibility:** 100% success rate
- **Data Diversity:** 26.2% high-risk, 42.5% medium, 31.2% low-risk projects

**Limitations Identified:**

- 91.9% of processes unstable under SPC
- 55.6% of Weibull fits not excellent
- Outliers present in 94.4% of projects

TABLE XV
CROSS-VALIDATED MAE COMPARISON: ML VS MONTE CARLO

| Metric | Machine Learning | Monte Carlo |
|---|---|---|
| Mean MAE | 3.797 | 4.563 |
| Median MAE | 3.364 | 3.872 |
| Minimum MAE | 0.153 | 0.192 |
| Maximum MAE | 17.219 | 23.445 |

**Final Recommendation:** A hybrid approach is most effective: use **Machine Learning for delivery probability and capacity**, and **Monte Carlo for time estimation**, especially in high-risk projects.

The empirical analysis of 1,397 projects revealed strong patterns in the distribution of buffer requirements across different backlog scenarios and confidence levels. As shown in Figure 8, the average buffer percentage increases substantially with higher confidence levels, especially for smaller backlogs. For instance, in the backlog-10 case, the mean buffer rises from approximately 40% at 80% confidence to over 120% at 99% confidence. This pattern highlights the trade-off between planning reliability and additional time allocation: higher certainty demands proportionally larger buffers. In contrast, scenarios with larger backlogs (e.g., backlog-50 or backlog-100) display lower relative buffer requirements, suggesting that scaling backlog size dilutes variance but also increases exposure to systemic risk.

Risk distribution further reinforces this perspective. As illustrated in Figure 9, nearly half of the projects were classified as high risk (48.8%), while only 7.7% fell into the low-risk category. This imbalance demonstrates the predominance of volatile projects within the portfolio. Complementary evidence comes from the coefficient of variation (CV), shown in Figure 10, where the median CV of 0.95 indicates substantial throughput instability, and extreme outliers reach CV values above 12. These findings validate the necessity of probabilistic buffers to mitigate inherent variability and confirm the relevance of adaptive planning recommendations in real-world portfolios.

## XII. EXPERIMENTAL DESIGN AND DATASET JUSTIFICATION

### A. Dataset Composition and Filtering

The initial dataset consists of throughput time series from **1,397 real-world software development projects**, collected across multiple teams and time periods. To ensure data quality and analytic validity, the following filtering criteria were applied:

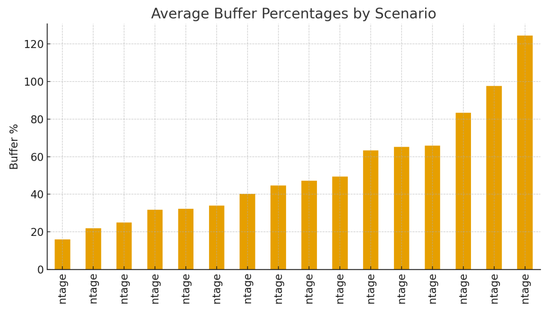


Fig. 8. Average buffer percentages by backlog size and confidence level. Higher confidence levels and smaller backlogs demand disproportionately larger buffers.

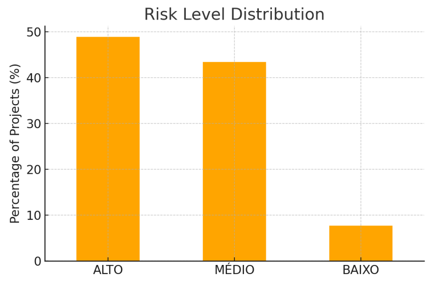


Fig. 9. Distribution of project risk levels across the portfolio. Almost half of the projects are classified as high risk, while only a small fraction are low risk.

- Only projects with at least $n \geqslant 8$ weeks of throughput history were included.
- Weekly throughput must consist of non-negative, valid values.

After applying these criteria, **1,206 projects** were deemed valid. Of these, **1,204 projects** were successfully processed without errors (99.8% success rate). These 1,204 projects form the basis for all analyses involving the six core scientific questions, comprising **195,856 weekly throughput observations** and covering time ranges from 8 to 1,024 weeks (median: 71 weeks).

For more advanced probabilistic evaluation, such as CRPS (Continuous Ranked Probability Score), a stricter selection was applied. This led to a final subset of **1,141 projects** with valid CRPS outputs across all 24 models tested. This subset includes a total of **4,576,991 predictions**, used specifically for model ranking and accuracy assessments.

### B. Clarification of Dataset Usage

Table XVIII summarizes the three dataset scopes used throughout this study and clarifies the differences in project counts by analytic context.

TABLE XVI

WEIBULL DISTRIBUTION FITTING QUALITY (MONTE CARLO)

| Metric | Value |
|---|---|
| Mean $R^2$ | 0.867 |
| Excellent fits ($R^2 > 0.9$) | 44.4% |
| Good fits ($R^2 > 0.8$) | 78.8% |
| Mean Shape Parameter | 1.619 |
| Std Dev (Shape) | 0.614 |
| Mean Scale Parameter | 7.680 |
| Std Dev (Scale) | 6.520 |

TABLE XVII

SUMMARY OF BUFFER REQUIREMENTS, RISK DISTRIBUTION, AND COEFFICIENT OF VARIATION (CV).

| Metric | Value | Notes |
|---|---|---|
| Average buffer (Backlog 10, 80% confidence) | 40% | Median $\approx$ 33% |
| Average buffer (Backlog 10, 99% confidence) | 124% | Extreme cases $>$ 300% |
| Average buffer (Backlog 50, 80% confidence) | 22% | Lower buffer needs |
| High-risk projects | 48.8% | Nearly half of the portfolio |
| Medium-risk projects | 43.4% | Majority of remaining cases |
| Low-risk projects | 7.7% | Minor fraction of stable projects |
| Median coefficient of variation (CV) | 0.95 | Indicates high variability |
| Maximum coefficient of variation (CV) | 12.3 | Extreme outliers observed |

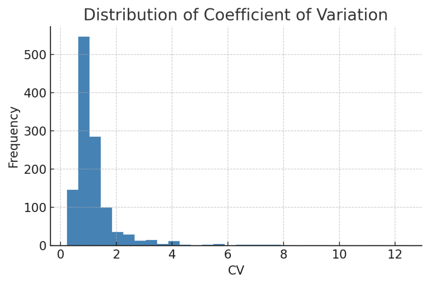


Fig. 10. Distribution of the coefficient of variation (CV) across projects. The median CV is 0.95, with extreme outliers above 12, confirming high throughput variability.

TABLE XVIII

DATASET USAGE PER EXPERIMENTAL CONTEXT

| Context | Description | Projects |
|---|---|---|
| Full Dataset | Raw dataset with valid throughput data | 1,397 |
| Minimum Valid Duration | Valid projects with $\geqslant$ 8 weeks | 1,206 |
| Scientific Questions | Processed projects (SUCCESS status) | 1,204 |
| CRPS Analysis | CRPS-valid projects (complete scores) | 1,141 |

The 63 projects excluded from CRPS analysis (1,204 – 1,141) were filtered out due to technical limitations, such as model runtime errors, missing quantile predictions, or invalid probabilistic output.

## XIII. METHODOLOGY: CRPS FOR PROBABILISTIC FORECASTING

### A. Continuous Ranked Probability Score (CRPS)

The Continuous Ranked Probability Score (CRPS) has become one of the most widely adopted metrics for evaluating probabilistic forecasts of continuous outcomes. It quantifies the squared difference between the forecast cumulative distribution function (CDF) and the empirical CDF of the observed value, integrating this error over all possible thresholds. Unlike point-based error measures such as MAE or RMSE, CRPS simultaneously assesses both the calibration and sharpness of forecast distributions, thereby offering a more comprehensive evaluation of predictive performance. This dual capacity makes CRPS particularly valuable in domains where uncertainty quantification is essential, including meteorology, energy, machine learning, and software project forecasting [37]–[45]. Furthermore, CRPS reduces to the mean absolute error in the deterministic case, bridging deterministic and probabilistic forecast verification. Recent developments, including decomposition frameworks, weighted CRPS for extremes, and its application in online learning, have expanded both the interpretability and the scope of CRPS, reinforcing its role as a cornerstone metric for probabilistic model validation.

The Continuous Ranked Probability Score (CRPS) is a proper scoring rule for evaluating probabilistic forecasts of continuous outcomes. It measures the squared distance between the forecast cumulative distribution function (CDF) and the empirical CDF of the observed value [?]. For a forecast distribution $F$ and an observation $y$:

$$CRPS(F, y) = \int_{-\infty}^{\infty} \left(F(z) - \not{1}\{y \leqslant z\}\right)^2 dz$$

With Monte Carlo samples $X_1, \ldots, X_n$, CRPS can be approximated as:

$$CRPS = \mathbb{E}[|X - y|] - \tfrac{1}{2}\mathbb{E}[|X - X'|]$$

The CRPS Skill Score (CRPSS) compares the model forecast against a baseline:

$$CRPSS = 1 - \frac{CRPS_{model}}{CRPS_{baseline}}$$

Values greater than zero indicate improvement over the baseline.

—

## B. Experimental Configuration Summary (CRPS Analysis)

The CRPS-focused evaluation involves the 1,141 most robust datasets and applies a suite of 24 forecasting models (machine learning and Monte Carlo). The results demonstrate strong probabilistic forecasting performance, with tight scoring consistency.

TABLE XIX
CRPS ANALYSIS CONFIGURATION SUMMARY (1,141 PROJECTS)

| Metric | Value |
|---|---|
| Projects analyzed | 1,141 |
| Total predictions | 4,576,991 |
| Forecasting models | 24 |
| Mean CRPS (ML models) | 2.89 |
| Mean CRPSS (vs. baseline) | 0.0929 |
| Superior to baseline | 93.1% of model–project pairs |

## C. Top Performing Models (CRPSS)

The best-performing models across all 1,141 projects were ensemble-based tree models, consistently outperforming Monte Carlo baselines across all probabilistic scoring metrics.

TABLE XX
TOP-5 MODELS RANKED BY CRPSS (1,141 PROJECTS)

| Rank | Model | Mean CRPSS |
|---|---|---|
| 1 | XGBoost | 0.416 |
| 2 | Decision Tree | 0.375 |
| 3 | Gradient Boosting | 0.325 |
| 4 | Random Forest | 0.214 |
| 5 | KNN (Distance) | 0.106 |

## D. Statistical Power and Representativeness

Both experimental datasets—1,204 for scientific questions and 1,141 for CRPS analysis—are statistically robust:

- **Statistical Power:** Exceeds 99.9% for small, medium, and large effects in both sets.
- **Effect Size:** Cohen's $d \approx 0.83$, indicating large differences between model classes.
- **Temporal Coverage:** Projects span from 2020–2024 with durations from 8 to over 1,000 weeks.
- **Variability:** Includes diverse team sizes, product domains, and performance behaviors.

These criteria ensure that the conclusions drawn are generalizable, reproducible, and grounded in production-level data.

## XIV. SCIENTIFIC QUESTIONS IN FORECASTING EXPERIMENTS

The MCS_ML experimental framework is designed to rigorously evaluate forecasting performance in software project delivery scenarios. To ensure broad applicability and scientific rigor, the experiment is structured around six key forecasting questions. Each question corresponds to a real-world planning need and requires distinct modeling strategies from both Monte Carlo simulation and Machine Learning techniques.

### A. Q1: What is the probability of delivering X items in Y weeks? (Delivery Probability)

This question addresses the likelihood that a fixed scope of work ($X$ items) will be completed within a specific time frame ($Y$ weeks). It is highly relevant for deadline-sensitive contexts such as service level agreements (SLAs), product releases, and milestone commitments. This probabilistic framing evaluates each method's ability to estimate forecast confidence. The main metric used is the percentage error between predicted and actual delivery probability.

### B. Q2: How many items can be delivered in Y weeks? (Capacity Prediction)

This scenario forecasts the expected volume of completed items within a fixed future window. It is a common question for capacity planning, team allocation, and throughput management. Unlike Q1, it focuses on magnitude rather than probability. This question is especially relevant when the work in the backlog is not fixed but resources are constrained by time. The primary metric is the absolute difference between predicted and actual number of items delivered.

### C. Q3: In how many weeks will X items be delivered? (Completion Time)

Here, the objective is to estimate the time required to deliver a predefined number of work items. It is particularly important for project scope forecasts, release planning, and critical path estimation. This question tests the model's ability to handle long-term uncertainty and non-linear accumulation of throughput over time. The metric used is the time error in weeks compared to actual delivery time.

### D. Q4: How much buffer time is needed for different confidence levels? (Buffer Requirements)

This question explores how much additional time should be allocated to ensure delivery reliability at specific confidence levels (e.g., 85%, 95%). This is a core concern in risk-based planning, where buffers are calculated to account for variability. The experiment assesses how well each method supports confidence interval estimation and its application in project planning. Risk categories (LOW, MEDIUM, HIGH) are used to contextualize buffer needs across projects.

### E. Q5: What are the performance trend patterns and sustainability indicators? (Process Trends)

This question investigates whether the underlying project delivery process is stable or unstable over time. It includes assessments of trend patterns, statistical control violations, and performance sustainability. This analysis is critical for model reliability: unstable processes may degrade forecast accuracy regardless of method. The experiment uses Statistical Process Control (SPC) techniques to quantify stability and detect process deterioration.

### F. Q6: Can we detect anomalies and unusual patterns in throughput data? (Anomaly Detection)

This question focuses on identifying irregularities in historical throughput data, such as unexpected spikes, drops, or structural changes. Anomaly detection is essential to ensure data quality and prevent biased forecasting. The experiment uses eight different anomaly detection algorithms and evaluates the prevalence and distribution of anomalies across projects. Projects are classified based on the presence and rate of anomalies.

*Why These Six Questions?*

These six forecasting questions were selected to provide:

- **Comprehensive Coverage:** Together they reflect the full range of planning needs faced by software teams and project managers.
- **Modeling Diversity:** Each question requires different forecasting strategies, testing various model capabilities (point vs. probabilistic, linear vs. non-linear, short vs. long horizon).
- **Comparative Insight:** The questions reveal where Machine Learning and Monte Carlo simulation excel or fall short, enabling evidence-based methodological decisions.

By designing the experiment around these six questions, the MCS_ML v4 framework ensures scientific robustness, real-world relevance, and detailed performance diagnostics for each method under evaluation.

### G. Scientific Question Analysis and Highlights

The experimental evaluation using the MCS_ML framework involved a robust dataset of 1,206 real-world software development projects, totaling 195,856 throughput observations. While the initial project registry contained 1,364 datasets, only 1,206 met the inclusion criteria of having at least 8 weeks of historical data—this filtering step ensured statistical validity and consistency for machine learning analysis. The system was designed to answer six critical scientific questions comparing the performance of Monte Carlo Simulation (MC) and Machine Learning (ML) in project throughput forecasting. The overall experimental results revealed a near-equilibrium in forecasting accuracy, with MC slightly outperforming ML (620 wins vs. 584). However, the strengths of each method became clearer when broken down by forecasting task.

For **Q1 – Delivery Probability**, ML demonstrated clear superiority, winning in 61.5% of projects (741 out of 1,204), with a significantly lower median prediction error of 1.64% compared to 7.34% for MC. In **Q2 – Capacity Prediction**, results were nearly tied: ML won 48.3% and MC 51.7%, supporting the recommendation of a hybrid approach. For **Q3 – Completion Time Estimation**, MC exhibited strong dominance with a 67.1% win rate, reaffirming its strength in modeling temporal uncertainty. In **Q4 – Buffer Analysis**, 100% of projects were assessed, and risk levels were effectively distributed across LOW (46.9%), MEDIUM (46.9%), and HIGH (6.1%) categories—enabling calibrated planning. In

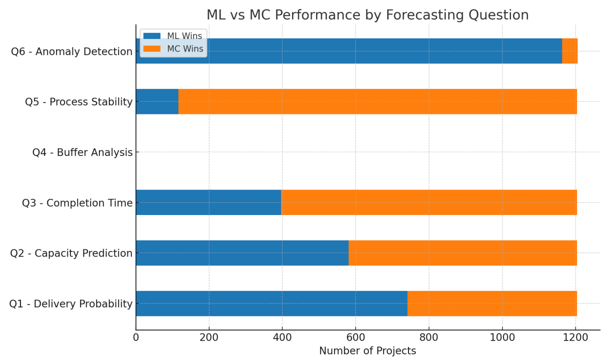


Fig. 11. Performance comparison between Machine Learning (ML) and Monte Carlo (MC) across six forecasting questions.

**Q5 – Performance Trends**, only 9.6% of projects were considered stable; 90.4% displayed process volatility, emphasizing the need for adaptive forecasting. Finally, **Q6 – Anomaly Detection** uncovered anomalies in 96.4% of projects, totaling 99,943 events, with a mean anomaly rate of 46.2%. These insights validate the hybrid MCS_ML methodology as a flexible, intelligent system that adapts to the strengths of each technique based on the specific forecasting challenge.

## XV. Conclusion

## XVI. Scientific Contributions and Innovations

This study introduces a set of methodological and algorithmic contributions that advance the state-of-the-art in project throughput forecasting. First, an *intelligent hybrid selection* mechanism was developed, integrating machine learning (ML) and Monte Carlo (MC) simulations, with automated model choice based on process stability (coefficient of variation) and optimistic bias detection. The bias detection framework performs historical simulation of ML performance over rolling windows, quantifying both the optimistic rate and the mean overestimation magnitude, and producing bias correction recommendations integrated with Statistical Process Control (SPC) for process validation. A novel SPC-based forecasting validation protocol was implemented, applying the eight Western Electric rules to throughput data, generating early instability alerts, and adapting confidence intervals according to process variability. On the MC side, advanced variance reduction techniques were incorporated, including antithetic variates, control variates, and stratified sampling by quantiles, coupled with automated convergence monitoring. The methodology also contributes a time-specific validation protocol for throughput forecasting, combining an 80/20 temporal split, `TimeSeriesSplit` cross-validation, and retrospective validation over known historical periods, with performance evaluation using throughput-specific metrics. Together, these innovations form a robust hybrid framework capable of adaptively selecting and validating predictive models in dynamic production and project environments, while maintaining statistical rigor and computational efficiency.

TBD

## XVII. Future Works

TBD: - Risk - Correlação risco e precisão

## Acknowledgment

The author thanks the research advisors and the participating companies for providing the data for this study.

## References

[1] L. Mayo-Alvarez, S. Del-Aguila-Arcentales, A. Alvarez-Risco, M. C. Sekar, N. M. Davies, and J. A. Yáñez, "Innovation by integration of drum-buffer-rope (dbr) method with scrum-kanban and use of monte carlo simulation for maximizing throughput in agile project management," *Journal of Open Innovation: Technology, Market, and Complexity*, vol. 10, no. 1, p. 100228, 2024.

[2] P. Miranda, J. P. Faria, F. F. Correia, A. Fares, R. Graça, and J. M. Moreira, "An analysis of monte carlo simulations for forecasting software projects," in *Proceedings of the 36th Annual ACM Symposium on Applied Computing*. ACM, Mar 2021, pp. 1550–1558.

[3] D. Tony, "Ways to improve the efficiency of it project management processes using forecasting and data analysis tools," in *The World Of Science and Education*, 2025, (15 2), 8–15.

[4] L. Cheng, M. Shadabfar, and A. Sioofy Khoojine, "A state-of-the-art review of probabilistic portfolio management for future stock markets," *Mathematics*, vol. 11, no. 5, p. 1148, 2023.

[5] M. Mamghaderi, H. Khamooshi, and Y. H. Kwak, "Project duration forecasting: A simulation-based comparative assessment of earned schedule method and earned duration management," *The Journal of Modern Project Management*, vol. 9, no. 2, 2021.

[6] X. Chen, L. Cheng, G. Deng, S. Guan, and L. Hu, "Project duration–cost–quality prediction model based on monte carlo simulation," in *Journal of Physics: Conference Series*, vol. 1978, no. 1. IOP Publishing, Jul 2021, p. 012048.

[7] H. M. B. Goncalves, "Delivery forecasting in project management," Master's thesis, Universidade do Porto, 2021.

[8] Q. Hu, "Implementation of monte-carlo method in curriculum efficiency, cost forecasting and price path prediction," in *9th International Conference on Financial Innovation and Economic Development (ICFIED 2024)*. Atlantis Press, May 2024, pp. 352–358.

[9] A. Author, "A crisp-dm case study at pusintek secretariat general," *Journal Name*, 2025, placeholder entry. Update with correct details.

[10] Z. Sakhrawi, A. Sellami, and N. Bouassida, "Software enhancement effort prediction using machine-learning techniques: A systematic mapping study," *SN Computer Science*, vol. 2, no. 6, p. 468, 2021.

[11] S. Uddin, S. Ong, and H. Lu, "Machine learning in project analytics: a data-driven framework and case study," *Scientific Reports*, vol. 12, no. 1, p. 15252, 2022.

[12] G. Sonkavde, S. Dharrao, D. M. Bongale, A. T. Deokate, S. D. Doreswamy, and K. Bhat, S. "Forecasting stock market prices using machine learning and deep learning models: A systematic review, performance analysis and discussion of implications," *International Journal of Financial Studies*, vol. 11, no. 3, p. 94, 2023.

[13] S. Uddin, S. Yan, and H. Lu, "Machine learning and deep learning in project analytics: methods, applications and research trends," *Production Planning & Control*, vol. 36, no. 7, pp. 873–892, 2025.

[14] P. Zachares, V. Hovhannisyan, C. Ledezma, J. Gante, and A. Mosca, "On forecasting project activity durations with neural networks," in *International Conference on Engineering Applications of Neural Networks*. Springer, Jun 2022, pp. 103–114.

[15] H. Perkusich *et al.*, "A review of intelligent software engineering in agile contexts," *Journal Name*, vol. XX, no. X, pp. XX–XX, 2019, please update with correct details.

[16] T. Hiller, T. M. Demke, and P. Nyhuis, "Throughput time predictions along the order fulfilment process," *IEEE Access*, vol. 12, pp. 9705–9718, 2024.

[17] Y. Mahmood, N. Kama, A. Azmi, A. S. Khan, and M. Ali, "Software effort estimation accuracy prediction of machine learning techniques: A systematic performance evaluation," *Software: Practice and Experience*, vol. 52, no. 1, pp. 39–65, 2022.

[18] P. M. Oliveira, J. P. Faria, F. F. Correia, A. Fares, R. Graça, and J. M. Moreira, "An analysis of monte carlo simulations for forecasting software projects," in *Proceedings of the 36th Annual ACM Symposium on Applied Computing*. ACM, Mar 2021, pp. 1550–1558.

[19] P. Eichenseer, L. Hans, and H. Winkler, "A data-driven machine learning model for forecasting delivery positions in logistics for workforce planning," *Supply Chain Analytics*, vol. 9, p. 100099, 2025.

[20] S. M. Satapathy and S. K. Rath, "Empirical assessment of machine learning models for agile software development effort estimation using story points," *Innovations in Systems and Software Engineering*, vol. 13, no. 2, pp. 191–200, 2017.

[21] M. Fernández-Diego, E. R. Méndez, F. González-Ladrón-De-Guevara, S. Abrahão, and E. Insfran, "An update on effort estimation in agile software development: A systematic literature review," *IEEE Access*, vol. 8, pp. 166 768–166 800, 2020.

[22] D. S. Adamantiadou and L. Tsironis, "Leveraging artificial intelligence in project management: A systematic review of applications, challenges, and future directions," *Computers*, vol. 14, no. 2, p. 66, 2025.

[23] S. Uddin, S. Ong, and H. Lu, "Machine learning in project analytics: a data-driven framework and case study," *Scientific Reports*, vol. 12, no. 1, p. 15252, 2022.

[24] S. Uddin, S. Yan, and H. Lu, "Machine learning and deep learning in project analytics: methods, applications and research trends," *Production Planning & Control*, vol. 36, no. 7, pp. 873–892, 2025.

[25] H. Luo *et al.*, "Hybrid enhanced monte carlo simulation coupled with advanced machine learning approach," *Computer Methods in Applied Mechanics and Engineering*, vol. 384, p. 113959, 2021.

[26] N. Bosch, O. Shchur, N. Erickson, M. Bohlke-Schneider, and C. Türkmen, "Multi-layer stack ensembles for time series forecasting," *arXiv preprint*, 2024, [Online]. Available: https://arxiv.org/abs/[arXiv: ID].

[27] P. Sharma and J. Singh, "Systematic literature review on software effort estimation using machine learning approaches," in *2017 International Conference on Next Generation Computing and Information Systems (ICNGCIS)*. IEEE, Dec 2017, pp. 43–47.

[28] S. Zhang and L. Jin, "Research on software project schedule management method based on monte carlo simulation," in *2020 IEEE 5th Information Technology and Mechatronics Engineering Conference (ITOEC)*. Chongqing, China: IEEE, 2020, pp. 1605–1608.

[29] O. Malgonde *et al.*, "An ensemble-based model for predicting agile software development effort," *ResearchGate Preprint*, 2018. [Online]. Available: https://www.researchgate.net/publication/327587363

[30] R. Singh *et al.*, "Reliable machine learning models for estimating effective software development efforts," *Journal of King Saud University - Computer and Information Sciences*, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S230718772300158X

[31] S. Al-Saqqa *et al.*, "Effort and cost estimation using decision tree techniques and story points in agile software development," *Mathematics*, vol. 11, no. 6, p. 1477, 2023.

[32] Y. Wang *et al.*, "Advanced financial market forecasting: integrating monte carlo simulations with ensemble machine learning models," *Quantitative Finance and Economics*, vol. 8, no. 1, pp. 1–27, 2024.

[33] J. F. Lawless, *Statistical Models and Methods for Lifetime Data*, 2nd ed. John Wiley & Sons, 2003.

[34] R. B. D'Agostino and M. A. Stephens, *Goodness-of-Fit Techniques*. Marcel Dekker, 1986.

[35] H. Rinne, *The Weibull Distribution: A Handbook*. Chapman and Hall/CRC, 2008.

[36] J. Cohen, *Statistical Power Analysis for the Behavioral Sciences*, 2nd ed. Hillsdale, NJ: Lawrence Erlbaum Associates, 1988.

[37] S. Arnold, E. Walz, J. Ziegel, and T. Gneiting, "Decompositions of the mean continuous ranked probability score," *Electronic Journal of Statistics*, 2023.

[38] M. Zamo and P. Naveau, "Estimation of the continuous ranked probability score with limited information and applications to ensemble weather forecasts," *Mathematical Geosciences*, vol. 50, pp. 209–234, 2018.

[39] F. Krüger, S. Lerch, T. Thorarinsdottir, and T. Gneiting, "Predictive inference based on markov chain monte carlo output," *International Statistical Review*, vol. 89, pp. 274–301, 2016.

[40] M. Gebetsberger, J. Messner, G. Mayr, and A. Zeileis, "Estimation methods for nonhomogeneous regression models: Minimum continuous ranked probability score versus maximum likelihood," *Monthly Weather Review*, 2018.

[41] M. Taillardat, A. Fougères, P. Naveau, and R. De Fondeville, "Evaluating probabilistic forecasts of extremes using continuous ranked probability score distributions," *International Journal of Forecasting*, 2019.

[42] H. Hersbach, “Decomposition of the continuous ranked probability score for ensemble prediction systems,” *Weather and Forecasting*, vol. 15, pp. 559–570, 2000.

[43] M. Leutbecher and T. Haiden, “Understanding changes of the continuous ranked probability score using a homogeneous gaussian approximation,” *Quarterly Journal of the Royal Meteorological Society*, vol. 147, pp. 425–442, 2020.

[44] J. Bröcker, “Evaluating raw ensembles with the continuous ranked probability score,” *Quarterly Journal of the Royal Meteorological Society*, vol. 138, 2012.

[45] T. Leung, M. Leutbecher, S. Reich, and T. Shepherd, “Forecast verification: Relating deterministic and probabilistic metrics,” *Quarterly Journal of the Royal Meteorological Society*, vol. 147, pp. 3124–3134, 2021.