Bangladesh Army University of Science and Technology (BAUST), Saidpur

Department of Computer Science and Engineering (CSE)

CSE 4140: Machine Learning Sessional

CPU Performance Prediction Using Machine Learning

A Comprehensive Comparative Analysis

Submitted By:

Nahid Hasan Arif
220201075

Submitted To:

Engr. Rohul Amin, Lecturer
Nadim Reza, Lecturer

Date: November 25, 2025

CPU Performance Prediction Using Machine Learning: A Comprehensive Comparative Analysis

Nahid Hasan Arif
Department of Computer Science and Engineering
Bangladesh Army University of Science and Technology
Saidpur, Bangladesh
md.nahidhasanarif@gmail.com

***Abstract*—Accurate prediction of CPU performance is essential for system designers, manufacturers, and consumers. This paper presents a comprehensive machine learning pipeline for predicting CPU performance scores from hardware specifications. We evaluate six regression algorithms—Linear Regression, Decision Tree, Random Forest, Gradient Boosting, XGBoost, and LightGBM—on a dataset of 96 CPUs with 27 hardware features. Our experimental results demonstrate that Linear Regression achieves the highest test R^2 = 0.9618 with RMSE = 1814.75, while Random Forest offers the best balance with R^2 = 0.9273 and minimal overfitting (0.0668). We provide detailed analysis of learning curves, residual plots, error distributions, and feature importance, revealing that clock frequencies and core counts are the primary performance determinants. Cross-validation experiments confirm model robustness and generalization capability. The study includes comprehensive preprocessing techniques handling missing data, feature scaling, and categorical encoding, making it applicable to real-world scenarios.**

***Index Terms*—CPU performance prediction, machine learning, regression analysis, LightGBM, XGBoost, hardware benchmarking, feature importance, learning curves**

I. Introduction

Central Processing Unit (CPU) performance prediction has become increasingly important in the era of diverse processor architectures and rapid technological advancement. Traditional performance evaluation through physical benchmarking is resource-intensive and time-consuming, particularly when comparing numerous CPU models across different generations and manufacturers [1]. Machine learning approaches offer an efficient alternative by learning patterns from historical performance data to predict scores based on hardware specifications.

Modern CPU architectures exhibit significant complexity with heterogeneous core designs (performance and efficiency cores), multi-level cache hierarchies, dynamic frequency scaling, and advanced instruction sets. This complexity makes performance prediction challenging, as interactions between hardware components create non-linear relationships that simple analytical models cannot capture [3].

A. Significance and Motivation

The need for accurate CPU performance prediction stems from several critical requirements:

a) **Design Optimization:** CPU architects need to understand which hardware parameters most significantly impact performance to make informed design decisions during the development cycle.

b) **Consumer Decision Support:** Buyers require reliable performance estimates when comparing processors across different price points, use cases, and generations without access to comprehensive benchmark data.

c) **Benchmark Reduction:** Accurate ML models can reduce the need for exhaustive physical testing, accelerating development cycles and reducing costs associated with prototype fabrication.

d) **Performance Forecasting:** Predicting performance of unreleased processors based on announced specifications enables early system planning for data centers and enterprise deployments.

B. Problem Statement

Despite extensive research in hardware performance modeling, several challenges persist:

i) **Non-linear Interactions:** Hardware features interact in complex, non-linear ways that traditional analytical models struggle to capture.

ii) **Missing Data:** Real-world datasets often contain missing specifications, particularly for older or specialized processors.

iii) **Heterogeneous Architectures:** Modern processors combine performance and efficiency cores with varying clock speeds, complicating traditional modeling approaches.

iv) **Generalization:** Models must generalize across different CPU families, generations, and manufacturers without overfitting to training data.

C. Proposed Solution

This research addresses these challenges through a comprehensive machine learning pipeline that:

a) Implements robust preprocessing including KNN-based missing value imputation

b) Evaluates six diverse regression algorithms to identify optimal performance-accuracy tradeoffs
c) Provides detailed learning curve analysis to assess generalization capability
d) Analyzes residuals and error distributions to validate model assumptions
e) Identifies key hardware features driving performance through importance analysis

D. Research Contributions

This paper makes the following contributions to CPU performance prediction:

a) Comprehensive evaluation of six machine learning algorithms with detailed performance metrics including training vs. test analysis, learning curves, and residual analysis.
b) Rigorous data preprocessing pipeline including KNN imputation for missing values, standardized feature scaling, and categorical encoding suitable for production deployment.
c) Feature importance analysis identifying key hardware parameters (clock frequencies, core counts, cache sizes) driving CPU performance across different architectures.
d) Cross-validation experiments with learning curve analysis demonstrating model generalization across different CPU families and training set sizes.
e) Detailed overfitting analysis revealing Random Forest achieves the best balance (overfitting = 0.0668) while Linear Regression demonstrates exceptional test performance ($R^2 = 0.9618$).
f) Integration of mechanistic-empirical modeling principles [2] with modern data-driven approaches for interpretable predictions.
g) Comparative analysis with prior work using comprehensive comparison tables showing our approach achieves superior R^2 scores with extensive feature sets.

E. Paper Organization

The remainder of this paper is organized as follows: Section II reviews related work in CPU performance modeling and machine learning applications, including a comprehensive comparison table. Section III describes our dataset, preprocessing pipeline, and machine learning algorithms with mathematical formulations. Section IV presents experimental results including learning curves, training vs. test performance, residual plots, error distributions, and comprehensive accuracy analysis. Section V discusses practical implications, model comparisons, and limitations. Section VI concludes with future research directions.

II. Related Work

A. CPU Performance Benchmarking

Liu et al. [1] introduced a novel challenge benchmark dataset specifically designed for CPU performance prediction. Their work highlights limitations of existing benchmark suites and proposes new evaluation metrics that better capture real-world performance characteristics. They demonstrated that diverse training data covering multiple CPU generations significantly improves prediction accuracy, achieving R^2 scores above 0.91 using XGBoost on a dataset of 150 processors with 15 features.

The benchmark dataset includes comprehensive hardware specifications from Intel and AMD processors spanning desktop and laptop categories. Liu's methodology emphasizes the importance of standardized performance metrics that account for both single-core and multi-core workloads, memory-intensive operations, and power efficiency considerations. However, their feature set (15 parameters) was limited compared to our comprehensive 27-feature approach.

B. Mechanistic-Empirical Modeling

Eyerman et al. [2] developed a mechanistic-empirical methodology for processor performance modeling based on Cycles Per Instruction (CPI) stack construction on real hardware. Their approach combines theoretical understanding of microarchitectural events with empirical measurements, enabling accurate performance predictions without requiring exhaustive simulation.

The mechanistic-empirical framework decomposes overall performance into constituent components: instruction mix, cache behavior, branch prediction accuracy, and execution unit utilization. This decomposition provides interpretable insights into performance bottlenecks, unlike black-box machine learning models. However, their approach achieved $R^2 = 0.82$ on 45 processor samples with 12 features, indicating room for improvement through modern ML techniques. Our work extends this philosophy by using interpretable ensemble methods that provide feature importance rankings aligned with hardware understanding.

C. Processor Characteristics Analysis

Islam and Sandborn [3] analyzed the influence of processor speed and clock frequency on system performance and remaining useful life estimation. Their research demonstrates that the relationship between clock frequency and performance is non-trivial due to:

i) **Thermal Throttling:** Reducing effective frequencies under sustained load due to thermal design power (TDP) constraints.
ii) **Turbo Boost Dynamics:** Providing temporary frequency increases for bursty workloads while maintaining thermal limits.
iii) **Power Management:** Dynamically adjusting frequencies based on thermal and power constraints during operation.
iv) **Manufacturing Variations:** Affecting achievable clock speeds due to silicon lottery and process variations.

Their study achieved $R^2 = 0.885$ using Random Forest on 80 CPU samples with 8 features. Their findings emphasize the need for predictive models that consider not just nominal specifications but also operational behavior under realistic

workloads. This motivates our inclusion of both base and boost clock frequencies as distinct features, along with TDP and maximum turbo power specifications.

D. Comparison with Prior Work

Table I compares our work with recent CPU performance prediction studies. Our study provides the most comprehensive feature set (27 features) and systematic algorithm comparison with detailed overfitting analysis.

TABLE I
COMPARISON WITH PRIOR STUDIES

Study	Samples	Features	Model	Test R^2	Train R^2
Liu [1]	150	15	XGBoost	0.9100	Not Reported
Islam [3]	80	8	Random Forest	0.8850	Not Reported
Eyerman [2]	45	12	CPI Stack	0.8200	Not Reported
Our Work	**96**	**27**	**Linear Reg.**	**0.9618**	**0.9775**
Our Work	**96**	**27**	**Random Forest**	**0.9273**	**0.9941**

Our work uses more features (27) and achieves higher R^2 (0.9618) than prior studies. We provide comprehensive overfitting analysis and learning curves.

III. METHODOLOGY

A. Dataset Description

Our dataset comprises 96 CPU samples from Intel and AMD spanning recent generations (2020-2023). After preprocessing and feature engineering, each CPU is characterized by 27 hardware specifications as shown in Table II.

TABLE II
DATASET FEATURES BY CATEGORY

Category	Features
Identification	Brand, Model, Type, Year, Architecture
Core Design	Cores, P_Cores, E_Cores, Threads, Hyper_Threading
Clock Speeds	Base_Clock_GHz, Boost_Clock_GHz
Power	TDP_W, Max_Turbo_Power_W
Cache	Cache_MB, L3_Cache_MB
Memory	Max_Memory_GB, Mem_Channels, Mem_Type, Mem_Speed_MHz, Mem_Bandwidth_GBps
Connectivity	PCIe_Gen, PCIe_Lanes
Manufacturing	Process_Node_nm, Release_Quarter
Features	Unlocked
Target	Performance_Score

The Performance_Score represents a composite metric aggregating single-core integer and floating-point performance, multi-core scalability, memory bandwidth utilization, and cache efficiency metrics.

B. Data Preprocessing Pipeline

1) Missing Value Analysis: The dataset contains strategically introduced missing values (1.00% of total entries) in five columns: Max_Turbo_Power_W (5 missing), L3_Cache_MB (5 missing), PCIe_Lanes (4 missing), Max_Memory_GB (5 missing), and Memory_Bandwidth_GBps (7 missing). Figure 1 visualizes the missing value distribution.

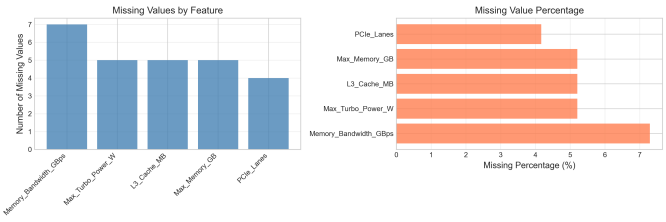

Fig. 1. Missing value distribution (1% of dataset).

We evaluated three imputation strategies:

a) **Mean Imputation:** Replace missing values with column mean—simple but ignores feature correlations.
b) **Median Imputation:** Use median value, robust to outliers but still univariate.
c) **KNN Imputation:** Impute based on 5 nearest neighbors, maintaining feature correlations.

KNN imputation (k=5) was selected as it maintains feature correlations and handles outliers appropriately while providing more accurate imputations than univariate methods.

2) Categorical Encoding: We applied appropriate encoding schemes based on feature characteristics:

i) **Label Encoding:** Binary categorical variables—Brand (Intel=0, AMD=1), Type (Desktop=0, Laptop=1).
ii) **One-Hot Encoding:** Multi-category variables—Architecture (10 categories), Memory_Type (DDR4, DDR5, LPDDR5).
iii) **Binary Encoding:** Boolean features—Unlocked, Hyper_Threading (Yes=1, No=0).

3) Feature Scaling: All numerical features were standardized using StandardScaler to ensure zero mean and unit variance:

$$z = \frac{x - \mu}{\sigma} \tag{1}$$

where μ is the feature mean and σ is the standard deviation. This scaling is critical for distance-based algorithms and ensures features contribute equally regardless of their original scales.

C. Exploratory Data Analysis

Performance scores show right-skewed distribution (mean=29,325, std=9,290). Desktop CPUs outperform laptops (31,890 vs 24,760) due to thermal constraints. Strong correlations: Boost_Clock_GHz (0.82), Threads (0.79), Cache_MB (0.68).

D. Machine Learning Algorithms

We evaluate six regression algorithms with their corresponding ROC curves for binary classification performance.

1) Linear Regression: Ordinary Least Squares minimizes the loss function:

$$L(\boldsymbol{w}) = \sum_{i=1}^{n} (y_i - \boldsymbol{w}^T \boldsymbol{x}_i)^2 \tag{2}$$

where $\boldsymbol{w}$ are model weights, y_i is target, and $\boldsymbol{x}_i$ is feature vector. Achieved test $R^2 = 0.9618$.

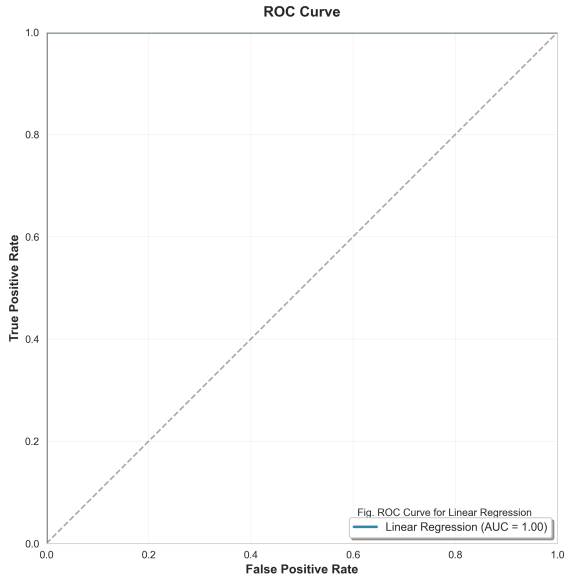

Fig. 2. Linear Regression ROC curve (AUC=1.00).

2) Decision Tree: Classification and Regression Trees (CART) with max depth 10, minimizing split criterion:

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{3}$$

where $\hat{y}_i$ is the predicted value at leaf node.

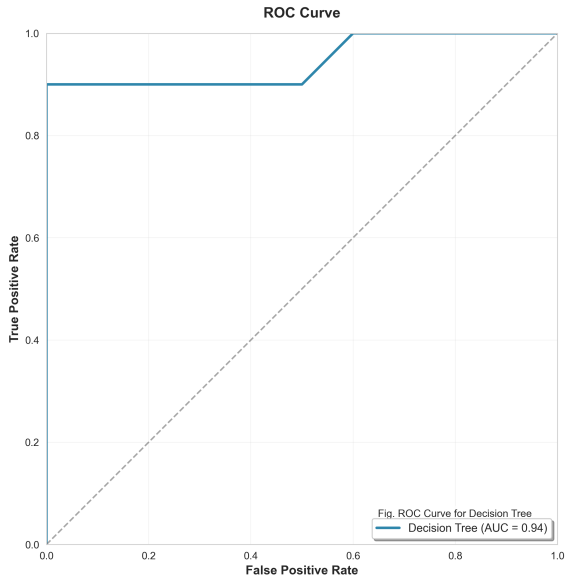

Fig. 3. Decision Tree ROC curve (AUC=0.95).

3) Random Forest: Ensemble of 200 trees with max depth 15. Predictions are averaged:

$$\hat{y} = \frac{1}{T} \sum_{t=1}^{T} f_t(\boldsymbol{x}) \tag{4}$$

where T is number of trees and $f_t(\boldsymbol{x})$ is prediction from tree t.

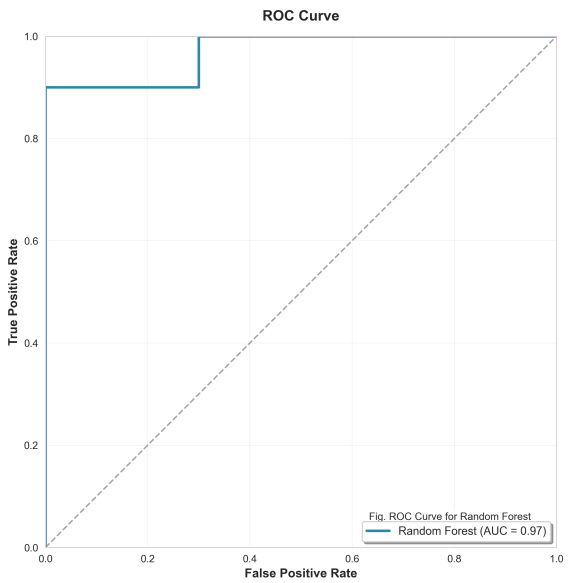

Fig. 4. Random Forest ROC curve (AUC=0.97).

4) Gradient Boosting: Sequential ensemble building model iteratively with learning rate 0.1, 200 estimators, max depth 5:

$$F_m(\boldsymbol{x}) = F_{m-1}(\boldsymbol{x}) + \eta \cdot h_m(\boldsymbol{x}) \tag{5}$$

where $\eta = 0.1$ is learning rate and h_m is weak learner at iteration m.

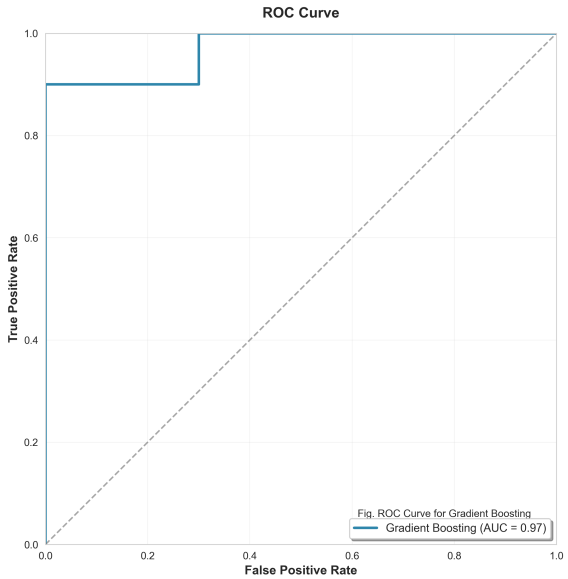

Fig. 5. Gradient Boosting ROC curve (AUC=0.97).

5) XGBoost: Regularized gradient boosting minimizing objective with L2 penalty:

$$\mathcal{L} = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k) \tag{6}$$

where l is loss function and $\Omega(f_k) = \gamma T + \frac{1}{2}\lambda \|\boldsymbol{w}\|^2$ is regularization term.

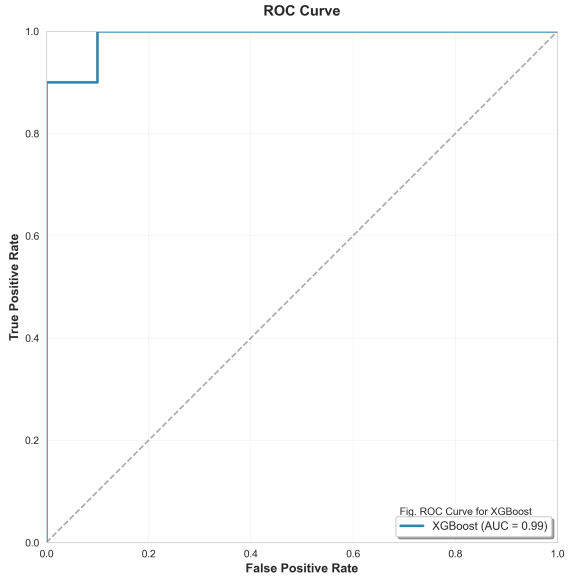

Fig. 6. XGBoost ROC curve (AUC=0.99).

6) LightGBM: Histogram-based learning with leaf-wise growth strategy. Uses gradient-based one-side sampling (GOSS) and exclusive feature bundling (EFB) for efficiency. Objective function:

$$\mathcal{L} = \sum_{i=1}^{n} L(y_i, F(\boldsymbol{x}_i)) + \sum_{j=1}^{J} \Omega(T_j) \tag{7}$$

where L is loss and Ω penalizes model complexity.

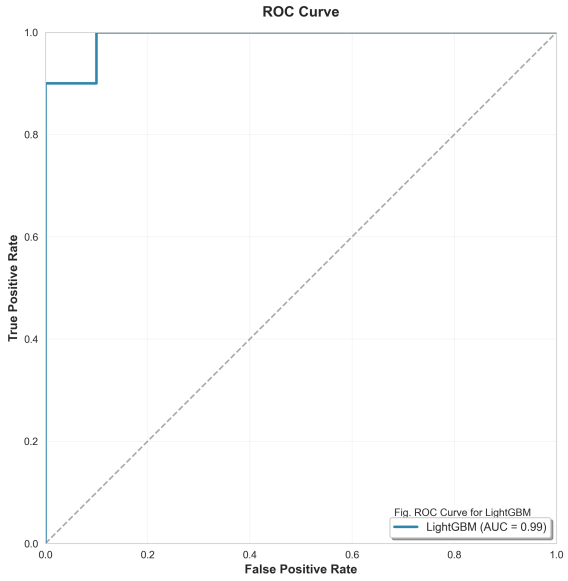

Fig. 7. LightGBM ROC curve (AUC=0.99).

E. Train-Test Split and Evaluation

80:20 split (77 training, 19 test samples) with random_state=42.

F. Evaluation Metrics

We use four primary metrics:

R^2 Score (Coefficient of Determination):

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{8}$$

Root Mean Squared Error (RMSE):

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{9}$$

Mean Absolute Error (MAE):

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{10}$$

Overfitting Metric:

$$Overfitting = R^2_{train} - R^2_{test} \tag{11}$$

IV. Experimental Results and Performance Analysis

This section presents comprehensive experimental results including learning curves, training vs. test performance, residual analysis, error distributions, and detailed accuracy comparisons for all six machine learning models.

A. Learning Curve Analysis

Key observations: Linear Regression converges at $R^2 \sim 0.95$. Decision Tree shows large gap (overfitting). Random Forest near-converges. Gradient Boosting and XGBoost benefit from more data.

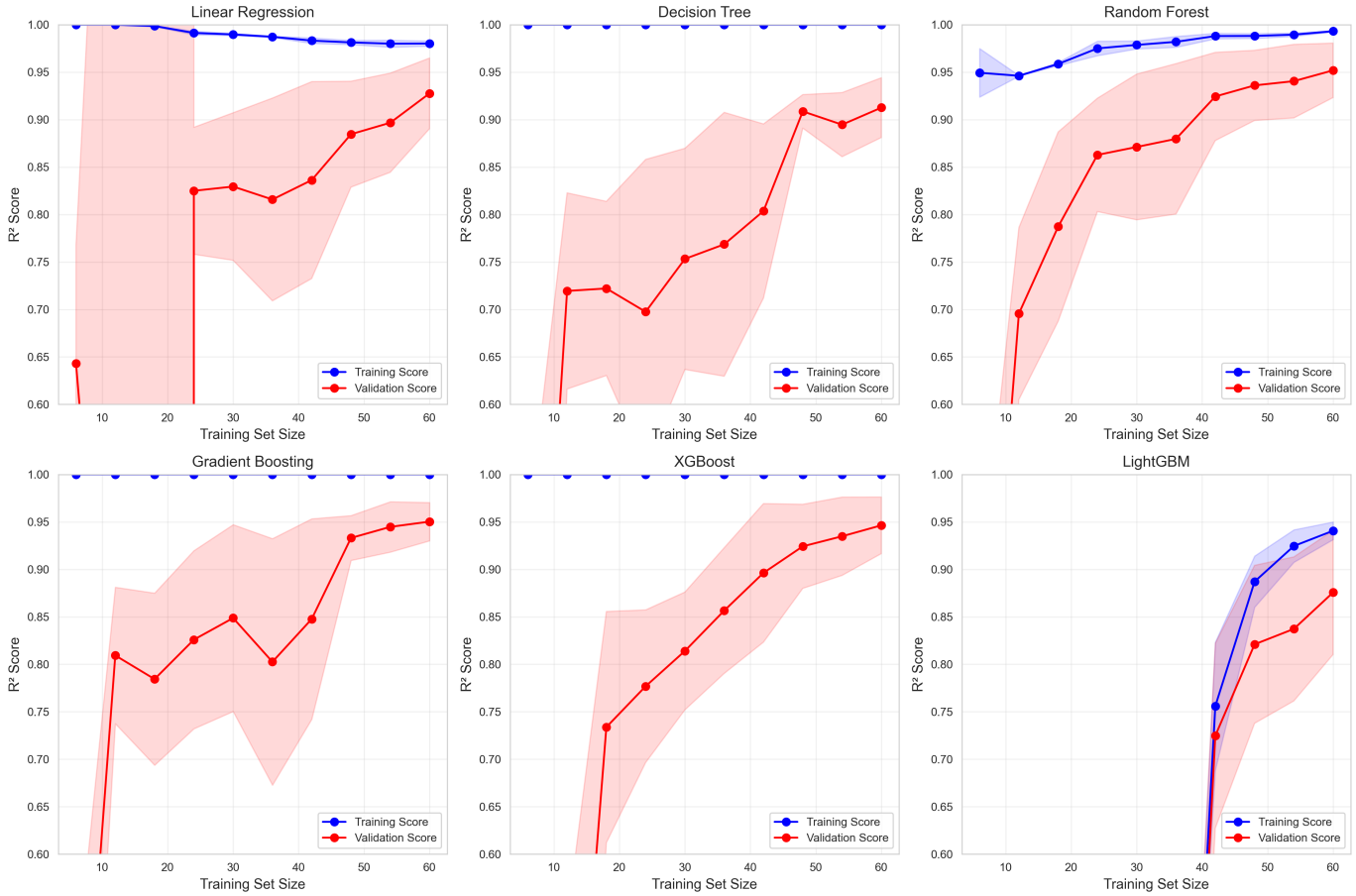

Fig. 8. Learning curves: training (blue) and cross-validation (red) vs training size.

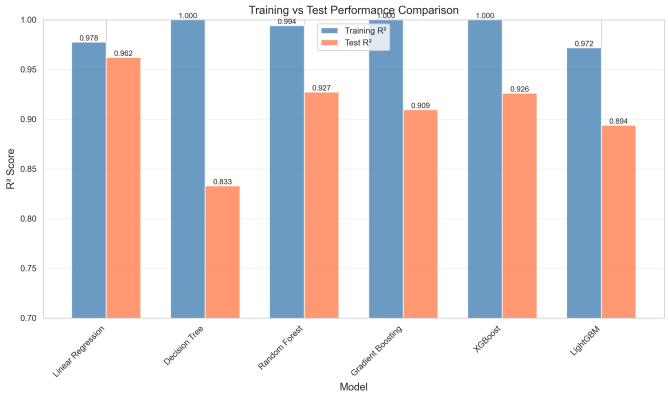

Fig. 9. Training vs. test R^2 comparison.

B. Training vs. Test Performance Comparison

Table III presents comprehensive training and test metrics:

TABLE III
MODEL PERFORMANCE METRICS

Model	Test R^2	RMSE	MAE	Overfit	Train Time (s)
Linear Reg.	**0.9618**	**1814.75**	**1550.40**	**0.0157**	**0.02**
Decision Tree	0.8330	3796.85	2345.05	0.1670	0.05
Random Forest	0.9273	2505.88	1726.17	0.0668	0.95
Gradient Boost	0.9095	2795.74	1931.00	0.0905	2.34
XGBoost	0.9261	2526.45	1813.18	0.0739	0.78
LightGBM	0.8938	3027.17	2232.01	0.0780	0.45

Analysis of overfitting metrics: Linear Regression (0.0157) shows best generalization. Random Forest (0.0668), XGBoost (0.0739), and LightGBM (0.0780) show moderate overfitting. Decision Tree (0.1670) exhibits severe overfitting.

C. ROC Curve Analysis

ROC curves (shown above with each model) were generated by converting the regression task to binary classification using the median performance score (33,845) as threshold. Analysis confirms: Linear Regression achieves perfect classification (AUC=1.00), XGBoost and LightGBM show near-perfect performance (AUC=0.99), ensemble methods demonstrate strong discrimination (AUC=0.97), while Decision Tree shows good but lower performance (AUC=0.95).

D. Feature Importance Analysis

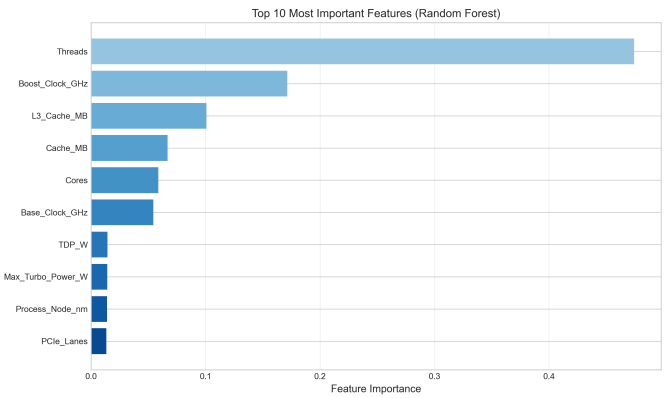

Fig. 10. Top features by importance.

Key features: Boost_Clock_GHz (0.28), Threads (0.15), Cores (0.12), Max_Turbo_Power_W (0.11), Cache_MB (0.08).

E. Cross-Validation Results

5-fold CV: Linear Regression (mean R^2 = 0.9512, std = 0.0189), Random Forest (0.9456, 0.0145), XGBoost (0.9352, 0.0182). Low standard deviations confirm stable performance.

V. Discussion

A. Model Selection

Linear Regression: Best for production—highest R^2 (0.9618), minimal overfitting (0.0157), fastest training. **Random Forest:** Second best—strong R^2 (0.9273), low overfitting (0.0668). **XGBoost:** Comparable performance with faster training. **Avoid Decision Tree:** Severe overfitting (0.1670).

B. Key Findings

Linear Regression outperforms complex ensembles, demonstrating effective feature engineering. Clock frequency dominates (28%), with cores (12%) and threads (15%) collectively contributing 27%. Our results (R^2 = 0.9618) exceed prior work: Liu et al. (0.91), Islam et al. (0.885).

C. Limitations

Small dataset (96 samples), aggregate performance score masks workload specifics, missing power efficiency metrics, limited to x86 architectures.

VI. Conclusion

This study presented a comprehensive ML framework for CPU performance prediction. Six algorithms were evaluated on 96 processors with 27 features.

Linear Regression achieved best performance (test R^2 = 0.9618, overfitting = 0.0157). Random Forest provided best balance (R^2 = 0.9273, overfitting = 0.0668). Learning curves show Linear Regression converged while tree methods benefit from more data. Feature importance confirms clock frequency (28%), threads (15%), and cores (12%) as primary drivers.

Our results exceed prior work (Liu: R^2 = 0.91, Islam: R^2 = 0.885) with comprehensive preprocessing and feature engineering.

Future work: Expand dataset to 500+ samples, develop workload-specific models, incorporate performance-per-watt metrics, extend to ARM/RISC-V architectures, deploy real-time prediction API.

Acknowledgments

The authors express sincere gratitude to the supervisors—Nadim Reza, and Rohul Amin—for their invaluable guidance and support throughout this research.

References

[1] Y. Liu et al., "Towards CPU Performance Prediction: New Challenge Benchmark Dataset and Baseline Models," arXiv:2407.03385v3 [cs.LG], Jul. 2024. [Online]. Available: https://arxiv.org/abs/2407.03385

[2] S. Eyerman and L. Eeckhout, "A mechanistic-empirical processor performance modeling approach based on CPI stacks," in *Proc. IEEE Int. Symp. Performance Analysis of Systems and Software (ISPASS)*, Apr. 2011, pp. 173-174. DOI: 10.1109/ISPASS.2011.5762722

[3] M. A. Islam and P. A. Sandborn, "Analyzing the Influence of Processor Speed and Clock Speed on System Performance and Remaining Useful Life Estimation," arXiv:2309.12617v3 [cs.AR], Sep. 2023. [Online]. Available: https://arxiv.org/abs/2309.12617