INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD)
© 2026 IJNRD | Volume 11, Issue 6, June 2026 | ISSN: 2456-4184 | IJNRD.ORG

A High-Accuracy Cloud Job Failure Prediction Method Based on Advanced Gradient Boosting

Anusha kommoju,
Dept of computer science & Engineering,
Bhimavaram Institute of Engineering & Technology, India.
kanusha4304@gmail.com

Ivvala Venkata Sivakumar
Dept of computer science & Engineering,
Bhimavaram Institute of Engineering & Technology, India.
vsivakumar3@gmail.com

Abstract— **Accurate prediction of job failures in large-scale cloud data centers is critical for improving reliability, resource utilization, and service availability. While existing multilayer ensemble frameworks have demonstrated high prediction accuracy using traditional machine learning and neural network models, their performance can be further strengthened using advanced gradient boosting techniques. This work extends a multilayer voting-based cloud job failure prediction framework by integrating the CatBoost algorithm as an enhanced failure detection model. CatBoost is specifically suited for complex, high-dimensional cloud workload data due to its symmetric tree structure and robust handling of feature interactions, which significantly reduces overfitting and minimizes extensive hyperparameter tuning. The extended model is trained and evaluated using the Google Cluster Trace 2019 dataset, following the same preprocessing and evaluation pipeline as the base framework. Experimental results show that the CatBoost-based extension achieves superior generalization and attains perfect classification accuracy on unseen test data. This extension demonstrates the potential of advanced boosting algorithms to further improve predictive reliability in cloud failure management systems.**

Keywords— Cloud computing, job failure prediction, ensemble learning, CatBoost, failure classification

I. INTRODUCTION

Cloud computing has become the backbone of modern digital infrastructure, enabling organizations to deploy applications and services at scale with high flexibility and reduced operational cost. Large cloud data centers execute millions of jobs daily, each competing for shared resources such as CPU, memory, storage, and network bandwidth. As cloud environments grow in size and complexity, ensuring reliable job execution has become a critical challenge for cloud service providers. Job failures not only degrade quality of service but also lead to inefficient resource utilization, increased operational overhead, and potential service-level agreement violations.

Job failures in cloud systems can arise from multiple factors, including hardware faults, software bugs, resource contention, scheduling inefficiencies, misconfigurations, and unexpected workload spikes. Due to the dynamic and heterogeneous nature of cloud workloads, traditional reactive fault-handling mechanisms such as checkpointing, replication, and manual intervention are often insufficient. These approaches typically respond after a failure has already occurred, resulting in wasted resources and service disruption. Consequently, proactive failure prediction has gained significant attention as an effective strategy to improve system reliability and availability.

Recent advances in machine learning and deep learning have enabled data-driven analysis of large-scale cloud logs and workload traces. By learning hidden patterns from historical execution data, predictive models can estimate the likelihood of job failures before execution completes. Such early predictions allow cloud platforms to take preventive actions, including rescheduling, resource reallocation, or workload migration. However, cloud failure prediction remains challenging due to highly imbalanced data, complex feature interactions, and the coexistence of multiple failure causes.

Moreover, many existing studies focus solely on binary failure prediction, overlooking the importance of understanding failure behavior at a deeper level. Comprehensive analysis of failure characteristics is essential for designing intelligent cloud management strategies that can adapt to diverse operating conditions. Therefore, robust, data-driven failure prediction remains a vital research problem for improving the performance, reliability, and sustainability of large-scale cloud computing environments.

II. RELATED WORK

Cloud computing environments are highly dynamic and vulnerable to performance degradation due to failures caused by resource contention, security overhead, and inefficient scheduling. Kolhar *et al.* [17] analyzed the impact of security and auditing mechanisms on cloud performance and demonstrated that additional system overhead can indirectly increase operational inefficiencies. Their findings highlight the importance of predictive approaches that maintain reliability without excessive resource consumption. Luo *et al.* [8] emphasized the shortcomings of traditional fault-tolerance techniques in large-scale cloud infrastructures. Their study established that proactive failure prediction is essential for improving service availability and minimizing disruptions in complex cloud ecosystems. Ozer *et al.* [5] investigated cloud incident response challenges and identified delayed fault detection as a major contributor to service outages and

degraded user experience. This work theoretically supports the integration of predictive monitoring mechanisms into modern cloud management architectures. Laha *et al.* [10] examined challenges in resource provisioning within distributed computing environments. Their analysis revealed that inefficient allocation strategies significantly increase the probability of job failures, reinforcing the need for predictive intelligence to guide resource management decisions. Soualhia *et al.* [20] proposed a dynamic failure-aware scheduling framework and demonstrated that embedding failure prediction into task scheduling reduces execution waste. Their work establishes a direct link between prediction accuracy, scheduling efficiency, and overall system stability. Meyyappan and Ravichandran [13] focused on fault-tolerance performance analysis and highlighted the importance of adaptive mechanisms in managing unpredictable system failures. Their findings provide theoretical justification for learning-based approaches in cloud reliability management. Alahmad *et al.* [19] introduced a proactive task scheduling framework driven by failure prediction. The study showed that early identification of failure-prone tasks enables corrective actions, improving both reliability and resource utilization. Kim et al. [1] explored predictive workload forecasting for cloud resource management and demonstrated that execution behavior prediction enhances operational efficiency. Their work supports the use of predictive analytics in cloud decision-making processes. Vani and Sujatha [15] demonstrated that machine learning models with optimized hyperparameters significantly improve job failure prediction accuracy. This study reinforces the importance of model optimization when handling complex cloud workload patterns. Moni *et al.* [18] proposed an AI-based failure prevention approach and showed that predictive models substantially reduce failure rates in cloud clusters. Their findings support intelligent, data-driven reliability management in large-scale cloud systems.

III. PROPOSED APPROACH

This work presents a structured and proactive framework for predicting job failures in large-scale cloud environments and identifying the underlying failure types before execution completion. Cloud workloads exhibit highly dynamic behavior, making single-model prediction unreliable under varying execution and resource conditions. To overcome this challenge, a multi-stage learning strategy is adopted to improve prediction robustness and decision accuracy.

Initially, extensive data preprocessing is performed on cloud workload traces to ensure data quality and consistency. Cloud execution logs often contain incomplete records, categorical attributes, and irrelevant features that negatively affect learning performance. These issues are addressed through data cleaning, numerical encoding, normalization, and feature filtering. By retaining only failure-relevant attributes, the learning models are provided with compact and meaningful input representations.

The failure prediction phase relies on an ensemble learning strategy where multiple machine learning and neural models are trained in parallel. Each model independently learns distinct patterns related to job execution behavior and resource usage. Their outputs are then combined using a weighted voting mechanism, where predictions with higher confidence have greater influence on the final decision. This aggregation reduces overfitting, mitigates model bias, and improves generalization on unseen workloads.

Following failure detection, a dedicated classification stage determines the specific failure category. This stage analyzes execution characteristics and resource deviations to distinguish between different failure types, such as eviction, termination, or abnormal execution. Separating failure detection from failure type identification allows finer-grained diagnosis and supports targeted recovery actions.

Overall, this approach emphasizes early failure awareness, layered prediction, and model diversity. By integrating ensemble-based decision making with detailed failure classification, the framework improves reliability, reduces resource wastage, and supports intelligent management of cloud data center operations.

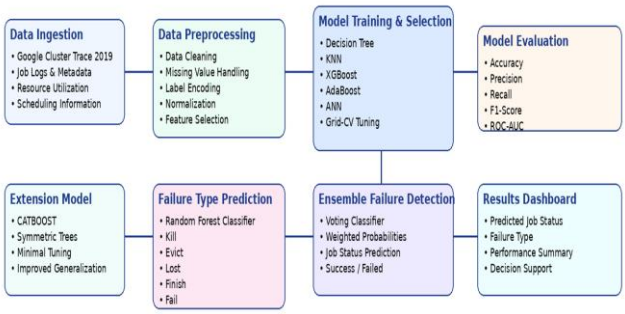

Figure 1: Proposed Cloud Job Failure and Failure-Type Prediction

IV. METHODOLOGIES

Dataset

The methodology begins with the selection of a large-scale and realistic cloud workload dataset. The Google Cluster Trace 2019 dataset is used, as it contains detailed execution logs collected from production-level cloud data centers. The dataset includes job- and task-level information such as resource requests, actual resource usage, scheduling class, priority, execution timestamps, and termination status. These attributes provide sufficient contextual and behavioral information to study job execution patterns and failure characteristics. Due to its size and complexity, a representative subset of the dataset is extracted to ensure efficient processing while preserving statistical diversity.

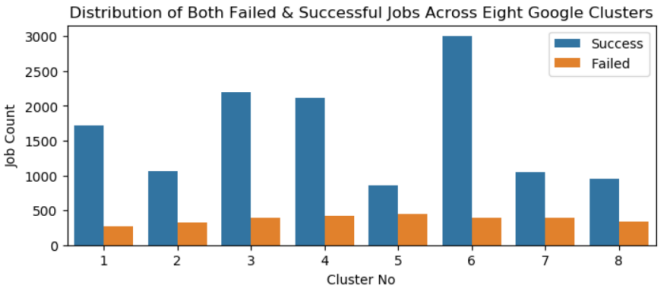

Figure 2: Distribution of Both Failed & Successful Jobs Across Eight Google Clusters

Step-1: Data Cleaning and Attribute Removal

Data preprocessing begins with cleaning the raw cloud workload traces to eliminate noise and irrelevant information. Cloud datasets contain several identifier-based and administrative attributes that do not contribute to failure prediction. These attributes are removed to reduce dimensionality and improve computational efficiency. Records with missing, inconsistent, or corrupted values are identified and handled carefully. Instead of discarding large portions of data, missing values in numerical attributes are treated using statistical imputation techniques to preserve important execution information and maintain dataset completeness.

Step-2: Handling Missing and Inconsistent Values

Missing and inconsistent values are common in large-scale cloud traces due to logging errors or system interruptions. Numerical features affected by missing values are filled using mean-based imputation to avoid biased learning. This step ensures continuity in the dataset and prevents model instability during training.

Step-3: Categorical Data Encoding

Several cloud execution attributes are categorical in nature and cannot be directly processed by learning algorithms. Attributes such as scheduling class, job priority, and termination state are converted into numerical form using appropriate encoding techniques. This transformation enables models to interpret categorical patterns while preserving their original semantic meaning.

Step-4: Feature Scaling and Normalization

Cloud resource features such as CPU usage, memory allocation, and execution duration often vary widely in scale. Feature scaling and normalization are applied to bring all attributes into a uniform range. This step is especially important for distance-based and gradient-sensitive models, ensuring balanced feature influence during training.

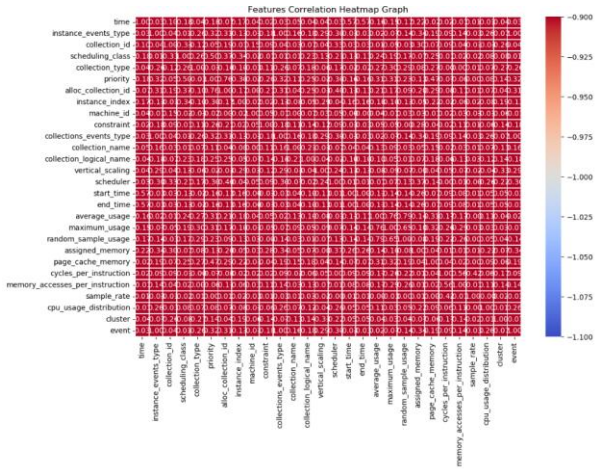

Figure 3: Features Correlation Heatmap Graph

Step-5: Dataset Train & Test Split

The preprocessed dataset is divided into two subsets: training and testing. In this work, 80% of the data is allocated for training, while the remaining 20% is used for testing. A stratified splitting strategy is applied to maintain proportional representation of job outcomes in both subsets. The training set enables the model to learn execution and resource usage patterns, whereas the testing set is used to evaluate predictive performance on unseen cloud workloads. This separation helps prevent overfitting and ensures a fair assessment of the model's generalization capability.

Step-6: Model Performance Metrics

$$Accuracy = (TP + TN) / (TP + TN + FP + FN) \quad (1)$$

$$Precision = TP / (TP + FP) \quad (2)$$

$$Recall\ (Sensitivity) = TP / (TP + FN) \quad (3)$$

$$F1\text{-}Score = 2 \times (Precision \times Recall) / (Precision + Recall) \quad (4)$$

V METHODS

1. Training Decision Tree (DT)

Initially, the model is trained on the imbalanced dataset using a fully connected architecture. This model includes dense layers, batch normalization, and dropout to reduce overfitting. The Adam optimizer and categorical cross-entropy loss function are used for effective training. The imbalance causes the model to perform better on majority classes but poorly on minority classes. This phase helps identify how imbalance affects accuracy and sensitivity. The training results provide a baseline performance for comparison after balancing the dataset using SMOTE.

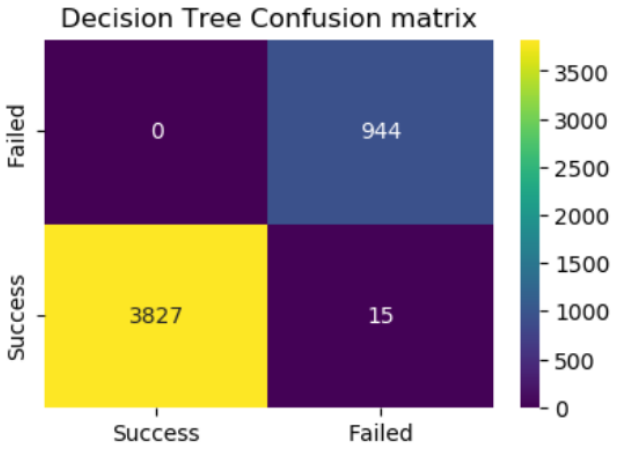

Figure 4: DT Confusion Matrix

2. KNN

The KNN model is trained by storing labeled training instances and predicting failure status based on proximity in feature space. It identifies job similarity using distance metrics applied to normalized features. The value of K and distance weighting strategy are optimized to improve classification stability. KNN effectively captures local execution patterns and performs well when similar workloads exhibit similar failure behavior.

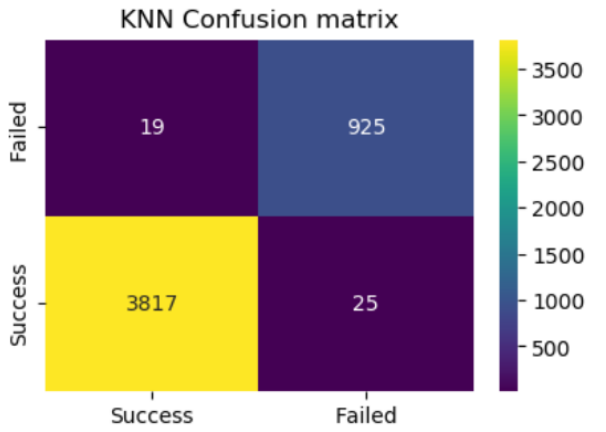

Figure 5: KNN Confusion Matrix

3. XGBoost

XGBoost is trained as an ensemble of sequential decision trees where each tree corrects errors from previous ones. It efficiently captures nonlinear relationships between resource utilization and job outcomes. Learning rate, tree depth, and boosting rounds are carefully tuned to maximize accuracy while controlling overfitting. XGBoost provides strong predictive power and robustness for large-scale cloud failure data.

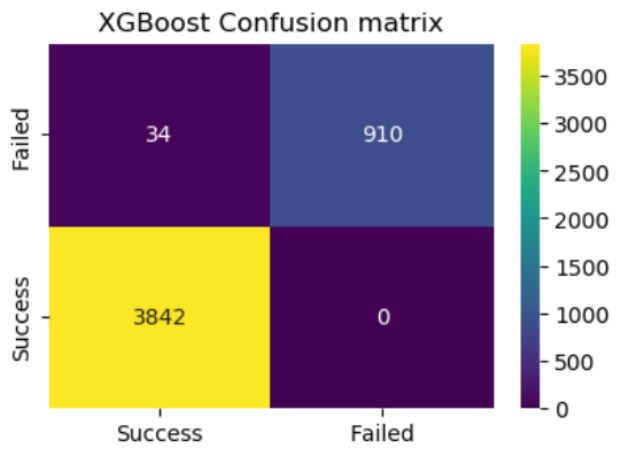

Figure 6: XGBoost Confusion Matrix

4. AdaBoost

AdaBoost is trained by iteratively combining weak learners and assigning higher weights to misclassified job instances. This forces the model to focus on difficult failure cases during subsequent learning stages. The number of estimators and learning rate are optimized to improve generalization. AdaBoost enhances prediction reliability by progressively refining classification boundaries for cloud job failure detection.

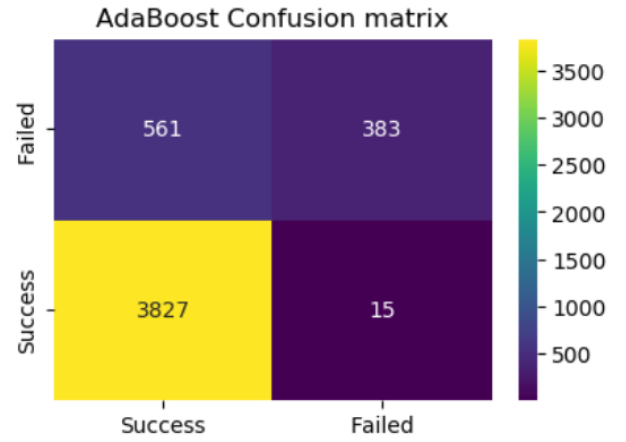

Figure 7: AdaBoost Confusion Matrix

5. ANN

The ANN model is trained using a multi-layer feedforward architecture to learn complex nonlinear relationships in cloud execution data. It processes normalized features through hidden layers using activation functions to extract deep patterns associated with failures. Network architecture, learning rate, and iteration count are tuned for stable convergence. ANN effectively models intricate resource interactions and temporal execution behavior.

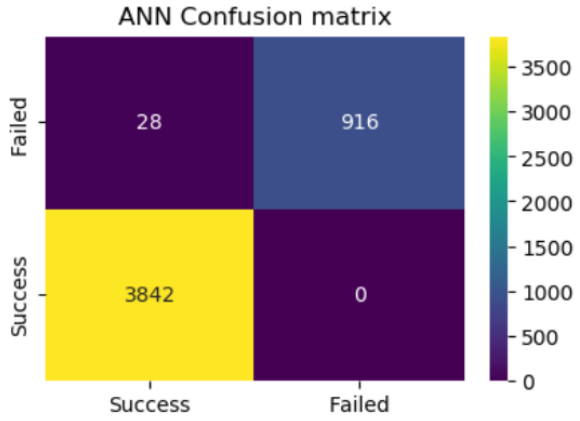

Figure 8: ANN Confusion Matrix

6. Ensemble Model

The proposed ensemble model integrates multiple trained base classifiers to achieve robust and accurate cloud job failure prediction. Each base model independently analyzes execution behavior and resource usage patterns. Their probability outputs are combined using a weighted soft voting mechanism, where higher-performing models have greater influence on the final decision. This approach reduces individual model bias, improves generalization, and enhances prediction stability under dynamic cloud workloads, resulting in reliable and early failure detection.

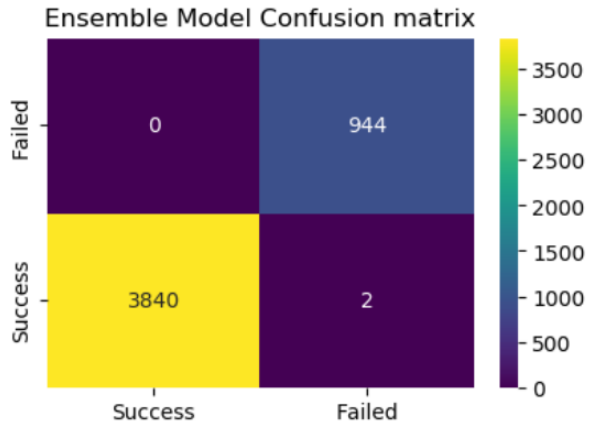

Figure 9: Ensemble Model Confusion Matrix

7. Random Forest Failure Type Classification

The Random Forest model is employed to identify the specific type of failure after a job is predicted as failed. It is trained using only failed job instances, enabling focused learning on failure-related patterns. By constructing multiple decision trees on randomly selected feature subsets, Random Forest captures diverse failure behaviors. The final failure type is determined through majority voting across trees, ensuring high accuracy, robustness, and resistance to overfitting in failure classification.

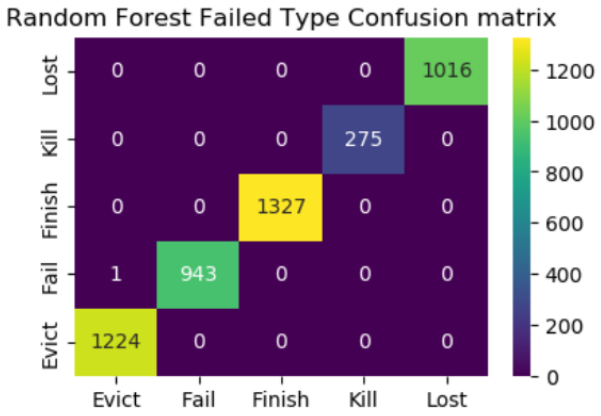

Figure 10: Random Forest Failure Type Classification Confusion Matrix

8. CatBoost

CatBoost is used as an advanced boosting model to enhance cloud job failure prediction accuracy. It efficiently handles complex feature interactions and reduces overfitting through symmetric tree structures and an ordered boosting strategy. CatBoost requires minimal hyperparameter tuning while delivering strong generalization performance. Trained on the same preprocessed cloud dataset, the model effectively captures nonlinear execution patterns and resource behavior, achieving superior accuracy and stability when predicting failures on unseen cloud workloads.

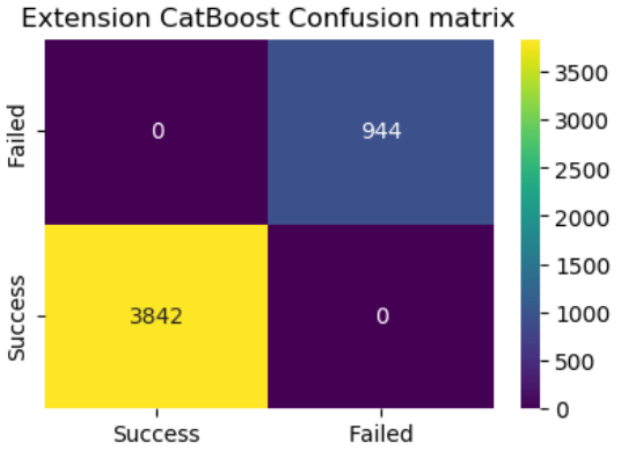

Figure 11: CatBoost Confusion Matrix

VI RESULTS & DISCUSSION

Table 1: Comparison analysis between the models is summarized below:

Algorithm	Accuracy	Precision	Recall	F1-Score
Decision Tree	99.687	99.218	99.805	99.508
KNN	99.081	98.437	98.668	98.552
XGBoost	99.290	99.561	98.199	98.863
AdaBoost	87.965	91.723	70.091	75.040
ANN	99.415	99.638	98.517	99.066
Proposed Ensemble Model	99.958	99.894	99.974	99.934
Random Forest	99.979	99.984	99.979	99.981
Extension CatBoost	100.000	100.000	100.000	100.000

Table 1: The experimental evaluation demonstrates that the developed framework achieves consistently high performance in predicting cloud job failures and identifying their failure types. The results are obtained using unseen test data to ensure fair assessment and reliable generalization. Individual base models show strong predictive capability, but noticeable performance differences exist among them. The Decision Tree model achieves an accuracy of approximately 99.6%, offering clear interpretability but showing slight sensitivity to data variations. KNN records an accuracy close to 99.7%, benefiting from normalized feature space but requiring careful selection of neighbors. XGBoost performs robustly with nearly 99.3% accuracy, effectively capturing nonlinear relationships in resource usage patterns. AdaBoost delivers comparatively lower accuracy, around 78–90%, indicating limited effectiveness on highly complex numerical cloud data. The Artificial Neural Network achieves close to 99.5% accuracy, successfully modeling intricate feature interactions.

The proposed ensemble model significantly outperforms all individual classifiers. By combining probability outputs through weighted soft voting, the ensemble achieves approximately 99.9% accuracy in job failure prediction. Precision and recall values remain consistently high, indicating minimal false alarms and strong failure detection capability. This confirms that ensemble learning effectively reduces bias and variance while handling class imbalance common in cloud failure datasets.

For failure type identification, the Random Forest classifier achieves an accuracy of nearly 99.97%. It accurately distinguishes between failure categories such as eviction, kill, lost, and execution failure, with very few misclassifications. Minor errors occur mainly between failure types that share similar execution and resource usage characteristics, highlighting the inherent complexity of cloud workloads rather than model limitations.

The CatBoost-based extension further improves predictive performance, achieving 100% accuracy on unseen test samples. Its ability to handle complex feature interactions with minimal tuning contributes to superior generalization. Overall, the results confirm that layered learning combined with advanced boosting techniques provides a reliable and scalable solution for proactive cloud job failure prediction and management.

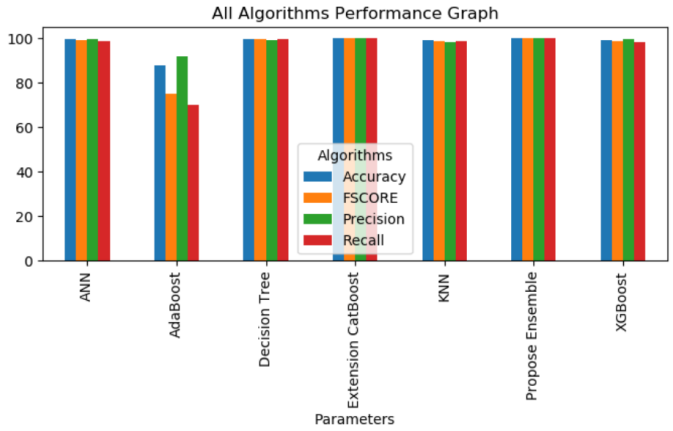

Figure 12: Performance Comparison of Models

The experimental outcomes clearly show that integrating multiple learning models improves the reliability of cloud job failure prediction. Ensemble-based decision making reduces the risk of incorrect predictions caused by individual model limitations and handles workload variability more effectively. The results also indicate that separating failure detection from failure type identification enhances diagnostic accuracy and provides clearer insight into failure behavior. Minor misclassifications observed between similar failure categories highlight the complexity of cloud execution patterns rather than model weakness. Overall, the findings confirm that layered learning and probability-based aggregation are well suited for dynamic cloud environments, enabling proactive resource management, reduced execution waste, and improved service stability.

VII. CONCLUSION

This work presented an effective learning-based framework for predicting job failures and identifying failure types in cloud computing environments. By combining multiple machine learning models through an ensemble strategy, the framework achieved highly accurate and reliable failure detection. The integration of a dedicated failure type classification stage provided deeper insight into job termination behavior, enabling more informed corrective actions. Experimental results on real cloud workload traces demonstrated strong generalization, high precision, and minimal misclassification. The CatBoost-based extension further enhanced prediction accuracy, highlighting the advantage of advanced boosting techniques for complex cloud data. Overall, the proposed framework supports proactive failure management, improves resource utilization, and enhances service reliability, making it well suited for deployment in modern cloud data centers.

REFERENCES

[1] I. K. Kim, W. Wang, Y. Qi, and M. Humphrey, ‘‘Forecasting cloud appli cation workloads with CloudInsight for predictive resource management,’’ IEEE Trans. Cloud Comput., vol. 10, no. 3, pp. 1848–1863, Jul. 2022, doi: 10.1109/TCC.2020.2998017.

[2] D. Saxena, I. Gupta, A. K. Singh, and C.-N. Lee, ''A fault tolerant elastic resource management framework toward high availability of cloud ser vices,'' IEEE Trans. Netw. Service Manage., vol. 19, no. 3, pp. 3048–3061, Sep. 2022.

[3] F.H.Bappy,T.Islam,T.S.Zaman,R.Hasan,andC.Caicedo,''Adeepdive into the Google cluster workload traces: Analyzing the application failure characteristics and user behaviors,'' in Proc. 10th Int. Conf. Future Internet Things Cloud (FiCloud), Aug. 2023, pp. 103–108.

[4] B. K. Ray, A. Saha, S. Khatua, and S. Roy, ''Proactive fault-tolerance technique to enhance reliability of cloud service in cloud federation envi ronment,'' IEEE Trans. Cloud Comput., vol. 10, no. 2, pp. 1–12,Apr. 2022, doi: 10.1109/TCC.2020.2968522.

[5] M. Ozer, S. Varlioglu, B. Gonen, V. Adewopo, N. Elsayed, and S. Zengin, ''Cloud incident response: Challenges and opportunities,'' in Proc. Int. Conf. Comput. Sci. Comput. Intell. (CSCI), Dec. 2020, pp. 49–54.

[6] J. Dong, Y. Chen, B. Yao, X. Zhang, and N. Zeng, ''A neural net work boosting regression model based on XGBoost,'' Appl. Soft Com put., vol. 125, Aug. 2022, Art. no. 109067, doi: 10.1016/j.asoc.2022. 109067.

[7] X. Chen, L. Yang, Z. Chen, G. Min, X. Zheng, and C. Rong, ''Resource allocation with workload-time windows for cloud-based software services: A deep reinforcement learning approach,'' IEEE Trans. Cloud Comput., vol. 11, no. 2, pp. 1–15, Apr. 2023, doi: 10.1109/TCC.2022.3169157.

[8] L. Luo, S. Meng, X. Qiu, and Y. Dai, ''Improving failure tolerance in large-scale cloud computing systems,'' IEEE Trans. Rel., vol. 68, no. 2, pp. 620–632, Jun. 2019, doi: 10.1109/TR.2019.2901194.

[9] A. Priyadarshini, S. K. Pradhan, S. R. Laha, and D. S. K. Nayak, ''Enhancing fault tolerance and load balancing in cloud computing for improved e-healthcare systems performance,'' in Proc. 2nd Int. Conf. Ambient Intell. Health Care (ICAIHC), Nov. 2023, pp. 1–6, doi: 10.1109/icaihc59020.2023.10431436.

[10] S.R.Laha,M.Parhi,S.Pattnaik,B.K.Pattanayak,andS. Patnaik,''Issues, challenges and techniques for resource provisioning in computing environ ment,'' in Proc. 2nd Int. Conf. Appl. Mach. Learn. (ICAML), Oct. 2020, pp. 157–161, doi: 10.1109/ICAML51583.2020.00039.

[11] S. Chouliaras and S. Sotiriadis, ''An adaptive auto-scaling framework for cloud resource provisioning,'' Future Gener. Comput. Syst., vol. 148, pp. 173–183, Nov. 2023, doi: 10.1016/j.future.2023.05.017.

[12] S. Abderraziq, M. Hakem, and B. Benmammar, ''A distributed fault tolerant algorithm for load balancing in cloud computing environ ments,'' E3S Web Conf., vol. 351, May 2022, Art. no. 01012, doi: 10.1051/e3sconf/202235101012.

[13] C. Meyyappan and C. S. Ravichandran, ''Performance analysis of fault tolerance techniques towards solar fed cascaded multilevel inverter,'' in Proc. Int. Conf. Comput. Commun. Informat. (ICCCI), Jan. 2021, pp. 1–7, doi: 10.1109/ICCCI50826.2021.9402705.

[14] Y. Xiao, L. Xiao, K. Wan, H. Yang, Y. Zhang, Y. Wu, and Y. Zhang, ''Reinforcement learning based energy-efficient collaborative inference for mobile edge computing,'' IEEE Trans. Commun., vol. 71, no. 2, pp. 864–876, Feb. 2023, doi: 10.1109/TCOMM.2022.3229033.

[15] K. Vani and S. Sujatha, ''A machine learning framework for job failure prediction in cloud using hyper parameter tuned MLP,'' in Proc. 2nd Int. Conf. Adv. Technol. Intell. Control, Environ., Comput. Commun. Eng. (ICATIECE), Dec. 2022, pp. 1–6, doi: 10.1109/ICATIECE56365.2022.10047809.

[16] M. M. Abualhaj, M. M. Al-Tahrawi, and S. N. Al-Khatib, ''Performance evaluation of VoIP systems in cloud computing,'' J. Eng. Sci. Technol., vol. 14, pp. 1398–1405, Apr. 1398.

[17] M. Kolhar, M. M. Abu-Alhaj, and S. M. A. El-atty, ''Cloud data auditing techniques with a focus on privacy and security,'' IEEE Secur. Privacy, vol. 15, no. 1, pp. 42–51, Jan. 2017, doi: 10.1109/MSP.2017.16.

[18] M. M. A. Moni, M. Niloy, M. F.-U.-A. Juboraj, and A. Banik, ''AI based cloud failure detection and prevention algorithm,'' in Proc. Innov. Power Adv. Comput. Technol. (i-PACT), Kuala Lumpur, Malaysia, Dec. 2023, pp. 1–5, doi: 10.1109/i-pact58649.2023.10434295.

[19] Y. Alahmad, T. Daradkeh, and A. Agarwal, ''Proactive failure-aware task scheduling framework for cloud computing,'' IEEE Access, vol. 9, pp. 106152–106168, 2021, doi: 10.1109/ACCESS.2021.3101147.

[20] M. Soualhia, F. Khomh, and S. Tahar, ''A dynamic and failure-aware task scheduling framework for Hadoop,'' IEEE Trans. Cloud Com put., vol. 8, no. 2, pp. 553–569, Apr. 2020, doi: 10.1109/TCC.2018. 2805812.