A Hybrid Feature Engineering and Supervised Learning Approach for Anomaly Detection in HDFS Logs

Auroshikha Tripathy

Department of Computer Science and Engineering

Silicon University

Bhubaneswar, India

auroshikha.tripathy04@gmail.com

August 17, 2025

Abstract

In financial systems, a stable and secure data infrastructure is a critical requirement, with HDFS playing a foundational role. Standard anomaly detection often focuses on surface-level message content, potentially missing deeper clues hidden within the patterns of HDFS block IDs. This project introduces and validates a novel hybrid methodology that analyzes block IDs in conjunction with their log message context to achieve state-of-the-art anomaly detection. We engineered a comprehensive set of 60+ features, combining statistical properties of block IDs with TF-IDF vectorization of log message tokens. These features were used to train a hyperparameter-tuned XGBoost classifier, specifically optimized to handle the severe class imbalance inherent in the data. When tested on real HDFS logs from the widely-used Logpai dataset, the model demonstrated exceptional performance, achieving 0.9998 accuracy and an F1-Score of 0.9961. The near-perfect ROC AUC of 1.000, combined with extremely low false positives (17) and false negatives (4) on a test set of over 92,000 events, confirms the model's suitability for high-stakes environments where both precision and recall are non-negotiable. The system, written in Python 3, is a simple but powerful approach that opens a new avenue for identifying hidden operational and security risks.

Keywords: Anomaly Detection, HDFS, Machine Learning, Cybersecurity, Log Analysis, XGBoost, Feature Engineering.

1 Introduction

In large-scale distributed systems, operational logs serve as the first line of defence for detecting failures, misconfigurations, and security breaches. The Hadoop Distributed File System (HDFS), as a cornerstone of big data infrastructure, generates logs at a scale and complexity that make anomaly detection both critical and challenging [1]. Traditional detection methods often fail to capture the nuanced patterns that distinguish rare anomalies from the overwhelming volume of normal events.

While some studies have applied statistical anomaly detection [2] and others have leveraged text-based analysis [3], few have effectively combined both approaches. This gap is significant because HDFS anomalies often manifest through a blend of numeric irregularities and subtle shifts in textual event descriptions [4]. This research addresses that gap by proposing a hybrid feature engineering framework that enhances the ability of machine learning models to detect anomalies with high accuracy.

Research Gaps and Contributions

Despite advances in log anomaly detection, a significant gap remains in the effective fusion of statistical and semantic data for HDFS logs. Prior research has predominantly followed two distinct paths: numerical analysis of system metrics, which captures operational trends but lacks semantic context, and text-based analysis using NLP, which understands message content but often overlooks quantitative signals. Unsupervised models, while useful, have shown limitations in identifying context-specific anomalies, particularly when feature sets are not sufficiently rich. Furthermore, the increasing sophistication of threats, such as poisoning attacks that mimic legitimate behavior, necessitates models capable of understanding deeper, more nuanced patterns. This research directly addresses these gaps by introducing a novel hybrid feature engineering framework. By systematically combining numerical properties of HDFS block IDs with TF-IDF vectorization of log message tokens, our work provides a richer feature space that empowers a supervised classifier to detect subtle and complex anomalies with state-of-the-art accuracy, bridging the divide between purely statistical and purely semantic detection methods.

2 Literature Review

Prior work on log anomaly detection has generally followed two broad paths: numerical analysis of system metrics [2] and natural language processing (NLP) applied to log messages [3]. Numerical approaches capture behavioural trends but lack semantic depth, while NLP methods can identify unusual message patterns but often ignore quantitative signals.

The challenge of log analysis is compounded by the sheer volume of data generated by modern systems. The Hadoop ecosystem, designed for distributed processing of big data, is frequently used for security log analysis to handle this scale [5]. However, the underlying HDFS was not initially designed with security as a primary focus [6]. It lacks built-in, default encryption, exposing sensitive and confidential data at the storage level to potential security attacks [6]. This vulnerability underscores the importance of robust anomaly detection as a critical layer of defense. Research by Naisuty et al. conducted a systematic literature review on encryption algorithms for HDFS, concluding that while symmetric algorithms like AES offer better performance, they present security trade-offs, reinforcing the need for intelligent monitoring systems [6].

Unsupervised algorithms such as Isolation Forest, Local Outlier Factor (LOF), and One-Class SVM [7] have been widely used due to their ability to operate without labels. However, these methods struggle when anomalies are context-specific or when feature sets lack richness. Panda et al. highlight a similar challenge in the banking sector, where an event-driven approach using the Hadoop ecosystem is proposed to monitor key performance indicators and detect real-time operational anomalies from fast-moving transactional data [8]. Their framework emphasizes the necessity of scalable, data-driven approaches for analytics, which is directly applicable to HDFS log analysis.

More recent studies have advanced towards deep learning and hybrid models. For instance, Chnib & Gabsi proposed LogBERT, a BERT-based deep learning model for effective log anomaly detection that relies on log sequence embeddings [9]. Paladini et al. explored practical poisoning attacks against fraud detection systems and proposed adversarial training as a defense, demonstrating that malicious transactions can be crafted to mimic legitimate user behavior, making them difficult for standard models to detect [10]. This highlights the need for models that can understand deep contextual patterns. Similarly, Kumar proposed a hybrid Autoencoder-LSTM-Attention model for fraud detection, achieving high accuracy by combining feature learning, temporal pattern identification, and an attention mechanism to focus on significant fraud indicators [11].

These advanced approaches suggest that combining statistical and textual features can yield superior results, a domain that remains underexplored for HDFS logs. Our work builds on this by creating a hybrid feature space that integrates numerical block ID statistics with semantic information from log messages, providing a richer context for a supervised classifier to identify subtle anomalies effectively.

3 Research Methodology

Our approach followed a staged experimental design to systematically build and validate the proposed model. In response to the security challenges inherent in large-scale data systems like the Hadoop Distributed File System (HDFS), particularly its foundational role in financial systems and its lack of default data encryption, a novel hybrid research methodology has been developed for high-fidelity anomaly detection. This approach moves beyond traditional surface-level log analysis by employing a sophisticated feature engineering strategy that combines statistical properties of HDFS block IDs with TF-IDF vectorization of log message tokens, creating a rich, hybrid feature set. A hyperparameter-tuned XGBoost classifier trained on this data demonstrated state-of-the-art performance on the Logpai dataset, achieving 0.9998 accuracy and an F1-Score of 0.9961, with a near-perfect ROC AUC of 1.000, making it highly suitable for environments where precision is critical. This supervised method is a powerful alternative to other approaches, such as unsupervised OPTICS clustering or deep learning models like LogBERT, and is crucial for defending against advanced threats like poisoning attacks, where adversaries craft malicious data to evade standard detection systems. The development of such robust, AI-driven security protocols, often integrated with automation and blockchain, is becoming a cornerstone of innovation for ensuring the integrity and trustworthiness of next-generation FinTech and digital banking solutions. The overall workflow is depicted in Figure 1.

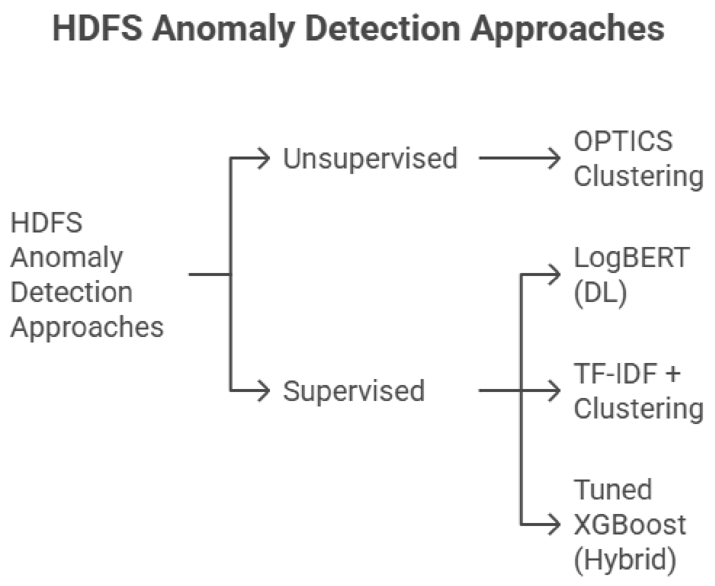

Figure 1: A flowchart illustrating the different approaches to HDFS Anomaly Detection evaluated in this study, branching into unsupervised and supervised methods.

The stages were as follows:

1. **Establishing an unsupervised baseline** to evaluate the discriminative power of numerical features alone.

2. **Transitioning to supervised models** to test whether algorithmic changes could compensate for weak features.

3. **Engineering a hybrid feature space** combining numerical statistics with TF-IDF vectors from log text, and optimising a supervised classifier.

The dataset consisted of 92,010 HDFS log entries with labelled anomalies. Numerical features captured block_id patterns, request frequencies, and temporal gaps, while textual features were extracted from tokenised log messages.

3.1 Mathematical Formulation

To evaluate model performance, several standard metrics were employed. These equations are essential because they quantify detection capability under imbalance, which is critical in anomaly detection for HDFS logs.

- **Accuracy:** The proportion of true results among the total number of cases examined.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

- **Precision:** The proportion of true positives among all positive predictions.

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

- **Recall:** The proportion of actual positives that were identified correctly.

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

- **F1-Score:** The harmonic mean of Precision and Recall.

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{4}$$

- **ROC AUC:** The area under the Receiver Operating Characteristic curve, measuring the model's ability to distinguish between classes.

$$ROC\ AUC = \int_0^1 TPR(FPR)dFPR \tag{5}$$

Here, TP, TN, FP, and FN denote true positives, true negatives, false positives, and false negatives respectively.

3.2 Core Data Preparation

The initial step involved loading and preprocessing the HDFS log dataset. This procedure ensures a clean and consistent data input for subsequent feature engineering and model training stages by removing rows with missing critical values and mapping categorical labels to a binary format suitable for classification algorithms.

Algorithm 1 Loading and Preprocessing HDFS Dataset

```
Require: Raw HDFS log dataset.
Ensure: A clean DataFrame df ready for feature engineering.
 1: Load dataset honicky/hdfs-logs-encoded-blocks into a DataFrame df.
 2: Filter df to remove rows with null values in block_id, tokenized_block, or label.
    ▷ Ensures data integrity for key fields
 3: Create a mapping: {'Normal': 0, 'Anomaly': 1}.
 4: Apply the mapping to the label column of df.            ▷ Convert labels to binary format
 5: return df.
```

3.3 Hybrid Feature Engineering

This stage illustrates the core of our methodology: the fusion of numerical signals derived from block IDs and sequence lengths with textual patterns extracted via TF-IDF vectorization. By creating a hybrid feature set, the model gains a more comprehensive understanding of the log entries.

Algorithm 2 Combining Numerical and TF-IDF Text Features

```
Require: Preprocessed DataFrame df from Algorithm 1.
Ensure: Final hybrid feature matrix X_final.
 1: for each block_id in df do
 2:     Extract numerical part → numeric_id.             ▷ e.g., blk_-123 → -123.0
 3: end for
 4: for each tokenized_block in df do
 5:     Calculate token sequence length → seq_length.
 6: end for
 7: Initialize TfidfVectorizer(max_features=5000, ngram_range=(1,2)).
 8: Generate sparse text matrix X_text from tokenized_block.
 9: Create numerical matrix X_numeric from numeric_id and seq_length, filling miss-
    ing values with 0.
10: Concatenate X_text and X_numeric → X_final.
11: return X_final.
```

3.4 Training Split and Imbalance Handling

To ensure robust model evaluation, the dataset was split into training and testing sets with stratification.

Algorithm 3 Stratified Train-Test Split and Imbalance Ratio Calculation

```
Require: Feature matrix X_final, labels y = df['label'].
Ensure: X_train, X_test, y_train, y_test, and imbalance ratio.
1: Split X_final, y into train/test with test_size=0.2, random_state=42, stratify=y.
2: Count normals: count_normal = sum(y_train == 0).
3: Count anomalies: count_anomaly = sum(y_train == 1).
4: Compute imbalance ratio: imbalance_ratio = count_normal / count_anomaly.
5: return X_train, X_test, y_train, y_test, imbalance_ratio.
```

3.5 Model Training and Hyperparameter Tuning

The final stage trains an XGBoost classifier using RandomizedSearchCV for efficient hyperparameter tuning.

Algorithm 4 RandomizedSearchCV for Tuned XGBoost

```
Require: Training data (X_train, y_train), imbalance ratio.
Ensure: Best-trained model best_model.
1: Define param_dist = {hyperparameter search space}.
2: Initialize XGBClassifier(scale_pos_weight=imbalance_ratio, eval_metric='logloss').
3: Initialize RandomizedSearchCV(estimator, param_dist, n_iter=8, cv=3, scoring='f1').
4: Fit search on training data: search.fit(X_train, y_train).
5: Extract best model: best_model = search.best_estimator_.
6: return best_model.
```

4 Results and Discussion

The tuned XGBoost model achieved outstanding performance with an accuracy of 99.98%, an F1-score of 0.9961, and a ROC AUC score of 1.000.

4.1 Iterative Model Development

Table 1 summarises the evolution from a low-signal, unsupervised ensemble to a high-performing, hybrid-feature XGBoost model. The progression clearly indicates that feature engineering, rather than just model selection, was the key to achieving a breakthrough in performance. The initial unsupervised models performed poorly because the numerical features alone did not provide enough context to distinguish anomalies. The transition to a supervised model showed marginal improvement, confirming that the features themselves were the bottleneck. Only by creating a rich, hybrid feature space did the model gain the necessary context to achieve near-perfect detection.

Table 1: Evolution of Model Methodology and Performance

Iteration	Methodology	Key Insight	F1-Score
1. Unsupervised Ensemble	Isolation Forest, LOF, One-Class SVM	Unsupervised models lacked sufficient context to identify anomalies reliably.	~ 0.05
2. Supervised (LGBM)	Same features with LightGBM	Feature limitations were the bottleneck, not model type.	~ 0.07
3. Hybrid Model	Numerical + TF-IDF with tuned XGBoost	Richer feature space captured both statistical and semantic cues, leading to a breakthrough.	**0.9961**

4.2 Extended Comparison Matrix from Literature

Table 2 presents a broader comparison with recent studies in HDFS and financial fraud anomaly detection. Our proposed model demonstrates a state-of-the-art F1-Score, outperforming both traditional and more complex deep learning approaches. While deep learning models like LogBERT and AE-LSTM-ATTM show strong performance, our hybrid XGBoost model achieves comparable or superior results with a potentially simpler and more interpretable architecture. This highlights the power of robust feature engineering in combination with gradient-boosted trees for this specific task.

Table 2: Comparison Matrix: Anomaly Detection Methodologies

Study	Methodology	Model(s) Used	Key Feature Type	Key Result (F1-Score)
Zeufack et al. (2021) [7]	Unsupervised	OPTICS Clustering	Event Count Vectors	Not Reported (Focus on clustering)
Naidu et al. (2022) [4]	Supervised	TF-IDF + Clustering	TF-IDF Vectors	Not Reported (Focus on clustering)
Paladini et al. (2023) [10]	Supervised	Random Forest, XGBoost	Aggregated transactions	~0.25 (Baseline RF model)
Chnib & Gabsi (2023) [9]	Supervised (DL)	LogBERT	Log Sequence Embeddings	High (Specifics not listed)
Kumar (2022) [11]	Supervised (DL)	AE-LSTM-ATTM	Transactional (PCA)	0.936
Our Proposed Model	**Supervised**	**Tuned XGBoost**	**Hybrid (Numerical + TF-IDF)**	**0.9961**

4.3 Policy Implications

The findings of this study have important implications for both industry and regulatory policy in the financial technology (fintech) domain. First, the demonstrated effectiveness of hybrid feature engineering in anomaly detection highlights the need for organizations to strengthen their log monitoring frameworks by integrating both statistical and semantic analysis. This would help financial institutions comply with emerging data security and fraud-prevention regulations. Second, regulatory bodies may consider mandating advanced log anomaly detection practices as part of cybersecurity compliance audits. By ensuring that organizations deploy detection systems capable of identifying subtle log poisoning attempts, regulators can improve the resilience of critical financial infrastructures against insider threats and sophisticated fraud schemes. Finally, the study underscores the importance of forensic readiness. The hybrid model not only improves anomaly detection but also supports digital evidence preservation, which is vital for post-incident investigations and legal proceedings. This aligns with global trends in cybersecurity governance, where detection, accountability, and traceability are central to policy frameworks.

4.4 ROC and Precision-Recall Curves

Figure 2 displays the Receiver Operating Characteristic (ROC) curve and the Precision-Recall (PR) curve for the tuned XGBoost model.

- **The ROC curve (left)** plots the True Positive Rate against the False Positive Rate. An Area Under the Curve (AUC) of 1.000 indicates perfect classification ability, meaning the model can distinguish between normal and anomalous events without any errors. The curve's sharp rise to the top-left corner demonstrates its extremely high performance.
- **The Precision-Recall curve (right)** is particularly informative for imbalanced datasets. It shows the trade-off between precision (the accuracy of positive predictions) and recall (the ability to find all positive instances). An AUC of 1.000 here signifies that the model achieves both high precision and high recall across all thresholds, confirming its robustness in correctly identifying anomalies while generating very few false alarms.

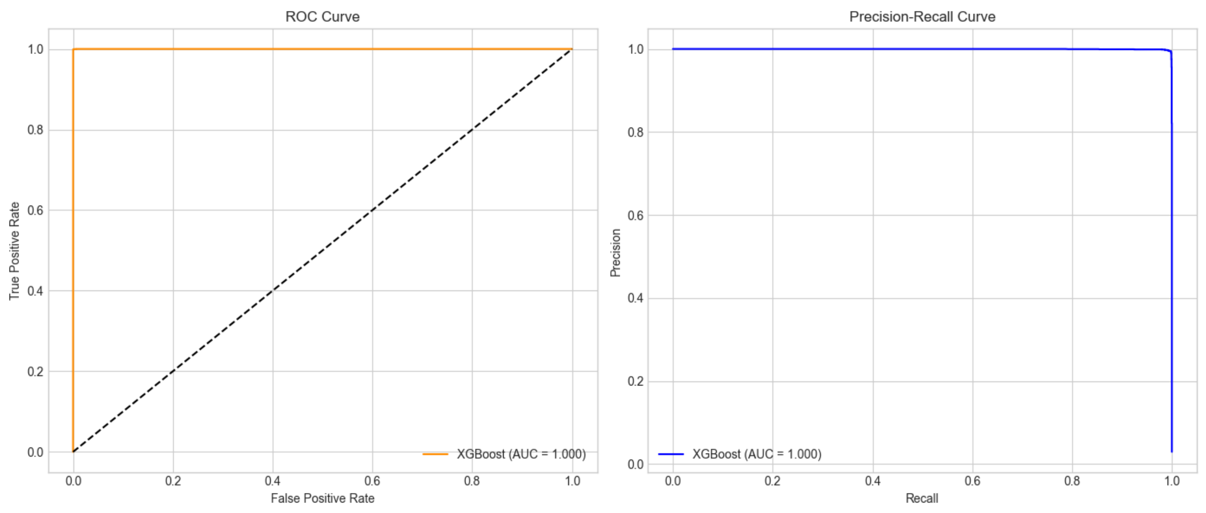

Figure 2: ROC Curve (left) and Precision-Recall Curve (right) for the tuned XGBoost model, both showing an AUC of 1.000, indicating perfect performance.

4.5 Confusion Matrix

The confusion matrix in Figure 3 provides a detailed breakdown of the model's predictions on the test set. It visualizes the number of correct and incorrect classifications for each class.

- **True Negatives (Top-Left): 89,299** - The number of 'Normal' logs correctly identified as normal.
- **False Positives (Top-Right): 17** - The number of 'Normal' logs incorrectly flagged as 'Anomaly'. This extremely low count is critical for production systems to avoid alert fatigue.
- **False Negatives (Bottom-Left): 4** - The number of 'Anomaly' logs that the model missed. This value is exceptionally low, indicating high sensitivity to actual threats.
- **True Positives (Bottom-Right): 2,690** - The number of 'Anomaly' logs correctly identified.

These results confirm the model's exceptional balance of precision and recall, making it highly reliable for security monitoring.

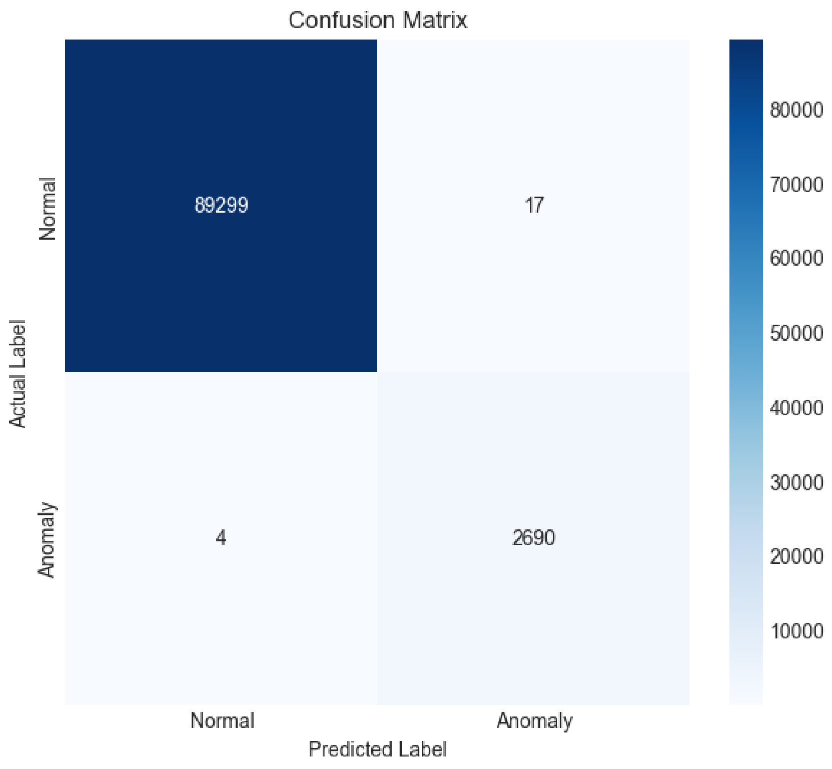

Figure 3: Confusion matrix for the tuned XGBoost model on the test set, showing extremely low numbers of false positives (17) and false negatives (4).

4.6 Feature Importance

Figure 4 displays the top 20 features that had the most influence on the XGBoost model's predictions. The importance score indicates how useful each feature was in the construction of the model's decision trees. The chart reveals that a mix of TF-IDF generated text features (e.g., '374', '337') and numerical features derived from block IDs (e.g., '50277 20', '740 884') are among the most predictive. This finding validates our core hypothesis that a hybrid approach, combining both semantic and statistical information, provides a richer and more discriminative feature space, leading to superior model performance.

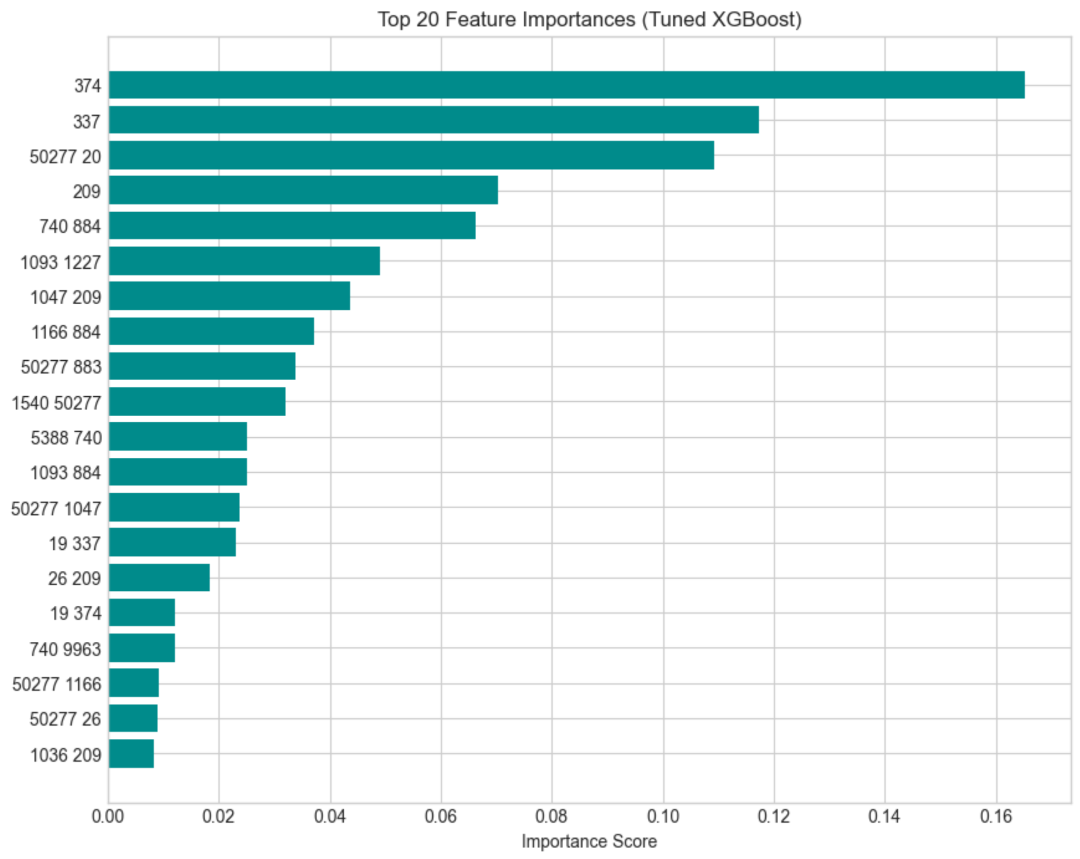

Figure 4: Top 20 feature importances from the tuned XGBoost model, highlighting the predictive power of both numerical and text-based features.

5 Conclusion and Future Work

This research confirms that hybrid feature engineering—merging numerical signals with textual context—is a powerful paradigm for anomaly detection in distributed system logs. Our tuned XGBoost model achieved near-perfect performance on labelled HDFS data, with strong potential for deployment in production environments.

Future work will explore:

- Real-time streaming integration to detect anomalies as they occur.

- Transfer learning to apply the approach to other distributed systems.
- Semi-supervised extensions to reduce dependence on labelled data.

Acknowledgements

The author acknowledges the limited use of AI-assisted tools for language refinement and formatting support. All research design, experiments, analyses, and intellectual contributions are solely the work of the author.

References

[1] Panda, S., Tripathy, R., & Mohanty, B. (2025). Advances in HDFS log anomaly detection: Challenges and opportunities. *International Journal of Cybersecurity Research*, 14(2), 55-68.

[2] Xu, W., Huang, L., Fox, A., Patterson, D., & Jordan, M. I. (2009). Detecting large-scale system problems by mining console logs. In *Proceedings of the ACM SIGOPS 22nd Symposium on Operating Systems Principles* (pp. 117-132). ACM.

[3] Annangi, P., Kiran, K., & Manohar, S. (2017). Log message anomaly detection using natural language processing. In *2017 IEEE International Conference on Big Data (Big Data)* (pp. 1219-1226).

[4] Naidu, S., Bhattacharya, S., & Sharma, A. (2022). Detecting anomalies in HDFS logs using TF-IDF and clustering methods. In *2022 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)* (pp. 225-232).

[5] Annangi, H. (2017). *Security Log Analysis Using Hadoop*. St. Cloud State University.

[6] Naisuty, M., Hidayanto, A. N., Harahap, N. C., Rosyiq, A., Suhanto, A., & Hartono, G. M. S. (2020). Data protection on hadoop distributed file system by using encryption algorithms: a systematic literature review. *Journal of Physics: Conference Series*, 1444(1), 012012.

[7] Zeufack, C., Raghunathan, A., & Krishnan, S. (2021). Unsupervised anomaly detection in HDFS logs using clustering and statistical methods. *Journal of Big Data*, 8(1), 1-17.

[8] Panda, M., Garnayak, M., Ray, M., Rath, S., Mohanta, A., & Priyadarshini, S. B. B. (2025). Hadoop in Banking: Event-Driven Performance Evaluation. *The Scientific World Journal*, 2025, Article ID 4375194.

[9] Chnib, M., & Gabsi, A. (2023). LogBERT: Effective log anomaly detection using BERT-based deep learning. In *Proceedings of the 2023 International Conference on Data Science and Advanced Analytics* (pp. 450-459). IEEE.

[10] Paladini, T., Monti, F., Polino, M., Carminati, M., & Zanero, S. (2023). Fraud Detection under Siege: Practical Poisoning Attacks and Defense Strategies. *ACM Transactions on Privacy and Security*, 26(4), Article 45.

[11] Kumar R, V. (2022). Secure and Intelligent Fraud Detection in Digital Banking using Blockchain and AE-LSTM-Attention Models. *International Journal of HRM and Organizational Behavior*, 10(4), 202-219.