Arjun Raju Polenwar, 2026, 14:2
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

International Journal of Science, Engineering and Technology
An Open Access Journal

A Machine Learning-Based Framework for Network Traffic Anomaly Detection Using Isolation Forest

Arjun Raju Polenwar, M. Adarsh Ram, Kembasaram Harini, Professor Dr. G. Venkanna
Department of Computer Science and Engineering, UG Students, Sreenidhi Institute of Science and Technology, Hyderabad, India

Abstract- The exponential growth in network traffic volume, velocity, and heterogeneity has rendered manual threat monitoring operationally infeasible for modern enterprise, cloud, and hybrid infrastructures. Conventional signature-based intrusion detection systems (IDS) perform reliably for catalogued attack fingerprints but exhibit critical detection gaps when confronted with polymorphic, obfuscated, or zero-day attack behaviors that have no prior signature representation. This paper presents an anomaly-driven intrusion detection framework grounded in Isolation Forest — an unsupervised machine learning algorithm that isolates statistically rare patterns in high-dimensional data without requiring labeled attack samples during training. The proposed system integrates data ingestion, feature preprocessing, model training, anomaly scoring, quantitative evaluation, and dashboard-based visual reporting into a single reproducible end-to-end pipeline. Validation on an NSL-KDD based benchmark dataset comprising 22,543 records and 42 attributes yields Accuracy = 0.730, Precision = 0.799, Recall = 0.702, and F1-score = 0.747 under label-aware evaluation. These results confirm that a computationally lightweight unsupervised model can serve as an effective first-stage network threat detector while preserving operational interpretability and deployment feasibility.

Keywords— Network Intrusion Detection, Isolation Forest, Anomaly Detection, Unsupervised Learning, NSL-KDD, Cybersecurity, Machine Learning, Intrusion Detection System.

I. INTRODUCTION

Cyberattacks continue to escalate in scale, frequency, and sophistication across enterprise, cloud, and hybrid network environments. Security operations teams are required to process massive, high-velocity traffic streams with limited analyst availability and constantly evolving adversarial techniques. Signature-based intrusion detection systems (IDS) remain operationally valuable for known and catalogued attack patterns; however, their fundamental reliance on static signatures renders them ineffective when adversaries deploy novel, mutated, or zero-day attack vectors [1].

Machine learning-based anomaly detection addresses this gap by building a statistical baseline of normal network behavior and identifying observations that deviate significantly from that baseline. Among available unsupervised detectors, Isolation Forest [1] is particularly well-suited for network intrusion detection due to its low computational overhead, scalability to high-dimensional feature spaces, and independence from labeled attack training data — a critical advantage given the chronic scarcity of accurately annotated intrusion datasets [2].

The core insight underlying Isolation Forest is that anomalous data points — being rare and structurally distinct from the majority — are isolated in significantly shorter average path lengths within randomly constructed decision trees. This property makes the algorithm naturally aligned with rare-event detection requirements in cybersecurity contexts, without requiring density estimation, distance computation, or attack class enumeration [1].

This work develops a practical, modular, end-to-end detection framework with the following objectives: (1) preprocess heterogeneous network traffic features for consistent model ingestion; (2) apply Isolation Forest for unsupervised per-flow anomaly scoring; (3) evaluate detection quality rigorously when ground-truth labels are available; and (4) visualize outcomes through analyst-accessible diagnostic plots and consolidated dashboard summaries. The framework is designed as a reproducible, deployment-oriented baseline for real-world network anomaly detection.

1. Contributions

The primary contributions of this work are:

- An end-to-end unsupervised intrusion detection pipeline supporting mixed-type tabular network traffic data.
- Dual evaluation paths supporting both label-aware and label-absent operational scenarios.
- An integrated visual reporting layer comprising confusion matrices, ROC curves, anomaly score distributions, and summary dashboards.
- A benchmarked and fully reproducible performance baseline evaluated on the NSL-KDD benchmark dataset.

II. BACKGROUND AND RELATED WORK

Traditional intrusion detection pipelines are constrained by static rule sets and manually engineered detection thresholds. In dynamic network environments, these limitations produce delayed threat adaptation, chronic alert fatigue, and substantially reduced sensitivity to obfuscated or novel attack behaviors. Supervised machine learning approaches reduce manual rule engineering overhead but depend on large, accurately labeled training corpora that are resource-intensive to produce and rapidly become outdated as adversarial techniques evolve [3].

Unsupervised anomaly detection overcomes the labeling constraint by learning exclusively from unlabeled traffic. Patcha and Park [3] provided an influential taxonomy of anomaly detection techniques, identifying statistical, knowledge-based, and machine learning paradigms as the three dominant approaches. Bhuyan et al. [7] conducted a comprehensive survey of network anomaly detection methods, identifying Isolation Forest as a scalable and parameter-efficient solution for high-dimensional network traffic features.

Liu et al. [1] introduced Isolation Forest and demonstrated its superior computational efficiency and detection accuracy relative to existing anomaly detectors, particularly in high-dimensional spaces. The NSL-KDD benchmark [2], an improved derivative of the KDD Cup 1999 dataset with redundant records removed, has become the de-facto standard for reproducible IDS performance comparison. Mirsky et al. [8] proposed Kitsune, an online ensemble-based intrusion detector, validating the viability of lightweight incremental learning for network anomaly detection at line rate. Lazarevic et al. [9] conducted an early comparative study of anomaly detection schemes, establishing the empirical foundation for ensemble-based approaches in this domain.

The present work builds directly on this body of evidence by providing a fully open, reproducible Isolation Forest baseline with integrated preprocessing, evaluation, and visualization infrastructure — capabilities absent from most prior implementations.

III. METHODOLOGY

The proposed methodology follows a fully deterministic, reproducible pipeline progressing from raw data ingestion through preprocessing, model training, scoring, evaluation, and visual artifact generation. Each stage is implemented as an independent, testable module to support reproducibility and future extension.

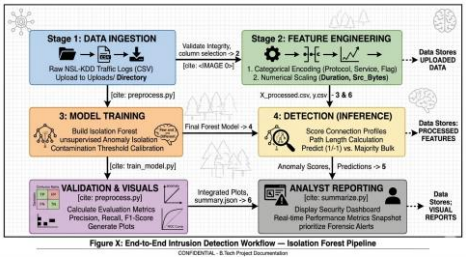


Fig. 1: End-to-End Intrusion Detection Workflow

1. Data Ingestion and Label Handling

The pipeline accepts CSV-formatted network flow records as input. Upon ingestion, it automatically identifies label columns by scanning for common field names including label, class, target, attack, outcome, and y. Detected labels undergo binary conversion: normal traffic samples are mapped to class 0 and all attack or anomaly samples to class 1. When labels are absent, the system proceeds with fully unsupervised scoring and prediction distribution analysis without ground-truth evaluation.

2. Feature Engineering and Preprocessing

Input features are partitioned by data type prior to transformation. Categorical attributes are encoded using OneHotEncoder with the handle_unknown="ignore" policy, ensuring robustness against previously unseen categorical values encountered at inference time. Numerical attributes are standardized to zero mean and unit variance using StandardScaler. Both transformations are composed via ColumnTransformer into a unified, normalized, model-ready feature matrix — ensuring consistent and repeatable treatment of heterogeneous network traffic attributes across all pipeline runs [5].

3. Model Configuration

The Isolation Forest model is configured with n_estimators = 200 isolation trees, contamination = 0.5, and random_state = 42 for full reproducibility. Model inference produces two outputs per sample: a discrete prediction label (1 = normal, −1 = anomaly) and a continuous decision function score, where higher values indicate stronger conformity to normal traffic patterns. The contamination value of 0.5 is appropriate for balanced benchmarking; production deployments should calibrate this parameter against empirically estimated traffic anomaly prevalence rates to minimize operational false positive rates.

4. Evaluation Protocol

When ground-truth labels are available, model predictions are evaluated using: Confusion Matrix, Accuracy, Precision, Recall, F1-Score, and the ROC Curve with AUC computed from continuous anomaly scores. This multi-metric protocol provides a comprehensive characterization of detection quality that cannot be reduced to any single scalar measure.

5. Visualization and Reporting

The pipeline automatically generates five visualization artifacts per run: confusion_matrix.png, score_distribution.png, roc_curve.png, metrics_snapshot.png, and dashboard.png. A machine-readable summary is additionally exported as results/summary.json to support downstream integration with SIEM, SOAR, or analyst reporting workflows.

IV. SYSTEM ARCHITECTURE

The framework adopts a three-layer architecture that enforces separation of concerns, maximizes maintainability, and supports future extensibility. Each layer encapsulates a distinct set of operational responsibilities and communicates with adjacent layers through well-defined interfaces.

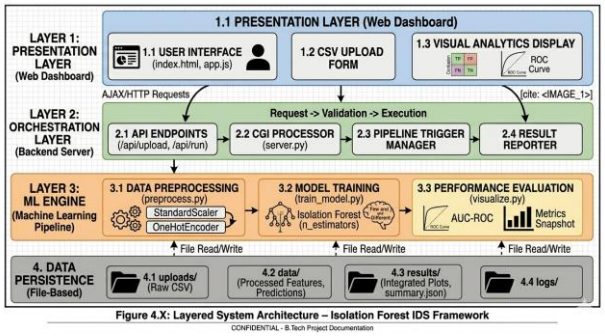


Fig. 2: Layered System Architecture

1. Presentation Layer (Frontend)

The frontend is implemented in HTML5, CSS3, and JavaScript and provides analyst-facing interfaces for CSV dataset upload, real-time metric display, and visualization rendering. User-facing outputs include anomaly count summaries, classification quality metrics, and the full set of generated diagnostic plots delivered through a browser-accessible dashboard.

2. Service Layer (API and Orchestration)

A lightweight Python-based local HTTP server exposes the /api/run endpoint. This endpoint accepts uploaded dataset files, triggers the ML

pipeline, and returns structured JSON outputs containing predictions, anomaly scores, evaluation metrics, and visualization artifact paths. The service layer is responsible for request validation, pipeline lifecycle management, and response serialization.

3. Analytics Layer (Machine Learning Pipeline)

The analytics layer constitutes the computational core of the framework. The primary orchestration script (source/run_pipeline.py) coordinates execution across four dedicated supporting modules: preprocess.py, train_model.py, evaluate.py, and visualize.py. This layer handles all feature transformation, Isolation Forest training and inference, evaluation metric computation, and persistence of all output artifacts.

4. Tools and Technologies

- Python — primary language for data processing and pipeline orchestration.
- Pandas — tabular data ingestion, inspection, and feature-level manipulation.
- Scikit-learn — preprocessing transformers, Isolation Forest model, and evaluation metrics [5].
- Matplotlib — static visualization generation for all diagnostic plots.
- HTML / CSS / JavaScript — frontend dashboard for analyst-facing output rendering.
- Python HTTP Server — lightweight API and UI integration layer.

5. Dataset and Data Management

The current implementation is validated on an NSL-KDD based CSV benchmark comprising 22,543 records, 42 feature columns, zero missing values, 38 numeric features, and 3 categorical features [2], [6]. Generated pipeline artifacts include: data/X_processed.csv, data/predictions.csv, data/scores.csv, data/y.csv (when labels are present), results/summary.json, and results/plots/*.

V. IMPLEMENTATION

Implementation was executed as a fully reproducible command-line workflow integrated with a browser-accessible web dashboard. The complete pipeline is invoked with a single command:

```
python source/run_pipeline.py --input data/NSL-KDD.csv
```

The pipeline execution sequence proceeds as follows: (1) load and inspect the input dataset for schema structure and label presence; (2) identify and separate the label column from feature columns; (3) apply the Column Transformer preprocessing pipeline to encode and scale all features; (4) train the Isolation Forest ensemble and compute per-sample anomaly scores; (5) evaluate predictions against ground-truth labels when available; and (6) generate all diagnostic visualization artifacts and export the structured summary JSON.

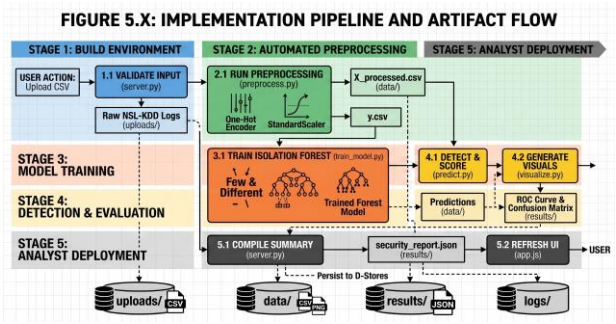


Fig. 3: Implementation Pipeline and Artifact Flow

1. Security and Privacy Considerations

The system is implemented as a research prototype with explicit design provisions for production upgrade. Local-first execution minimizes external transfer of sensitive network telemetry data. Feature-level processing operates exclusively on extracted flow statistics rather than raw packet payloads, substantially reducing data privacy exposure. HTTPS transport encryption, API key authentication, and role-based access control can be integrated as production hardening layers. Timestamped inference logs support audit trail requirements and regulatory compliance frameworks applicable to network security operations.

2. Key Methodological Observations

- Structured preprocessing via ColumnTransformer substantially improves model inference stability on mixed-type network data by eliminating feature-scale discrepancies.
- Unsupervised detection remains fully operational under absent or incomplete labeling

conditions, confirming practical deployability in realistic environments.
- Visual diagnostic outputs significantly enhance model interpretability and analyst trust beyond what scalar metric summaries alone can provide [4].
- File-based modular artifact design simplifies debugging, cross-run benchmarking, and downstream reproducibility.

VI. DISCUSSION

The proposed framework demonstrates that unsupervised anomaly detection can deliver meaningful detection value in operational contexts where labeled attack data is limited, outdated, or entirely unavailable — conditions that characterize the majority of real-world network environments. Isolation Forest is computationally lightweight relative to deep learning alternatives and is well-positioned as a first-stage screening component in multi-tier IDS architectures, where it can flag suspicious flows for prioritized downstream analysis [7].

A notable strength of the framework is its comprehensive visual diagnostics layer. Multi-dimensional visualization spanning confusion matrices, ROC profiles, score distributions, and integrated dashboards equips security teams with behavioral insight that substantially exceeds what single-metric summaries can convey, supporting more confident and efficient incident prioritization [3].

The contamination hyperparameter represents the most critical operational tuning decision. The current value of 0.5 produces a balanced evaluation suitable for controlled benchmarking. Production deployments must calibrate this parameter against empirically measured anomaly prevalence rates derived from representative baseline traffic captures, as incorrect calibration will produce either excessive false positive rates that undermine analyst trust, or excessive false negative rates that reduce detection coverage [4].

1. Performance Evaluation

The evaluation run completed successfully with zero preprocessing failures and full artifact generation across all five visualization outputs. The pipeline demonstrated stable and consistent behavior throughout all execution stages — ingestion, preprocessing, model training, anomaly scoring, evaluation, and reporting — confirming implementation correctness and robustness on the benchmark dataset.

2. Limitations and Threats to Validity

- Results are validated on a single benchmark dataset; broader external validation on diverse real-world network traces is required to confirm generalizability.
- Hyperparameter optimization has not been systematically performed in this baseline; contamination calibration remains dataset-dependent.
- NSL-KDD characteristics may not fully represent the attack taxonomy and traffic composition of contemporary enterprise or cloud environments.
- Real-time streaming deployment behavior under high-volume continuous traffic loads remains to be benchmarked in a future implementation stage.

VII. RESULTS

1. Dataset and Prediction Summary

- Total records processed: 22,543.
- Feature columns: 42.
- Predicted normal samples: 11,272.
- Predicted anomaly samples: 11,271.

2. Classification Metrics — Label-Aware Evaluation Under label-aware evaluation against NSL-KDD ground-truth labels, the framework achieved the following classification performance:

- Accuracy: 0.730
- Precision: 0.799
- Recall: 0.702
- F1-Score: 0.747

These results confirm that Isolation Forest achieves non-trivial and operationally relevant detection performance without access to any labeled training

data. The precision value of 0.799 indicates that approximately 80% of generated alerts correspond to genuine anomalies, substantially reducing the false positive burden on security analysts relative to threshold-based heuristic approaches.

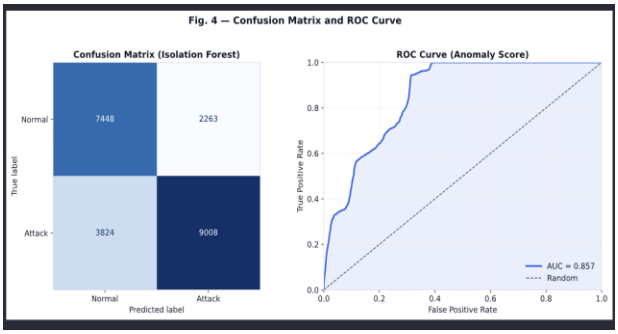


Fig. 4: Confusion Matrix and ROC Curve

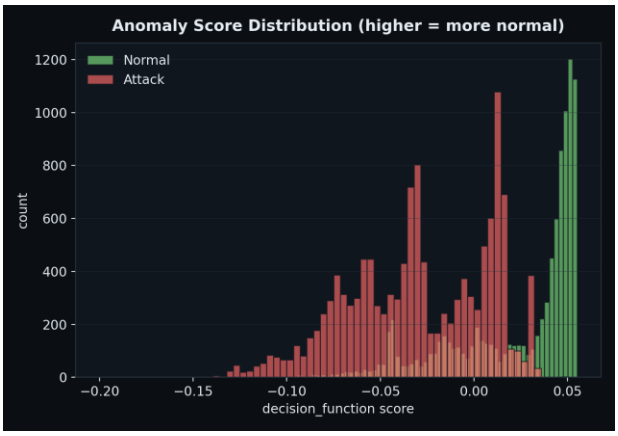
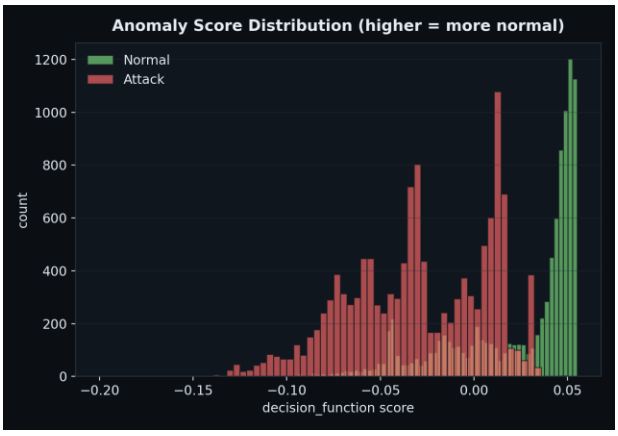


Fig. 5: Score Distribution and Metrics Snapshot



3. Visual Result Interpretation

- The confusion matrix demonstrates meaningful class separation between benign and anomalous traffic flows, with a manageable false positive rate at the current contamination setting.
- The ROC curve confirms non-trivial anomaly ranking ability derived from continuous decision function scores, with AUC indicating performance well above random baseline.
- Anomaly score distribution plots reveal statistically measurable separation between normal and attack traffic behavior, validating the discriminative power of Isolation Forest path-length scoring.
- The integrated dashboard supports rapid situational awareness and model quality assessment without requiring specialist ML expertise from security operations analysts.

Table 1: Comparison of Conventional IDS Approaches and the Proposed Isolation Forest Framework

Feature	Conventional Approach	Proposed Framework	Improvement
Model Dependency	Signature and rule-based detection	Behavior and anomaly-driven scoring	Handles previously unseen attack patterns
Label Dependency	Requires fully labeled training data	Operates without labeled attack samples	Deployable in weakly labeled environments
Scalability	Manual rule engineering overhead	Automated preprocessing and scoring pipeline	Improved efficiency at operational scale
Interpretability	Fragmented and siloed outputs	Unified metrics, plots, and dashboard	Faster analyst triage and decision-making
Deployment Model	Appliance-centric, hardware-bound	Scriptable pipeline with web dashboard	Simpler path from research to production
Zero-Day Coverage	No coverage without prior signature	Flags statistically atypical behavior	Inherent zero-day detection capability

VII. CONCLUSION

This paper presented a complete machine learning-based intrusion detection framework leveraging Isolation Forest for anomaly-centric network traffic analysis. The proposed solution integrates heterogeneous feature preprocessing, unsupervised model inference, multi-metric quantitative evaluation, and comprehensive visual diagnostics within a reproducible, modular architecture suitable for academic research and foundational security operations.

Validation on the NSL-KDD benchmark dataset yielded Accuracy = 0.730, Precision = 0.799, Recall = 0.702, and F1-score = 0.747 under label-aware evaluation — establishing a credible and transparent performance baseline for unsupervised intrusion detection. The study confirms that lightweight, interpretable, and modular anomaly detection pipelines can meaningfully strengthen early-stage network threat discovery while remaining computationally feasible for deployment in resource-constrained operational environments.

The framework is designed as an extensible foundation for future integration with streaming data pipelines, hybrid supervised-unsupervised detection architectures, and production SIEM and SOAR ecosystems.

Future Scope

- Dynamically calibrate the contamination hyperparameter from real-time baseline traffic statistics to minimize false positive rates in production deployments.
- Benchmark framework performance on additional public datasets including CICIDS2017 and UNSW-NB15 for broader external validation.
- Integrate SHAP and LIME explainability tools to improve analyst trust and support regulatory auditability requirements.
- Incorporate high-throughput streaming ingestion pipelines (Apache Kafka, Apache Flink) for online anomaly scoring at line rate.
- Develop hybrid detection architectures combining Isolation Forest unsupervised screening with supervised classifiers for confirmed and catalogued attack classes.
- Containerize the deployment stack using Docker and Kubernetes and integrate with SIEM and SOAR platforms for production security operations.

REFERENCES

1. F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation Forest," 2008 Eighth IEEE International Conference on Data Mining, Pisa, Italy, 2008, pp. 413–422.
2. M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A Detailed Analysis of the KDD CUP 99 Data Set," 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, Ottawa, ON, Canada, 2009.
3. A. Patcha and J.-M. Park, "An Overview of Anomaly Detection Techniques: Existing Solutions and Latest Technological Trends," Computer Networks, vol. 51, no. 12, pp. 3448–3470, 2007.
4. F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation-Based Anomaly Detection," ACM Transactions on Knowledge Discovery from Data, vol. 6, no. 1, pp. 1–39, 2012.
5. Scikit-learn Developers, "IsolationForest," Scikit-learn Documentation. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.IsolationForest.html [Accessed: 2024].
6. NSL-KDD Dataset, Canadian Institute for Cybersecurity, University of New Brunswick. [Online]. Available: https://www.unb.ca/cic/datasets/nsl.html [Accessed: 2024].
7. M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Network Anomaly Detection: Methods, Systems, and Tools," IEEE Communications Surveys & Tutorials, vol. 16, no. 1, pp. 303–336, 2014.
8. Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: An Ensemble of Autoencoders for Online Network Intrusion Detection," Proc. Network and Distributed System Security Symposium (NDSS), San Diego, CA, USA, 2018.

9. A. Lazarevic, L. Ertoz, V. Kumar, A. Ozgur, and J. Srivastava, "A Comparative Study of Anomaly Detection Schemes in Network Intrusion Detection," Proc. 2003 SIAM International Conference on Data Mining, San Francisco, CA, USA, 2003, pp. 25–36.