A Comparative Analysis of Machine Learning Classifiers for Network Infiltration Detection

Author: Mukesh T

Institution: Collaborative Research Initiative

Date: March 11, 2024

Abstract

The escalating sophistication of cyber threats necessitates a paradigm shift from traditional signature-based intrusion detection systems to more dynamic, data-driven approaches. This paper presents a comprehensive empirical study on the application of machine learning for network infiltration detection. We investigate the binary classification of network log entries as either "Benign" or "Infiltration" by leveraging fundamental traffic attributes such as port, protocol, and flow duration. Two canonical yet fundamentally different classification algorithms, Logistic Regression (a linear model) and the Decision Tree Classifier (a non-linear model), are implemented and subjected to a rigorous comparative evaluation. The methodology encompasses a complete machine learning pipeline, including data preprocessing, model training, and performance validation using a suite of metrics (Accuracy, Precision, Recall, F1-Score) derived from the confusion matrix. Our experimental results reveal the superior efficacy of the Decision Tree Classifier, which achieved an accuracy of 92.3% and a recall of 0.90, significantly outperforming Logistic Regression. This performance disparity underscores the inherently non-linear characteristics of the feature space in network intrusion detection. The paper concludes with a critical discussion of these findings, an analysis of the study's limitations, and a proposed roadmap for future work, emphasizing the integration of Explainable AI (XAI) and a robust MLOps framework for operationalization.

Keywords: *Network Security, Intrusion Detection, Machine Learning, Logistic Regression, Decision Trees, MLOps, Cybersecurity.*

1. Introduction

The proliferation of interconnected digital ecosystems has rendered network security a cornerstone of modern organizational strategy. However, the threat landscape is in a state of perpetual evolution, with malicious actors continuously devising novel attack vectors that circumvent traditional security measures. Signature-based Intrusion Detection Systems (IDS), while effective against known threats, are fundamentally reactive and often fail to identify zero-day exploits and polymorphic attacks. This limitation has catalyzed research into Anomaly-based IDS, with machine learning (ML) emerging as a particularly promising frontier.

By learning complex patterns directly from network traffic data, ML models can develop a nuanced understanding of "normal" behavior and, by extension, identify subtle deviations indicative of malicious activity. This research contributes to this domain by conducting a foundational comparative analysis between two distinct classes of supervised learning algorithms for the task of network infiltration detection.

The core research question guiding this study is: *To what extent do linear versus non-linear models differ in performance and practical utility for the classification of network infiltration attempts?* We hypothesize that the complex, high-dimensional nature of network traffic data contains non-linear relationships that will be better captured by non-linear models.

Contributions of this paper:

1. An empirical evaluation of Logistic Regression and a Decision Tree Classifier on a real-world network log dataset.
2. A detailed analysis of performance trade-offs, with a specific focus on the importance of the recall metric in a security context.
3. A forward-looking discussion on the operationalization of such models through a structured MLOps framework.

This paper is organized as follows: Section 2 reviews related work. Section 3 details the methodology. Section 4 presents the experimental results and analysis. Section 5 discusses the findings and limitations. Finally, Section 6 concludes and outlines future research directions.

2. Related Work

The application of machine learning to network security is a well-established field. Early research focused on statistical methods and classical algorithms such as Naïve Bayes and K-Nearest Neighbors for traffic classification. More advanced techniques, including Support Vector Machines (SVMs), were later shown to be highly effective by identifying an optimal separating hyperplane in the feature space.

More recently, the focus has shifted towards ensemble methods. Techniques like Random Forests, which construct a multitude of decision trees to mitigate overfitting and improve generalization, have demonstrated state-of-the-art performance on various intrusion detection benchmarks. Similarly, Gradient Boosting machines have proven highly effective. This project situates itself within this body of work by revisiting two foundational models to establish a clear performance baseline and explicitly investigate the impact of model linearity on this specific classification task.

3. Methodology

3.1. Dataset and Feature Description

The study utilizes a dataset of system network logs containing labeled examples of benign and infiltration traffic. The following features were selected for their established relevance in intrusion detection:

- **'Dst Port'**: The destination port, which often indicates the target application and can signal reconnaissance scans or attacks on specific services.
- **'Protocol'**: The network protocol (e.g., TCP, UDP), which governs data transmission rules.
- **'Flow Duration'**: The temporal length of the connection. Unusually short or long durations can be anomalous.
- **'Tot Bwd Pkts'**: The total number of backward packets, which can be indicative of data exfiltration.
- **'ACK Flag Cnt' & 'PSH Flag Cnt'**: The frequency of specific TCP flags, which can reveal patterns related to network scanning or specific exploits.

3.2. Data Preprocessing Pipeline

A standardized preprocessing pipeline was designed to ensure data quality and model compatibility.

1. **Missing Value Imputation**: Rows with missing feature values were expunged from the dataset.
2. **Label Encoding**: The binary categorical target variable ('Benign', 'Infiltration') was encoded into a numerical format (0, 1).
3. **Feature Scaling**: The StandardScaler from Scikit-learn was employed to transform all features to have a mean of zero and a standard deviation of one. This is crucial for models like Logistic Regression that are sensitive to the scale of input features.
4. **Data Splitting**: The dataset was partitioned into a 70% training set and a 30% testing set.

3.3. Algorithmic Framework

3.3.1. Logistic Regression
Logistic Regression is a parametric, linear classification algorithm. It models the probability of a binary outcome by applying the logistic (sigmoid) function to a linear combination of the input features. The predicted probability is given by:
$$ P(Y=1|X) = \sigma(z) = \frac{1}{1 + e^{-z}} $$
where $z=\beta_0+\sum_{i=1}^{n}\beta_i X_i$. The coefficients (β) are learned by minimizing a cost function, typically the Log Loss (Binary Cross-Entropy), using optimization techniques like gradient descent.
3.3.2. Decision Tree Classifier
A Decision Tree is a non-parametric algorithm that creates a hierarchical, tree-like structure of decision rules. The tree is constructed via recursive partitioning, where each node splits the data based on the feature and threshold that yields the highest Information Gain, which is a measure of the reduction in impurity. This study uses the Gini Impurity as the splitting

criterion:
Gini(S) = 1 - \sum_{i=1}^{c} p_i^2 $$
Where pi is the proportion of samples belonging to class i at a given node. The algorithm aims to create terminal nodes (leaves) that are as "pure" as possible.

3.4. Evaluation Framework

Model performance was evaluated using metrics derived from the confusion matrix.

- **Accuracy**: The proportion of total correct predictions.
- **Precision**: The ability of the model not to label a negative sample as positive. Precision=TP+FPTP.
- **Recall (Sensitivity)**: The ability of the model to find all the positive samples. Recall=TP+FNTP. This is often the most critical metric in security, as it directly measures the model's ability to detect threats (minimizing false negatives).
- **F1-Score**: The harmonic mean of Precision and Recall. F1=2×Precision+RecallPrecision×Recall.

4. Experimental Results & Analysis

4.1. Quantitative Results

The performance of the trained models on the unseen test set is summarized in Table 1.

Table 1: Performance Metrics of Classification Models

Metric	Logistic Regression	Decision Tree Classifier
Accuracy	85.6%	92.3%
Precision	0.88	0.94
Recall (Sensitivity)	0.82	0.90
F1-Score	0.85	0.92

4.2. Comparative Analysis

The results clearly indicate that the Decision Tree Classifier significantly outperforms the Logistic Regression model across all evaluated metrics. The most critical distinction lies in the **Recall** score (0.90 vs. 0.82), which translates to the Decision Tree successfully identifying a substantially higher percentage of actual infiltration attempts.

This performance gap can be attributed to the inherent linearity of Logistic Regression. It constructs a single linear decision boundary in the feature space. The results strongly suggest that the boundary separating benign and malicious traffic is complex and non-linear. The Decision Tree, through its recursive partitioning, is capable of learning a much more complex, piece-wise decision boundary that better fits the underlying data structure, leading to superior classification performance.

5. Discussion

5.1. Limitations of the Study

This study is subject to several limitations. First, the analysis was conducted on a single, static dataset; the models' generalizability to different network environments is unverified. Second, the study was limited to two models; more advanced ensemble or deep learning models may yield further performance improvements. Finally, the system does not account for **concept drift**, where the statistical properties of network traffic and attack vectors change over time.

5.2. MLOps and Future Enhancements

Transitioning this research into a production-grade system requires a robust MLOps strategy.

- **Productionalization Strategy**: A future system should incorporate a CI/CD pipeline for automated testing and deployment, a model registry (e.g., MLflow) for versioning, and a monitoring framework to track performance and detect concept drift. An automated retraining pipeline would be essential to keep the model current with the evolving threat landscape.
- **Advanced Research Directions**:
    1. **Model Ensembling**: Future work should prioritize the implementation of Random Forest or Gradient Boosting classifiers to enhance robustness.
    2. **Explainable AI (XAI)**: Integrating frameworks like SHAP or LIME is crucial for providing human-interpretable explanations for model predictions, enabling security analysts to trust and act upon the system's alerts.
    3. **Real-Time Streaming Architecture**: The ultimate goal is to develop a scalable, real-time system using a streaming architecture (e.g., Apache Kafka and Spark Streaming) to provide immediate threat intelligence.

6. Conclusion

This paper presented a comparative analysis of Logistic Regression and a Decision Tree Classifier for network infiltration detection. Our findings demonstrate that the non-linear Decision Tree model is significantly more effective for this task, achieving superior performance across all metrics, most notably a recall of 0.90. This result underscores the importance of selecting models that can capture the inherent complexity of cybersecurity data. While foundational, this work validates the efficacy of a data-driven approach to intrusion detection and provides a clear roadmap for future research, focusing on more advanced models and the critical integration of MLOps for creating robust, real-world security solutions.

7. References

- Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.
- Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: Data mining, inference, and prediction*. Springer Science & Business Media.
- Scikit-learn Documentation: https://scikit-learn.org/stable/documentation.html
- Towards Data Science: https://towardsdatascience.com/