

**DEVELOPMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING
BASED CYBERSECURITY TECHNOLOGIES FOR DETECTING THREATS,
PROTECTING SENSITIVE GOVERNMENT DATA, AND STRENGTHENING THE
RESILIENCE OF DIGITAL INFORMATION INFRASTRUCTURE**

Rasel Hossain Babu¹, Shafi Alam², Md Zahid Hassan³, Md Asaduzzaman Nopu⁴ & Md
Shadman Soumik⁵

¹Master of Science in Cybersecurity, Department of Information Sciences and Engineering,
Bay Atlantic University, Washington, D.C., USA. Email: raselhossainbabu5@gmail.com;
rbabu@stu.bau.edu

²Master of Science in Big Data Analytics, Department of Information Sciences and
Engineering, Bay Atlantic University, Washington, D.C., USA. Email:
shafialam001@gmail.com; salam@stu.bau.edu

³Master of Science in Cybersecurity, Department of Information Sciences and Engineering,
Bay Atlantic University, Washington, D.C., USA. Email: Zahid.hassan.msc@gmail.com
mdhassan@stu.bau.edu

⁴Master of Science in Cybersecurity, Department of Information Sciences and Engineering,
Bay Atlantic University, Washington, D.C., USA. Email: asaduzzamannopu@gmail.com
mnopu@stu.bau.edu

⁵Master of Science in Information Technology, Department of Information
Technology (MSIT), Washington University of Science and Technology (WUST), 2900
Eisenhower Ave, Alexandria, VA 2231, USA. Email: msoumik.student@wust.edu

Abstract:

Government agencies that hold sensitive data, and the digital information infrastructure that supports them, face a steady increase in the volume, speed, and variety of cyber threats, and manual analysis can no longer keep pace with the telemetry that modern networks produce. Machine learning (ML) is widely proposed as a way to detect intrusions, malware, and anomalous behavior at scale, yet the literature often reports accuracy figures that do not translate into operational value because they ignore class imbalance, unequal error costs, and the fragility of models under changing conditions. This educational article presents a framework for developing and evaluating AI and ML-based threat detection technologies in settings where sensitive data must be protected and service continuity matters. The framework combines standard classification metrics with a cost-sensitive decision model, a threshold-selection rule derived from Bayes decision theory, and a simple robustness protocol based on feature noise. To demonstrate the framework without using any real organizational data, we generated a synthetic imbalanced dataset of 20,000 records with approximately ten percent malicious samples and compared a logistic regression baseline, a random forest, a gradient boosting classifier, a multilayer perceptron, and an unsupervised isolation forest. In this synthetic setting the multilayer perceptron achieved the highest F1 score (0.792), the random forest achieved the highest precision (0.961), and tuning the decision threshold of the random

forest to minimize expected cost reduced that cost by roughly 63 percent relative to the default threshold of 0.5. Performance of all supervised models degraded as input noise increased. These results should not be generalized to real traffic; their purpose is to show how the evaluation pipeline works and what questions practitioners should ask of any reported detector.

Keywords: artificial intelligence, machine learning, cybersecurity, threat detection, government data protection, intrusion detection, class imbalance, cost-sensitive learning, anomaly detection, resilience

1. INTRODUCTION

Government agencies, hospitals, utilities, and financial institutions depend on digital information infrastructure whose compromise can expose personal records, interrupt essential services, and erode public trust. Defenders of these systems must monitor large volumes of network flows, authentication events, endpoint logs, and application traces, and they must do so while attackers continually adapt their methods. Signature-based tools remain valuable for recognizing known threats, but they are poorly suited to novel or obfuscated attacks. This gap has motivated a large body of research that applies machine learning to cybersecurity tasks, including intrusion detection, malware classification, phishing detection, and user behavior analytics (MahdaviFar & Ghorbani, 2019; Sarker et al., 2020).

Despite this activity, the transition from laboratory results to dependable operational tools has been difficult. Sommer and Paxson (2010) argued that anomaly detection in security faces a fundamentally different problem from the one faced in many other ML applications, because the cost of errors is asymmetric, the data distribution shifts over time, and meaningful ground truth is scarce. More recently, Arp et al. (2022) catalogued recurring pitfalls in published security-ML research, such as sampling bias, data snooping, inappropriate baselines, and metrics that do not reflect deployment conditions. Reviews of intrusion detection with ML similarly note that many studies rely on a small number of dated benchmark datasets and report high aggregate accuracy that can mask poor performance on rare attack classes (Dina & Manivannan, 2021; Ferrag et al., 2020).

Public-sector systems raise the stakes of this problem. Government networks hold tax, health, immigration, law-enforcement, and defense-related records whose disclosure can cause lasting harm to individuals and to national interests, and they are persistent targets for criminal groups and state-linked actors. These systems also tend to combine modern cloud services with long-lived legacy components, which widens the attack surface and complicates monitoring. Technologies for detecting threats in this setting must therefore satisfy requirements beyond raw accuracy: they must operate under strict data-governance rules, support auditable decisions, tolerate changing conditions, and contribute to the ability of the infrastructure to keep serving citizens during and after an attack.

The present article takes a deliberately modest approach to this challenge. Rather than claiming a new detection algorithm, it describes a development and evaluation framework that practitioners and students can reuse, and it illustrates that framework with a fully synthetic experiment whose every number can be reproduced from a fixed random seed. The framework is organized around three principles. First, performance must be reported with metrics that

remain meaningful under strong class imbalance. Second, the decision threshold of a detector is a policy choice that should follow from the relative costs of missed attacks and false alarms. Third, a detector's behavior under perturbation should be examined before it is trusted in an environment that attackers can influence.

The contributions of the article are as follows. We formalize the detection problem as cost-sensitive binary classification and derive the optimal decision rule for calibrated probabilities. We describe a reproducible experimental protocol, including data generation, preprocessing, model training, and evaluation. We present results for five representative models on synthetic data, and we discuss why those results, while instructive, cannot substitute for evaluation on real telemetry. Finally, we outline limitations and directions for responsible future work, including validation on public intrusion-detection corpora such as NSL-KDD and CIC-IDS2017 (Sharafaldin et al., 2018; Tavallaei et al., 2009).

The remainder of the article is organized as follows. Section 2 reviews related work. Section 3 formalizes the problem and introduces the mathematical model. Section 4 describes the data and experimental setup. Section 5 reports results. Section 6 discusses implications for protecting sensitive data and strengthening resilience. Section 7 states the limitations, and Section 8 concludes.

2. RELATED WORK

2.1 Machine Learning Approaches to Intrusion Detection

Intrusion detection systems (IDS) are commonly divided into signature-based and anomaly-based designs. Signature-based systems match observed activity against patterns of known attacks and offer low false-alarm rates for those attacks but cannot recognize unfamiliar ones. Anomaly-based systems model normal behavior and flag deviations, which in principle allows detection of novel attacks at the price of more false positives (Chandola et al., 2009). Supervised learning approaches train on labeled examples of benign and malicious activity and have used decision trees, support vector machines, ensembles, and neural networks. Surveys of the field describe how these techniques have been applied to network traffic features and compare their reported performance on public datasets (Dina & Manivannan, 2021; Sarker et al., 2020).

Deep learning has attracted particular attention because it can learn feature representations directly from raw or minimally processed data. MahdaviFar and Ghorbani (2019) surveyed deep learning applications across cybersecurity tasks, while Ferrag et al. (2020) compared deep architectures for intrusion detection across several datasets. These surveys also caution that results depend heavily on the dataset, the preprocessing pipeline, and the evaluation protocol, so that direct numerical comparison across studies is often unreliable.

2.2 Tree Ensembles and Boosting

Tree-based ensembles are a strong default for tabular security data. Random forests average many decorrelated decision trees trained on bootstrap samples and random feature subsets, which reduces variance and gives a natural estimate of feature importance (Breiman, 2001). Gradient boosting builds trees sequentially, each one correcting the residual errors of its

predecessors, and often achieves high accuracy with modest tuning (Friedman, 2001). Both families handle mixed feature types, require little scaling, and are comparatively inexpensive to train, which makes them attractive baselines against which more complex models should be compared (Hastie et al., 2009).

2.3 Unsupervised and Semi-Supervised Detection

Labeled attack data are expensive to obtain and quickly become outdated, so unsupervised and semi-supervised detectors are of practical interest. Isolation forests identify anomalies by exploiting the observation that unusual points are easier to separate from the rest of the data using random partitions (Liu et al., 2008). Because they require no attack labels, they can be trained on traffic assumed to be mostly benign. However, their effectiveness depends on whether malicious activity is in fact easy to isolate, and attacks designed to resemble normal behavior may defeat them. Chandola et al. (2009) provide a broad taxonomy of anomaly detection methods and the assumptions behind each.

2.4 Evaluation Under Imbalance and Unequal Costs

In most operational settings malicious events are rare. When a class is rare, overall accuracy is a misleading metric because a model that labels everything benign can score very highly. Precision, recall, and the F1 score focus on the minority class, and precision-recall curves are generally more informative than receiver operating characteristic (ROC) curves when positives are scarce (Saito & Rehmsmeier, 2015). Beyond metric choice, the consequences of errors are unequal: a missed intrusion may lead to data exfiltration, while a false alarm consumes analyst time. Cost-sensitive learning formalizes this asymmetry and shows that the optimal decision threshold depends on the ratio of error costs rather than on a fixed probability of one half (Elkan, 2001).

2.5 Robustness and Adversarial Considerations

Security is an adversarial domain, so detectors may face inputs crafted to evade them. Goodfellow et al. (2015) showed that small, deliberately chosen perturbations can cause neural networks to misclassify inputs, and subsequent surveys have catalogued attacks against and defenses for ML systems used in security settings (Hu et al., 2022). A full adversarial evaluation requires an explicit threat model and attack algorithm. In this article we adopt a much simpler robustness probe based on random noise, which is a necessary but not sufficient condition for resilience, and we state clearly that it does not measure adversarial robustness.

3. PROBLEM FORMULATION AND MATHEMATICAL MODEL

3.1 Classification Setting

Let $\mathbf{x} \in \mathbb{R}^d$ denote a feature vector describing one observation, for example a network flow summarized by d statistics, and let $y \in \{0, 1\}$ denote its label, where $y = 1$ indicates malicious activity and $y = 0$ indicates benign activity. Let $\pi = P(y = 1)$ be the prevalence of malicious events. A detector produces a score $s(\mathbf{x}) \in [0, 1]$ intended to increase with the likelihood of malice, and an alert is raised when $s(\mathbf{x}) \geq \tau$ for a threshold τ . For a supervised probabilistic classifier, $s(\mathbf{x})$ estimates the posterior probability $P(y = 1 \mid \mathbf{x})$.

Given a threshold τ , the true positive rate (recall) and the false positive rate are defined as

$$TPR(\tau) = P(s(x) \geq \tau \mid y = 1), \quad FPR(\tau) = P(s(x) \geq \tau \mid y = 0) \quad (1)$$

and precision is the fraction of alerts that correspond to real attacks:

$$Precision(\tau) = \pi \cdot TPR(\tau) / [\pi \cdot TPR(\tau) + (1 - \pi) \cdot FPR(\tau)] \quad (2)$$

Equation 2 makes explicit why a detector with a seemingly small false positive rate can still overwhelm analysts when π is small: the false alerts are generated by the large benign majority. The F1 score is the harmonic mean of precision and recall,

$$F1 = 2 \cdot Precision \cdot Recall / (Precision + Recall). \quad (3)$$

3.2 Cost-Sensitive Decision Rule

Let C_{FN} denote the cost of a missed attack and C_{FP} the cost of a false alarm, with correct decisions assumed to cost nothing. The expected cost per observation at threshold τ is

$$R(\tau) = C_{FN} \cdot \pi \cdot (1 - TPR(\tau)) + C_{FP} \cdot (1 - \pi) \cdot FPR(\tau). \quad (4)$$

If the score is a calibrated posterior probability $p(x) = P(y = 1 \mid x)$, the decision that minimizes expected cost for each observation is to raise an alert whenever the expected cost of ignoring exceeds the expected cost of alerting, that is, when $C_{FN} \cdot p(x) > C_{FP} \cdot (1 - p(x))$. Solving for $p(x)$ gives the threshold

$$\tau^* = C_{FP} / (C_{FP} + C_{FN}). \quad (5)$$

This result, discussed in the cost-sensitive learning literature (Elkan, 2001), shows that when a missed attack is ten times as costly as a false alarm, the optimal threshold for a calibrated model is $1/11 \approx 0.091$ rather than 0.5. In practice many classifiers, including random forests and boosted trees, do not produce calibrated probabilities, so the best empirical threshold is found by minimizing Equation 4 on held-out validation data rather than by applying Equation 5 directly. Equation 5 nevertheless provides a principled starting point and a sanity check: the empirically optimal threshold should be much lower than 0.5 whenever C_{FN} greatly exceeds C_{FP} .

3.3 Models Considered

Logistic regression models the log-odds of malice as a linear function of the features, $\log[p/(1 - p)] = w \cdot x + b$, and is fitted by maximum likelihood with regularization (Hastie et al., 2009). It serves as an interpretable baseline. The random forest and gradient boosting models are the tree ensembles described in Section 2.2. The multilayer perceptron (MLP) is a feed-forward neural network with two hidden layers of 64 and 32 units, rectified linear activations, and a logistic output unit, trained by stochastic gradient-based optimization. The isolation forest assigns each point an anomaly score

$$s(x, n) = 2^{-(E[h(x)] / c(n))}, \quad (6)$$

where $h(x)$ is the path length needed to isolate x in a random tree, $E[h(x)]$ is its average across trees, and $c(n)$ is the average path length of an unsuccessful search in a binary tree built on n

points (Liu et al., 2008). Scores near one indicate points that are isolated quickly and are therefore treated as anomalous.

3.4 Robustness Probe

To examine sensitivity to input perturbation, we corrupt each standardized test feature vector with independent Gaussian noise of standard deviation σ , $\tilde{x} = x + \varepsilon$ with $\varepsilon \sim N(0, \sigma^2 I)$, and recompute F1 for each σ in a fixed grid. Define the relative degradation of a model at noise level σ as

$$D(\sigma) = [F1(0) - F1(\sigma)] / F1(0). \quad (7)$$

A model with small $D(\sigma)$ at moderate σ is less sensitive to this form of noise. We emphasize that random noise is not a substitute for adversarial testing: an attacker would choose perturbations deliberately, and the damage can be far greater than that caused by random noise of the same magnitude (Goodfellow et al., 2015).

4. DATA AND EXPERIMENTAL SETUP

4.1 Synthetic Dataset

No real organizational or government data were used. We generated a synthetic binary classification dataset with the `make_classification` routine of the scikit-learn library (Pedregosa et al., 2011). The dataset contains 20,000 records with 30 numeric features, of which 14 are informative, 6 are linear combinations of informative features, and the remainder are noise. Each class is composed of three Gaussian clusters, and 2 percent of labels were flipped at random to mimic labeling error. The class weights were set so that about 90 percent of records are benign and 10 percent are malicious. Because the features are abstract, they are labeled `x1` through `x30` and have no operational meaning. Figure 1 shows the class distribution and a two-dimensional principal component projection of the test set, which illustrates that the classes overlap substantially.

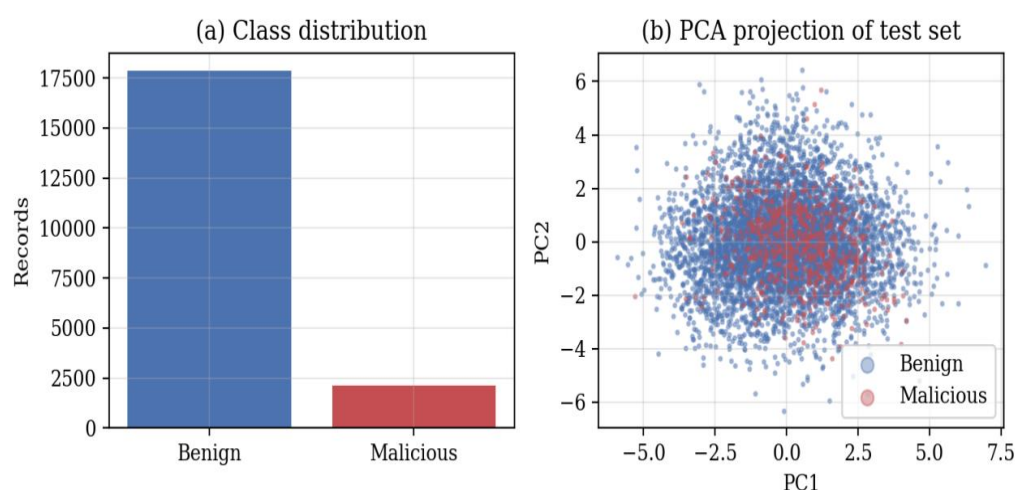


Figure 1. Illustrative synthetic data. (a) Class distribution of the full dataset. (b) Projection of the standardized test set onto its first two principal components. The substantial overlap indicates that the detection task is not trivially separable.

4.2 Protocol

The data were split into a training set of 14,000 records and a test set of 6,000 records using stratified sampling so that both retain the same class proportions. Features were standardized using the mean and standard deviation of the training set only, which prevents information from the test set leaking into preprocessing, a pitfall highlighted by Arp et al. (2022). Supervised models were trained on the labeled training data with default or lightly specified hyperparameters and without extensive tuning, because the aim is to illustrate the evaluation framework rather than to maximize performance. The isolation forest was trained only on benign training records with 200 trees, and its alert threshold was set so that the top 10 percent of test scores are flagged, which matches the assumed prevalence. A single random seed (42) was used for data generation, splitting, and model initialization, so all reported numbers are reproducible. **Table 1** summarizes the setup.

Table 1. Summary of the synthetic dataset and experimental configuration.

Item	Value
Total records	20,000
Training / test records	14,000 / 6,000
Features (informative / redundant / noise)	30 (14 / 6 / 10)
Approximate malicious prevalence	10%
Label noise	2% random flips
Preprocessing	Standardization fitted on training set
Supervised models	Logistic regression, random forest (200 trees), gradient boosting, MLP (64-32)
Unsupervised model	Isolation forest (200 trees), trained on benign records only
Default alert threshold	0.5 for supervised models; top 10% of scores for isolation forest
Random seed	42

4.3 Evaluation Metrics

We report precision, recall, and F1 at the default threshold, together with two threshold-independent summaries: the area under the ROC curve (AUC) and the average precision (AP), which summarizes the precision-recall curve. Given the imbalance, we regard recall, precision, F1, and AP as more informative than accuracy, and we do not report accuracy at all (Saito & Rehmsmeier, 2015). For the cost analysis we assume $C_{FN} = 10$ and $C_{FP} = 1$ in arbitrary cost units, which expresses the belief that a missed attack is ten times as damaging as a false alarm; these values are assumptions chosen for illustration, not estimates from any real organization.

5. RESULTS

5.1 Overall Detection Performance

Table 2 reports the performance of the five models on the held-out test set. The MLP achieved the best F1 score (0.792) and the best AUC (0.944) and AP (0.847), reflecting a balance of precision (0.823) and recall (0.763). The random forest attained the highest precision (0.961) but a recall of only 0.348 at the default threshold, meaning that it missed roughly two thirds of malicious records while rarely raising false alarms. Gradient boosting behaved similarly, with precision 0.853 and recall 0.317. The logistic regression baseline performed poorly at the default threshold, with recall of only 0.062, which is consistent with a linear boundary being inadequate for data built from multiple clusters per class. The unsupervised isolation forest performed worst (F1 = 0.168, AUC = 0.573), only modestly better than chance in ranking, because the malicious records in this synthetic dataset are not simple outliers relative to the benign class.

Table 2. Test-set performance of the evaluated models on the synthetic dataset (illustrative only). P = precision, R = recall, AUC = area under the ROC curve, AP = average precision.

Model	P	R	F1	AUC	AP
Logistic Regression	0.541	0.062	0.112	0.763	0.307
Random Forest	0.961	0.348	0.511	0.932	0.789
Gradient Boosting	0.853	0.317	0.462	0.908	0.681
MLP (64-32)	0.823	0.763	0.792	0.944	0.847
Isolation Forest (unsupervised)	0.173	0.162	0.168	0.573	0.140

The ROC curves in Figure 2 show that the MLP and random forest dominate across most operating points, while the isolation forest lies close to the diagonal. The precision-recall curves in Figure 3 separate the models more clearly. Because positives constitute about one tenth of the data, the no-skill precision is approximately 0.10, shown as the dashed horizontal line, and

the gap between the best and worst models is visibly larger in this view than in the ROC view, which is consistent with the guidance that precision-recall analysis is more informative under imbalance (Saito & Rehmsmeier, 2015).

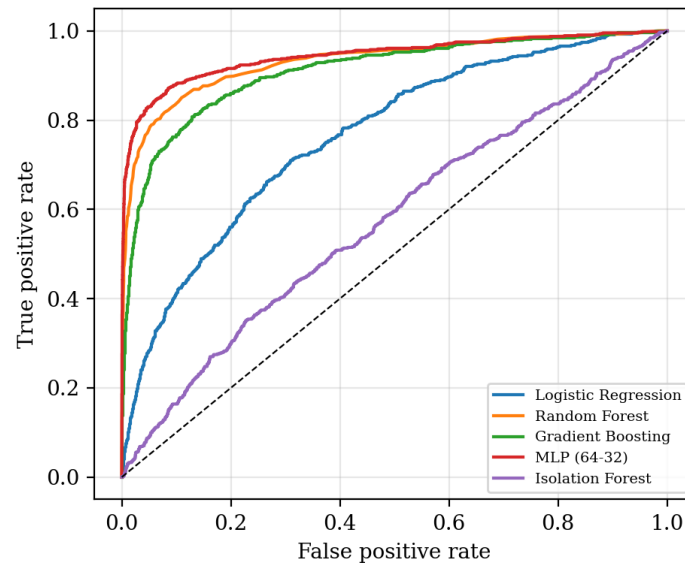


Figure 2. Receiver operating characteristic curves for all models on the synthetic test set (illustrative only).

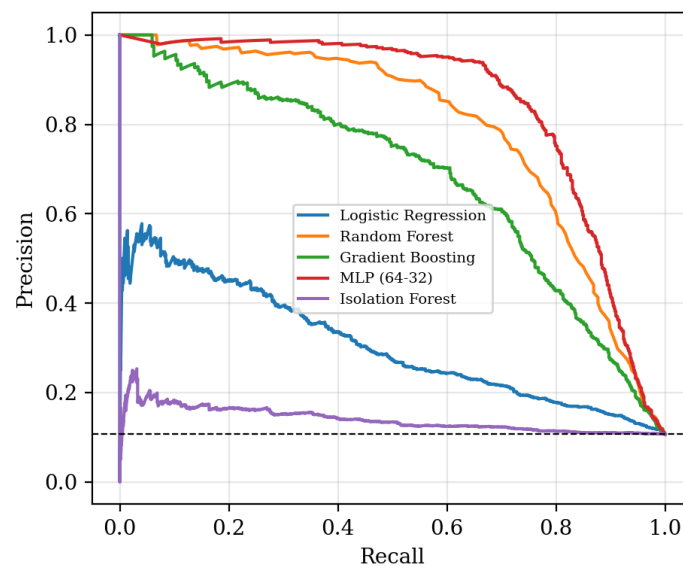


Figure 3. Precision-recall curves for all models on the synthetic test set. The dashed line marks the prevalence-based no-skill baseline (illustrative only).

5.2 Cost-Sensitive Threshold Selection

The default threshold of 0.5 is rarely appropriate when errors have unequal costs. Using the random forest scores and the assumed costs $C_{FN} = 10$ and $C_{FP} = 1$, we swept the threshold from 0.02 to 0.98 and computed the expected cost of Equation 4 at each value. Figure 4 shows

that the expected cost is minimized at $\tau = 0.17$, considerably below 0.5, and Table 3 compares the two operating points. At the default threshold the forest produced only 9 false positives but 418 false negatives among 6,000 test records, giving an expected cost of 0.698 per record. At the cost-optimal threshold the false negative rate fell to 15.0 percent at the price of a false positive rate of 11.0 percent, and the expected cost dropped to 0.258, a reduction of about 63 percent.

The empirical optimum of 0.17 is higher than the value $1/11 \approx 0.091$ predicted by Equation 5 for a calibrated model. This is expected: random forest vote fractions are not calibrated probabilities, so the theoretical threshold does not transfer exactly. The result supports the practical recommendation in Section 3.2 to use Equation 5 as a guide and to select the final threshold on validation data.

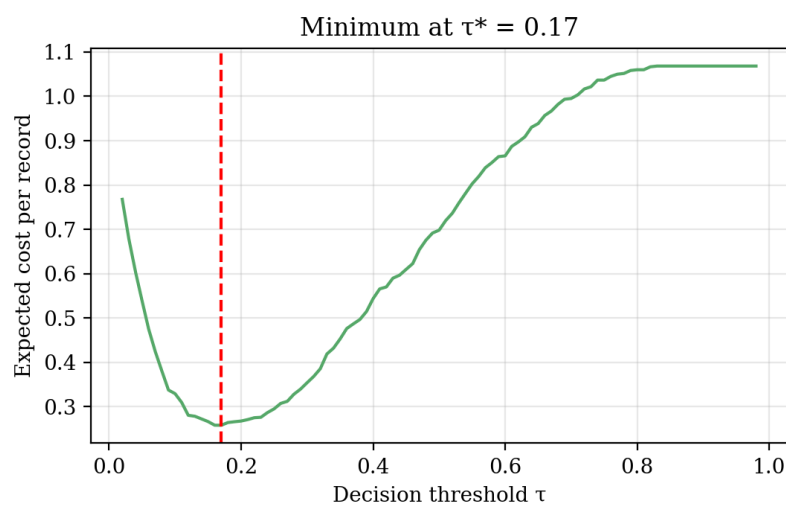


Figure 4. Expected cost per record of the random forest as a function of the decision threshold, with $C_{FN} = 10$ and $C_{FP} = 1$ (assumed values). The dashed line marks the minimizing threshold (illustrative only).

Table 3. Random forest operating points on the synthetic test set under assumed costs $C_{FN} = 10$ and $C_{FP} = 1$ (illustrative only).

Operating point	Threshold	False positive rate	False negative rate	Expected cost per record
Default	0.50	0.2%	65.2%	0.698
Cost-minimizing	0.17	11.0%	15.0%	0.258

5.3 Sensitivity to Input Noise

Figure 5 and Table 4 report F1 as Gaussian noise of increasing standard deviation is added to the standardized test features. All four supervised models lost performance as noise increased. The MLP, which was the strongest model on clean data, degraded the most in relative terms: its F1 fell from 0.792 without noise to 0.598 at $\sigma = 0.5$ and to 0.367 at $\sigma = 1.0$, a relative degradation $D(1.0)$ of about 54 percent by Equation 7. The random forest degraded more gently, with F1 declining from 0.504 to 0.379 over the same range (about 25 percent). At the highest noise level the MLP and random forest scored similarly, so the MLP's advantage on clean data disappeared. The logistic regression baseline did not decline because its F1 was already very low; its small increase is an artifact of noise pushing some scores across the threshold and should not be interpreted as improved robustness. Small differences between the clean-data F1 values in Table 2 and the $\sigma = 0$ column of Table 4 arise from how ties at the 0.5 threshold are handled by the two code paths and are immaterial.

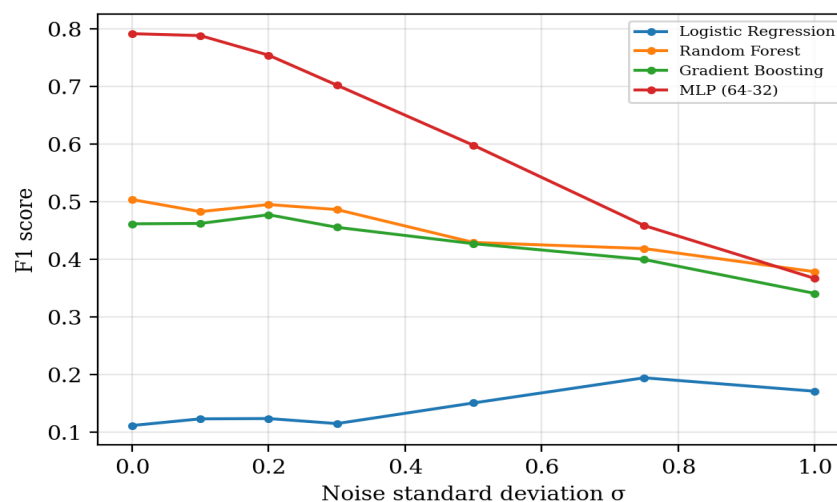


Figure 5. F1 score of supervised models as Gaussian noise of standard deviation σ is added to the standardized test features (illustrative only). This is a random-noise probe and not an adversarial attack.

Table 4. F1 score under increasing input noise (illustrative only).

Model	$\sigma = 0$	$\sigma = 0.1$	$\sigma = 0.2$	$\sigma = 0.3$	$\sigma = 0.5$	$\sigma = 0.75$	$\sigma = 1$
Logistic Regression	0.112	0.123	0.124	0.115	0.151	0.195	0.171
Random Forest	0.504	0.483	0.495	0.487	0.430	0.419	0.379
Gradient Boosting	0.462	0.463	0.478	0.456	0.427	0.400	0.341

Model	$\sigma = 0$	$\sigma = 0.1$	$\sigma = 0.2$	$\sigma = 0.3$	$\sigma = 0.5$	$\sigma = 0.75$	$\sigma = 1$
MLP (64-32)	0.792	0.789	0.755	0.703	0.598	0.459	0.367

6. DISCUSSION

6.1 What the Synthetic Results Illustrate

The experiments are not evidence about any real system, but they do illustrate several lessons that appear repeatedly in the applied literature. The first lesson is that a single headline number can conceal very different behaviors. The random forest and the MLP have similar AUC values (Table 2), yet at the default threshold one rarely raises false alarms and misses most attacks, while the other catches most attacks at the cost of more false alarms. Which behavior is preferable depends entirely on the relative costs of errors, which is why the cost-sensitive view of Section 3.2 is more informative than any ranking by a single metric.

The second lesson is that the decision threshold is a policy lever. Moving the random forest's threshold from 0.5 to the cost-minimizing value reduced expected cost far more than the choice between most of the candidate models would have. In a security operations center this lever can be adjusted over time as analyst capacity, threat levels, and the value of protected assets change. Reporting only performance at a default threshold deprives decision makers of this flexibility, and reporting only threshold-free summaries such as AUC hides the operating point they must actually choose.

The third lesson is that models that look strongest under clean conditions are not necessarily the most stable. The MLP lost most of its advantage under moderate noise. In deployment, feature distributions change as networks evolve, sensors are reconfigured, and attackers modify their behavior, so a detector's performance on a static test set is an optimistic estimate of its future performance. Continuous monitoring of model performance, periodic retraining, and drift detection are therefore part of the detector rather than optional extras (Sommer & Paxson, 2010; Arp et al., 2022).

6.2 Implications for Protecting Sensitive Data

Systems that hold sensitive government or citizen data impose additional requirements on detection pipelines. Because the data are sensitive, the telemetry used to train and operate detectors must itself be protected, minimized, and governed by access controls and retention limits, and training workflows should avoid copying raw sensitive content into analytic environments when derived features suffice. Detection should be one layer among several. The NIST Cybersecurity Framework organizes security activities into functions for identifying, protecting, detecting, responding to, and recovering from incidents, and an ML detector contributes primarily to the detection function while depending on sound practices in the others (National Institute of Standards and Technology [NIST], 2018). A highly accurate detector is of limited value if alerts are not triaged, if response playbooks are untested, or if backups cannot be restored.

Developing detection technology for government environments also involves design choices that go beyond model selection. Feature pipelines should be built from minimized telemetry, such as flow statistics, authentication metadata, and endpoint event counts, rather than from the content of sensitive records. Where data cannot leave a protected enclave, models can be trained in place, and where multiple agencies wish to learn from one another without pooling raw logs, privacy-preserving approaches can be considered, although these were not evaluated here. Every alert should be traceable to the model version, the input features, and the threshold policy in force, so that decisions can be audited and, where necessary, explained to oversight bodies. Human analysts remain accountable for response actions, and the detector should be designed to support their judgment, not to replace it.

6.3 Implications for Resilience

Resilience concerns the ability of an infrastructure to continue operating, and to recover quickly, in the presence of attacks. From this perspective, a detector's contribution is measured not only by whether it identifies attacks but by how quickly and reliably it enables containment, and by how gracefully it fails. Cost-sensitive thresholds allow organizations to tilt toward higher recall for critical assets, accepting more false alarms where the consequences of a miss are severe, and toward higher precision for lower-value assets. Combining detectors with different strengths, for example a supervised ensemble for known attack families and an anomaly detector for novelty, can improve coverage, although in our synthetic experiment the unsupervised model added little because the simulated attacks were not outliers. The robustness probe in Section 5.3 also suggests that organizations should test candidate detectors under degraded and shifted inputs before relying on them.

6.4 Practical Recommendations

Several practical recommendations follow from the framework. Practitioners should report precision, recall, F1, and precision-recall summaries rather than accuracy; they should state the cost assumptions behind any threshold and show how conclusions change when those assumptions change; they should evaluate on data that are temporally separated from the training data wherever possible; and they should compare against simple baselines such as tuned tree ensembles before adopting more complex architectures (Hastie et al., 2009; Arp et al., 2022). Finally, they should treat explanations and analyst feedback as part of the system, since a detector that analysts do not trust will not be used effectively.

7. LIMITATIONS

This article has important limitations that readers should keep in mind. The most significant is that all data are synthetic. The simulated features have no semantic meaning, the class structure is generated from Gaussian clusters, and real network traffic exhibits temporal dependence, heavy-tailed distributions, protocol structure, and strategic adversaries that the simulator does not capture. Consequently, the relative ranking of the models here, including the success of the MLP and the failure of the isolation forest, may differ substantially on real data, and no claim is made that these rankings would hold in practice.

Second, the experiments use a single random seed, a single train-test split, and untuned models, so we report no confidence intervals or significance tests. Differences between models of a few percentage points should not be regarded as meaningful. A rigorous study would use repeated cross-validation or multiple seeds, tune hyperparameters on validation data, and report uncertainty. Third, the cost values are assumed rather than elicited from a real risk analysis, and Equation 4 ignores the fact that the cost of a missed attack varies with the asset affected and the time to detection. Fourth, the robustness probe uses random noise and not adversarial perturbations, so it does not establish resilience against a motivated attacker (Goodfellow et al., 2015; Hu et al., 2022). Fifth, we did not address calibration, explainability, concept drift, or the computational constraints of real-time deployment, each of which is important in operational settings.

Future work should repeat the evaluation on public intrusion-detection corpora such as NSL-KDD and CIC-IDS2017 (Sharafaldin et al., 2018; Tavallaei et al., 2009), taking care to follow the methodological guidance of Arp et al. (2022) regarding data splits and realistic baselines. It should add probability calibration, time-aware validation, and explicit adversarial evaluation, and it should examine privacy-preserving training approaches appropriate to sensitive data.

8. CONCLUSION

Machine learning can strengthen the detection of cyber threats, but its value depends on how it is evaluated and deployed. This article presented an educational evaluation framework that combines imbalance-aware metrics, a cost-sensitive decision rule, and a simple robustness probe, and it illustrated the framework on a synthetic dataset. In that setting the neural network gave the best balance of precision and recall, tuning the decision threshold under asymmetric costs reduced expected cost by about 63 percent relative to the default threshold, and every supervised model lost accuracy as input noise increased. These findings are illustrative and must not be mistaken for results about real systems. The enduring takeaways are methodological: report metrics that match the operational problem, choose thresholds by cost, test behavior under change, and treat the detector as one component of a broader program for protecting sensitive data and sustaining the resilience of digital infrastructure.

REFERENCES

- [1.] Arp, D., Quiring, E., Pendlebury, F., Warnecke, A., Pierazzi, F., Wressnegger, C., Cavallaro, L., & Rieck, K. (2022). Dos and don'ts of machine learning in computer security. In Proceedings of the 31st USENIX Security Symposium (pp. 3971–3988). USENIX Association.
- [2.] Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), 5–32. <https://doi.org/10.1023/A:1010933404324>
- [3.] Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. *ACM Computing Surveys*, 41(3), Article 15. <https://doi.org/10.1145/1541880.1541882>
- [4.] Dina, A. S., & Manivannan, D. (2021). Intrusion detection based on machine learning techniques in computer networks. *Internet of Things*, 16, Article 100462. <https://doi.org/10.1016/j.iot.2021.100462>

- [5.] Elkan, C. (2001). The foundations of cost-sensitive learning. In Proceedings of the 17th International Joint Conference on Artificial Intelligence (pp. 973–978). Morgan Kaufmann.
- [6.] Ferrag, M. A., Maglaras, L., Moschogiannis, S., & Janicke, H. (2020). Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study. *Journal of Information Security and Applications*, 50, Article 102419. <https://doi.org/10.1016/j.jisa.2019.102419>
- [7.] Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5), 1189–1232. <https://doi.org/10.1214/aos/1013203451>
- [8.] Goodfellow, I. J., Shlens, J., & Szegedy, C. (2015). Explaining and harnessing adversarial examples. In International Conference on Learning Representations. <https://arxiv.org/abs/1412.6572>
- [9.] Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: Data mining, inference, and prediction* (2nd ed.). Springer. <https://doi.org/10.1007/978-0-387-84858-7>
- [10.] Hu, Y., Kuang, W., Qin, Z., Li, K., Zhang, J., Gao, Y., Li, W., & Li, K. (2022). Artificial intelligence security: Threats and countermeasures. *ACM Computing Surveys*, 55(1), Article 20. <https://doi.org/10.1145/3487890>
- [11.] Liu, F. T., Ting, K. M., & Zhou, Z.-H. (2008). Isolation forest. In Proceedings of the 2008 Eighth IEEE International Conference on Data Mining (pp. 413–422). IEEE. <https://doi.org/10.1109/ICDM.2008.17>
- [12.] MahdaviFar, S., & Ghorbani, A. A. (2019). Application of deep learning to cybersecurity: A survey. *Neurocomputing*, 347, 149–176. <https://doi.org/10.1016/j.neucom.2019.02.056>
- [13.] National Institute of Standards and Technology. (2018). Framework for improving critical infrastructure cybersecurity (Version 1.1). <https://doi.org/10.6028/NIST.CSWP.04162018>
- [14.] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.
- [15.] Saito, T., & Rehmsmeier, M. (2015). The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. *PLOS ONE*, 10(3), Article e0118432. <https://doi.org/10.1371/journal.pone.0118432>
- [16.] Sarker, I. H., Kayes, A. S. M., Badsha, S., Alqahtani, H., Watters, P., & Ng, A. (2020). Cybersecurity data science: An overview from machine learning perspective. *Journal of Big Data*, 7, Article 41. <https://doi.org/10.1186/s40537-020-00318-5>

- [17.] Sharafaldin, I., Lashkari, A. H., & Ghorbani, A. A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. In Proceedings of the 4th International Conference on Information Systems Security and Privacy (pp. 108–116). SciTePress. <https://doi.org/10.5220/0006639801080116>
- [18.] Sommer, R., & Paxson, V. (2010). Outside the closed world: On using machine learning for network intrusion detection. In 2010 IEEE Symposium on Security and Privacy (pp. 305–316). IEEE. <https://doi.org/10.1109/SP.2010.25>
- [19.] Tavallae, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009). A detailed analysis of the KDD CUP 99 data set. In 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications (pp. 1–6). IEEE. <https://doi.org/10.1109/CISDA.2009.5356528>