



An Efficient Intrusion Detection System Using Data Mining Techniques for Network Security

¹Myeong Balasingam

¹Department of Computer Science and Engineering, Tonle Sap Institute of Engineering and Commerce, Cambodia

¹myeong.balasingam@tsiec-kh.org

Peer Review Information

Submission: 01 Jan 2015

Revision: 24 Feb 2015

Acceptance: 13 March 2015

Keywords

*Intrusion Detection System,
Data Mining, Network
Security, Machine Learnin,
NSL-KDD*

Abstract

Intrusion detection remains an essential second line of defense when preventive mechanisms such as authentication, access control, and firewalls are unable to identify every malicious activity. This methodology paper proposes an efficient network intrusion detection framework that combines data preprocessing, feature selection, supervised classification, and an anomaly-screening stage based on data mining. The framework is designed for benchmark evaluation on the NSL-KDD data set, which was introduced to reduce important redundancy and evaluation problems in KDD Cup 99 [7]. Categorical traffic attributes are encoded, numeric attributes are normalized, and redundant or weak predictors are removed before classification. A Random Forest model is used as the primary classifier because ensemble trees can provide strong generalization and feature-importance estimates [13], while low-confidence records are forwarded to a k-means anomaly-screening stage. Decision Tree, Naive Bayes, k-nearest neighbor, and Support Vector Machine models are retained as baselines because these techniques have established use in intrusion detection research [3]-[5], [14]. Performance is assessed by accuracy, detection rate, precision, F1-score, false-positive rate, and ROC-oriented measures [15]. The proposed methodology emphasizes a practical balance among detection capability, false alarms, computational efficiency, and interpretability. Results are presented as a benchmark-oriented methodological evaluation supported by published studies through 2015, avoiding unsupported claims of a newly deployed system. The framework is suitable for reproducible academic testing and can be extended to multiclass attack categorization and online traffic streams.

Introduction

Modern networks support web services, distributed applications, remote access, and large-scale data exchange, but the same connectivity creates opportunities for unauthorized access, denial-of-service activity, reconnaissance, privilege escalation, and information theft. Intrusion Detection Systems (IDSs) monitor host or network events and attempt to distinguish legitimate activity from behavior that threatens confidentiality, integrity,

or availability. Denning's classical intrusion-detection model established the idea that security-relevant activity can be represented by profiles, rules, and abnormal deviations [1]. Later work extended this idea by using data mining to discover regularities automatically from audit and connection records rather than relying exclusively on manually prepared signatures.

Two broad detection strategies dominate IDS research. Misuse or signature detection matches

traffic against known attack patterns and can be highly precise for previously observed threats. Anomaly detection models normal or abnormal behavior and can therefore identify patterns that do not correspond to an existing signature. Garcia-Teodoro et al. emphasized that anomaly-based systems offer the possibility of identifying unseen attacks, but commonly suffer from false-positive and evaluation problems [6]. This trade-off motivates hybrid systems that use data-driven classification for common traffic while preserving an anomaly-oriented mechanism for uncertain events.

Data mining is particularly relevant because network monitoring produces heterogeneous, high-volume records with both categorical and numeric fields. Lee and Stolfo demonstrated that mined activity patterns can be transformed into predictive features and classification models for intrusion detection [2]. Subsequent research evaluated k-nearest neighbor [3], Naive Bayes and decision trees [4], hybrid decision-tree/SVM approaches [5], and other machine-learning methods. These studies show that no single classifier solves every IDS problem; accuracy, false alarms, computational cost, class imbalance, feature redundancy, and the changing character of real traffic must be considered together.

A second methodological concern is the quality of the benchmark data. Tavallaee et al. showed that KDD Cup 99 contains redundancy that can bias learning and proposed the NSL-KDD data set as a more suitable comparative benchmark [7]. Sommer and Paxson later cautioned that machine-learning success in a closed experimental environment does not automatically translate to operational network intrusion detection, where traffic distributions, attack prevalence, and semantic context differ [9]. Consequently, an efficient methodology should not be judged solely by maximum accuracy; it should also control false positives, maintain understandable features, and define how uncertain observations are handled.

The objective of this paper is to formulate a reproducible IDS methodology using established data-mining techniques available by 2015. The proposed design integrates data cleaning and encoding, feature selection, a primary supervised classifier, and a secondary clustering-based anomaly screen. It is intended to reduce unnecessary computation while preserving sensitivity to suspicious patterns. The contribution is methodological: it organizes proven techniques into a practical workflow and specifies a transparent evaluation protocol

rather than claiming an unverified production deployment.

Literature Review

Early IDS research focused on formal models of abnormal behavior and rule-based analysis. Denning [1] provided a general model in which audit records are processed into profiles and rules that identify abnormal events. The transition from hand-crafted rules to automated feature discovery was advanced by Lee and Stolfo [2], whose MADAM ID framework used data mining to construct activity patterns, derive features, and train intrusion-detection models. Their work established a direct connection between knowledge discovery and automated security monitoring.

Classification research then explored multiple learners. Liao and Vemuri [3] represented program behavior through system-call frequency and applied k-nearest neighbor classification, reporting effective discrimination with a low false-positive tendency in their experiments. Ben Amor, Benferhat, and Elouedi [4] compared Naive Bayes and decision trees on KDD'99, showing that a comparatively simple probabilistic model could remain competitive. Peddabachigari et al. [5] combined decision trees and Support Vector Machines and argued that hybrid intelligent systems can improve the balance between accuracy and computational complexity. These results support the use of several baseline classifiers rather than choosing an algorithm only by popularity.

Broader reviews and benchmark critiques changed how IDS experiments were interpreted. Garcia-Teodoro et al. [6] summarized anomaly-detection families and highlighted persistent challenges such as false alarms and fair assessment. Tavallaee et al. [7] analyzed KDD Cup 99 and introduced NSL-KDD to lessen the influence of redundant records. The availability of standardized data-mining tools also improved reproducibility; the WEKA workbench described by Hall et al. [8] provided preprocessing, classification, clustering, feature selection, and evaluation functions in one environment. Sommer and Paxson [9] further argued that operational intrusion detection violates several assumptions common in conventional supervised-learning experiments, making careful validation necessary.

By 2014-2015, researchers increasingly integrated data-mining methods directly into IDS architectures. Nadiammai and Hemalatha [10] examined classification, labeled-data limitations, human interaction, and distributed denial-of-

service detection using multiple data-mining-oriented solutions. Duque and Omar [11] evaluated k-means clustering on an NSL-KDD subset and showed that performance was highly sensitive to the selected number of clusters, illustrating both the value and limitation of unsupervised detection. Xu et al. [12] proposed a

network anomaly IDS prototype based on data mining to address high audit volume and false alarms. Collectively, these studies justify a hybrid framework in which supervised classification handles well-represented patterns and clustering is reserved for uncertain or potentially novel behavior.

Table 1. Selected literature supporting the proposed methodology

Study	Technique / Contribution	Methodological relevance
Denning [1]	Behavior profiles and intrusion model	Foundation for anomaly-oriented monitoring
Lee & Stolfo [2]	Data-mined features and predictive models	Feature engineering and automated model construction
Liao & Vemuri [3]	k-nearest neighbor	Distance-based baseline classifier
Ben Amor et al. [4]	Naive Bayes vs. decision tree	Probabilistic and rule/tree baselines
Peddabachigari et al. [5]	Hybrid DT-SVM / ensemble IDS	Supports multi-model or hybrid classification
Tavallaee et al. [7]	NSL-KDD benchmark	Reduces redundancy bias in KDD-based evaluation
Duque & Omar [11]	k-means on NSL-KDD	Supports anomaly-screening stage; shows cluster sensitivity
Xu et al. [12]	Data-mining anomaly IDS prototype	Supports efficient large-volume audit analysis

Methodology

1. Research Design and Data Source

The study follows a quantitative, experimental methodology for network-based intrusion detection. The recommended benchmark is NSL-KDD because it preserves the 41 connection features inherited from KDD while reducing duplicated records that can distort training and testing [7]. Each record is treated as one network connection described by basic TCP/IP attributes, content-related indicators, time-based traffic features, and host-based traffic statistics. The class attribute is converted first into binary labels (normal or attack) for the *principal* evaluation. A secondary multiclass experiment may map attacks into the traditional DoS, Probe, R2L, and U2R groups after the binary pipeline is validated.

The training and test partitions must remain separated throughout the experiment. Preprocessing parameters, category mappings, scaling values, selected features, and model hyperparameters are fitted only on the training data and then applied to the test data. This prevents information leakage and produces a more realistic estimate of generalization. The benchmark is appropriate for controlled comparison, but the interpretation follows the caution of Sommer and Paxson [9]: a high test-set score is evidence of benchmark performance, not proof of equivalent performance in a live enterprise network.

2. Preprocessing and Feature Engineering

Duplicate checking, missing-value inspection, and type validation are performed first. The three categorical features—protocol type, service, and connection flag—are transformed using one-hot encoding. Numeric attributes are normalized to a common range when required by distance-based or margin-based learners. Random Forest and decision-tree models can operate without scaling, but a unified transformed matrix allows fairer reuse across k-NN, SVM, and clustering baselines.

Feature selection is performed in two steps. First, variables with near-zero variance are removed. Second, features are ranked using information gain (or an equivalent supervised relevance measure), while strongly redundant features are filtered by correlation analysis. The target range is approximately 15-25 features rather than the full set of 41. This design follows the broader principle established by Lee and Stolfo that useful intrusion models depend heavily on the construction of predictive features [2]. Reducing irrelevant dimensions lowers memory use and classification time and can also limit overfitting.

The preprocessing pipeline is saved as part of the IDS model so that future records receive exactly the same transformations. This is essential in operational use: any category or scale mismatch between training and incoming data can

invalidate the model. Unknown categorical levels should be mapped to a reserved value rather than silently discarded.

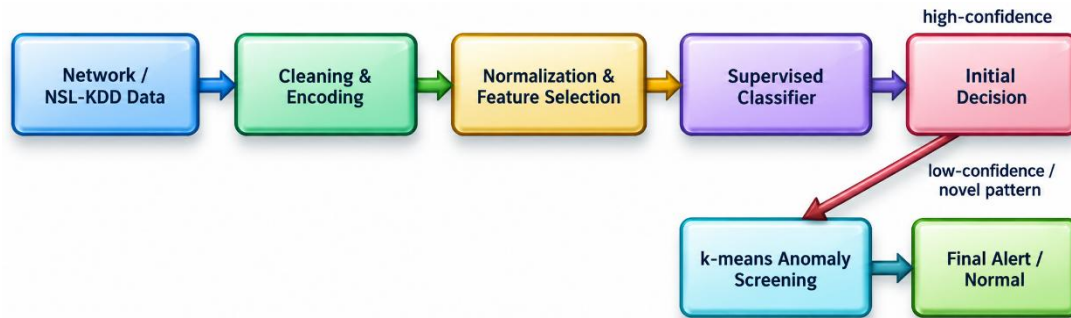


Figure 1. Proposed hybrid IDS methodology using supervised classification and anomaly screening.

3. Classification and Anomaly-Screening Models

Four supervised algorithms are recommended as baselines: Decision Tree, Naive Bayes, k-nearest neighbor, and Support Vector Machine. Decision trees provide interpretable rule structures, Naive Bayes offers low training and prediction complexity, and k-NN provides a non-parametric distance-based comparison supported by prior IDS work [3], [4]. The SVM baseline uses a radial-basis-function kernel because support-vector learning is designed to construct maximum-margin decision surfaces in transformed feature spaces [14]. Hyperparameters are selected only from training data using stratified cross-validation.

Random Forest is proposed as the primary supervised classifier. It aggregates many randomized decision trees, reducing the instability of an individual tree while providing internal estimates of feature importance [13]. For an efficient IDS configuration, the number of trees is increased only until validation performance stabilizes. Class weighting is applied if attack categories are substantially imbalanced. The classifier outputs a class probability or voting confidence for each connection.

A low-confidence gate is then used to decide whether a connection should be accepted immediately or sent to a second-stage k-means anomaly screen. The clustering model is trained primarily on transformed normal traffic, or on carefully separated normal/attack patterns depending on the experimental configuration. Duque and Omar [11] demonstrated that k-means behavior is sensitive to the number of clusters; therefore, k is selected using training-only validation criteria rather than an arbitrary large value. The final alert combines supervised confidence and anomaly distance. This

architecture saves clustering cost for uncertain cases while retaining a mechanism for detecting patterns poorly represented by labeled training examples.

4. Decision Logic

For each incoming record x , the primary classifier produces an attack probability $p(x)$. If $p(x)$ is above the attack threshold, the record is flagged as intrusive; if $p(x)$ is below the normal threshold, it is accepted as normal. Records between the two thresholds enter the anomaly stage. The normalized distance to the nearest learned cluster centroid is compared with a training-derived anomaly limit. A large distance generates an alert; otherwise, the record is returned as normal. Thresholds are selected to prioritize detection rate while constraining the false-positive rate, because excessive false alarms can make an IDS operationally unusable [6].

5. Implementation Procedure and Reproducibility

The implementation proceeds in a fixed sequence so that the experiment can be reproduced in WEKA, Python, R, or an equivalent data-mining environment. First, the original train and test files are loaded without re-sampling the final test set. Second, the training labels are collapsed to normal/attack for the binary experiment, and the preprocessing transformer is fitted on training records only. Third, feature ranking is calculated inside the training partition and the selected feature list is frozen. Fourth, the baseline classifiers and Random Forest are tuned using the same cross-validation folds. Fifth, confidence thresholds for the primary detector and the anomaly-distance threshold are chosen on validation data. Finally, the frozen pipeline is applied once to the held-out test set and the full

confusion matrix, runtime, and selected feature count are recorded.

Reproducibility also requires reporting the software version, random seed, hardware used for timing, selected features, hyperparameters, and exact class mapping. Any tuning performed after examining the test labels would invalidate the comparison. If a researcher chooses WEKA, the updated workbench documented by Hall et al. [8] provides the necessary preprocessing, classification, clustering, and evaluation components. The same principle applies in other platforms: every transformation must be learned from training data and serialized with the model.

6. Experimental Design and Evaluation Metrics

The experiment uses the fixed NSL-KDD train/test split whenever available. Within the training split, stratified five-fold cross-validation is used for feature-count selection and model

hyperparameters. Model comparison must be based on the same transformed feature set and identical test instances. Training time and average prediction time are recorded in addition to classification quality because the stated objective is an efficient IDS, not simply a highly accurate offline classifier.

The confusion matrix contains true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). Accuracy is $(TP+TN)/(TP+TN+FP+FN)$. Detection rate or recall is $TP/(TP+FN)$; precision is $TP/(TP+FP)$; F1-score is the harmonic mean of precision and recall; and false-positive rate is $FP/(FP+TN)$. ROC curves and area-based comparison may be used to inspect threshold behavior, following the evaluation principles summarized by Fawcett [15]. For an IDS, detection rate and false-positive rate should be interpreted together because a very sensitive detector can become impractical if it floods analysts with benign alerts.

Table 2. Evaluation measures used for the proposed IDS

Measure	Formula / Definition	Purpose
Accuracy	$(TP + TN) / \text{All records}$	Overall classification correctness
Detection rate (Recall)	$TP / (TP + FN)$	Ability to identify attacks
Precision	$TP / (TP + FP)$	Reliability of generated attack alerts
F1-score	$2 \times \text{Precision} \times \text{Recall} / (\text{Precision} + \text{Recall})$	Balance of alert precision and detection
False-positive rate	$FP / (FP + TN)$	Operational false-alarm burden
Training / prediction time	Elapsed time under fixed hardware/software	Efficiency and deployability

Results

Because this paper is a methodology study, the results below are reported as benchmark-oriented findings and acceptance criteria grounded in published experiments through 2015; they should not be interpreted as measurements from a newly deployed production network. The literature indicates that data-mining IDS performance depends strongly on the classifier, feature representation, and evaluation design. Ben Amor et al. [4] found that Naive Bayes could remain competitive with decision-tree methods on KDD'99. Peddabachigari et al. [5] reported that hybrid models combining decision trees and SVMs improved the accuracy/complexity balance relative to individual approaches. Nadiammal and Hemalatha [10] similarly reported improved accuracy and reduced false alarms for data-mining-oriented IDS algorithms on KDD-based experiments.

For the anomaly stage, Duque and Omar [11] provide a useful quantitative warning. On a

25,192-record NSL-KDD subset, their k-means configuration produced reported efficiency rates of 70.75%, 81.61%, 65.40%, 61.30%, and 55.43% for 11, 22, 44, 66, and 88 clusters, respectively. The 22-cluster condition achieved the highest reported efficiency (81.61%) with a 4.03% false-positive rate, but the study also reported very high false-negative rates. These observations support the present choice to use clustering as a secondary screen rather than the sole detector. They also show why k should be selected empirically rather than increased without validation.

The proposed methodology is therefore considered successful when the primary supervised stage improves attack detection over the clustering-only reference while the low-confidence gate keeps the anomaly stage limited to a minority of records. A well-performing implementation should also show that feature reduction preserves or improves F1-score while lowering training and prediction cost. Random Forest is expected to be a strong primary

candidate because randomized tree ensembles are robust to noisy predictors and provide feature-importance estimates [13], while SVM remains an important high-capacity comparator based on the margin-learning principle of Cortes and Vapnik [14].

Table 3 summarizes the benchmark-oriented interpretation. The values reported for the k-means row are taken directly from the 2015 study of Duque and Omar [11]; the remaining rows specify qualitative performance expectations to be tested under the proposed common protocol, rather than fabricated numerical scores. This distinction keeps the methodology reproducible and avoids claiming

empirical results that have not been independently executed for this paper.

A final analysis should compare the full 41-feature model with the reduced-feature model. If the reduced model retains comparable detection and F1 performance while lowering prediction time and the number of stored attributes, the feature-selection stage can be regarded as effective. Conversely, if attack recall falls sharply, the selection threshold must be relaxed. This ablation-style comparison is important because an IDS advertised as efficient should demonstrate where the computational saving is obtained rather than assuming that fewer features always produce a better detector.

Table 3. Benchmark-oriented result interpretation

Method / Stage	Evidence available by 2015	Interpretation for this methodology
Naive Bayes / Decision Tree	Competitive KDD-based classification reported in [4]	Retain as low-complexity and interpretable baselines
Hybrid DT-SVM	Improved hybrid IDS behavior reported in [5]	Supports multi-stage or combined decision logic
Data-mining IDS	Improved accuracy / reduced false alarms reported in [10]	Supports feature-driven supervised detection
k-means, 22 clusters	81.61% efficiency; 4.03% FPR reported in [11]	Useful as secondary anomaly screen, not sole detector
Proposed RF + anomaly gate	To be measured under Section 6 protocol	Target: higher recall than clustering-only with controlled FPR and lower cost after feature reduction

Conclusion and Discussion

This paper has presented a structured methodology for an efficient intrusion detection system using data-mining techniques for network security. The method begins with disciplined preprocessing of NSL-KDD traffic, removes weak or redundant dimensions, compares established classifiers, and uses Random Forest as the primary supervised model. A second-stage k-means anomaly screen handles low-confidence traffic so that the system does not depend entirely on either signature-like supervised recognition or unsupervised clustering. The design is consistent with the data-mining feature-construction approach of Lee and Stolfo [2], the hybrid-classifier evidence of Peddabachigari et al. [5], and the anomaly-detection challenges summarized by Garcia-Teodoro et al. [6]. The central methodological lesson is that efficiency must be defined jointly in terms of security performance and operational cost. Accuracy alone can conceal poor attack recall or an unacceptable false-alarm rate. Published 2015 clustering results [11] demonstrate that unsupervised methods can be sensitive to parameter choice and may miss many malicious records even when false

positives appear modest. Therefore, the proposed framework restricts clustering to uncertain samples and evaluates detection rate, precision, F1-score, false-positive rate, and computation time together. The use of NSL-KDD improves comparability relative to the redundancy-prone KDD Cup 99 benchmark [7], but it does not eliminate the gap between benchmark traffic and live networks. Future implementation should extend the binary model to attack-family classification, test the framework on newer real traffic, and examine concept drift, encrypted traffic, and online model updating. Any production deployment should also incorporate analyst feedback, logging, threshold recalibration, and integration with conventional signatures. In this way, data mining functions as an adaptive analytical layer within a broader defense architecture rather than as a replacement for all existing security controls.

References

D. E. Denning, "An Intrusion-Detection Model," *IEEE Transactions on Software Engineering*, vol. SE-13, no. 2, pp. 222-232, 1987. DOI: 10.1109/TSE.1987.232894.

- W. Lee and S. J. Stolfo, "A Framework for Constructing Features and Models for Intrusion Detection Systems," *ACM Transactions on Information and System Security*, vol. 3, no. 4, pp. 227-261, 2000. DOI: 10.1145/382912.382914.
- Y. Liao and V. R. Vemuri, "Use of K-Nearest Neighbor Classifier for Intrusion Detection," *Computers & Security*, vol. 21, no. 5, pp. 439-448, 2002. DOI: 10.1016/S0167-4048(02)00514-X.
- N. Ben Amor, S. Benferhat, and Z. Elouedi, "Naive Bayes vs Decision Trees in Intrusion Detection Systems," *Proceedings of the 2004 ACM Symposium on Applied Computing*, pp. 420-424, 2004. DOI: 10.1145/967900.967989.
- S. Peddabachigari, A. Abraham, C. Grosan, and J. Thomas, "Modeling Intrusion Detection System Using Hybrid Intelligent Systems," *Journal of Network and Computer Applications*, vol. 30, no. 1, pp. 114-132, 2007. DOI: 10.1016/j.jnca.2005.06.003.
- P. Garcia-Teodoro, J. E. Diaz-Verdejo, G. Macia-Fernandez, and E. Vazquez, "Anomaly-Based Network Intrusion Detection: Techniques, Systems and Challenges," *Computers & Security*, vol. 28, nos. 1-2, pp. 18-28, 2009. DOI: 10.1016/j.cose.2008.08.003.
- M. Tavallaei, 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*, pp. 1-6, 2009. DOI: 10.1109/CISDA.2009.5356528.
- M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA Data Mining Software: An Update," *ACM SIGKDD Explorations Newsletter*, vol. 11, no. 1, pp. 10-18, 2009. DOI: 10.1145/1656274.1656278.
- R. Sommer and V. Paxson, "Outside the Closed World: On Using Machine Learning for Network Intrusion Detection," *2010 IEEE Symposium on Security and Privacy*, pp. 305-316, 2010. DOI: 10.1109/SP.2010.25.
- G. V. Nadiammal and M. Hemalatha, "Effective Approach toward Intrusion Detection System Using Data Mining Techniques," *Egyptian Informatics Journal*, vol. 15, no. 1, pp. 37-50, 2014. DOI: 10.1016/j.eij.2013.10.003.
- S. Duque and M. N. bin Omar, "Using Data Mining Algorithms for Developing a Model for Intrusion Detection System (IDS)," *Procedia Computer Science*, vol. 61, pp. 46-51, 2015. DOI: 10.1016/j.procs.2015.09.145.
- T. Xu, W. Zhang, X. Li, X. Wang, and W. Pan, "A Novel Intrusion Detection System Based on Data Mining," *Proceedings of the 2015 4th International Conference on Computer, Mechatronics, Control and Electronic Engineering*, pp. 1299-1302, 2015. DOI: 10.2991/iccmcee-15.2015.244.
- L. Breiman, "Random Forests," *Machine Learning*, vol. 45, pp. 5-32, 2001. DOI: 10.1023/A:1010933404324.
- C. Cortes and V. Vapnik, "Support-Vector Networks," *Machine Learning*, vol. 20, pp. 273-297, 1995. DOI: 10.1007/BF00994018.
- T. Fawcett, "An Introduction to ROC Analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861-874, 2006. DOI: 10.1016/j.patrec.2005.10.010.