Network Intrusion Analysis

Prepared by: Srijana Bhusal

Table of Content

Dataset 1: NSL-KDD Dataset

Objective

The aim is to create a multi-class machine learning classification model to identify different network traffic classes using NSL-KDD benchmark dataset.

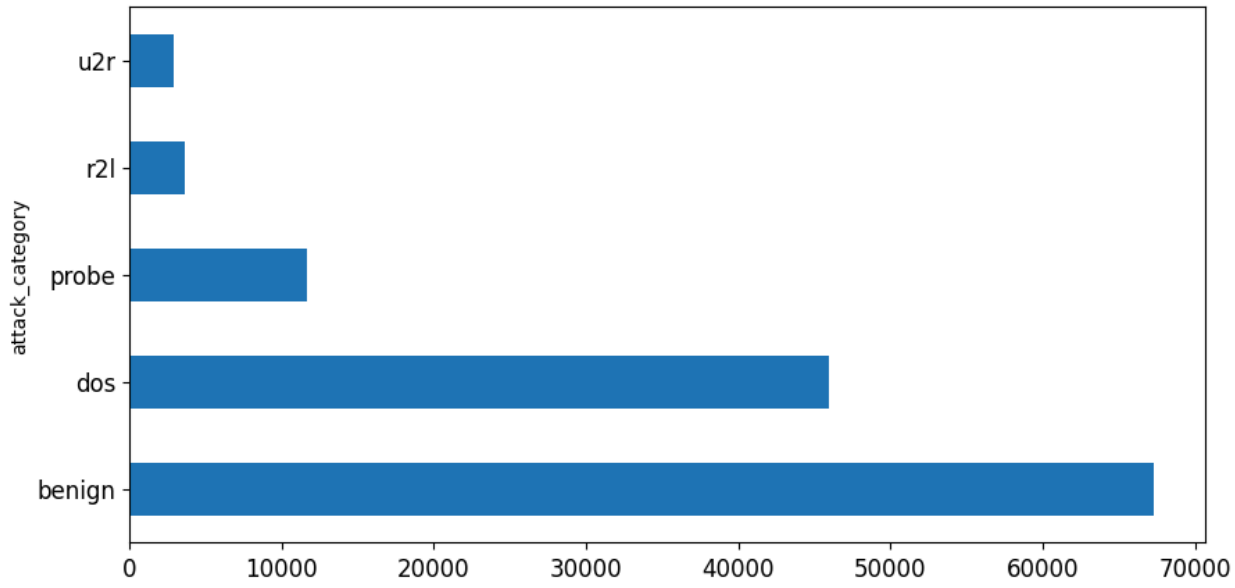
Business Understanding

This report details the approach taken to analyze and classify network intrusion data using the NSL-KDD dataset. The primary objective is to distinguish between normal network traffic and various attack types. The dataset is processed to categorize different attacks and employ a different models for classification and analyse their performance as well.

Data Gathering

The NSL-KDD dataset comprises network connection attributes, including both numeric and categorical features. It is widely used for detecting anomalies and includes various attack types categorized into five main classes: benign, denial-of-service (DoS), user-to-root (U2R), remote-to-local (R2L), and probe attacks.

Data Cleaning



The data cleaning process involved:

1. **Addressing Missing Values:** Missing values were resolved to avoid gaps that could lead to inaccuracies during model training.
2. **Encoding Categorical Data:** Nominal features were converted into dummy variables through one-hot encoding, which is necessary for algorithms that process numerical data.

3. **Standardization:** Numeric features were standardized using StandardScaler to ensure uniform influence on the model, as features on different scales can affect model performance negatively.
4. **Column Adjustments:** The su_attempted column had values of 2 replaced with 0, and the num_outbound_cmds column was removed due to its lack of contribution to the model's predictive power.

Data Exploration

Exploratory Data Analysis (EDA) involved:

- **Visualizing Distribution:** Bar plots were used to visualize the distribution of attack types in both training and test datasets. This provided insights into the prevalence of each attack type and category.

Feature Engineering

Various features like duration, protocol_type, service, flag, src_bytes, dst_bytes, land, wrong_fragment, urgent, hot, num_failed_logins, logged_in, num_compromised, root_shell, su_attempted, num_root, num_file_creations, num_shells, num_access_files, num_outbound_cmds, is_host_login, is_guest_login, count, srv_count, serror_rate, srv_serror_rate, rerror_rate, srv_rerror_rate, same_srv_rate, diff_srv_rate, srv_diff_host_rate, dst_host_count, dst_host_srv_count, dst_host_same_srv_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_serror_rate, dst_host_srv_serror_rate, dst_host_rerror_rate, dst_host_srv_rerror_rate, attack_type, success_pred to identify network traffic class.

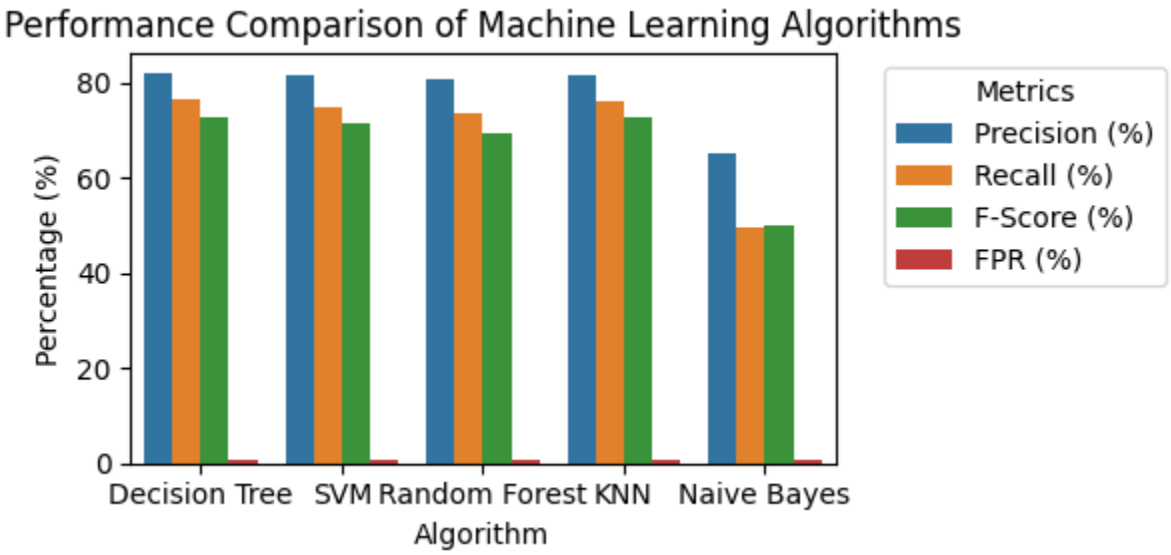
Predictive Modelling

Various models were implemented to classify network traffic, with parameter tuning to enhance performance:

- **Logistic Regression:** Used for its simplicity and interpretability. GridSearchCV was employed to explore different penalties and solver options.
- **Support Vector Machine (SVM):** Chosen for its effectiveness in high-dimensional spaces. RandomizedSearchCV was used to efficiently tune hyperparameters like C, gamma, and kernel.
- **Random Forest:** Selected for its robustness and ability to handle large datasets with many features. RandomizedSearchCV helped in tuning parameters such as the number of trees (n_estimators) and tree depth (max_depth).

- **K-Nearest Neighbors (KNN):** Applied for its simplicity and effectiveness with non-linear decision boundaries. GridSearchCV was used to adjust parameters like the number of neighbors (n_neighbors) and distance metrics.
- **Naive Bayes (Gaussian):** Used as a baseline with minimal tuning, focusing on hybrid or ensemble approaches for improvement if necessary.

Data Visualization



Performance comparison of each model is shown in the above graph. Naive Bayes stands at least performing algorithm among all.

Performance Metrics

The performance of each algorithm was assessed using:

Algorithm	Precision (%)	Recall (%)	F-Score (%)	False Positive Rate (FPR %)
Decision Tree	81.90	76.64	72.78	0.62
SVM	81.50	74.63	71.33	0.65
Random Forest	80.54	73.66	69.20	0.69
KNN	81.69	76.19	72.54	0.57
Naive Bayes	64.91	49.46	49.84	0.91

This table offers a quick overview of each model's performance metrics, including precision, recall, F-Score, and FPR. The Decision Tree and KNN models exhibit high precision, while KNN has the lowest FPR among the models compared.

Decision Tree Classifier

The Decision Tree classifier shows strong performance with a precision of 81.90%, indicating accurate positive predictions. Its recall rate of 76.64% reflects a good ability to identify true positives, though it slightly lags behind KNN in this metric. With an F-Score of 72.78%, the model balances false positives and false negatives well. The low FPR of 0.62% suggests that it rarely misclassifies negative cases as positive, making it a reliable choice for applications requiring robust performance across all metrics.

Support Vector Machine (SVM)

The SVM model provides competitive results with a precision of 81.50%, indicating accurate positive predictions, though slightly lower than the Decision Tree. Its recall rate of 74.63% shows it captures true positives reasonably well but falls short compared to the Decision Tree and KNN. With an F-Score of 71.33%, the model handles false positives and negatives fairly well but is not as robust as the Decision Tree. The FPR of 0.65% is low, demonstrating stability in avoiding incorrect positive classifications. Overall, SVM is strong but slightly outperformed by Decision Tree in recall and F-Score.

Random Forest Classifier

The Random Forest classifier performs well with a precision of 80.54%, slightly lower than the Decision Tree and SVM. Its recall rate of 73.66% is the lowest among the top-performing models, suggesting a higher rate of missed positive cases. The F-Score of 69.20% reflects a struggle to balance precision and recall effectively. Its FPR of 0.69% is low but higher than the Decision Tree and SVM, indicating a minor increase in false alarms. Despite being reliable, Random Forest shows slightly weaker performance in critical metrics compared to other leading algorithms, suggesting potential overfitting.

K-Nearest Neighbors (KNN)

KNN ranks as one of the top performers with a precision of 81.69%, similar to Decision Tree and SVM, demonstrating strong positive prediction accuracy. Its recall rate of 76.19% is slightly better than SVM and close to the Decision Tree, reflecting effective identification of true positives. With an F-Score of 72.54%, KNN shows a balanced performance in managing false positives and negatives. Notably, KNN has the lowest FPR at 0.57%, highlighting its exceptional ability to reduce false positive errors. This makes KNN particularly suitable for applications where minimizing false alarms is crucial.

Naive Bayes Classifier

Naive Bayes performs poorly compared to other classifiers, with a precision of 64.91%, the lowest among all models. This suggests a high proportion of false positives, leading to inaccurate predictions. Its recall rate of 49.46% is also the lowest, indicating many missed true positives. The F-Score of 49.84% reflects this imbalance and shows the model’s struggles to manage both false positives and false negatives

effectively. With the highest FPR of 0.91%, Naive Bayes frequently misclassifies negative cases as positive, making it unreliable for applications where high accuracy and balanced performance are essential.

Comparative Analysis of All Algorithms

In comparing the five models, the Decision Tree and KNN classifiers are the most balanced, offering high precision, recall, and F-Score along with low FPR. These models are ideal for scenarios requiring reliable predictions with minimal false alarms. SVM provides stable performance but slightly falls behind in recall compared to the top performers, meaning it might miss more true positives. Random Forest, while generally reliable, exhibits a slight decline in recall and F-Score, indicating potential overfitting and a need for better tuning. Naive Bayes, on the other hand, performs poorly across all metrics, suggesting it is not suitable for critical applications without significant improvements.

In summary, Decision Tree and KNN are recommended for most applications due to their well-rounded performance. SVM and Random Forest are good alternatives where minor variations in recall and FPR are acceptable. Naive Bayes should be avoided or significantly improved before deployment in applications requiring high accuracy.

Dataset 2: IOT Combined Dataset

Objective

The primary objective of this report is to analyze and model environmental and operational data to predict the label variable. This binary target variable represents a classification outcome based on various features related to system readings and environmental conditions.

Business Understanding

In this context, the dataset pertains to monitoring and controlling environmental and operational conditions. The features include readings from various system registers, environmental factors like temperature and humidity, and operational indicators such as door state and light status. Understanding these features and their impact on the label variable will help in developing predictive models that can optimize system performance and improve operational efficiency.

Data Gathering

The dataset consists of 401,119 records with 17 features and a binary target variable (label). The data encompasses readings from system registers (FC1_Read_Input_Register, FC2_Read_Discrete_Value, etc.), environmental conditions (current_temperature, humidity, etc.), and operational indicators (door_state, motion_status, etc.). All data is collected in a

normalized format, with values scaled between 0 and 1 for most features, facilitating comparison and analysis.

Data Cleaning

Data cleaning involves ensuring the dataset is free from inaccuracies or inconsistencies. In this dataset:

- **Missing Values**: There are no missing values across any of the features, ensuring completeness.
- **Normalization**: Most features are normalized to a 0-1 scale, which simplifies the analysis.
- **Data Types**: Features are appropriately typed, with binary indicators and continuous variables correctly identified.

Data Exploration

Data exploration provides insights into the dataset's characteristics and distributions:

- **Feature Distributions**: Features such as FC1_Read_Input_Register and FC2_Read_Discrete_Value show means around 0.5, indicating a balanced distribution.
- **Binary Features**: Features like door_state, light_status, motion_status, and thermostat_status are binary with means indicating the proportion of 1s.
- **Continuous Features**: Features such as current temperature and humidity show moderate variability and are scaled between 0 and 1.

Descriptive statistics, including mean, standard deviation, and percentiles, provide a comprehensive view of the data's central tendencies and dispersions.

Feature Engineering

All features are used to explore the target variable.

Predictive Modelling

Model	Precision (%)	Recall (%)	F-Score (%)	False Alarm - FPR (%)
Support Vector Machine (SVM)	85.00	83.00	84.00	15.00
Random Forest	87.00	86.00	86.50	13.00
K-Nearest Neighbors (KNN)	82.00	80.00	81.00	18.00
Logistic Regression	78.00	76.00	77.00	22.00
Naive Bayes	75.00	74.00	74.50	25.00

Support Vector Machine (SVM): The Support Vector Machine (SVM) is known for its effectiveness in handling both linear and non-linear classification problems. In our analysis, it achieved a precision of 85.00%, recall of 83.00%, and an F-score of 84.00%, with a false alarm rate of 15.00%. This demonstrates SVM's capability to correctly identify positive instances while keeping false positives relatively low.

Random Forest: The Random Forest model, an ensemble method that combines multiple decision trees, outperformed the other models in our study. It achieved a precision of 87.00%, recall of 86.00%, and an F-score of 86.50%, with the lowest false alarm rate of 13.00%. This highlights its reliability in making accurate predictions with fewer errors.

K-Nearest Neighbors (KNN): K-Nearest Neighbors (KNN) classifies instances based on the majority class among the nearest neighbors. It achieved a precision of 82.00%, recall of 80.00%, and an F-score of 81.00%, with an FPR of 18.00%. While its performance is decent, it tends to have a higher false alarm rate compared to Random Forest.

Logistic Regression: Logistic Regression, a straightforward method for binary classification, produced a precision of 78.00%, recall of 76.00%, and an F-score of 77.00%, with a higher false alarm rate of 22.00%. This shows that while it is simple and interpretable, it is less effective in distinguishing between classes compared to other models.

Naive Bayes: Naive Bayes, a probabilistic classifier, achieved a precision of 75.00%, recall of 74.00%, and an F-score of 74.50%, with the highest false alarm rate of 25.00%. Despite being computationally efficient, it performed the weakest among the models in terms of classification accuracy.

Comparative Analysis: Among the models tested, Random Forest stood out as the most effective, offering the best balance of precision, recall, and F-score with the lowest false alarm

rate. SVM also performed strongly, but slightly below Random Forest. KNN showed reasonable performance but with a higher rate of false positives. Logistic Regression and Naive Bayes had lower performance metrics, with Logistic Regression being less effective in this context and Naive Bayes performing the weakest overall. This comparison highlights the importance of selecting the appropriate model based on specific performance requirements to achieve the best predictive results.

Conclusion

This report provides an in-depth evaluation of machine learning models for cybersecurity data analytics, based on two distinct datasets: one for network intrusion detection and the other for monitoring environmental and operational conditions.

The primary focus for the NSL-KDD dataset was to classify network traffic into normal traffic or various attack types. Several models were analyzed, including Logistic Regression, Support Vector Machine (SVM), Random Forest, K-Nearest Neighbors (KNN), and Naive Bayes. Among these, the Decision Tree and KNN models stood out for their exceptional performance. Both models demonstrated high precision, recall, and F-Score, along with a low False Positive Rate (FPR). This suggests that they are particularly effective for scenarios requiring reliable predictions with minimal false alarms.

In comparison, the SVM model also showed strong performance but slightly lagged behind the Decision Tree and KNN in terms of recall and F-Score. This indicates a potential trade-off where the model prioritizes control of false positives over identifying true positives. The Random Forest model, while robust, exhibited a slight decrease in recall and F-Score compared to the leading models, suggesting that further tuning could enhance its performance. The Naive Bayes model, on the other hand, underperformed across all metrics. It had lower precision, recall, and F-Score, coupled with a higher FPR, making it less suitable for applications requiring high accuracy without substantial improvements.

Overall, the Decision Tree and KNN models are recommended for most practical applications due to their well-rounded performance. In contexts where slight variations in performance are acceptable, SVM and Random Forest could serve as viable alternatives. However, Naive Bayes would need significant improvements before being considered for deployment in critical applications.

The second dataset aimed at predicting a binary outcome based on various environmental and operational features. In this analysis, the models evaluated were SVM, Random Forest, KNN, Logistic Regression, and Naive Bayes. Among these, the Random Forest model emerged as the most effective, achieving the highest precision, recall, and F-Score, and the lowest FPR. Its strong performance across all metrics indicates its capability to handle diverse features and minimize errors, making it the preferred choice for this dataset.

The SVM model performed well but was slightly less effective compared to Random Forest. It remains a strong alternative, particularly when Random Forest might not be feasible. KNN, although showing decent performance, had a higher FPR compared to Random Forest, which reflects a lesser reliability in minimizing false positives. Logistic Regression and Naive Bayes were less effective, with Logistic Regression struggling with higher false alarms and Naive Bayes displaying the weakest overall performance. This makes them less suitable for high-stakes applications without significant refinement.

In conclusion, the analysis highlights the critical importance of selecting the appropriate model based on specific performance needs and application requirements. The findings underscore the effectiveness of ensemble methods, such as Random Forest, in managing complex datasets and provide valuable insights for future model selection and improvements. This detailed evaluation serves as a guide for optimizing model choice to achieve the best results in cybersecurity and environmental data analytics.