Project Report: Anomaly Detection using UNSW-NB15 Dataset

Purna Jadhav, Daniel Cersosimo, Harshini Akuleti, Naga Himachand Pasupuleti, Avantika Deshmukh

1. Introduction

1.1 Background

In today's digital landscape, cybersecurity is of utmost importance due to the increasing frequency and sophistication of cyber threats. Intrusion detection systems (IDS) are pivotal in safeguarding network infrastructure by identifying and thwarting potential threats. Leveraging machine learning techniques on extensive network traffic datasets can significantly bolster the effectiveness of IDS. In this project, we aim to delve into the UNSW-NB15 dataset, curated by Dr. Nour Moustafa, which provides detailed insights into various network intrusion occurrences. By harnessing this dataset, our objective is to construct a robust machine learning model capable of accurately identifying anomalous activity.

1.2 Objectives

Our project's primary objectives are as follows:

- Data Exploration and Preprocessing: Thoroughly explore the UNSW-NB15 dataset, perform comprehensive data preprocessing to ensure data quality, and prepare the dataset for machine learning analysis. In addition, we seek to conduct exploratory data analysis to understand the statistical nature of the data to execute the necessary preprocessing steps
- Machine Learning Modeling: Develop, evaluate, and optimize machine learning models, including k-NN, Random Forest, SVM, Logistic Regression, XGBoost, and LightGBM, to classify network traffic as normal or anomalous

● Insights and Interpretation: Analyze model results, derive meaningful insights regarding network intrusion detection, and interpret the implications for network security and potential deployment scenarios

1.3 Scope

Our project's scope encompasses the following key aspects:

●Data Acquisition and Preprocessing: Acquire the UNSW-NB15 dataset and conduct thorough preprocessing to handle missing values, outliers, and inconsistencies.

●Exploratory Data Analysis (EDA): Employ various EDA techniques to gain insights into the dataset's underlying patterns and distributions.

●Machine Learning Modeling: Implement a diverse set of machine learning algorithms to develop robust intrusion detection models, considering both accuracy and computational efficiency.

●Performance Evaluation: Evaluate the performance of each model using appropriate evaluation metrics and select the optimal model for deployment.

●Insights and Interpretation: Derive actionable insights from model results and provide meaningful interpretations regarding network intrusion detection.

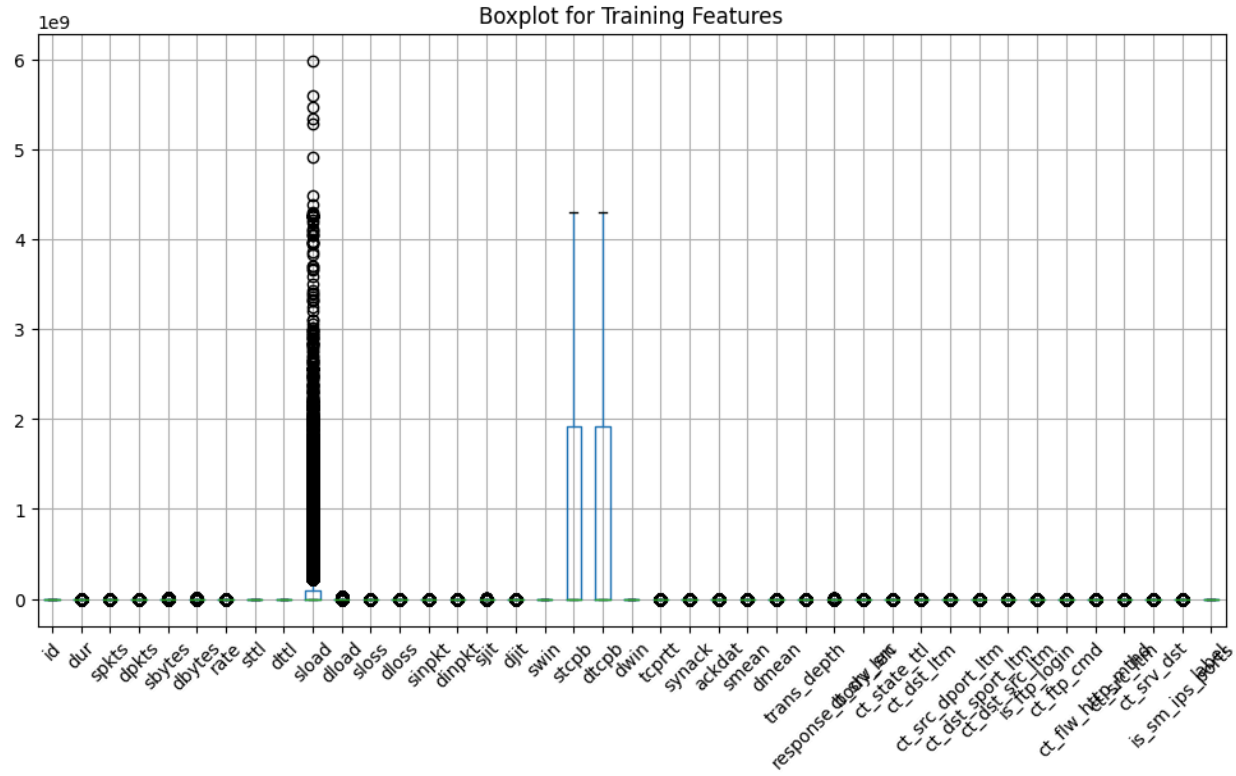
2. Data Preprocessing

2.1 Appropriateness of Data Cleaning Techniques

To ensure the reliability of our analysis, we applied standard data cleaning techniques, including:

● Handling Missing Values: Assessed the dataset for missing values of which there were none

● Outlier Detection: Identified outliers using statistical methods and opted to include in modeling as to prevent the loss of potential integral patterns for anomalous activity



Above depicts the boxplot for the training set, the test set is visually identical so we will include one to avoid redundancy. Most of the variables have values very close to zero which indicates that for these variables, either the data points are all very small, or there is very little variation among them. One variable 'sload' is with a vastly different scale than the others. It has a highly skewed distribution with a long tail of high values. The majority of the data is concentrated within a relatively narrow range, but there are significant outliers that extend far beyond this range. This suggests that the variable may have extreme values that could potentially influence the analysis.stcpb and dtcpb have distributions that are less skewed compared to sload. Although they have a wider range of values compared to most other variables, their scales are smaller, suggesting that the variability within these variables is not as extreme. The skew towards lower

values indicates that the majority of the data falls within lower ranges, with fewer outliers extending towards higher values. With the above in mind, given the context of this project to build models for anomaly detection, outlier results are often strong indicators of anomalies and therefore, given the goal of the project, may be integral in identifying instances as anomalies. Removal of outliers can be considered and would be especially viable in many contexts however, the desire to identify anomalies presents a unique situation. This is further corroborated by the vast amount of qualifying outliers based on IQR computations, and this could impact the distribution of the anomalies and normal instances if removed as well as simply being a substantial amount of data. Doing so may result in a large loss of important patterns in the data and negatively impact results. As a result, we opt to continue with the dataset as is, finalize preprocessing, and build models for anomaly detection. If performance consistently suffers, we may revisit this and employ suitable methods at that point.

●Handling Inconsistencies: Resolved inconsistencies in the dataset to ensure consistency and accuracy.

2.2 Transformation of Raw Data

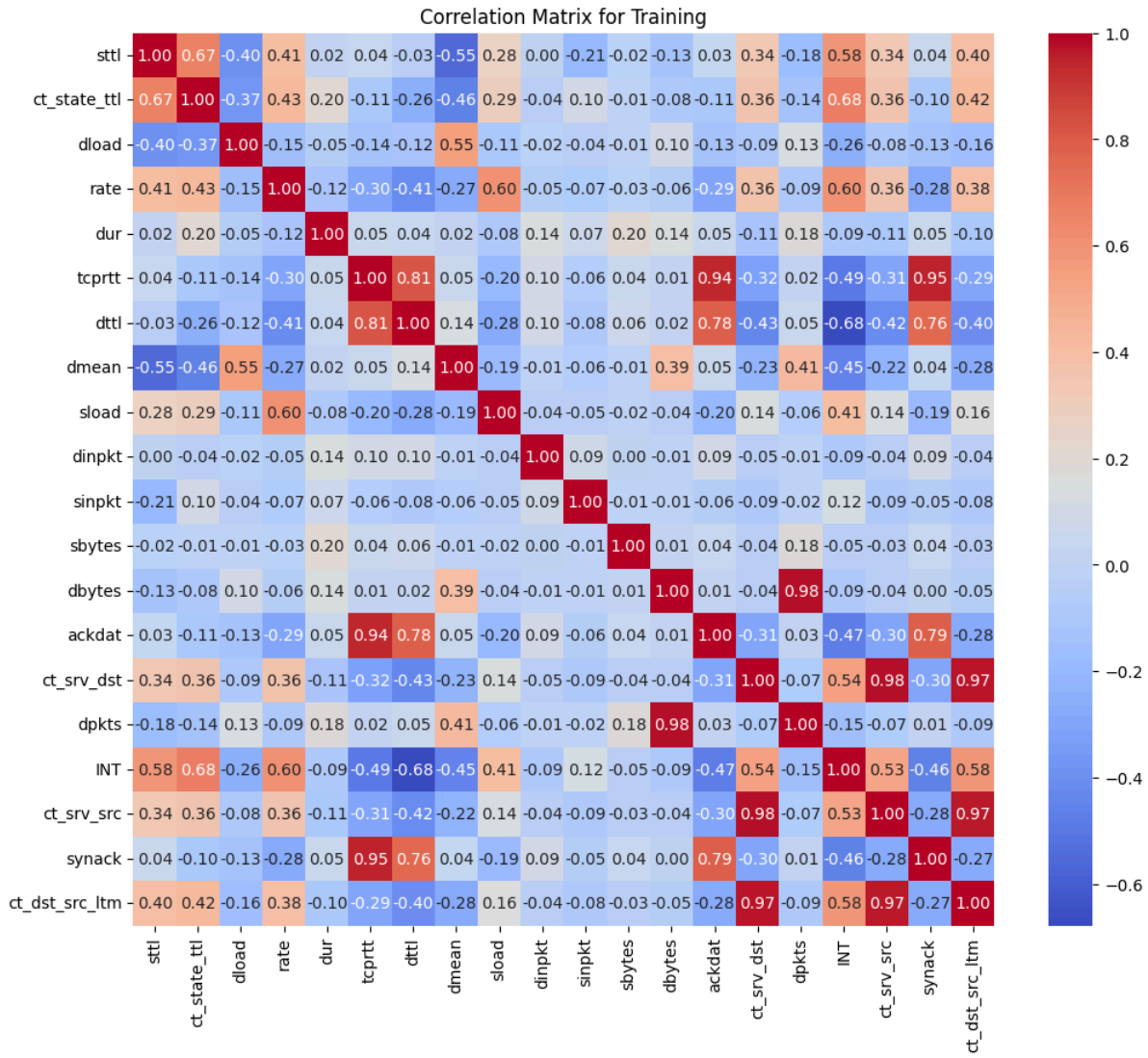
The raw data from the UNSW-NB15 dataset underwent transformation to prepare it for machine learning modeling. This involved:

● Encoding Categorical Variables: Converted categorical variables into numerical representations using techniques such as one-hot encoding after having ensured consistency in the categories between the training and test samples

● Dropped Irrelevant Features: Removed the ID column from the features as this has not bearing on the nature of the traffic instances

2.3 Feature Correlation Analysis

● Visualized Correlation Matrix: We specifically portrayed the training set correlation matrix, again similar to the test data to avoid redundancy, to analyze the nature of the relationships between the features and identify multicollinearity. There are several pairs of variables with very high positive correlation (close to 1). For example: ct_state_ttl and sttl ackdat and smean synack and smean There are also pairs with high negative correlation (values close to -1). For instance: dmean and sttl Many pairs of variables have a correlation close to 0, indicating no strong linear relationship between them. For example: dload and sbytes rate and sbytes. While we can note the presence of correlation among the features, the far majority do not exhibit such and therefore, in an effort to preserve the patterns in the natural state of this dataset as is, we opt to progress to the modeling with this preprocessed data. Again, as per above, if the models suffer, we may pivot back to this and implement necessary measures regarding multicollinearity however currently, the presence is not notable and given this dataset is specifically curated for analogous projects, we seek to at least attempt to initially model in alignment with how it was crafted. The correlation matrix is shown on the following page.



3.Machine Learning Analysis

In our machine learning analysis, we embarked on a comprehensive exploration of various algorithms to develop effective classifiers for network intrusion detection. This process involved a systematic approach, considering the strengths and limitations of each algorithm, and leveraging appropriate tools and techniques to achieve optimal performance. Let's delve into the details:

3.1.Selection and Application of Machine Learning Algorithms

k-NN (k-Nearest Neighbors):

- Selection Rationale: We opted for k-NN due to its simplicity and intuitive classification approach, which relies on the proximity to labeled instances. This made it a suitable candidate for initial exploration.
- Application: Initially, we set k=7 as a starting point. However, recognizing the need for further optimization, we sought to conduct grid search hyperparameter tuning for k however this was not viable given the computational overhead with such high dimensional data. This provided us with initial inspiration to consider PCA which will be discussed later. Without gridsearch here, this model saw .78 accuracy which compared to the later models is poor and thus we will focus on those until we conduct PCA

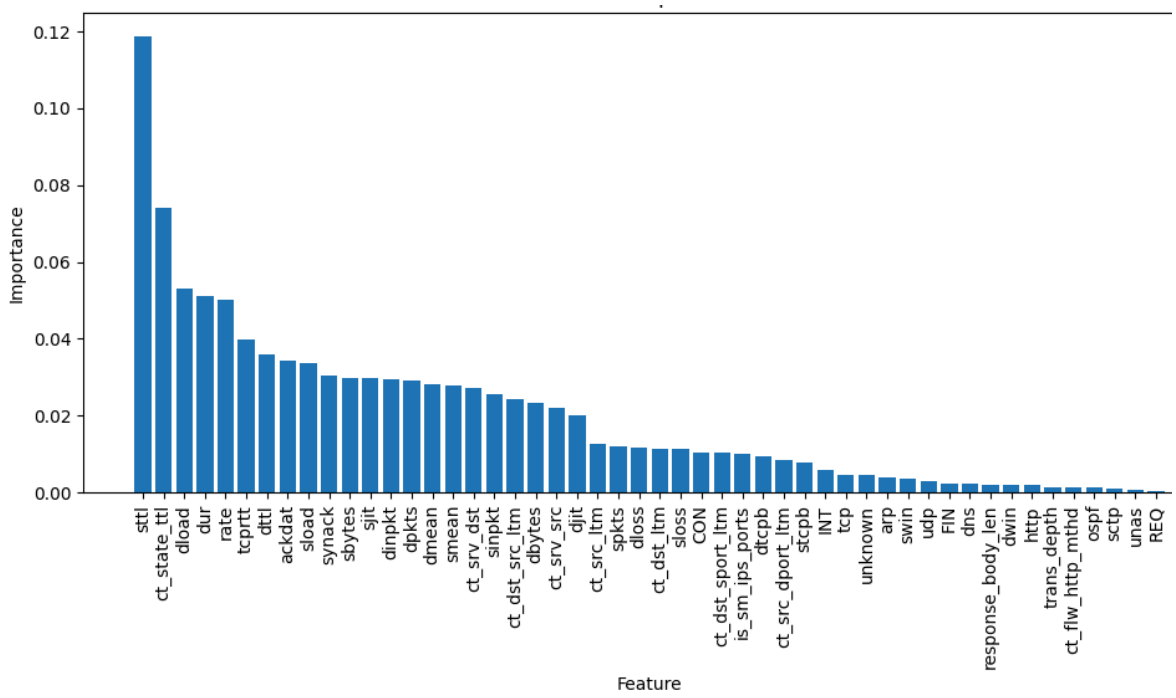
Logistic Regression:

- Selection Rationale: Logistic Regression, known for its simplicity and effectiveness in binary classification tasks, was employed to provide a baseline for comparison with more complex algorithms.
- Application: We trained a Logistic Regression model to assess its performance and interpretability, serving as a benchmark for evaluating other models' efficacy. This saw poor performance, .71 accuracy, and therefore we considered other methods

Random Forest:

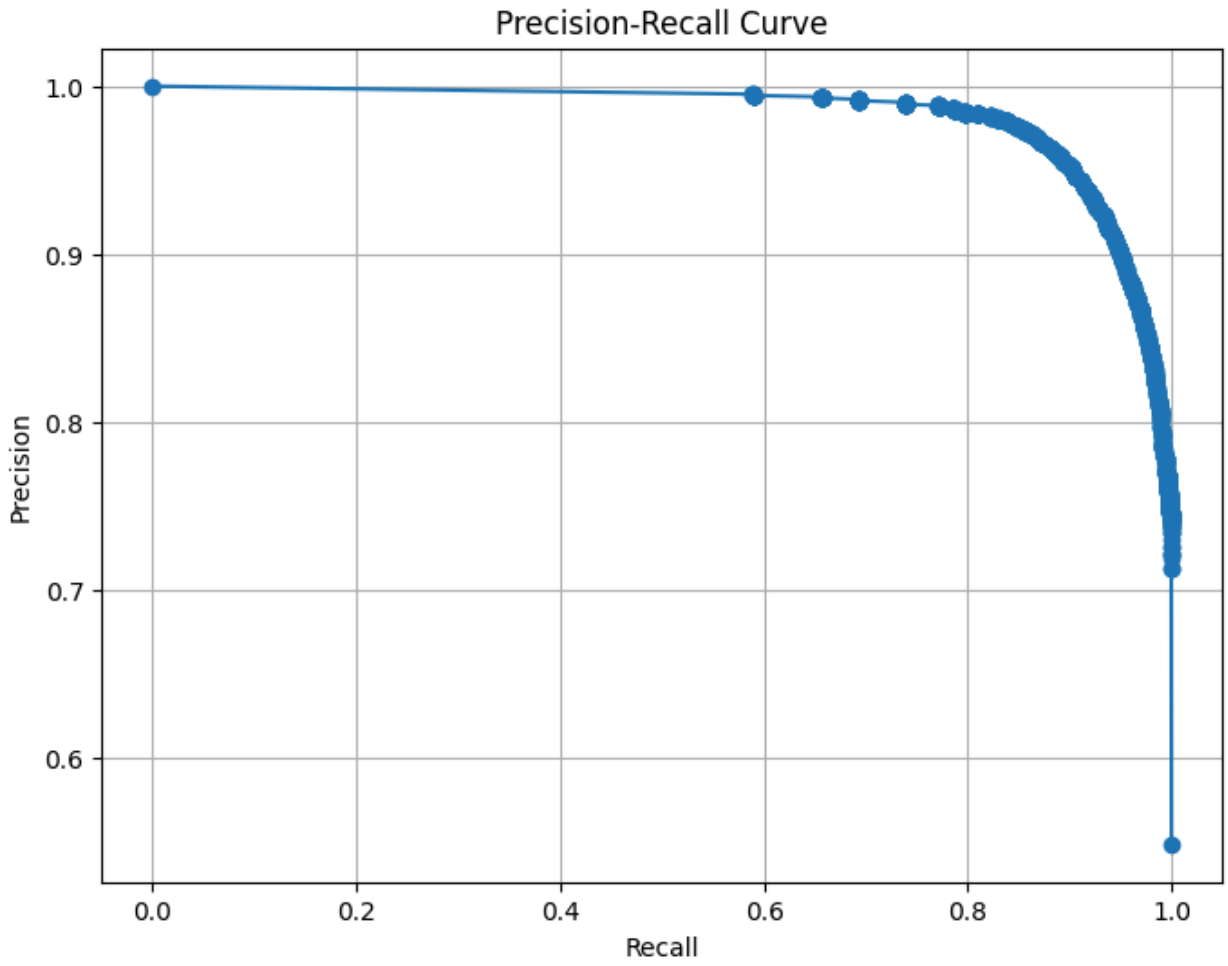
- Selection Rationale: Random Forest was chosen for its capability to handle high-dimensional data and capture complex relationships between features and the target variable.

- Application: We constructed an initial ensemble of 100 decision trees and computed feature importance to identify key features. By setting a threshold and removing less influential features, we enhanced the model's robustness and interpretability.



Subsequently, we executed a manual rough estimating procedure for the optimal number of trees in the random forest. Manually engaging in the practice as seen above is limited as we cannot encompass every value and this procedure would be automated via gridsearch or self coded logic however due to computational limitations, this is not suitable. As a result, we pivoted to test some ranges of values to roughly find the optimal number of trees in a balance with computation. Following this process, we decided on 150 trees since it offers the improved performance of the larger tree testings while offering the best balance of performance and computational overhead. Following this we executed a positive classifying threshold optimization procedure by computing the f1-score for the model at every range of possible threshold values. By threshold

we are referring to the probability of anomaly needed to be classified as such. The Precision-Recall curve for this process is visualized below.



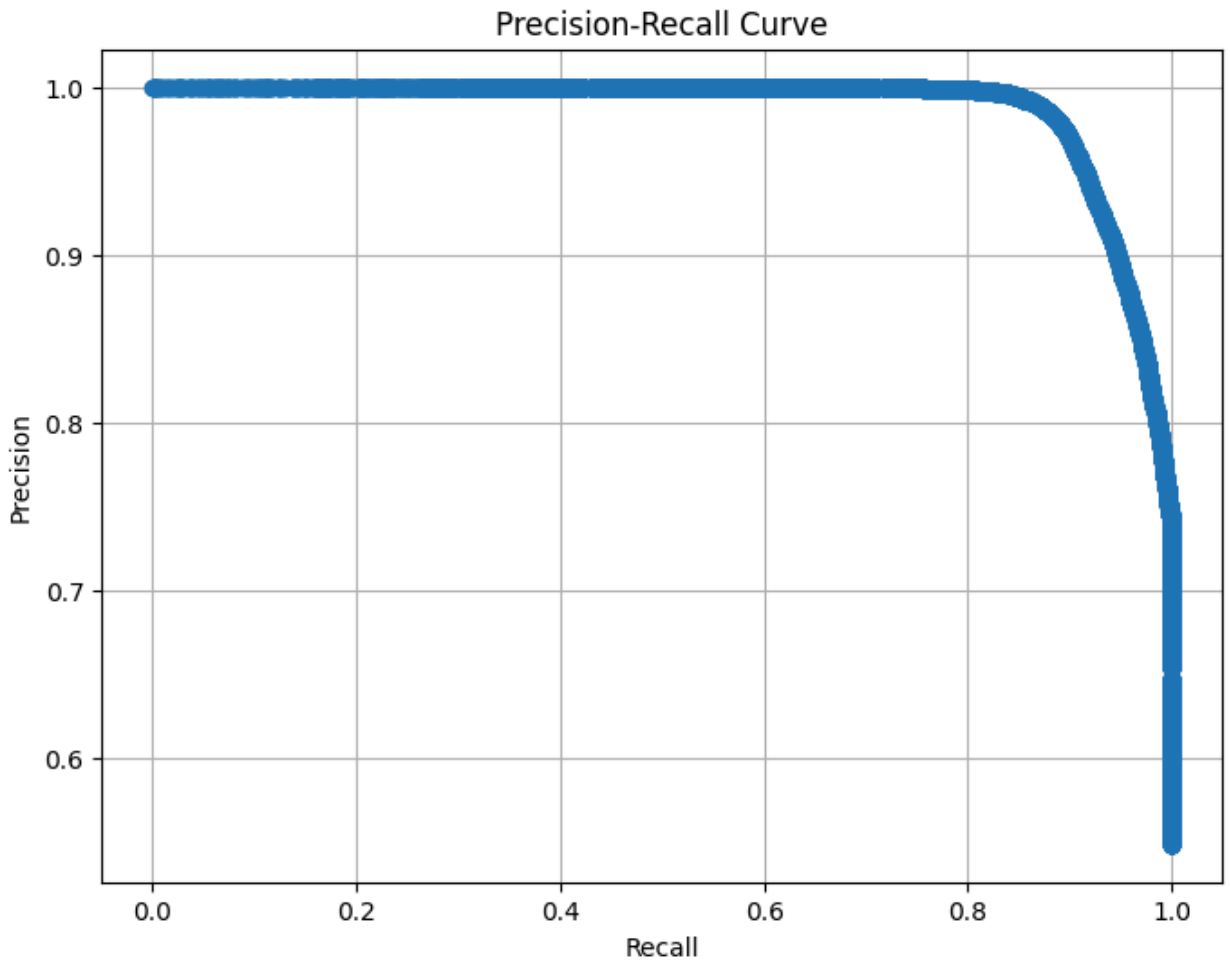
Following this, we sorted these thresholds based on their f1-score performance and selected the optimal one at 0.78. This indicates that the model sees optimal f1-score performance when a higher threshold of anomaly probability from the algorithm is needed to qualify as such. As a result, we utilized the Random Forest model to classify in accordance with this threshold and saw improved results here

	precision	recall	f1-score	support
0	0.92	0.91	0.91	18581
1	0.92	0.93	0.93	22581
accuracy			0.92	41162
macro avg	0.92	0.92	0.92	41162
weighted avg	0.92	0.92	0.92	41162

We can note the strong support for both anomalous and normal traffic instances which therefore deems accuracy as a viable metric alongside the precision, recall, and f1-score.

XGBoost:

- Selection Rationale: XGBoost was incorporated to enhance weak learners' performance, specifically decision trees, through ensemble techniques, aiming to improve classification accuracy.
- Application: XGBoost is employed to evaluate its effectiveness in improving classification performance, considering both accuracy, f1-score and computational efficiency. In order to optimize this model we conducted the same optimization practice as for the Random Forest. The Precison Recall Curve is shown below:



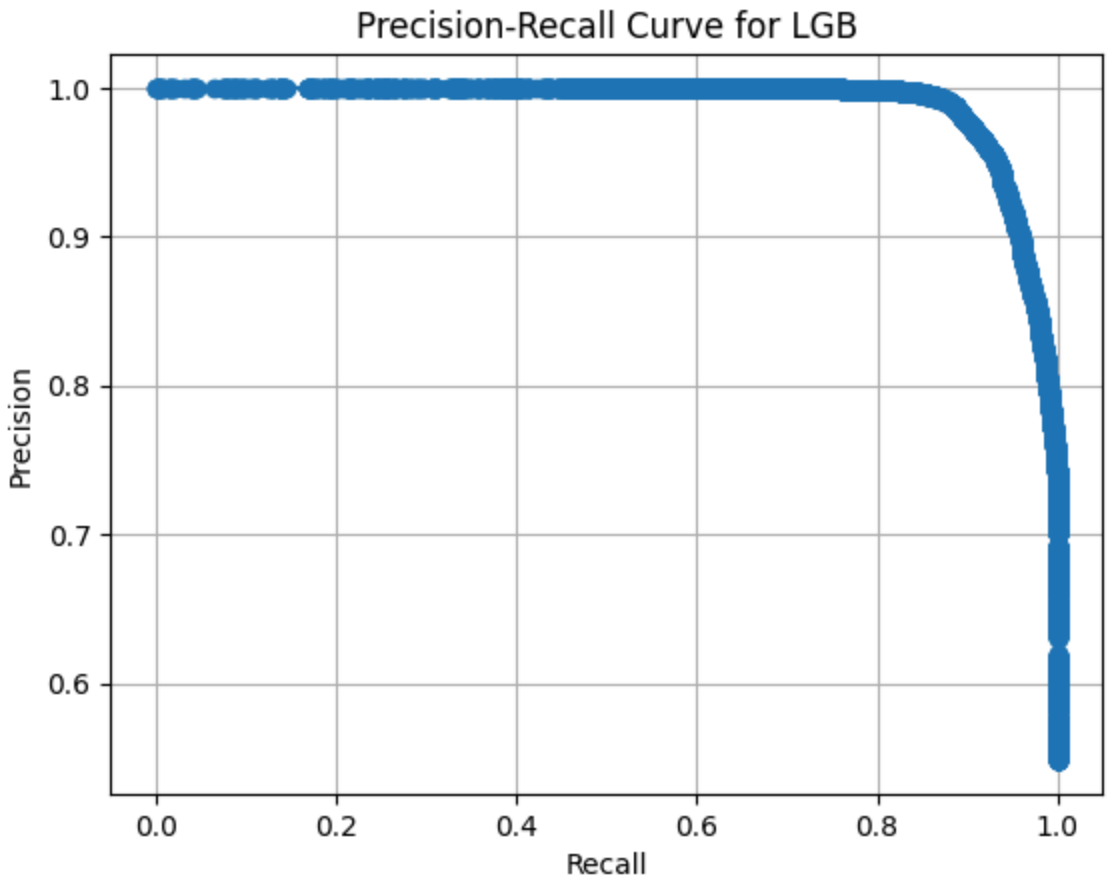
Following this, we determined the optimal threshold at .91. This indicates that for an instance to be classified as anomalous from this model, 91 percent certainty of this class if needed to be deemed as such and in turn render optimal performance. The results of applying this threshold are shown below:

	precision	recall	f1-score	support
0	0.89	0.96	0.93	18581
1	0.97	0.90	0.93	22581
accuracy			0.93	41162

macro avg	0.93	0.93	0.93	41162
weighted avg	0.93	0.93	0.93	41162

LightGBM binary classifier:

- Selection Rationale: LightGBM was incorporated to enhance weak learners' performance, specifically decision trees, through ensemble techniques, aiming to improve classification accuracy. This employs a similar approach to that of XGboost however the differences between led us to consider both to see which performs best
- Application: LightGBM is employed as a similar boosting technique to XG boost to simply compare their performances between each other and the other models. In order to optimize this model we conducted the same optimization practice as for the Random Forest and XGBoost. The Precision Recall Curve is shown below:

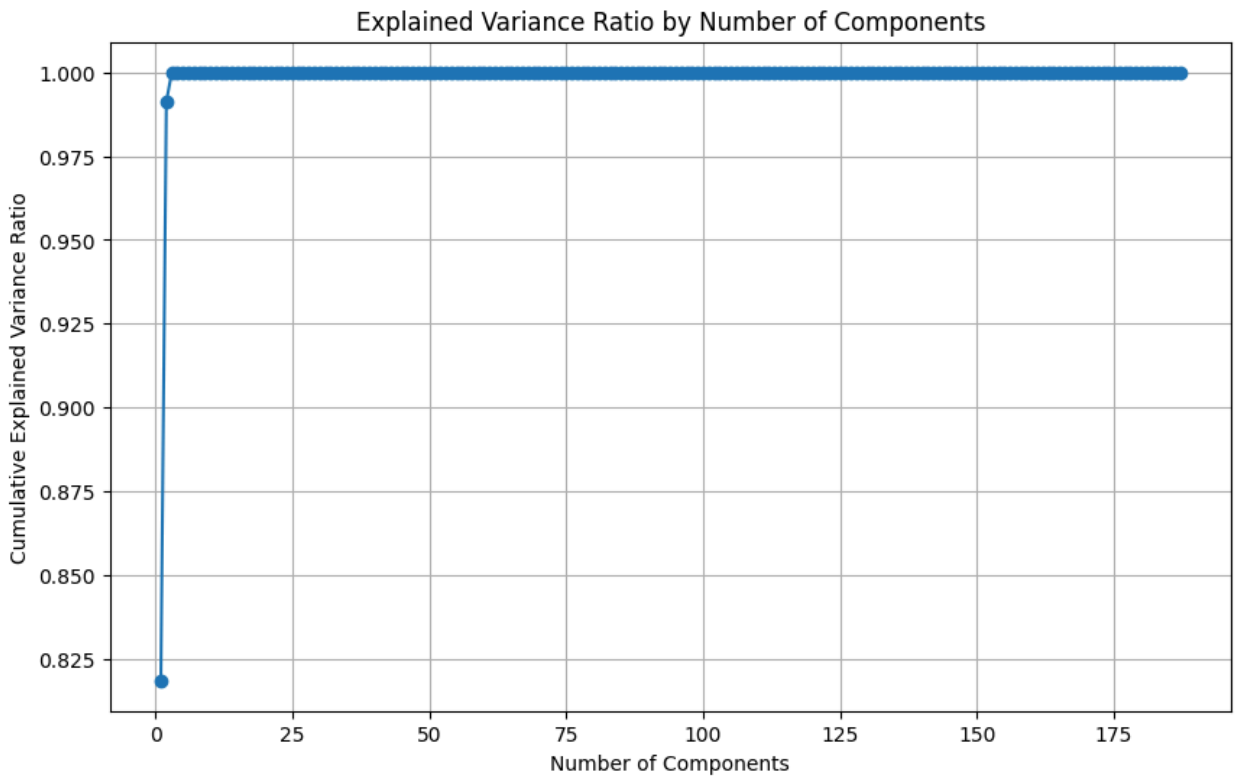


After this, we compute the threshold to f1-score associations and determine the optimal threshold at .84. Similar to above, 84 percent certainty behind the anamoy class for an instance is needed to be categorized as such. After ascribing this to for new predictions, we see the following results as the best performing model:

	precision	recall	f1-score	support
0	0.92	0.95	0.93	18581
1	0.95	0.93	0.94	22581
accuracy			0.94	41162
macro avg	0.94	0.94	0.94	41162
weighted avg	0.94	0.94	0.94	41162

3.3 Experimentation with PCA:

Following the production of these three robust models, we decided to conduct PCA and explore if the dimensionally reduced inputs would render other methods as string performers as well. We pivot to apply PCA onto the features to reduce the dimensions in an effort to potentially improve model performance and computational intensivity. We can project that some models such as k-NN will improve due to sensitivity to large scale spatiality regarding distance between features when dimensions are high as k-NN is entirely based on distance. Logistic regression could improve in scenarios of multicollinearity however these features do not bear strong correlation amongst each other and therefore we do not expect Logistic regression to improve substantially. Random forest performs well with high dimensional data and given the inevitable tradeoff of some patterns/relationships in the data via PCA, we expect Random forest to decline in performance. For the XGBoots and LightGBM boost, we expect these to decline as well as they are based in decision trees and therefore are suitable for high dimensional data with the inevitable loss of some patterns/relationships potentially damaging performance in a similar light to random forest. In addition to these models, we will also incorporate a linear SVC model which, with the previous high dimensional data, suffered from tremendous computation. In order to determine the optimal principal components, we will employ the elbow method to select this value as seen below



From the visual above and the numerical outputs themselves, we determine that 3 Principal Components is suitable as per this method and as a result we initialize, fit, and transform our input features to three PCs. Following this, we then run through each of our models again and also an SVM via a linear SVC. After this modeling, we see none of the models exceed .80 accuracy with the only model seeing improvement being the k-nn by .01 accuracy, with the incorporation of a gridsearch resulting in a k of 13 deemed as optimal and thus employed for this .79 accuracy. The SVM sees an accuracy of .77, similarly poor as per the other models from the PCA input. As a result, we consolidate our focus on our three strong performing models above.

3.3 Proficiency in Using Tools and Languages

Python, pandas, numpy, scikit-learn, xgboost, and lightgbm libraries, served as our primary toolkit for implementing machine learning algorithms. These libraries provided a rich set of tools

for data preprocessing, model training, hyperparameter tuning, and performance evaluation. Leveraging these tools enabled us to efficiently explore various algorithms and methodologies, ensuring a robust analysis. Matplotlib and seaborn also facilitate our means of developing visualizations of said analysis as seen above and below

3.4. Complexity and Accuracy of Models

In evaluating model performance, we meticulously considered the balance between model performance, computational overhead, and complexity. Hyperparameters were carefully tuned to optimize model performance while ensuring these other factors were considered. Models were assessed using multiple performance metrics, including accuracy, precision, recall, and F1 score. This comprehensive evaluation allowed us to gain insights into each model's effectiveness for network intrusion detection, considering both predictive performance and computational efficiency. Overall, our machine learning analysis involved a rigorous and systematic approach, aimed at developing effective classifiers for network intrusion detection. By leveraging a diverse range of algorithms and methodologies, we aimed to identify the most suitable model for deployment, contributing to enhanced network security and threat detection capabilities

4. Results and Final Visualization

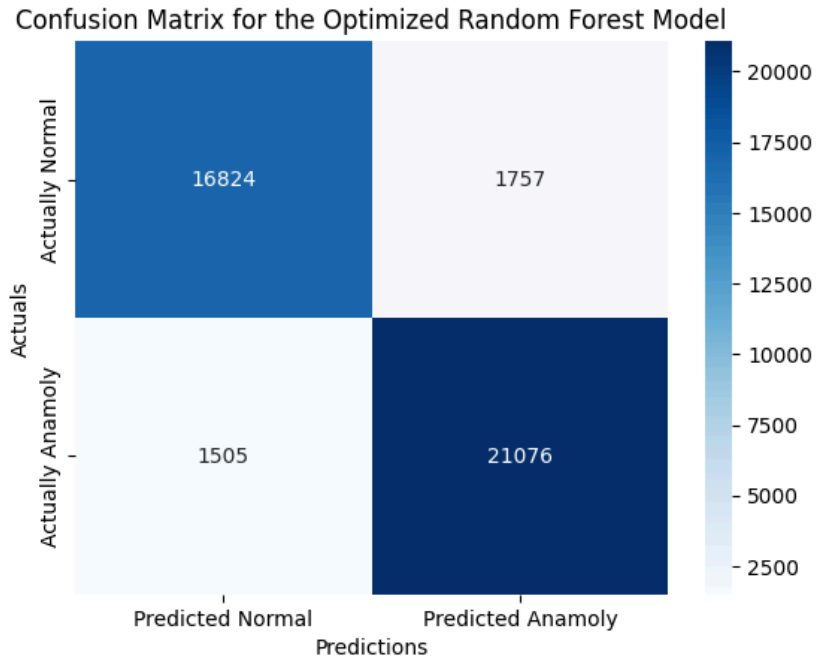
4.1 Results:

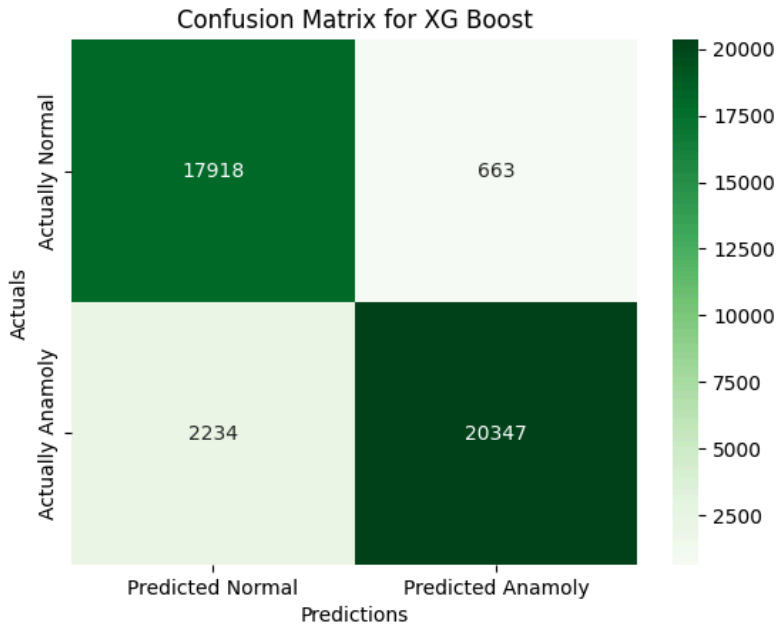
Following the comprehensive modeling above, we have a myriad of models with a variation in performance. Three specific models perform virtually the same as the top performers. We will go through each of these, in order of execution and display 3 numbers for each: Accuracy, F1-Score for normal state, and F1-Score for anomalous state. The Optimized Random Forest model for the

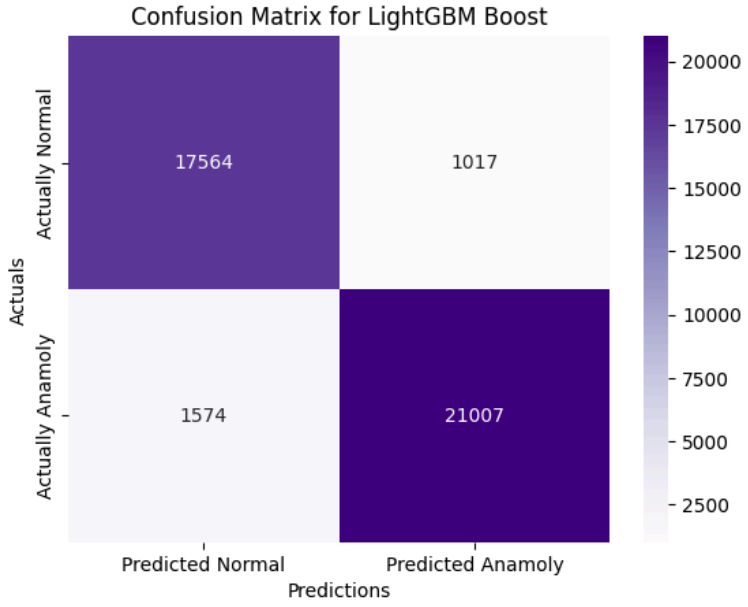
high dimensional data with feature selection: .92, .91, .93. The XG boosted binary classifier: .93, .93, .93. The LightGBM boosted binary classifier: .94, .93, .94. Other models fell below .80 accuracy Overall, the best performing models were those conducted without PCA as described above. The goal of utilizing PCA was to see if we could produce models with close to or better performance compared to those with the high dimensional features. As expected, the models executed with PCA saw faster runtimes however their performance suffered notably. The lack of multicollinearity within the input features likely diminished the positive performance impact which PCA would have on several models. k-NN, highly sensitive to large spatial distance between high dimensional features, actually saw improved performance with the PCA however even this was minimal. Overall, our three most robust models, as noted above, are all based in ensembles of decision trees, a technique which itself is quite good for high dimensional data. As a result, we expected these models to suffer in performance and this nation was corroborated with the results as each of these three saw substantial drops in performance. In sum, we can conclude that PCA in the context of this data and the models we have built mostly sees a negative impact on our models and therefore we pivot back to focus on our robust high dimensional models. That being said, the optimized random forest model incorporates dimensionality reduction in of itself as we removed the features with minimal impact on information gain with the model retaining its high end performance. In the visualizations, we will focus on these three to provide an emphasis on the most useful models in terms of performance; however , feel free to look through the performance of them all in the above classification reports for our main models. Interestingly, while all three models see similar performance, the Optimized Random Forest sees a strong balance in the small error between predicting True anomalies and True normal instances while XGBoost sees noticeably more misclassifications for

predicting True normal instances and less for true anomalies. Lightgbm sees a performance between the two in this regard, with a slight increased magnitude for misclassifications in predicting true normal instances compared to a close but smaller amount of True anomaly misclassifications. for A confusion matrix for each of the select four models will be portrayed below detailing the results just described.

4.2 Final Visualizations:







5. Conclusion:

In sum, we were effectively able to develop three robust models as detailed in the confusion matrices above. We are quite satisfied with these models as their measures in accuracy, recall,

precision, and f1-score are excellent. On top of this, each of the three models is computationally efficient and this is especially of note given the context of this project being anomaly detection. In a hypothetical deployment scenario, these models could be incorporated into real time IDS to support an organization's SIEM. As a result, the high accuracy and computational efficiency would allow for rapid determinations regarding if a network traffic instance is anomalous or not. This could be highly beneficial for such systems and increase their ability to act with speed to such occurrences. As an example, referencing the third reading from class, these three models due to their lightweight computation and robust performance could be viable for real time anomaly detection in similar systems which rely on instantaneous detection and alerts. The LightGBM model performed the best by a small margin and therefore, given their similar computation times, would advise this model for deployment scenarios however such a suggestion would need further analysis of any specific context in which a deployment would occur. For further research, we would consider testing models of this configuration on data of different network traffic features as well as multiclass attack classification given time to further engage in additional intricacies for this dataset and potentially others as well. Overall we enjoyed this project and are fascinated by the potential for the further research and real world deployment scenarios as discussed above.