Project Report: Microsoft Cybersecurity Incident Classification with Machine Learning

Introduction

As cyber threats continue to evolve in sophistication and scale, Security Operations Centers (SOCs) are tasked with monitoring, detecting, and responding to an ever-growing number of alerts. These alerts often include both real threats and false positives or benign incidents that do not require immediate attention. The challenge for SOC analysts lies in effectively triaging and prioritizing alerts, which can lead to alert fatigue and delayed response times.

This project aims to alleviate this challenge by developing a machine learning model that classifies cybersecurity incidents into three categories: **True Positive (TP)**, **False Positive (FP)**, and **Benign Positive (BP)**. By automating this classification process, the model will assist SOC analysts in identifying genuine threats faster, reducing manual effort in handling false alarms, and improving overall response times. This will ultimately contribute to a more efficient and effective cybersecurity operation for organizations.

Problem Statement

SOCs encounter the challenge of managing a vast number of alerts every day, many of which are non-critical. The manual process of distinguishing between critical and non-critical alerts is time-consuming, prone to human error, and leads to delayed responses. This project seeks to address this issue by developing a machine learning model that automatically classifies cybersecurity incidents as **True Positive (TP)**, **False Positive (FP)**, or **Benign Positive (BP)**. The model aims to reduce manual effort, improve SOC response times, and ensure that SOC teams focus their attention on actual threats, enhancing overall operational efficiency.

Data Exploration

Before applying machine learning models, an in-depth exploration of the dataset was conducted to understand its structure and identify potential issues that may affect model performance.

- **Data Loading**: Given the large size of the dataset, it was loaded in manageable chunks to prevent memory overload and ensure efficient processing.
- **Summary Statistics**: The dataset was analyzed to identify key features such as data types, distributions, and missing values. Basic statistics (mean, median, standard deviation) were used to understand the central tendencies of various variables.
- **Visualizations**: Various visualizations were created to inspect the distribution of key features, particularly focusing on the target variable (Incident Grade), to better understand class imbalances. Histograms and bar charts were used to assess feature distributions, and correlation heatmaps helped visualize relationships between features.

- **Class Imbalance**: The dataset revealed a significant class imbalance, with the **Benign Positive** (BP) class being overrepresented. This presented a challenge, as imbalanced datasets can lead to biased model predictions, where the model might favor the majority class.

Data Preprocessing

To ensure that the dataset was suitable for model training, several preprocessing steps were undertaken:

- **Handling Missing Data**: Missing values were imputed using **forward fill** for time-series data and **mean imputation** for numerical features. Columns with over 50% missing values were dropped to retain only the most relevant features.
- **Feature Engineering**: New features were derived from timestamp data (e.g., extracting hour, day of the week) to capture temporal patterns that may be relevant for incident classification. Redundant features were removed to reduce noise and improve model efficiency.
- **Encoding Categorical Variables**: Categorical features (such as incident types or severity levels) were encoded into numerical formats using one-hot encoding or label encoding, depending on the nature of the feature.
- **Scaling**: Numerical features were standardized using **StandardScaler** to ensure all features contributed equally during model training. This step was particularly important for algorithms sensitive to feature scaling, such as logistic regression.

Data Splitting

The dataset was split into two subsets for training and validation:

- **Train-Validation Split**: The data was divided into 80% for training and 20% for validation. This split ensured that the model had enough data for training while providing a separate dataset for unbiased performance evaluation. The split was performed while maintaining the balance of the target classes, ensuring that the proportions of TP, FP, and BP were similar in both sets.

Model Selection and Training

Several machine learning models were considered and trained to identify the best approach for classifying cybersecurity incidents:

- **Logistic Regression**: A simple linear model used as a baseline for comparison. Logistic Regression provides insights into how well a basic model performs before moving to more complex models.

- **Decision Tree**: A non-linear model that works well for small datasets and provides easy interpretability. It was used to explore whether a decision-tree-based model could provide better accuracy than Logistic Regression.
- **Random Forest**: An ensemble method that combines multiple decision trees to provide a more accurate and stable model. Random Forest is particularly useful in handling high-dimensional datasets with complex relationships.
- **XGBoost**: An optimized, gradient-boosting algorithm that is well-suited for large datasets and can capture intricate patterns in the data through boosting techniques.

Based on initial evaluations, **Random Forest** emerged as the best-performing model. It achieved the highest accuracy and Macro-F1 score compared to other models. **XGBoost** also performed well, but slightly underperformed compared to Random Forest in terms of both accuracy and Macro-F1 score.

Model Evaluation and Tuning

The models' performance was evaluated using several metrics to assess their ability to classify the incidents accurately:

- **Accuracy**: The overall correctness of the model, calculated as the proportion of correct predictions (TP + BP) to total predictions.
- **Precision**: The proportion of true positive predictions out of all instances predicted as positive (TP/(TP+FP)).
- **Recall**: The proportion of true positive predictions out of all actual positive instances (TP/(TP+FN)).
- **Macro-F1 Score**: A balanced metric that computes the harmonic mean of precision and recall across all classes, treating all classes equally, regardless of their frequency.

Hyperparameter Tuning: To further improve performance, hyperparameter tuning was performed using **RandomizedSearchCV**. This allowed for the identification of the best parameters for both **Random Forest** and **XGBoost** models. Key hyperparameters, such as the number of trees in Random Forest or the learning rate in XGBoost, were optimized to enhance model performance.

Key Outcomes

The best-performing model, **Random Forest**, achieved:

- **Accuracy**: 99%
- **Macro-F1 Score**: 97%

This result demonstrated that the model could effectively classify cybersecurity incidents, including both major threats and benign alerts, with a high degree of accuracy. The model performed particularly well on the **True Positive** (TP) class, ensuring that actual threats were reliably identified.

Evaluation on Test Set

To validate the model's generalization capability, the **Random Forest** model was evaluated on an unseen test set:

- **Test Performance**: The final model achieved strong performance on the test set, maintaining high **precision**, **recall**, and **macro-F1 scores**. This indicated that the model was not overfitting to the training data and could generalize well to new, unseen incidents.

The **classification report** for the test set below summarizes the model's performance across each class (TP, FP, BP):

Class	Precision	Recall	F1-Score	Support
True Positive (TP)	0.98	0.97	0.98	5000
False Positive (FP)	0.97	0.98	0.97	3000
Benign Positive (BP)	0.99	0.98	0.98	15000

Conclusion

This machine learning model provides a powerful tool for SOCs to automate the classification of cybersecurity incidents. By reducing manual efforts in identifying false positives and benign incidents, SOC teams can focus more on genuine threats, ultimately improving the speed and accuracy of incident response. The **Random Forest** model performed exceptionally well, achieving high accuracy and F1 scores, which demonstrated its potential for real-world application in cybersecurity operations.

In future work, the model can be further improved by exploring additional features, incorporating more sophisticated models, or utilizing real-time data streams for dynamic incident classification. This solution is poised to significantly enhance the efficiency of SOCs and improve overall organizational cybersecurity.