**Project Report**

**Microsoft Cybersecurity Incident Classification with Machine Learning**

**1. Introduction:**

The rise in cyber threats has significantly increased the workload for Security Operations Centers (SOCs), responsible for monitoring, detecting, and responding to cybersecurity incidents. Many alerts are false positives or benign incidents that do not require immediate attention, which causes inefficiency in triaging true threats. This project aims to address these challenges by developing a machine learning model that classifies cybersecurity incidents into three categories: **True Positive (TP)**, **False Positive (FP)**, and **Benign Positive (BP)**. Automating this classification process will allow SOC analysts to identify real threats faster, thereby improving the security posture and reducing the workload caused by false alarms.

**2. Problem Statement:**

SOCs are overwhelmed with a large number of alerts daily, many of which are non-critical. Manually identifying the alerts that require attention is time-consuming and error-prone. This project seeks to alleviate that issue by automating the classification process, making it more efficient and accurate. The goal is to develop a machine learning model that can classify incidents as True Positive (TP), False Positive (FP), or Benign Positive (BP).

**3. Data Exploration:**

* **Data Loading**: The dataset was loaded in chunks to efficiently handle its large size.
* **Summary Statistics**: The data was analyzed to check for data types, structure, and missing values.
* **Visualizations**: Distribution of key features, including the target variable (Incident Grade), was visualized to identify class imbalances.
* **Class Imbalance**: The **Benign Positive (BP)** class was overrepresented in comparison to the **True Positive (TP)** and **False Positive (FP)** classes, presenting a challenge in model training.

**4. Data Preprocessing:**

* **Handling Missing Data**: Missing values were addressed using forward-fill and mean imputation strategies. Columns with more than 50% missing values were dropped.
* **Feature Engineering**: Timestamp-based features were derived, and redundant columns were removed.
* **Encoding Categorical Variables**: Categorical features were converted into numerical formats through encoding techniques.
* **Scaling**: Numerical features were standardized to ensure they contributed equally during model training.

**5. Data Splitting:**

The data was split into training (80%) and validation (20%) sets while maintaining the balance of classes to ensure the model does not learn biased patterns.

**6. Model Selection and Training:**

Several models were trained and evaluated, including:

* **Logistic Regression**: A simple baseline model for comparison.
* **Decision Tree**: A non-linear model known for its interpretability.
* **Random Forest**: An ensemble model of decision trees offering more stability and accuracy.
* **XGBoost**: A powerful, gradient boosting algorithm known for handling large datasets effectively.

Among these models, **Random Forest** performed the best, achieving high accuracy and macro-F1 scores.

**7. Model Evaluation and Tuning:**

* **Cross-validation**: The model’s performance was evaluated using cross-validation, ensuring it generalizes well across different subsets of data.
* **Hyperparameter Tuning**: **RandomizedSearchCV** was used to find the best parameters for both Random Forest and XGBoost.

**8. Metrics Used:**

* **Accuracy**: Measures the overall correctness of the model.
* **Precision**: Indicates how many positive predictions were correctly identified.
* **Recall**: Measures the model's ability to correctly identify all actual positive instances.
* **Macro-F1 Score**: A balanced metric that treats all classes equally, even in the case of imbalanced classes.

**9. Key Outcomes:**

* **Random Forest** emerged as the best-performing model, achieving an **accuracy of 91%** and a **macro-F1 score of 0.62** on the test dataset.
* **XGBoost** also performed well but slightly underperformed compared to Random Forest.

**10. Evaluation on Test Set:**

* The model was evaluated on a test set that had not been seen during training. This ensures that the model generalizes well to unseen data.
* The final evaluation on the test set showed strong performance, despite some challenges due to class imbalance.

**Classification Report on Test Data:**

| **Class** | **Precision** | **Recall** | **F1-Score** | **Support** |
| --- | --- | --- | --- | --- |
| 0 | 0.59 | 0.80 | 0.68 | 24,124 |
| 1 | 0.86 | 0.12 | 0.22 | 21,252 |
| 2 | 0.94 | 0.97 | 0.96 | 303,765 |
| **Accuracy** |  |  | **91%** | 349,141 |
| **Macro avg** | 0.80 | 0.63 | 0.62 | 349,141 |
| **Weighted avg** | 0.91 | 0.91 | 0.89 | 349,141 |

**Macro-F1 Score: 0.62**

**Macro Precision: 0.80**

**Macro Recall: 0.63**

**11. Confusion Matrix on Test Data:**

[[ 19200 345 4579]

[ 4693 2655 13904]

[ 8802 71 294892]]

* The **True Positives (TP)** are predominantly identified under class 2 (the largest class), with good recall.
* **False Positives (FP)** are more significant in class 1, showing that the model is incorrectly identifying many instances of class 1 as class 2.
* The **Benign Positives (BP)** are mostly predicted correctly in class 0, but still show some misclassifications in classes 1 and 2.

**12. Conclusion:**

The **Random Forest** model performed well for incident classification in the cybersecurity context. However, further improvements can be made by addressing the class imbalance more effectively through techniques like **SMOTE** or **class weighting**. Model performance can also be optimized by fine-tuning hyperparameters further and testing other ensemble methods or advanced techniques like **XGBoost** or **LightGBM**.

**13. Future Work:**

* Apply advanced techniques like **SMOTE** to balance the classes and improve recall for False Positive (FP) and Benign Positive (BP) classes.
* Experiment with other models such as **Neural Networks** or **Deep Learning** approaches for better performance.
* Incorporate real-time alert data to continuously train and update the model to handle evolving threats.