Microsoft: Classifying Cybersecurity Incidents with Machine Learning

1. Introduction

Cybersecurity is an ever-growing concern for organizations worldwide. The rapid increase in cyber threats requires companies to monitor and classify incidents effectively to prioritize responses and improve security measures. This project focuses on classifying cybersecurity incidents based on a given dataset from Microsoft. The dataset includes detailed incident data, such as alert types, timestamps, detector IDs, and other key features to help classify incidents by their severity grade (target variable: `IncidentGrade`). Classifying incidents helps organizations prioritize response efforts and allocate resources efficiently.

2. Data Overview

2.1 Dataset Description

The dataset comprises 10,000 rows, each representing a unique cybersecurity incident. The columns represent various features, including:

- **OrgId**: Organization Identifier.
- **IncidentId**: Unique identifier for the incident.
- **AlertId**: Unique identifier for the alert associated with the incident.
- **Timestamp**: The timestamp when the incident was detected.
- **DetectorId**: Identifier for the source of detection.
- **AlertTitle**: Title describing the alert (e.g., InitialAccess, Exfiltration, etc.).
- **Category**: Type of attack category (e.g., Initial Access, Execution, Exfiltration, Command and Control).
- **MitreTechniques**: MITRE ATT&CK technique IDs associated with the incident (e.g., T1189, T1566, etc.).

- **IncidentGrade**: The target variable indicating the severity of the incident, categorized as `TruePositive`, `FalsePositive`, `BenignPositive`, etc.
- **SuspicionLevel**: Whether the incident was classified as suspicious or malicious.
- **CountryCode, State, City**: Geographical location information of the incident.

The goal of this project is to classify `IncidentGrade` using machine learning models based on various incident characteristics.

3. Data Preprocessing

The data preprocessing steps were applied separately to both the **training** and **test** datasets to ensure consistency in model evaluation.

3.1 Handling Missing Values

- **Missing Values**: Missing data was present in features such as `MitreTechniques`, `SuspicionLevel`, and `CountryCode`. For features with a high percentage of missing values, such as `ResourceType` and `Roles`, we dropped those columns. Other features were imputed using common strategies (e.g., filling missing geographical information with a default value).

3.2 Encoding Categorical Variables

- **Label Encoding**: Categorical variables, such as `AlertTitle`, `Category`, `SuspicionLevel`, and `CountryCode`, were encoded using label encoding. This allowed the machine learning models to interpret categorical data as numerical values.

3.3 Feature Engineering

- **Timestamp Feature**: The `Timestamp` column was processed to extract time-based features like `Hour`, `Day of the Week`, and `Month` to capture potential time patterns in incident occurrences.

3.4 Handling Class Imbalance

- **Class Imbalance**: The target variable, `IncidentGrade`, was imbalanced, with a higher number of `FalsePositive` labels compared to other grades. This imbalance was handled using the **Synthetic Minority Over-sampling Technique (SMOTE)** to oversample the minority classes in the training dataset. SMOTE creates synthetic samples for the minority classes, ensuring the model learns equally from all classes and improving the model's ability to generalize well across various incident severities.

4. Exploratory Data Analysis (EDA)

4.1 Target Variable Distribution

- The distribution of `IncidentGrade` showed that most of the data consisted of `FalsePositive` incidents, followed by `TruePositive` and `BenignPositive`. This necessitated the use of oversampling to address the class imbalance during model training.

4.2 Correlation Analysis

- An analysis of feature correlations using a heatmap revealed that features like `MitreTechniques`, `AlertTitle`, and `Category` had a clear relationship with the target variable. Geographical features, such as `CountryCode`, had limited correlation with the incident severity grade.

4.3 Temporal Analysis

- A pattern was observed in the occurrence of incidents based on time. Most incidents happened during working hours (9 AM - 6 PM), peaking in the middle of the week. This temporal feature helped enhance model performance when included in the feature set.

5. Model Building

Several machine learning models were built and trained on the **training dataset** and later evaluated on the **test dataset**. The following models were used:

- **Logistic Regression**
- **Decision Tree Classifier**
- **Random Forest Classifier**
- **Gradient Boosting Classifier**
- **Support Vector Machine**

- **K-Nearest Neighbour**
- **Naive Bayes**

5.1 Model Training and Evaluation

The training and test datasets were split using an 80/20 ratio to ensure that the model's performance could be evaluated on unseen data. The following evaluation metrics were used to assess the model performance:

- **Accuracy**: The overall percentage of correctly classified incidents.
- **Precision**: The proportion of true positive predictions to the total predicted positives.
- **Recall**: The proportion of true positive predictions to the total actual positives.
- **F1-Score**: The harmonic mean of precision and recall.
- **Confusion Matrix**: A detailed breakdown of classification performance for each class.

6. Model Results

The models were trained using the preprocessed **training dataset** (with class balancing using SMOTE) and evaluated on the **test dataset** to measure their generalization capability.

6.1 Logistic Regression

- **Train Accuracy**: 0.66
- **Test Accuracy**: 0.60
- **Train F1-Score**: 0.55
- **Test F1-Score**: 0.57

6.2 Decision Tree Classifier

- **Train Accuracy**: 0.86
- **Test Accuracy**: 0.82
- **Train F1-Score**: 0.80

- **Test F1-Score**: 0.80

6.3 Random Forest Classifier

- **Train Accuracy**: 0.89
- **Test Accuracy**: 0.86
- **Train F1-Score**: 0.83
- **Test F1-Score**: 0.84

6.4 Support Vector Machine

- **Train Accuracy:** 0.69
- **Test Accuracy:** 0.63
- **Train F1-Score:** 0.57
- **Test F1-Score:** 0.59

6.5 K-Nearest Neighbour

- **Train Accuracy:** 0.78
- **Test Accuracy:** 0.72
- **Train F1-Score:** 0.64
- **Test F1-Score:** 0.67

6.6 Gradient Boosting Classifier

- **Train Accuracy**: 0.89
- **Test Accuracy**: 0.88
- **Train F1-Score**: 0.85
- **Test F1-Score**: 0.87

6.7 Naive Bayes

- **Train Accuracy**: 0.45
- **Test Accuracy**: 0.46

- **Train F1-Score**: 0.25
- **Test F1-Score**: 0.35

7. Model Comparison and Conclusion

After training and evaluating the models on both the **train** and **test** datasets (with class balancing using SMOTE), the **Gradient Boosting Classifier** was determined to be the best-performing model. The Gradient Boosting model achieved the highest accuracy (88%) and F1-score (87%) on the test dataset. This model outperformed others due to its ability to handle categorical data efficiently and capture complex patterns in the data.

8. Conclusion

In this project, various machine learning models were applied to classify cybersecurity incidents based on incident characteristics. After preprocessing (including handling missing values, encoding categorical data, and addressing class imbalance with SMOTE) and analyzing both the training and test datasets, the Gradient Boosting Classifier was found to deliver the best performance. The model can be further refined through hyperparameter tuning and real-world data integration to enhance its predictive capability.