Project Documentation: Microsoft Cybersecurity Incident Classification

By : Sripathi V R

1. Project Overview

The primary objective of this project is to develop a machine learning model that enhances the efficiency of Security Operation Centers (SOCs) by accurately predicting the triage grade of cybersecurity incidents. The model classifies incidents as True Positive (TP), Benign Positive (BP), or False Positive (FP) based on historical data and customer feedback. This classification helps SOC analysts prioritize incidents and respond to threats more efficiently.

2. Data Exploration and Understanding

Purpose:

To gain an initial understanding of the dataset, including its structure, feature types, distribution of the target variable, and any inherent patterns or anomalies.

Key Steps:

Loading the Dataset: The dataset train.csv was loaded into the notebook to examine its features and target variables.

Initial Inspection: Analyzed the dataset's shape and the data types of each feature to understand the structure and composition.

Exploratory Data Analysis (EDA): Conducted EDA using statistical summaries and visualizations to identify patterns, correlations, and potential anomalies, focusing on understanding the data distribution, especially the target variable's imbalance.

3. Data Preprocessing and Feature Engineering

Purpose:

Prepare the dataset for modeling by addressing missing values, encoding categorical variables, normalizing numerical features, and engineering new features to enhance model performance.

Key Steps:

Handling Missing Data: Identified missing values and handled them appropriately, using imputation methods or removing affected rows to maintain data integrity.
Feature Engineering: Created new features or modified existing ones, such as deriving new time-based features or combining related features to capture more information.
Encoding Categorical Variables: Converted categorical variables into numerical formats using techniques like one-hot encoding and label encoding to make them suitable for machine learning algorithms.

4. Data Splitting

Purpose:
Split the dataset into training and validation sets to evaluate model performance before testing on unseen data.

Key Steps:
Train-Validation Split: The dataset was split into training (80%) and validation (20%) sets to ensure that model evaluation is robust and reliable.
Stratification: Applied stratified sampling to maintain the class distribution in both the training and validation sets, which is crucial due to the imbalance in the target variable.

5. Model Selection and Training

Purpose:
Identify the most suitable machine learning models for the classification task and train them on the preprocessed dataset.

Models Used:
Baseline Model: Started with simple models like logistic regression and decision trees to establish a baseline performance. This helps in determining the complexity needed for more advanced models.

Advanced Models:

Random Forests: An ensemble model that builds multiple decision trees and merges them to get a more accurate and stable prediction.
Gradient Boosting Machines (GBM): Models like XGBoost and LightGBM, which build trees sequentially, each trying to correct the errors of the previous ones. LightGBM, in particular, was used due to its efficiency with large datasets.
Neural Networks: Tested for their ability to capture complex patterns in data, especially if non-linear relationships are present.

Key Steps:
Hyperparameter Tuning: Used RandomizedSearchCV for hyperparameter optimization. Randomized search is computationally more efficient than grid search and helps in finding the optimal parameters faster.
Cross-Validation: Implemented k-fold cross-validation to assess model performance consistently across different subsets of the data, reducing overfitting risk.

6. Model Evaluation and Tuning

Purpose:
Evaluate the models using appropriate metrics and tune them to enhance performance.

Key Metrics Used:
Macro-F1 Score: Provides a balanced evaluation metric by considering both precision and recall across all classes.
Precision and Recall: Specifically focused on minimizing false positives and maximizing true positives, which are critical in cybersecurity contexts.

Key Steps:
Handling Class Imbalance: Applied SMOTE (Synthetic Minority Over-sampling Technique) to balance the dataset, as there were significant disparities in class frequencies. This helps in improving the model's ability to correctly predict the minority class.
Performance Evaluation: Evaluated the models based on the validation set performance using the metrics mentioned above.

7. Model Interpretation and Feature Importance

Purpose:

Understand which features have the most impact on model predictions to provide transparency and insight into the model's decision-making process.

Method Used:

LightGBM Built-in Feature Importance: Used LightGBM's built-in feature importance method to rank features based on their contribution to the model's predictions. This method is faster and more suitable for large datasets.

Key Steps:

Calculating Feature Importance: After training the model, extracted the feature importance scores provided by LightGBM.

Analysis of Feature Importance: Analyzed the top features to understand their impact on model performance and ensure that the most informative features are used in decision-making.

8. Error Analysis

Purpose:

Identify and analyze misclassified cases to understand the model's weaknesses and areas for potential improvement.

Key Steps:

Misclassification Identification: Examined the validation set to identify cases where the model predictions did not match the actual labels.

Analysis of Common Misclassifications: Studied these cases to detect patterns or commonalities that could indicate areas for feature improvement or model refinement.

9. Final Evaluation on Test Set

Purpose:

Evaluate the finalized model on a separate test set to assess its generalizability and robustness to new data.

Key Steps:

Testing: Evaluated the model on the test set and reported final performance metrics to understand how well the model generalizes to unseen data.

Comparison to Baseline: Compared the final test results to the baseline model to ensure consistent performance improvement.

10. Reporting

Purpose:

For Provided a comprehensive report documenting the entire process, including methodologies, decisions, challenges, and outcomes.

Sections:

Model Documentation: Documented each step, including the rationale behind choosing specific models, handling challenges like data imbalance, and optimizing model performance.

Recommendations:

Integration into SOC Workflows: Suggested ways to integrate the model into SOC workflows for automated triage and incident prioritization.

Future Improvements: Highlighted areas for future work, such as continuous model retraining, further feature engineering, and exploring additional data sources.

Recommendations

Integration into SOC Workflows: The model can be integrated into existing SOC workflows to automatically triage incidents, allowing analysts to focus on critical threats.

Continuous Model Improvement: Regular updates with new data will ensure that the model adapts to evolving threat landscapes and maintains its accuracy.

Feature Engineering and Model Refinement: Additional features, particularly those capturing temporal patterns or correlations between features, could further enhance model performance.

Handling Class Imbalance with Advanced Techniques: Explore other methods beyond SMOTE, such as cost-sensitive learning or ensemble methods, to further improve model performance on minority classes.

Real-World Deployment Considerations: When deploying the model, consider its computational requirements and ensure it is capable of real-time data processing to provide timely insights for incident response.