National College of Ireland

Configuration Manual

MINIMIZING FALSE POSITIVE IN SOC USING AI/ML

MSc in Cybersecurity

Kiran Sreekumar
Student ID: x23279851

School of Computing
National College of Ireland

Supervisor: Michael Prior

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Kiran Sreekumar		
Student ID:	x23279851		
Programme:	MSc in Cybersecurity	**Year:**	September 2024
Module:	MSc (Research) Practicum		
Lecturer:	Michael Prior		
Submission Due Date:	11th August 2025		
Project Title:	MINIMIZING FALSE POSITIVES IN SOC USING AI/ML		
Word Count:	341	**Page Count:**	3

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Kiran Sreekumar
Date:	11th August 2025

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Kiran Sreekumar
Student ID: x23279851

1 System and Hardware Configuration

This section details the hardware and software configuration used for the development and implementation of the machine learning models for minimizing false positives in Security Operations Centres (SOCs).

1.1 Laptop Specifications

- Model: HP Pavilion x360 Convertible 14-dv0053TU
- Processor: Intel Core i5, 11th Gen
- RAM: 8 GB DDR4
- Storage: 512 GB SSD
- Operating System: Windows 11 Home 64-bit

1.2 Software Environment

- Programming Language: Python 3.10+
- Development Environment: Jupyter Notebook (Anaconda distribution)
- IDE Used: Jupyter Notebook via Anaconda Navigator

1.3 Python Libraries

The following Python libraries were installed and imported to facilitate the model implementation:

- pandas – data manipulation
- numpy – numerical computations
- matplotlib, seaborn – data visualization
- scikit-learn – model building and evaluation (classification, preprocessing, metrics)
- LabelEncoder, RandomizedSearchCV – for preprocessing and hyperparameter tuning

2 Dataset Preparation and Preprocessing

This section outlines how datasets were handled before modeling.

2.1 Datasets Used

Two datasets were used in the project:

- GUIDE_Test.csv – Microsoft Security Incident Prediction Dataset (Test)
- cybersecurity_attacks.csv – Simulated real-world SOC alert data

2.2 Preprocessing Steps

- Missing Values: Replaced using mode imputation (fillna(mode))
- High Cardinality Columns: Dropped irrelevant or sparse columns such as "Payload Data", "User Information", etc.
- Encoding:
  1. Label Encoding for categorical features and target variable (LabelEncoder)
  2. One-Hot Encoding for test dataset (pd.get_dummies)
- Feature Engineering: Numerical features were separated, and correlations were visualized.
- Data Splitting: 80/20 train-test split for model evaluation

3 Model Development and Evaluation

This section explains the machine learning models, training strategies, and evaluation methods.

3.1 Models Implemented

- **Random Forest Classifier**
- **Gradient Boosting Classifier**

3.2 Training and Tuning

- **Train-Test Split**: 80% training, 20% testing
- **Hyperparameter Tuning**:
  - Utilized RandomizedSearchCV for both models
  - Parameters tuned include n_estimators, max_depth, min_samples_split, and learning rate (for Gradient Boosting)

References

Saabith, S., Vinothraj, T., & Fareez, M. (2021). A review on Python libraries and Ides for Data Science. *Int. J. Res. Eng. Sci*, *9*(11), 36-53. https://www.researchgate.net/profile/Vinothraj-Thangarajah/publication/357898994_A_Review_on_Python_Libraries_and_IDEs_for_Data_Science/links/620249344d89183b338b49c2/A-Review-on-Python-Libraries-and-IDEs-for-Data-Science.pdf

Durvasulu, M. B. T. (2021). Building efficient storage architectures with Python. *Int J Adv Res Educ Technol*, *8*(03). https://ijarety.in/admin/img/18_Building.pdf

Larralde, M., & Zeller, G. (2023). PyHMMER: a Python library binding to HMMER for efficient sequence analysis. *Bioinformatics*, *39*(5), btad214. https://academic.oup.com/bioinformatics/article-abstract/39/5/btad214/7131068

https://youtu.be/HLD-Ll_-IT4?si=JhKy1_p9Eyd7rEWe