# Detecting Recon Cyberattacks with Machine Learning

The objective of this project is to develop a model that can reliably predict the occurrence of Recon Cyberattacks in the context of the CICIOT 2023 dataset.

The idea is to specialize models to detect one kind of cyberattack, thus augmenting their performance, as the researchers also pledged:

"*Finally, although we are focusing on 33 different attacks, future directions could also be tailored to address issues related to individual attacks or categories.*"

Neto, E.C.P.; Dadkhah, S.; Ferreira, R.; Zohourian, A.; Lu, R.; Ghorbani, A.A. CICIoT2023: A Real-Time Dataset and Benchmark for Large-Scale Attacks in IoT Environment. Sensors 2023, 23, 5941. https://doi.org/10.3390/s23135941

## The CICIOT Dataset

This dataset is the result of a research from the University of New Brunswick Centre for Cybersecurity.

It has extracted CSV features on network traffic across 105 Internet of Things (IoT) devices with 33 cyberattacks run on them. 7 types of attacks were run: distributed denial of service (DDoS), denial of service (DoS), reconnaissance, web-based, brute-force, spoofing, and the Mirai botnet.

Link to Kaggle https://www.kaggle.com/datasets/madhavmalhotra/unb-cic-iot-dataset.

## Data Dictionary

| Feature | Data Type | Description |
|---|---|---|
| **flow_duration** | float64 | Total duration of the network flow in seconds. |
| **Header_Length** | float64 | The length of the packet header in bytes. |
| **Protocol Type** | float64 | Numerical representation of the network protocol used. |
| **Duration** | float64 | Time to live (TTL) in seconds |
| **Rate** | float64 | The rate of packet transmission over the network in packets per second. |
| **Srate** | float64 | The rate of outbound packets in the flow, indicating data sent from the source. |
| **Drate** | float64 | The rate of inbound packets in the flow, indicating data received by the destination. |
| **fin_flag_number** | float64 | Number of packets with the FIN flag set, indicating the end of data communication. |
| **syn_flag_number** | float64 | Number of packets with the SYN flag set, used to initiate a TCP connection. |
| **rst_flag_number** | float64 | Number of packets with the RST flag set, used to reset the connection. |
| **psh_flag_number** | float64 | Number of packets with the PSH flag set, indicating the push function. |
| **ack_flag_number** | float64 | Number of packets with the ACK flag set, used to acknowledge the receipt of packets. |

| | | |
|---|---|---|
| **ece_flag_number** | float64 | Number of packets with the ECE flag set, indicating Explicit Congestion Notification Echo. |
| **cwr_flag_number** | float64 | Number of packets with the CWR flag set, used to signal congestion window reduced. |
| **ack_count** | float64 | The total number of acknowledgment packets within the flow. |
| **syn_count** | float64 | The total number of synchronization packets within the flow. |
| **fin_count** | float64 | The total number of finish packets within the flow. |
| **urg_count** | float64 | The total number of urgent packets within the flow. |
| **rst_count** | float64 | The total number of reset packets within the flow. |
| **HTTP** | float64 | Indicator of HTTP traffic (1 for HTTP traffic, 0 otherwise). |
| **HTTPS** | float64 | Indicator of HTTPS traffic (1 for HTTPS traffic, 0 otherwise). |
| **DNS** | float64 | Indicator of DNS traffic (1 for DNS traffic, 0 otherwise). |
| **Telnet** | float64 | Indicator of Telnet traffic (1 for Telnet traffic, 0 otherwise). |
| **SMTP** | float64 | Indicator of SMTP traffic (1 for SMTP traffic, 0 otherwise). |
| **SSH** | float64 | Indicator of SSH traffic (1 for SSH traffic, 0 otherwise). |
| **IRC** | float64 | Indicator of IRC traffic (1 for IRC traffic, 0 otherwise). |
| **TCP** | float64 | Indicator of TCP protocol usage (1 for TCP, 0 otherwise). |
| **UDP** | float64 | Indicator of UDP protocol usage (1 for UDP, 0 otherwise). |
| **DHCP** | float64 | Indicator of DHCP traffic (1 for DHCP traffic, 0 otherwise). |
| **ARP** | float64 | Indicator of ARP traffic (1 for ARP traffic, 0 otherwise). |
| **ICMP** | float64 | Indicator of ICMP traffic (1 for ICMP traffic, 0 otherwise). |
| **IPv** | float64 | Indicator of IPv4 or IPv6 traffic (1 for IP traffic, 0 otherwise). |
| **LLC** | float64 | Indicator of LLC traffic (1 for LLC traffic, 0 otherwise). |
| **Tot sum** | float64 | The total size of the packets transferred in the flow. |
| **Min** | float64 | The minimum size of packets in the flow. |
| **Max** | float64 | The maximum size of packets in the flow. |
| **AVG** | float64 | The average size of packets in the flow. |
| **Std** | float64 | The standard deviation of packet sizes in the flow. |
| **Tot size** | float64 | Total size of the flow in bytes. |
| **IAT** | float64 | Inter-Arrival Time of the packets in the flow. |
| **Number** | float64 | Total number of packets in the flow. |
| **Magnitue** | float64 | A derived metric indicating the magnitude of the flow. |
| **Radius** | float64 | A derived metric indicating the radius of the flow. |

| | | |
|---|---|---|
| **Covariance** | float64 | The covariance of packet sizes in the flow. |
| **Variance** | float64 | The variance of packet sizes in the flow. |
| **Weight** | float64 | A weight metric related to the flow. |
| **label** | object | Categorical label of the traffic type. |

## Simplified Data Features Overview

***Communication Patterns***

**- Duration Measures:** These features tell us how long a communication takes place. It's like timing a phone call to see if it's a quick hello or a long conversation.

- `flow_duration`, `Duration`

**- Transmission Rates:** These are like the speed of conversation, telling us how fast data is being sent and received.

- `Rate`, `Srate`, `Drate`

***Traffic Signs***

**- Flag Counts:** Just like flags used in sports to indicate different events, these counts tell us how many times certain types of network "signals" occur.

- `fin_flag_number`, `syn_flag_number`, `rst_flag_number`, etc.

**- Packet Counts:** These give us the total number of "letters" sent and received during the communication.

- `ack_count`, `syn_count`, `fin_count`, etc.

***Communication Types***

**- Protocol Indicators:** These indicate the method of communication being used, similar to different social media platforms like email or instant messaging.

- `HTTP`, `HTTPS`, `DNS`, `Telnet`, etc.

***Conversation Statistics***

**- Size Measures:** These features measure the "volume" of the conversation, or how much information is being exchanged.

- `Tot sum`, `Min`, `Max`, `AVG`, `Std`, `Tot size`

**- Timing Measures:** These tell us about the timing between exchanges, like the pauses between sentences in a conversation.

- `IAT` (Inter-Arrival Time)

***Interaction Complexity***

**- Complexity Metrics:** These are advanced measures that summarize the overall complexity and pattern of the communication.

- `Magnitue`, `Radius`, `Covariance`, `Variance`, `Weight`

***Nature of Traffic***

**- Traffic Category:** This is the label that tells us whether the communication is regular and expected (`BenignTraffic`) or potentially suspicious (`ReconAttack`).

- `label`

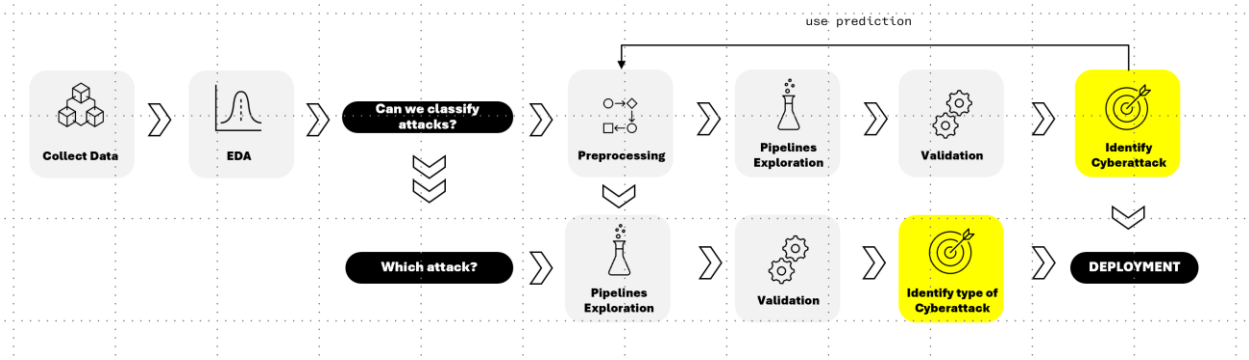
## Directory Structure

The current directory contains the following items:

- The notebooks representing three different stages of the project
    - *Cybersec_eda.ipynb* contains initial data exploration and EDA
    - *Cybersec_binary_modelling.ipynb* contains EDA refinement and binary classification modelling
    - *Cybersec_multiclass_modelling.ipynb* contains multiclass classification modelling, inference details and final considerationsbinary
- *Pipeline_adaboost.zip* contains the binary classification (first-stage) model. It is saved as a zip file to avoid github's limitation of size.
- *Gridsearch_xgboost.pkl* contains the multiclassification (second-stage) model.
- *Deployment.py* contains the final inference file where the two stage model actually makes predictions on sample of the test data. Conveniently it loads adaboost model directly from the zip folder.

## Process description

The main steps of the project are described in the diagram below:




- First I collected the data from CCIOT 2023, filter it for Recon Attacks only and preprocess it for interpretability and efficiency.
- Secondly I performed EDA.
- Then I experimented different pipelines to understand the effectiveness of different types of preprocessing steps and models for the binary classification (identifying cyberattacks from benign traffic).
- Subsequently I performed cross validation and hyperparameter tuning. The output model ('pipeline_adaboost') is saved in the directory. However, Random Forest as defined in the notebook could also be employed. I encourage further exploration in the optimization of these two models.
- After that I turned to multiclass problem and developed a two staged-model with adaboost being the binary classifier and XG Boost the multiclass one.
- The output was saved in the directory ('gridsearch_xgboost').
- Finally, a guideline for inference of the two staged model is provided in *deployment.py*.

## Notes

- In Sprint 2 notebook there are cells encapsulated with triple quotations. This is because the time of execution of the cell is very long.
- I deliberately didn't apply feature selection steps that could make inference process less intuitive. The plan is in fact to apply another modelling layer to better predict device-based attacks.