**Network Traffic Anomaly Detection Case Study**

**Introduction:**

Network security is a critical concern in today's time, with organizations facing increasing threats from cyber-attacks and malicious activities. One of the key challenges in maintaining network security is the detection of anomalous network traffic, which may indicate potential security breaches or unauthorized access.

**Aim:**

The aim of this case study is to detect network traffic anomalies using machine learning.

**Dataset Information:**

The dataset used in this study contains information about network traffic, including various features and a target variable indicating whether the network traffic is anomalous or not. Here's a brief overview of the dataset:

* **SourceIP**: IP address of the source device.
* **DestinationIP**: IP address of the destination device.
* **SourcePort**: Port number used by the source device.
* **DestinationPort**: Port number used by the destination device.
* **Protocol**: Network protocol used (e.g., TCP, UDP).
* **BytesSent**: Number of bytes sent from the source device.
* **BytesReceived**: Number of bytes received by the destination device.
* **PacketsSent**: Number of packets sent from the source device.
* **PacketsReceived**: Number of packets received by the destination device.
* **Duration**: Duration of the network communication.
* **IsAnomaly**: Target variable indicating whether the network traffic is anomalous (1) or normal (0).

**Dataset Link:** [**https://www.kaggle.com/code/vidhikishorwaghela/network-traffic-anomaly-detection/input**](https://www.kaggle.com/code/vidhikishorwaghela/network-traffic-anomaly-detection/input)

**Code:**

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy\_score, classification\_report, confusion\_matrix

from sklearn.impute import SimpleImputer

data = pd.read\_csv("synthetic\_network\_traffic.csv")

data.dropna(inplace=True)

# Separate features and target variable

X = data.drop(columns=['IsAnomaly'])

y = data['IsAnomaly']

# Split the data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Impute missing values with the mean

imputer = SimpleImputer(strategy='mean')

X\_train\_imputed = imputer.fit\_transform(X\_train)

X\_test\_imputed = imputer.transform(X\_test)

# Initialize the Random Forest classifier

clf = RandomForestClassifier(n\_estimators=100, random\_state=42)

clf.fit(X\_train\_imputed, y\_train)

y\_pred = clf.predict(X\_test\_imputed)

accuracy = accuracy\_score(y\_test, y\_pred)

print("\nAccuracy:", accuracy)

# Print classification report

print("\nClassification Report:")

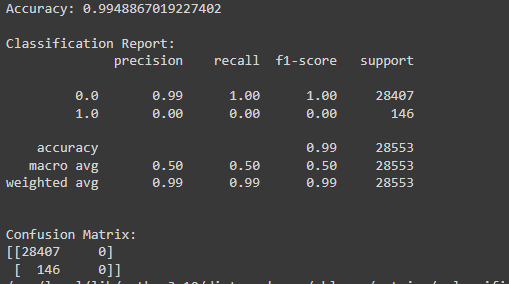
print(classification\_report(y\_test, y\_pred))

# Print confusion matrix

print("\nConfusion Matrix:")

print(confusion\_matrix(y\_test, y\_pred))

**Output:**

****

**Result Interpretation**

The model was able to achieve an impressive accuracy of around 99.49%, correctly classifying the majority of instances in the dataset. This suggests that it is highly effective at identifying normal network traffic patterns. With its high accuracy, the model demonstrates its potential to contribute to network security by efficiently recognizing and managing regular network activities. So it is able to identify which patterns in the network traffic are normal which is a crucial task to keep our etwok safe from anomalies.

**F1-score:** The F1-score, which represents the harmonic mean of precision and recall, is reported as 0. This means that in this scenario, the model's primary objective is to accurately classify normal network traffic

These results are reflecting the model's reliability in distinguishing between normal and anomaly network behaviors, thereby contributing to enhance the network security.

Top of Form