UNIFIED MENTOR

YOUR SKILL, SUCCESS & JOURNEY

DATA SCIENCE INTERNSHIP

Project 3 – Cyber security: Suspicious Web Threat Interactions

UMID03072548194

Anjali Shibu

anjalishi1994@gmail.com

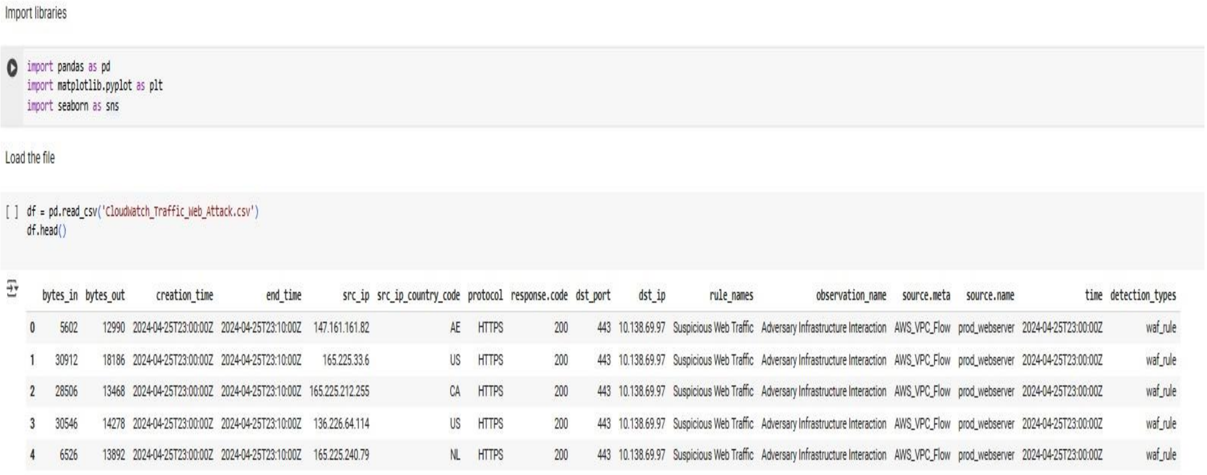
Table of Contents

Introduction 3
Import libraries and load the file as pandas DataFrame 3
Variable description 4
Unique values. 5
Data cleaning 5
Summary statistics 5
EDA 6
Time series analysis and visualization 8
Rolling average of incoming traffic 8
Geographic and IP based analysis 9
Anomaly detection with machine learning 10
Predictive modelling for traffic forecasting 12
Conclusion 13

Introduction

Cybersecurity is very important these days. The given data set contains web traffic records collected through AWS CloudWatch , aimed at detecting suspicious activities and potential attack attempts. Data is loaded as pandas dataframe,cleaned, analysed and meaningful insights are derived. This project includes time series analysis,anomaly detection with machine learning and predictive modelling for traffic forecasting .

Import libraries and load the file as pandas DataFrame



Import libraries

```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

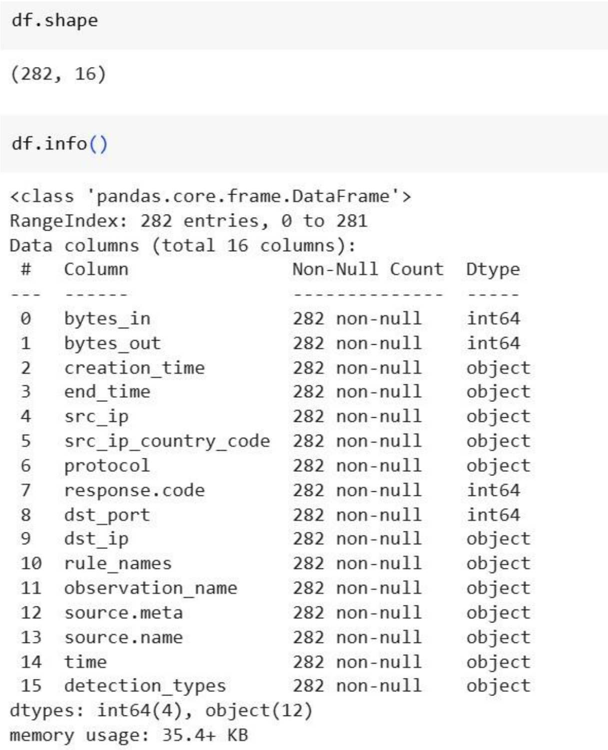
Load the file

```python
df = pd.read_csv('CloudWatch_Traffic_Web_Attack.csv')
df.head()
```

	bytes_in	bytes_out	creation_time	end_time	src_ip	src_ip_country_code	protocol	response.code	dst_port	dst_ip	rule_names	observation_name	source.meta	source.name	time	detection_types
0	5602	12990	2024-04-25T23:00:00Z	2024-04-25T23:10:00Z	147.161.161.82	AE	HTTPS	200	443	10.138.69.97	Suspicious Web Traffic	Adversary Infrastructure Interaction	AWS_VPC_Flow	prod_webserver	2024-04-25T23:00:00Z	waf_rule
1	30912	18186	2024-04-25T23:00:00Z	2024-04-25T23:10:00Z	165.225.33.6	US	HTTPS	200	443	10.138.69.97	Suspicious Web Traffic	Adversary Infrastructure Interaction	AWS_VPC_Flow	prod_webserver	2024-04-25T23:00:00Z	waf_rule
2	28506	13468	2024-04-25T23:00:00Z	2024-04-25T23:10:00Z	165.225.212.255	CA	HTTPS	200	443	10.138.69.97	Suspicious Web Traffic	Adversary Infrastructure Interaction	AWS_VPC_Flow	prod_webserver	2024-04-25T23:00:00Z	waf_rule
3	30546	14278	2024-04-25T23:00:00Z	2024-04-25T23:10:00Z	136.226.64.114	US	HTTPS	200	443	10.138.69.97	Suspicious Web Traffic	Adversary Infrastructure Interaction	AWS_VPC_Flow	prod_webserver	2024-04-25T23:00:00Z	waf_rule
4	6526	13892	2024-04-25T23:00:00Z	2024-04-25T23:10:00Z	165.225.240.79	NL	HTTPS	200	443	10.138.69.97	Suspicious Web Traffic	Adversary Infrastructure Interaction	AWS_VPC_Flow	prod_webserver	2024-04-25T23:00:00Z	waf_rule

Here we use dataframe df to load the file and see first 5 rows of the data using df.head().

To get details of shape of the dataframe we use df.shape and to get attribute information we use df.info().



```
df.shape
```

```
(282, 16)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 282 entries, 0 to 281
Data columns (total 16 columns):
 #   Column               Non-Null Count  Dtype 
---  ------               --------------  ----- 
 0   bytes_in             282 non-null    int64 
 1   bytes_out            282 non-null    int64 
 2   creation_time        282 non-null    object
 3   end_time             282 non-null    object
 4   src_ip               282 non-null    object
 5   src_ip_country_code  282 non-null    object
 6   protocol             282 non-null    object
 7   response.code        282 non-null    int64 
 8   dst_port             282 non-null    int64 
 9   dst_ip               282 non-null    object
 10  rule_names           282 non-null    object
 11  observation_name     282 non-null    object
 12  source.meta          282 non-null    object
 13  source.name          282 non-null    object
 14  time                 282 non-null    object
 15  detection_types      282 non-null    object
dtypes: int64(4), object(12)
memory usage: 35.4+ KB
```

From the above figure we see the dataset has 282 rows and 16 columns. There are no missing values in columns. Most of the data types are object. Some data cleaning is necessary for gaining insights from it.

Variable description

16 variables are present, the details of the variables are given below.

bytes_in: Bytes received by the server.

bytes_out: Bytes sent from the server.

creation_time: Timestamp of when the record was created.

end_time: Timestamp of when the connection ended.

src_ip: Source IP address.

src_ip_country_code: Country code of the source IP.

protocol: Protocol used in the connection.

response.code: HTTP response code.

dst_port: Destination port on the server.

dst_ip: Destination IP address.

rule_names: Name of the rule that identified the traffic as suspicious.

observation_name: Observations associated with the traffic.

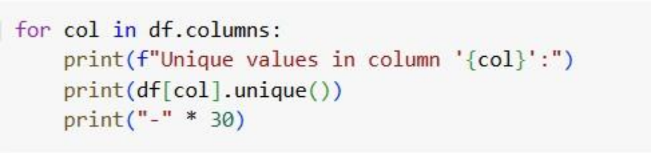
source.meta: Metadata related to the source.

source.name: Name of the traffic source.

time: Timestamp of the detected event.

detection_types: Type of detection applied

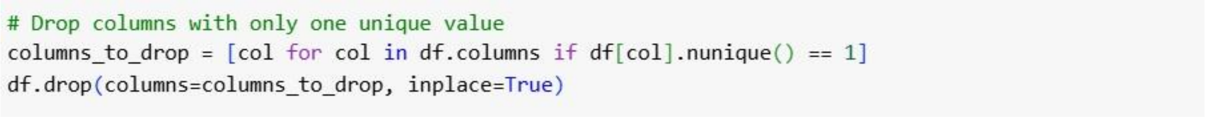
Unique values



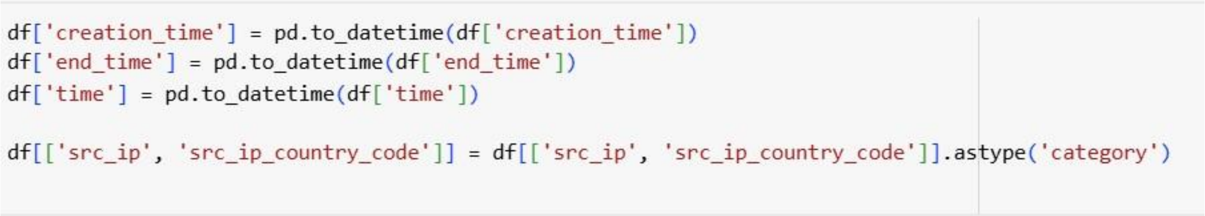
```
for col in df.columns:
    print(f"Unique values in column '{col}':")
    print(df[col].unique())
    print("-" * 30)
```

While checking for unique values in each variable ,we find some variables have just a single value which do not contribute much to gain insights. Some data cleaning is necessary.

Data cleaning

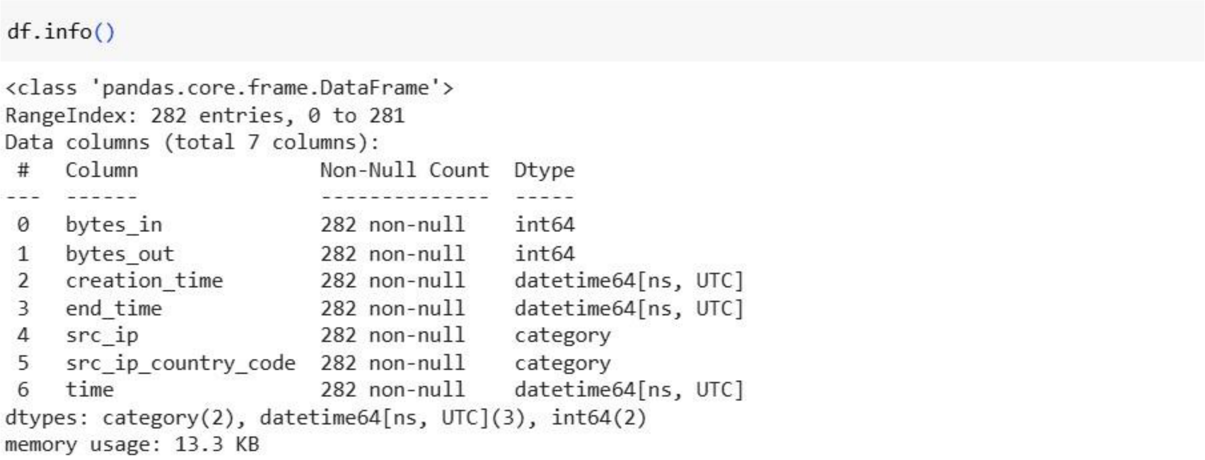


```
# Drop columns with only one unique value
columns_to_drop = [col for col in df.columns if df[col].nunique() == 1]
df.drop(columns=columns_to_drop, inplace=True)
```



```
df['creation_time'] = pd.to_datetime(df['creation_time'])
df['end_time'] = pd.to_datetime(df['end_time'])
df['time'] = pd.to_datetime(df['time'])

df[['src_ip', 'src_ip_country_code']] = df[['src_ip', 'src_ip_country_code']].astype('category')
```



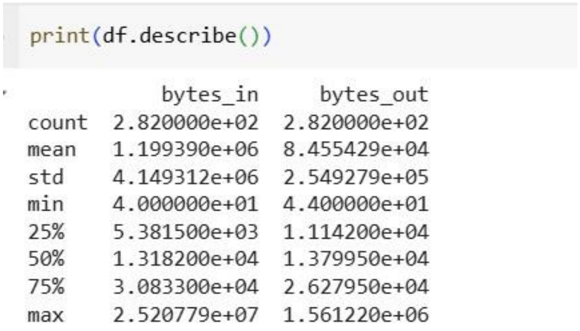
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 282 entries, 0 to 281
Data columns (total 7 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   bytes_in             282 non-null    int64
 1   bytes_out            282 non-null    int64
 2   creation_time        282 non-null    datetime64[ns, UTC]
 3   end_time             282 non-null    datetime64[ns, UTC]
 4   src_ip               282 non-null    category
 5   src_ip_country_code  282 non-null    category
 6   time                 282 non-null    datetime64[ns, UTC]
dtypes: category(2), datetime64[ns, UTC](3), int64(2)
memory usage: 13.3 KB
```

We drop columns that have the same values .Creation time , end time and time is converted into date time format for temporal analysis and forecasting. Other variables are converted into category type for further analysis.

Summary statistics

We use df.describe() to get summary statistics for each numeric column.



```
print(df.describe())
```

```
          bytes_in     bytes_out
count  2.820000e+02  2.820000e+02
mean   1.199390e+06  8.455429e+04
std    4.149312e+06  2.549279e+05
min    4.000000e+01  4.400000e+01
25%    5.381500e+03  1.114200e+04
50%    1.318200e+04  1.379950e+04
75%    3.083300e+04  2.627950e+04
max    2.520779e+07  1.561220e+06
```

Dataset contains 282 network traffic observations.

Inbound traffic is significantly higher than outbound traffic on average :

Mean inbound : 1199390 bytes

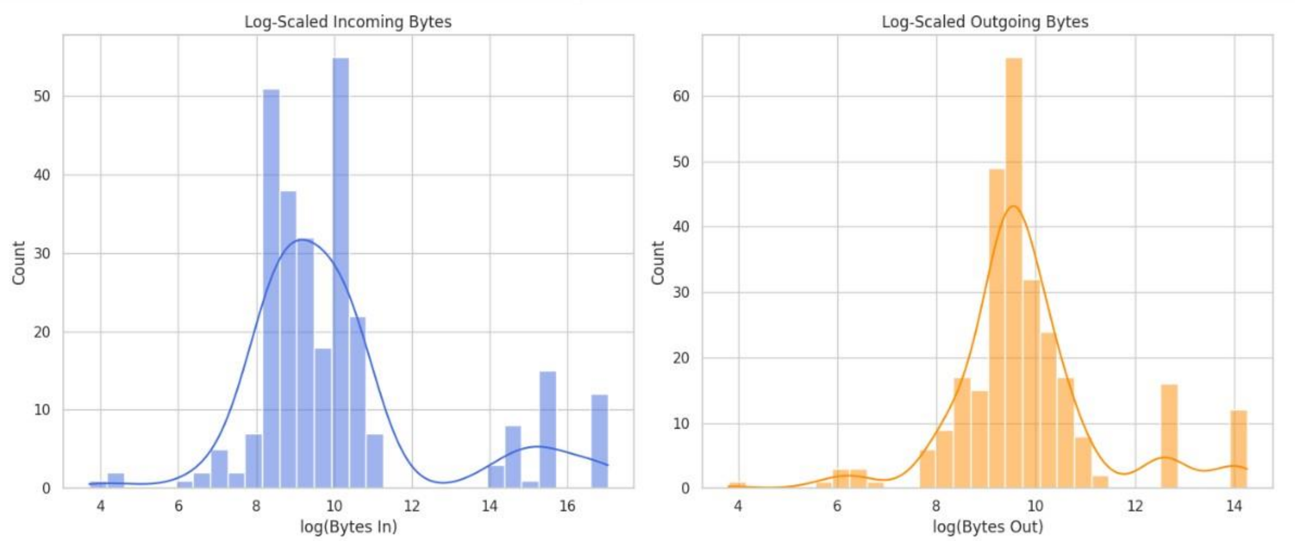
Mean outbound : 84554.29 bytes

High variability in traffic is seen as both metrics show large standard deviations. Bytes_in =4.1M bytes, bytes_out – 255k bytes.

Extreme outliers are present. The maximum values are orders of magnitude larger than the 75th percentile. Peak inbound 25.2M bytes and peak outbound 1.56M bytes. Suggests potential anomalies like large file transfers, DDoS attempts or misconfigured devices.

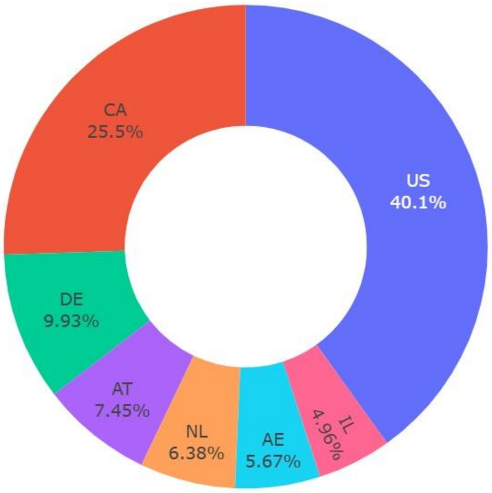
EDA

Distribution of bytes in and bytes out log scaled



The log-transformed distributions of incoming and outgoing bytes reveal that both metrics follow approximately log-normal patterns, typical for network traffic data. Incoming bytes show a wider spread compared to outgoing bytes,reinforcing the earlier observation of higher variability in inbound traffic. Both distributions peak at moderate values ,indicating most connections involve mid-sized data transfers. The long right tails confirm rare but extreme outliers, likely large file transfers or anomalies, that warrant further investigation.Log scaling help visualize these heavy-tailed distribution more clearly, masking skewness while preserving critical trends.

Suspicious traffic by country



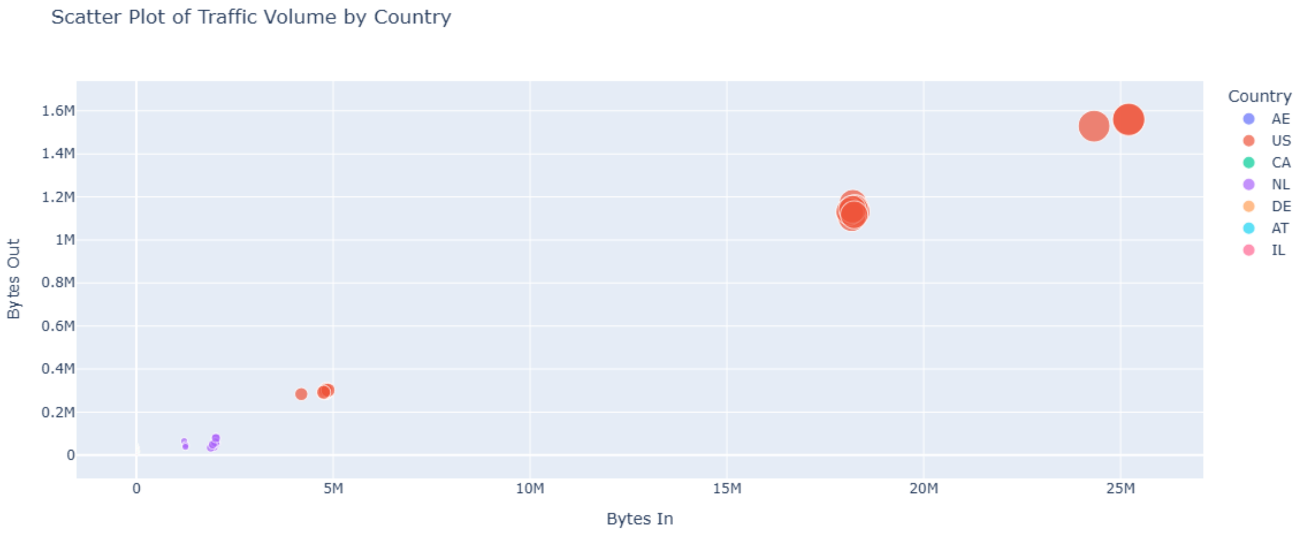
Network traffic originates predominantly from the United States(40.1%) and Canada (25.5%) reflecting strong North American engagement. European contributions are significant, led by Germany(9.93%) , Austria(7.45%) and the Netherlands (6.38%). The presence of the United Arab Emirates (5.67%) and Israel (4.96%) indicates activity in the Middle East.

US dominance suggests major operations, cloud providers , or user activity there.

European traffic may reflect distributed services or regional hubs.

Middle Eastern entries should be validated if unexpected.

Bytes in and out by country



In this scatterplot ,each data point represent traffic volume from a specific country.The size of the points corresponds to the relative volume of traffic, and colours distinguish countries.

The large bubbles toward the right-hand side of the chart indicate potentially significant servers or data centres with large-scale network activity.

Near the origin there are several smaller points .While they appear almost invisible on the static chart, interactive Plotly features allow hovering to inspect values in detail.These small volumes may represent either low-activity users or potential scanning/probing attempts.

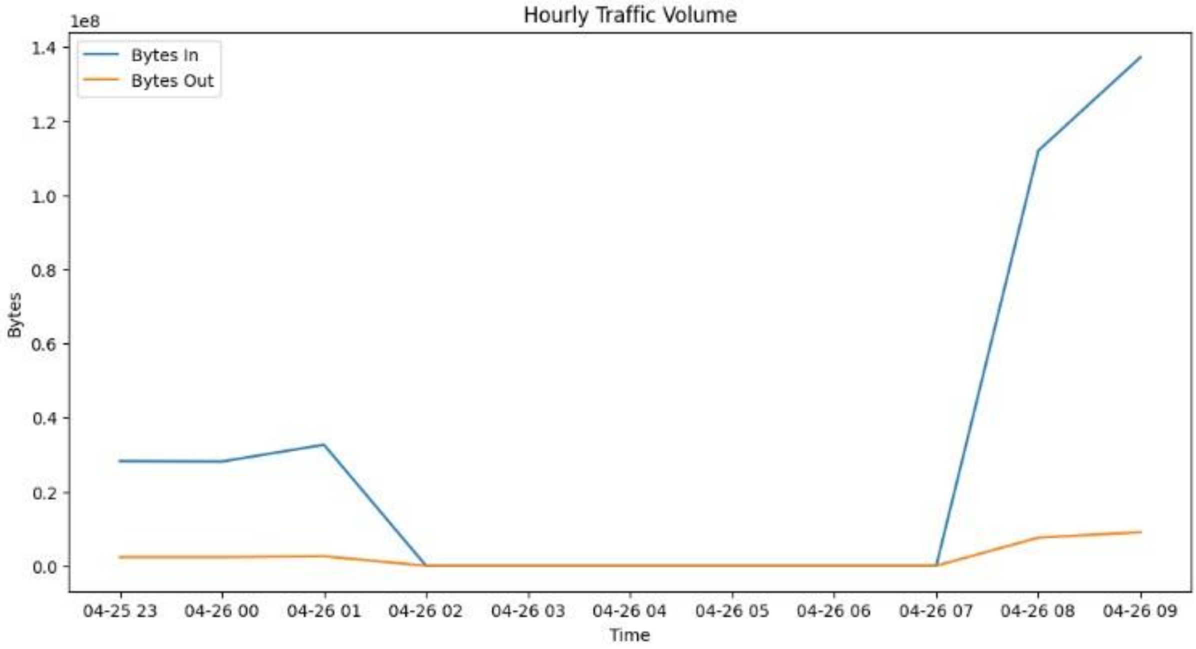
Cyber security insight – The dominance of few countries in traffic volume might warrant further investigation to determine whether this is normal operational activity or if anomalous spikes indicate suspicious connections or exfiltration attempts.

Low volume but frequent connections from smaller countries should also be investigated, as they could be reconnaissance attempts or distributed probes.

Time series analysis and visualization

Here we detect temporal patterns ,like increased activities at certain hours of the day.

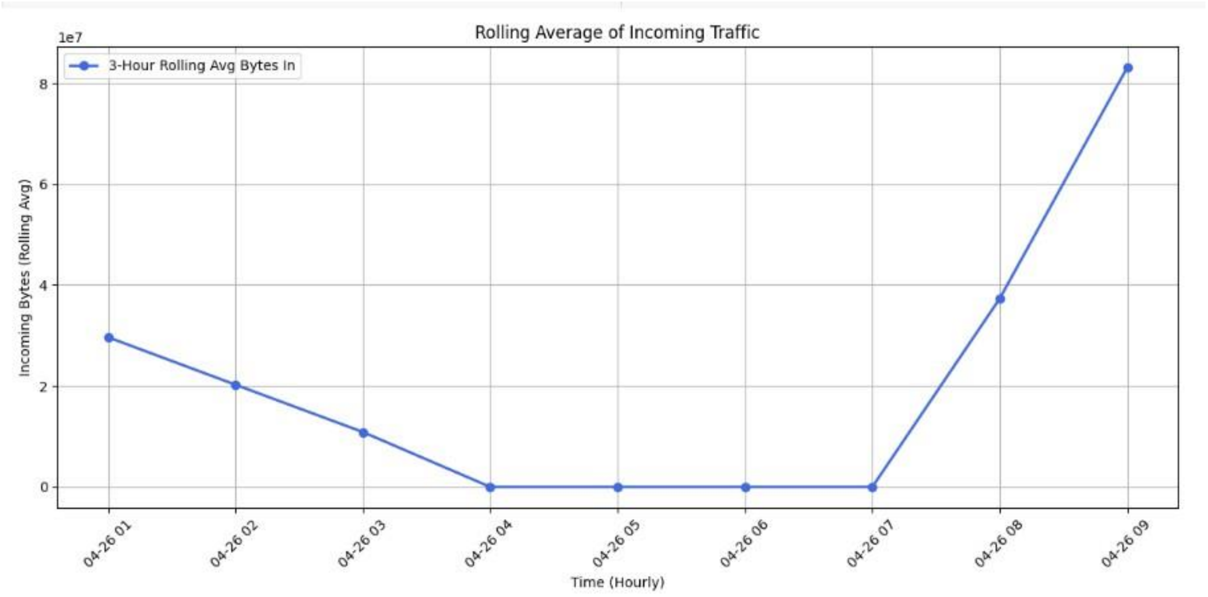
So we resample dataset into hourly intervals using pandas resample(). This is crucial for detecting peak attack periods.



A clear trend is observed where traffic remains relatively stable before dropping to near zero between 02:00 and 07:00 hours, followed by a sudden spike in inbound traffic(Bytes In) starting around 07:00, peaking sharply by 09:00. Outbound traffic (Bytes Out) remains consistently low throughout, with a minor increase during the same spike perios. These patterns are crucial from a cyber security perspective, as such sharp fluctuations may indicate coordinated attack attempts, unusual data transfers , or system misconfigurations occurring at specific times of the day. I dentifying these peak activity hours allows security teams to strengthen monitoring and response mechanisms during high-risk periods.

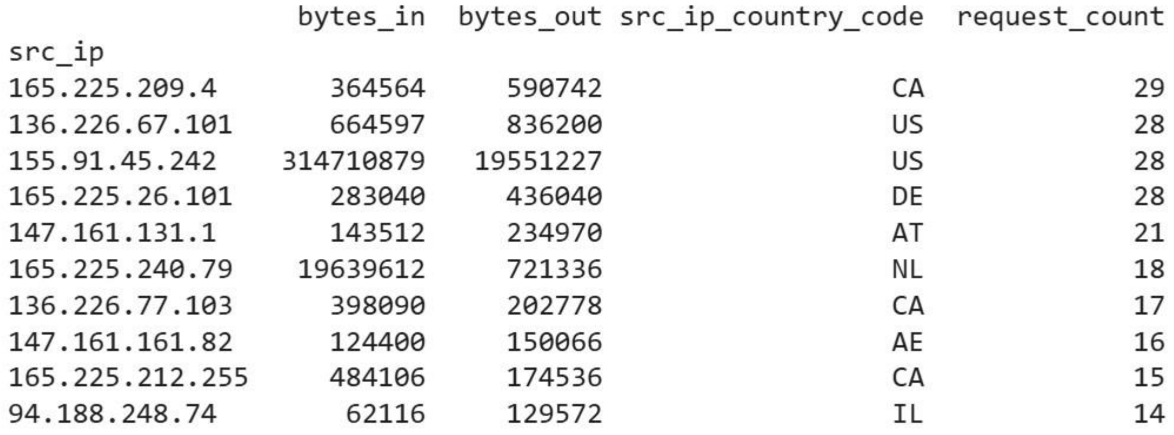
Rolling average of incoming traffic

To capture broader patterns and reduce noise in the dataset, a 3-hour rolling average of incoming traffic was calculated using Pandas rolling() method.

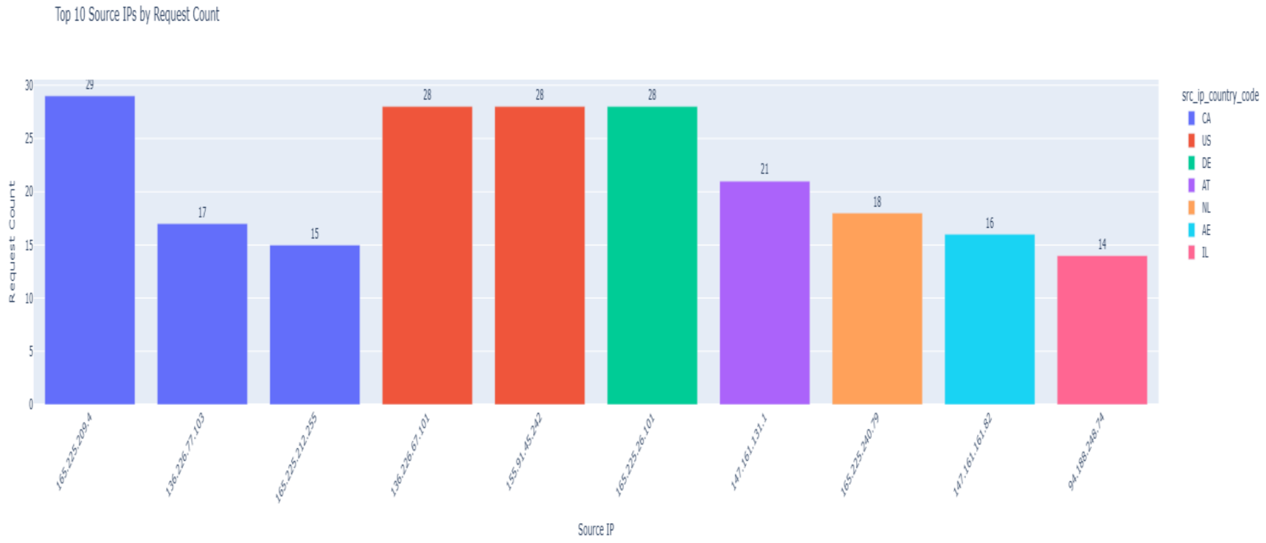


As shown in the plot above , the rolling average highlights a decline in incoming traffic between 01:00 and 04:00 hours, reaching almost zero, followed by a sharp rise starting from 07:00, peaking at 09:00. This smoothed trend reinforces earlier observations from the hourly traffic plot, confirming that network activity is highly time-dependent with concentrated bursts of data flow during specific periods. From a cyber security perspective , rolling averages are particularly useful for detecting sustained anomalies or attack windows, as they filter out short-lived fluctuations and emphasize persistent traffic surges that may require closer investigation.

Geographic and IP based analysis



```
                    bytes_in  bytes_out src_ip_country_code  request_count
src_ip
165.225.209.4         364564     590742                  CA             29
136.226.67.101        664597     836200                  US             28
155.91.45.242      314710879   19551227                  US             28
165.225.26.101        283040     436040                  DE             28
147.161.131.1         143512     234970                  AT             21
165.225.240.79      19639612     721336                  NL             18
136.226.77.103        398090     202778                  CA             17
147.161.161.82        124400     150066                  AE             16
165.225.212.255       484106     174536                  CA             15
94.188.248.74          62116     129572                  IL             14
```



The data highlights the top 10 source IP addresses by request count , revealing significant activity from diverse geographic locations. The most active IP is 165.225.209.4 (Canada) ,made 29 requests, followed by three IP's 136.226.67.101(US), 155.91.45.242 (US) and 165.225.26.101(Germany) each with 28 requests. Notably, 155.91.45.242(US) stands out with exceptionally high data transfer (314 MB in / 19.5 MB out), warranting further investigation for potential anomalies. Geographic distribution shows concentrated traffic from North America (US, Canada) and Europe(Germany, Austria, Netherlands), with additional activity from the Middle East (UAE,Israel). This analysis suggests monitoring high-volume IPs for unusual patterns and considering regional access controls if needed.

Anomaly detection with machine learning

To detect anomalies in the data , I implemented an unsupervised machine learning approach using Isolation Forest algorithm.This is effective for identifying rare and unusual patterns in datasets such as potential security threats orabnormal user behaviour.

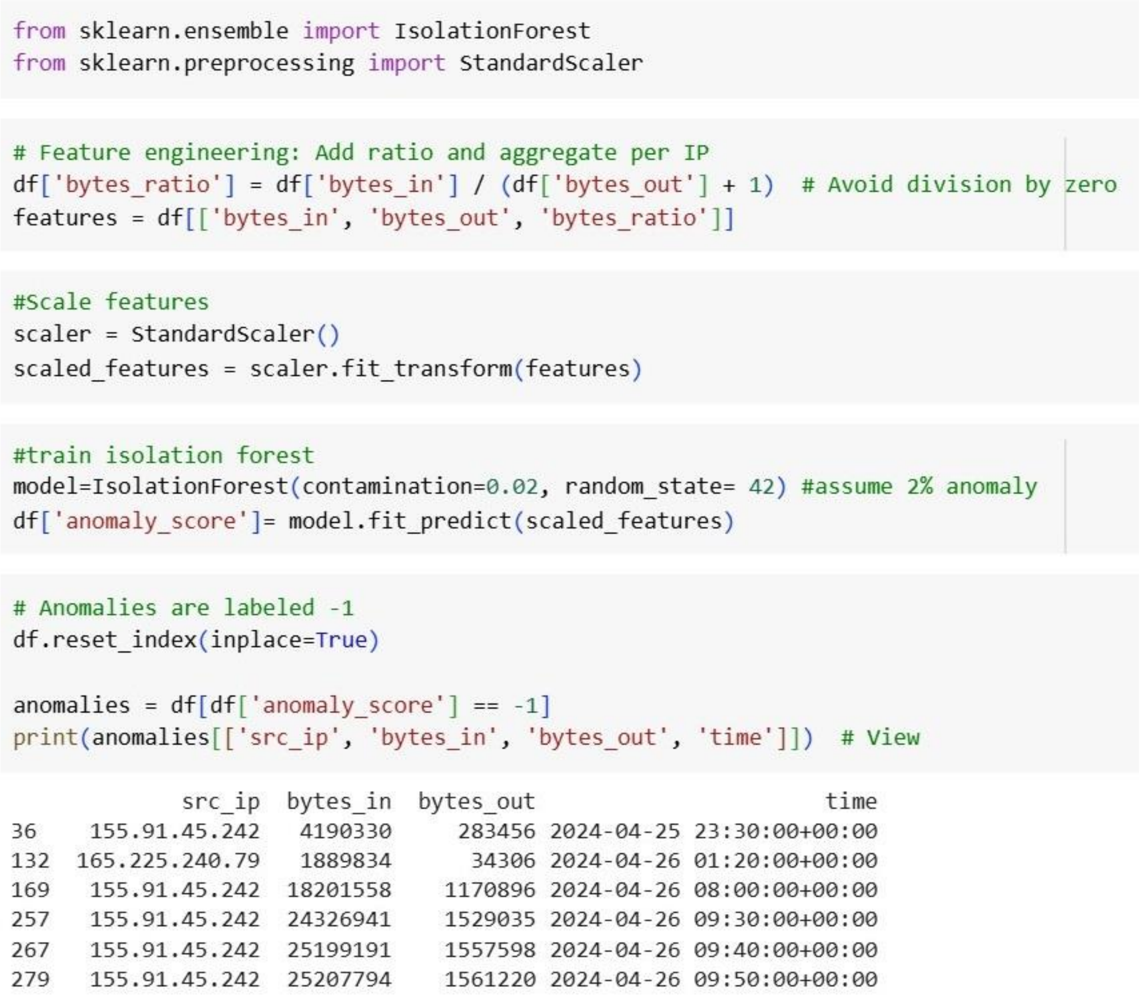
Steps taken are :

1) Feature engineering – created a new feature bytes_ratio to capture the relationship between incoming and outgoing traffic.

2)Feature selection and scaling – selected relevant features and applied StandardScaler to normalize the data for better model performance.

3)Model training –Trained an Isolation Forest model assuming 2% contamination rate.

4)Anomaly identification- anomalies were labelled a score of -1. Extracted and printed the anomalous records for further analysis.



```
from sklearn.ensemble import IsolationForest
from sklearn.preprocessing import StandardScaler
```

```
# Feature engineering: Add ratio and aggregate per IP
df['bytes_ratio'] = df['bytes_in'] / (df['bytes_out'] + 1)  # Avoid division by zero
features = df[['bytes_in', 'bytes_out', 'bytes_ratio']]
```

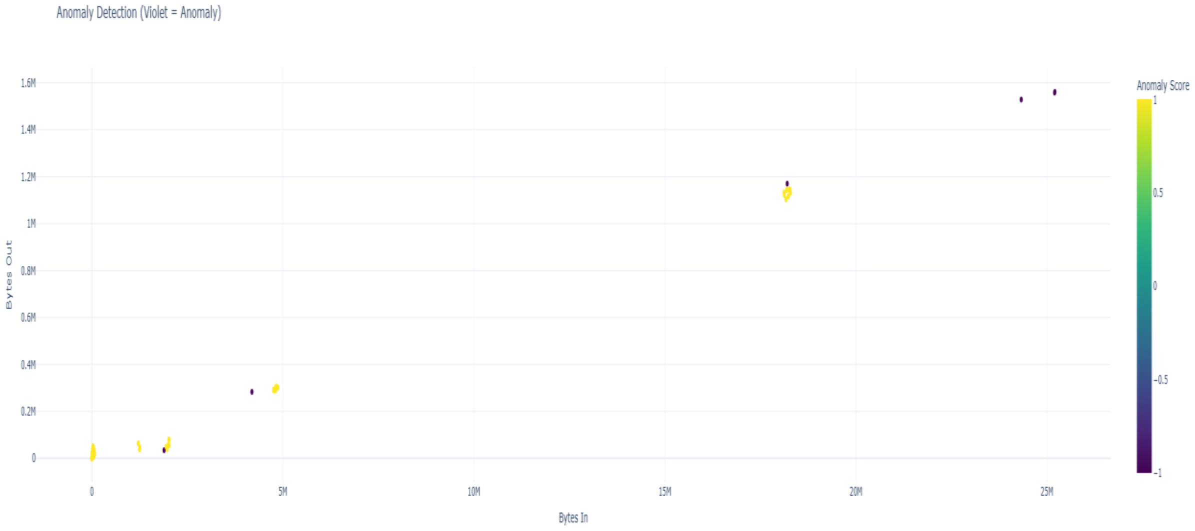
```
#Scale features
scaler = StandardScaler()
scaled_features = scaler.fit_transform(features)
```

```
#train isolation forest
model=IsolationForest(contamination=0.02, random_state= 42) #assume 2% anomaly
df['anomaly_score']= model.fit_predict(scaled_features)
```

```
# Anomalies are labeled -1
df.reset_index(inplace=True)

anomalies = df[df['anomaly_score'] == -1]
print(anomalies[['src_ip', 'bytes_in', 'bytes_out', 'time']])  # View
```

```
             src_ip  bytes_in  bytes_out                      time
36     155.91.45.242   4190330     283456 2024-04-25 23:30:00+00:00
132  165.225.240.79   1889834      34306 2024-04-26 01:20:00+00:00
169    155.91.45.242  18201558    1170896 2024-04-26 08:00:00+00:00
257    155.91.45.242  24326941    1529035 2024-04-26 09:30:00+00:00
267    155.91.45.242  25199191    1557598 2024-04-26 09:40:00+00:00
279    155.91.45.242  25207794    1561220 2024-04-26 09:50:00+00:00
```



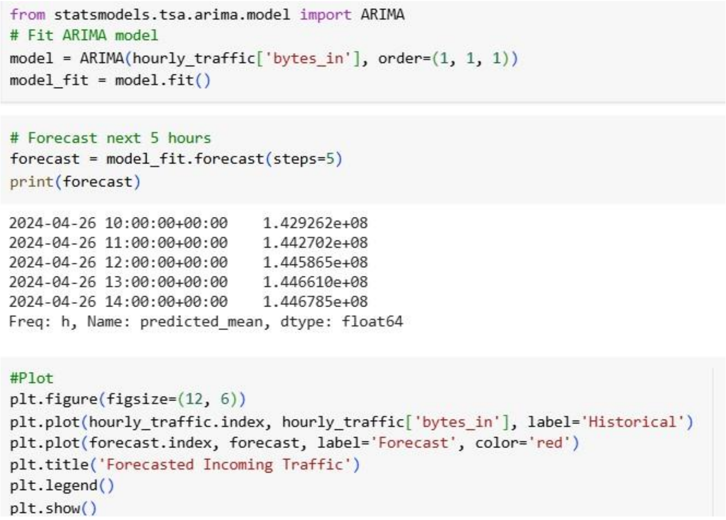
Most data points are clustered near the origin, suggesting typical low-volume traffic. A few points are spread out in the higher range of Bytes in(20M-30M) , which may represent large data transfers. The scatter plot above displays anomaly scores over time, with violet coloured spikes indicating detected anomalies scattered across both high and low traffic volumes. These anomalies can be due to unusual user behaviour, potential data exfiltration, misconfigured systems or spikes in usage.

Further investigation into the violet-labelled points is recommended to determine if they represent benign anomalies or potential security incidents.

Predictive modelling for traffic forecasting

Forecasting incoming traffic using ARIMA(AutoRegressive Integrated Moving Average)

To anticipate future network load and detect potential capacity issues, I implemented a time series forecasting model using ARIMA. This helps in proactive resource planning and anomaly detection based on expected traffic patterns



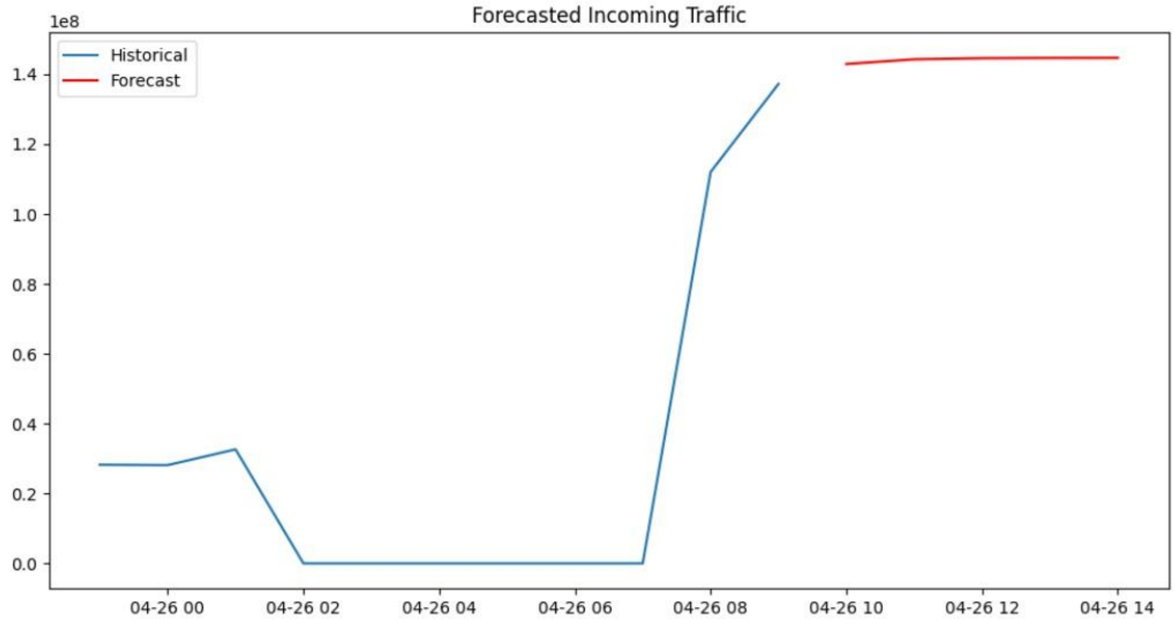
```
from statsmodels.tsa.arima.model import ARIMA
# Fit ARIMA model
model = ARIMA(hourly_traffic['bytes_in'], order=(1, 1, 1))
model_fit = model.fit()
```

```
# Forecast next 5 hours
forecast = model_fit.forecast(steps=5)
print(forecast)
```

```
2024-04-26 10:00:00+00:00    1.429262e+08
2024-04-26 11:00:00+00:00    1.442702e+08
2024-04-26 12:00:00+00:00    1.445865e+08
2024-04-26 13:00:00+00:00    1.446610e+08
2024-04-26 14:00:00+00:00    1.446785e+08
Freq: h, Name: predicted_mean, dtype: float64
```

```
#Plot
plt.figure(figsize=(12, 6))
plt.plot(hourly_traffic.index, hourly_traffic['bytes_in'], label='Historical')
plt.plot(forecast.index, forecast, label='Forecast', color='red')
plt.title('Forecasted Incoming Traffic')
plt.legend()
plt.show()
```

ARIMA model was chosen due to its effectiveness in modeling temporal dependencies in univariate time series data.I used the bytes_in metric aggregated hourly to capture traffic trends. Configured the model with parameters p=1,d=1,q=1. These parameters were tuned based on autocorrelation and partial autocorrelation plots.Then we forecast next 5 hours of incoming traffic. The forecast is plotted.



The forecast shows a sharp recovery in incoming traffic after a drop , followed by a stable high volume trend. Using ARIMA(1,1,1) model , we predicted traffic for the next 5 hours.The forecast suggests sustained activity, which may reflect normal operations or require further investigation depending on context. This visualization supports proactive monitoring and resource planning.

Conclusion

This cybersecurity analytics project successfully identified key patterns and threats in network traffic through comprehensive exploratory analysis, anomaly detection, and predictive modeling. The Isolation Forest algorithm flagged high-risk IPs with unusual data transfers, while ARIMA forecasting enabled proactive traffic monitoring. Geographic analysis revealed concentrated traffic from North America and Europe, with specific IPs warranting further investigation. The project demonstrates how data science can enhance threat detection combining statistical analysis, machine learning and time series modeling to transform raw logs into actionable security insights.