Cyber Attack Data Exploration

```
#importing all the necessary Python libraries and the dataset
import warnings
warnings.filterwarnings('ignore')
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns

data=pd.read_csv("cybersecurity_attacks.csv")
```

```
import warnings
warnings.filterwarnings('ignore')
```

```
data
```

	Timestamp	Source IP Address	Destination IP Address	Source Port	Destination Port	Protocol	Packet Length
0	2023-05-30 06:33:58	103.216.15.12	84.9.164.252	31225	17616	ICMP	503

```
# general information
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 138 entries, 2020-01-31 to 2023-10-31
Data columns (total 2 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   Attack Type  138 non-null    object
 1   value        138 non-null    int64 
dtypes: int64(1), object(1)
memory usage: 3.2+ KB
```

10:38:46

EDA(Exploratory data analysis)

```
# View the first few rows of the dataset
data.head()
```

	Timestamp	Source IP Address	Destination IP Address	Source Port	Destination Port	Protocol	Packet Length	Pa
0	2023-05-30 06:33:58	103.216.15.12	84.9.164.252	31225	17616	ICMP	503	
1	2020-08-26 07:08:30	78.199.217.198	66.191.137.154	17245	48166	ICMP	1174	
2	2022-11-13 08:23:25	63.79.210.48	198.219.82.17	16811	53600	UDP	306	C
3	2023-07-02 10:38:46	163.42.196.10	101.228.192.255	20018	32534	UDP	385	
4	2023-07-16 13:11:07	71.166.185.76	189.243.174.238	6131	26646	TCP	1462	

5 rows × 25 columns

```
# View the last few rows of the dataset
data.tail()
```

	Timestamp	Source IP Address	Destination IP Address	Source Port	Destination Port	Protocol	Packet Length
2343	2021-04-28 15:27:26	9.183.106.168	108.119.119.108	44392	3212	UDP	860
2344	2020-08-28 05:23:21	49.8.82.156	84.60.91.246	38941	45682	ICMP	308
2345	2023-07-13 03:46:53	34.179.210.92	104.164.142.98	54875	12814	UDP	1126
2346	2020-02-28 05:01:36	129.73.142.165	64.182.219.85	2467	58445	UDP	701

```
# give the number of rows and columns
data.shape
```

```
(2348, 25)
```

```
#give the number of rows
data.shape[0]
```

```
2348
```

```
#give the number of columns
data.shape[1]
```

```
25
```

```
# extract all columns of the dataset
data.columns
```

```
Index(['Timestamp', 'Source IP Address', 'Destination IP Address',
       'Source Port', 'Destination Port', 'Protocol', 'Packet Length',
       'Packet Type', 'Traffic Type', 'Payload Data', 'Malware Indicators',
       'Anomaly Scores', 'Alerts/Warnings', 'Attack Type', 'Attack Signature',
       'Action Taken', 'Severity Level', 'User Information',
       'Device Information', 'Network Segment', 'Geo-location Data',
       'Proxy Information', 'Firewall Logs', 'IDS/IPS Alerts', 'Log Source'],
      dtype='object')
```

Data Cleaning

```
# check for null values
data.isna().sum()
```

```
Timestamp                    0
Source IP Address            0
Destination IP Address       0
Source Port                  0
Destination Port             0
Protocol                     0
Packet Length                0
Packet Type                  0
Traffic Type                 0
Payload Data                 0
Malware Indicators        1176
Anomaly Scores               0
Alerts/Warnings           1187
Attack Type                  0
Attack Signature             0
Action Taken                 0
Severity Level               0
```

```
User Information             0
Device Information           0
Network Segment              1
Geo-location Data            1
Proxy Information         1148
Firewall Logs             1197
IDS/IPS Alerts            1163
Log Source                   1
dtype: int64
```

```
data.isnull()
```

	Timestamp	Source IP Address	Destination IP Address	Source Port	Destination Port	Protocol	Packet Length	Packet Type
0	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...
2343	False	False	False	False	False	False	False	False
2344	False	False	False	False	False	False	False	False
2345	False	False	False	False	False	False	False	False
2346	False	False	False	False	False	False	False	False
2347	False	False	False	False	False	False	False	False

2348 rows × 25 columns

```
df.dropna(inplace=True)
```

```
df.isnull().sum()
```

```
Source Port         0
Destination Port    0
Packet Length       0
Anomaly Scores      0
dtype: int64
```

```
# calculate the mean , std, min, max and count of every attributes
data.describe()
```

	Source Port	Destination Port	Packet Length	Anomaly Scores
count	2348.000000	2348.000000	2348.000000	2348.000000
mean	32314.992760	32838.227428	788.548552	50.104647
std	18781.241745	18571.544069	413.974742	28.932539
min	1031.000000	1030.000000	64.000000	0.060000
25%	15965.500000	17096.250000	428.750000	24.657500
50%	31733.000000	32502.000000	786.000000	50.520000
75%	48357.000000	49076.500000	1143.250000	75.160000
max	65521.000000	65535.000000	1500.000000	99.990000

```
# Check for duplicate values
data.duplicated().sum()
```

```
0
```

```
# Checking Skewness from 'Source Port' to 'Anomaly Scores'

df=data.loc[:,'Source Port':'Anomaly Scores']
df=df.select_dtypes([np.int, np.float])
for i, col in enumerate(df.columns):
    print("\nSkewness of "+col +" is", df[col].skew()) #measures skewness
```

```

Skewness of Source Port is 0.05946941104053445
```

```
    Skewness of Destination Port is 0.03858148110055104

    Skewness of Packet Length is -0.017914267379731216

    Skewness of Anomaly Scores is -0.02083080381904235
```

```
# Check unique values
data["Traffic Type"].unique()
```

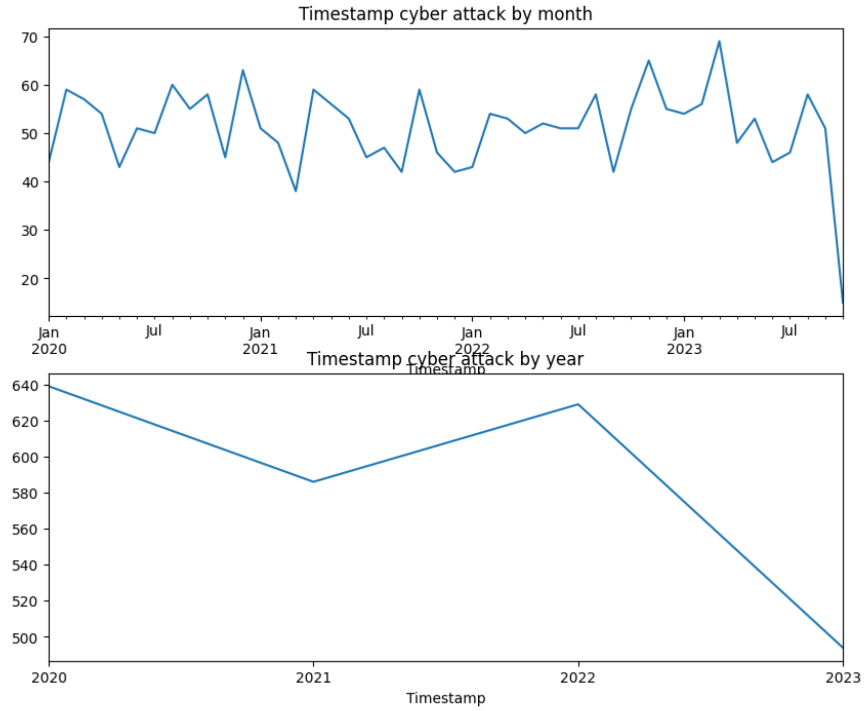
```
    array(['HTTP', 'DNS', 'FTP'], dtype=object)
```

```
data["Attack Type"].unique()
```

```
    array(['Malware', 'DDoS', 'Intrusion'], dtype=object)
```

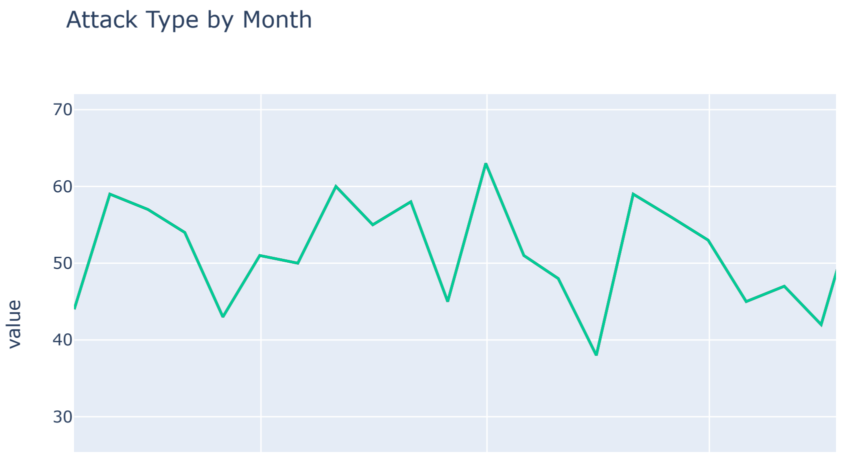
```
# Assuming 'data' is your DataFrame and 'Timestamp' is the column containing timestamps
data['Timestamp'] = pd.to_datetime(data['Timestamp'])

fig = plt.figure(figsize=(10, 8))
fig.add_subplot(211)
data.resample('M', on='Timestamp')['Attack Type'].count().plot(title='Timestamp cyber attack by month')


fig.add_subplot(212)
data.resample('Y', on='Timestamp')['Attack Type'].count().plot(title='Timestamp cyber attack by year')
```

```
    <Axes: title={'center': 'Timestamp cyber attack by year'}, xlabel='Timestamp'>
```


Timestamp cyber attack by month

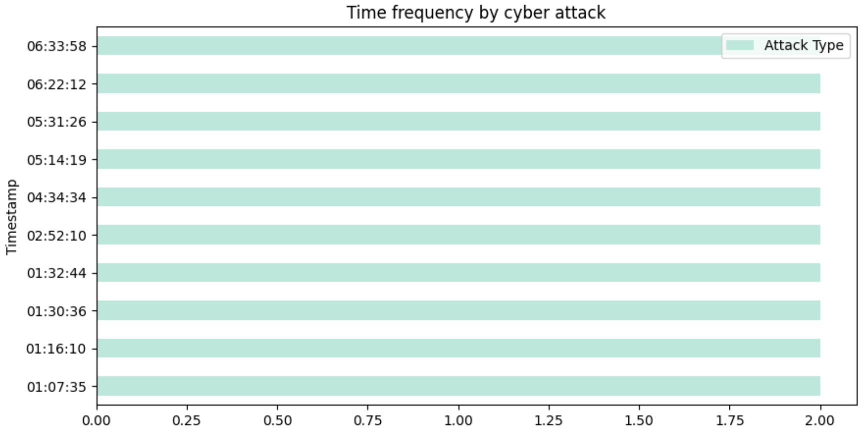
Timestamp cyber attack by year

```
import plotly.express as px

data = pd.crosstab(data['Timestamp'], data['Attack Type']).resample('M').count().melt(ignore_index=False)
px.line(data, x=data.index, y='value', color='Attack Type', title='Attack Type by Month').show()
```



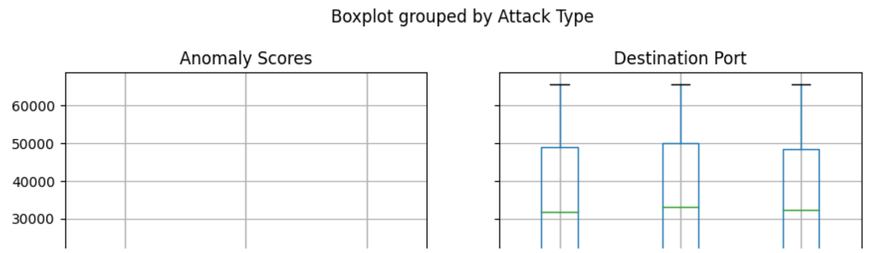
```
data['Timestamp'] = pd.to_datetime(data['Timestamp'])
data.groupby(data['Timestamp'].dt.time).agg({'Attack Type': 'count'}).nlargest(10, 'Attack Type').plot(kind='barh', figsize=(10, 5), co
```

```
<Axes: title={'center': 'Time frequency by cyber attack'}, ylabel='Timestamp'>
```

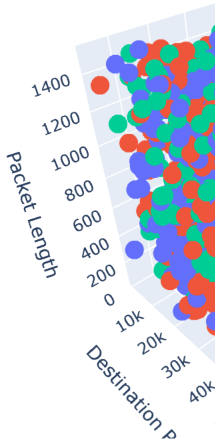


```
data.boxplot(figsize=(10,8), by='Attack Type')
```

```
array([[<Axes: title={'center': 'Anomaly Scores'}, xlabel='[Attack Type]'>,
        <Axes: title={'center': 'Destination Port'}, xlabel='[Attack Type]'>],
       [<Axes: title={'center': 'Packet Length'}, xlabel='[Attack Type]'>,
        <Axes: title={'center': 'Source Port'}, xlabel='[Attack Type]'>]],
      dtype=object)
```

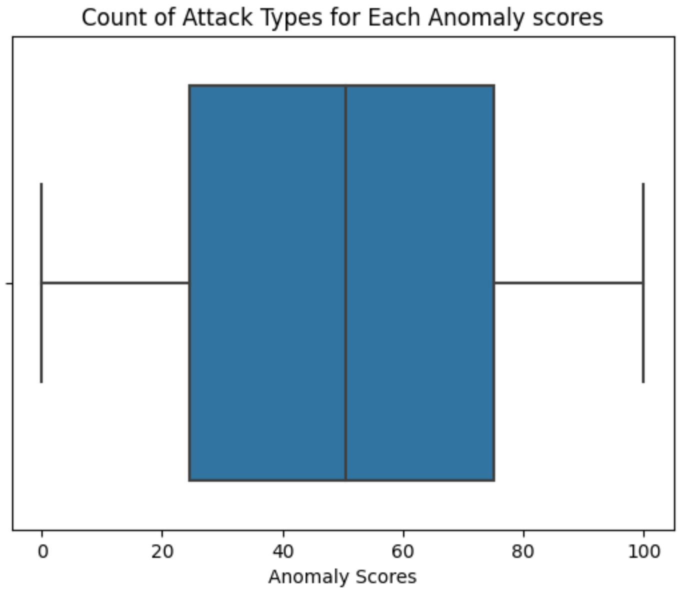


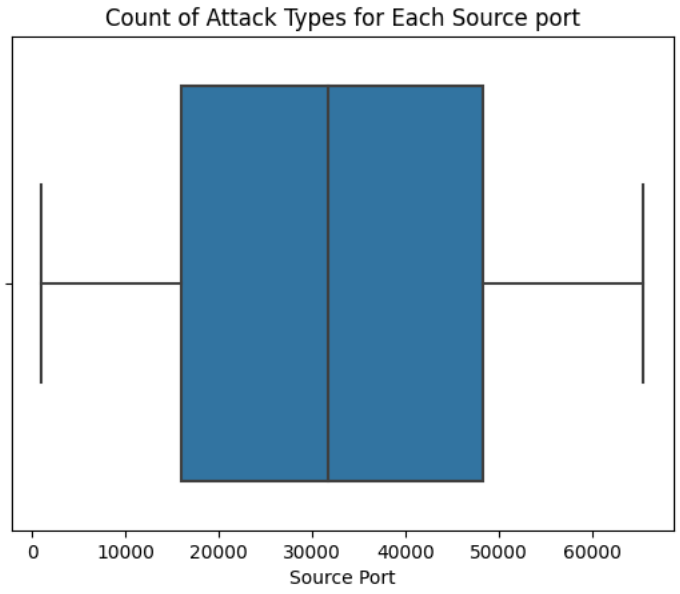
```
px.scatter_3d(data, x='Source Port', y='Destination Port', z='Packet Length', color='Protocol').show()
```

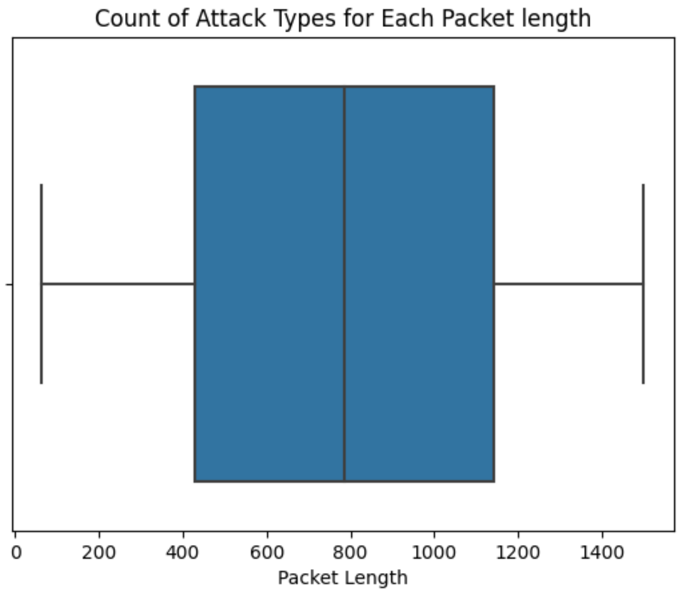


```
pd.crosstab(data['Geo-location Data'], data['Attack Type']).nlargest(5, columns='Malware')['Malware'].plot(kind='barh',figsize=(8,7), t:
    <Axes: title={'center': 'Malware Attack by location'}, xlabel='Geo-location Data'>
```

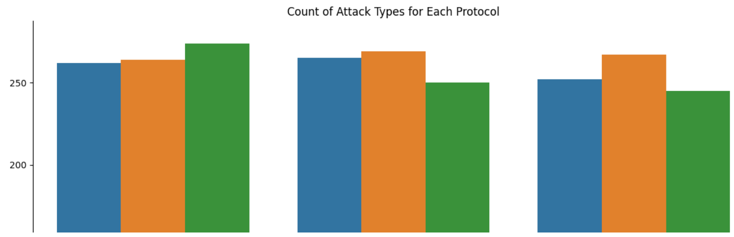
```
columns = ['Anomaly Scores','Source Port','Packet Length']

for col in columns:
    sns.boxplot(data=data, x=col, hue='Attack Type')
    plt.title(f'Count of Attack Types for Each {col.capitalize()}')
    plt.show()
```







```
columns_to_plot = ['Protocol','Attack Signature','Action Taken','Network Segment']
for col in columns_to_plot:
    sns.catplot(data=data, x=col, hue='Attack Type', kind='count', height=8, aspect=1.5)
    plt.title(f'Count of Attack Types for Each {col.capitalize()}')
    plt.show

warnings.filterwarnings("default")
```



Data Visualization

```
data['Payload Data'].dtype
```

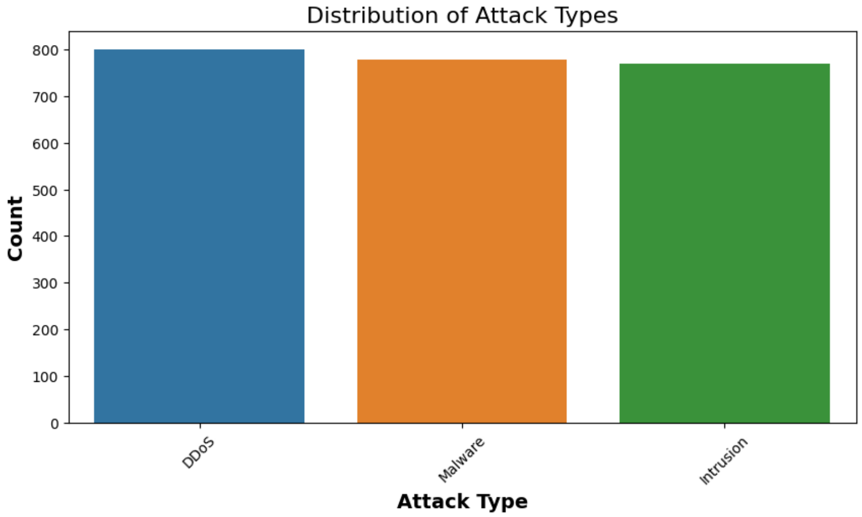
```
    dtype('O')
```

```
# Convert data to a string

text = str(data['Payload Data'])
```

+

Bar Chart

```
# Visualize the distribution of attack types

attack_counts = data['Attack Type'].value_counts()

plt.figure(figsize=(10, 5))
sns.barplot(x=attack_counts.index , y=attack_counts)

plt.xlabel('Attack Type',fontsize=14, fontweight='bold')
plt.ylabel('Count',fontsize=14, fontweight='bold')
plt.title('Distribution of Attack Types', fontsize=16)

plt.xticks(rotation=45)
plt.show()

print(attack_counts)
```



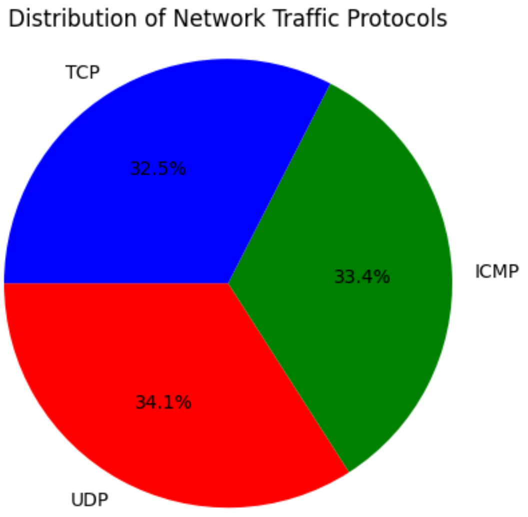
```
DDoS         800
Malware      779
Intrusion    769
Name: Attack Type, dtype: int64
```

Pie Charts

```
data['Protocol'].value_counts()
```

```
ICMP    800
UDP     784
TCP     764
Name: Protocol, dtype: int64
```

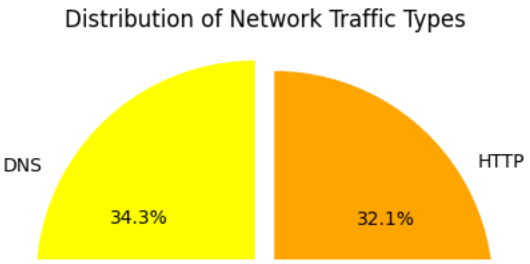
```
labels = ['UDP', 'ICMP', 'TCP']
sizes = data['Protocol'].value_counts()  # Proportional sizes of each category
colors = ['red', 'green', 'blue']  # Color for each category segment
explode = (0.1, 0, 0)  # Explode a slice if needed (0 means no explosion)


# Create a pie chart
plt.pie(sizes, labels=labels, colors=colors,  autopct='%1.1f%%', startangle=180)

plt.axis('equal')
plt.title('Distribution of Network Traffic Protocols')
plt.show()
```



```
data['Traffic Type'].value_counts()
```

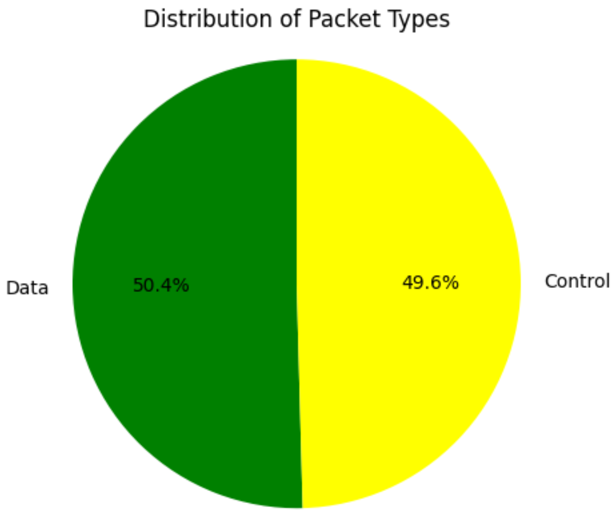
```
HTTP    805
DNS     790
FTP     753
Name: Traffic Type, dtype: int64
```

```
# Data for the pie chart

labels =['DNS','FTP','HTTP']
sizes = data['Traffic Type'].value_counts()
colors = ['yellow', 'green', 'orange']
explode = (0.1, 0, 0)

# Create a pie chart
plt.pie(sizes, labels=labels, colors=colors, explode=explode, autopct='%2.1f%%', startangle=90)

plt.axis('equal')
plt.title('Distribution of Network Traffic Types')

plt.show()
```



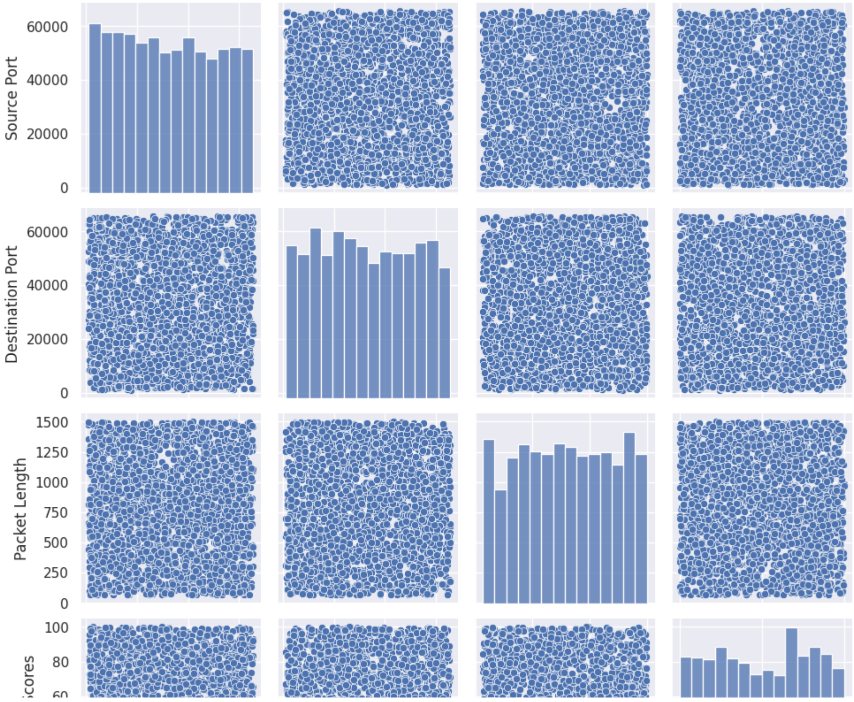
```
data['Packet Type'].value_counts()

     Control    1184
     Data       1164
     Name: Packet Type, dtype: int64
```

```
# Data for the pie chart
labels =['Data','Control']
sizes = data['Packet Type'].value_counts()
colors = ['green', 'yellow']
explode = (0, 0)
plt.pie(sizes, labels=labels, colors=colors, explode=explode, autopct='%1.1f%%', startangle=90)
plt.axis('equal')
plt.title('Distribution of Packet Types')

plt.show()
```



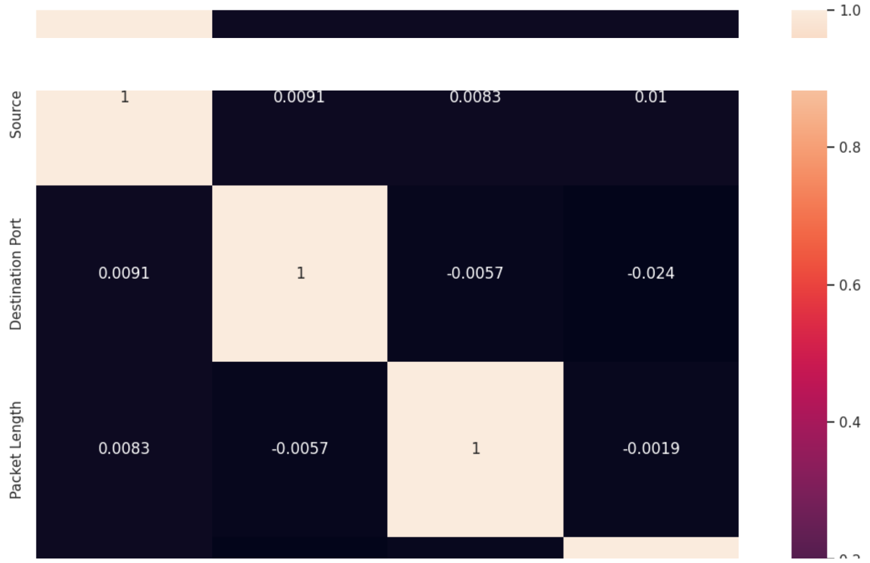
```
#Scatter plots of all columns
sns.set()
cols = ['Source Port','Destination Port','Packet Length','Anomaly Scores']
sns.pairplot(data[cols], size = 2.5)
plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/seaborn/axisgrid.py:2095: UserWarning:

The `size` parameter has been renamed to `height`; please update your code.
```



```
import warnings
warnings.filterwarnings('ignore')

#Correlation matrix
cormat = data.corr()
f, ax = plt.subplots(figsize = (15, 10))
sns.heatmap(corrmat, vmax = 1, square = True, annot = True)
```

```
<Axes: >
```


Source
Destination Port
Packet Length
1
0.0091
0.0083
0.01
0.0091
1
-0.0057
-0.024
0.0083
-0.0057
1
-0.0019
1.0
0.8
0.6
0.4