A look into global
cybersecurity threats

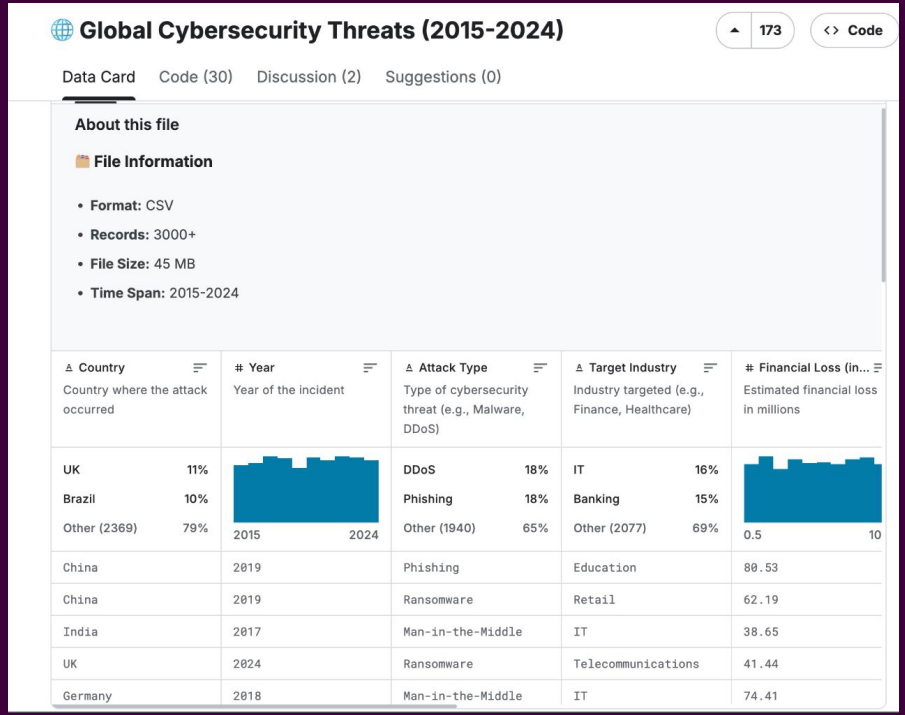
# CYBER SAFE

Intro to Data Science Math 104 Final- Sarah Collier

# This Dataset

This dataset was found on kaggle.com "The Global Cybersecurity Threats Dataset (2015-2024) provides extensive data on cyberattacks, malware types, targeted industries, and affected countries. It is designed for threat intelligence analysis, cybersecurity trend forecasting, and machine learning model development to enhance global digital security."

https://www.kaggle.com/datasets/atharvasoundankar/global-cybersecurity-threats-2015-2024



# DATA UNDERSTANDING AND CLEANING

SECTION 1

# Understanding the Dataset

*Data Understanding and Cleaning*

[35]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

[36]:
```
df = pd.read_csv("Global_Cybersecurity_Threats_2015-2024.csv")
```

[37]:
```
df.head()
df.info()
df.isnull().sum()
df.columns
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3000 entries, 0 to 2999
Data columns (total 10 columns):
 #   Column                                Non-Null Count  Dtype
---  ------                                --------------  -----
 0   Country                               3000 non-null   object
 1   Year                                  3000 non-null   int64
 2   Attack Type                           3000 non-null   object
 3   Target Industry                       3000 non-null   object
 4   Financial Loss (in Million $)         3000 non-null   float64
 5   Number of Affected Users              3000 non-null   int64
 6   Attack Source                         3000 non-null   object
 7   Security Vulnerability Type           3000 non-null   object
 8   Defense Mechanism Used                3000 non-null   object
 9   Incident Resolution Time (in Hours)   3000 non-null   int64
dtypes: float64(1), int64(3), object(6)
memory usage: 234.5+ KB
```

[37]:
```
Index(['Country', 'Year', 'Attack Type', 'Target Industry',
       'Financial Loss (in Million $)', 'Number of Affected Users',
       'Attack Source', 'Security Vulnerability Type',
       'Defense Mechanism Used', 'Incident Resolution Time (in Hours)'],
      dtype='object')
```

- Used basic functions like df.info() and df.columns to organize and understand data
- 3,000 rows
- 10 columns
- Visible data types

# Cleaning Requirements

```
columns_to_clean = ['Country', 'Attack Type', 'Target Industry', 'Attack Source', 'Security Vulnerability Type', 'Defense Mechanism Used']

for col in columns_to_clean:
    df[col] = df[col].str.strip().str.title()
```

```
df.head()
```

| | Country | Year | Attack Type | Target Industry | Financial Loss (in Million $) | Number of Affected Users | Attack Source | Security Vulnerability Type | Defense Mechanism Used | Incident Resolution Time (in Hours) |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | China | 2019 | Phishing | Education | 80.53 | 773169 | Hacker Group | Unpatched Software | Vpn | 63 |
| 1 | China | 2019 | Ransomware | Retail | 62.19 | 295961 | Hacker Group | Unpatched Software | Firewall | 71 |
| 2 | India | 2017 | Man-In-The-Middle | It | 38.65 | 605895 | Hacker Group | Weak Passwords | Vpn | 20 |
| 3 | Uk | 2024 | Ransomware | Telecommunications | 41.44 | 659320 | Nation-State | Social Engineering | Ai-Based Detection | 7 |
| 4 | Germany | 2018 | Man-In-The-Middle | It | 74.41 | 810682 | Insider | Social Engineering | Vpn | 68 |

```
num_duplicates = df.duplicated().sum()
print("Number of duplicate rows", num_duplicates)
df = df.drop_duplicates(keep='first')
print("New dataframe shape:", df.shape)
```

```
Number of duplicate rows 0
New dataframe shape: (3000, 10)
```

```
df.head()
```

| | Country | Year | Attack Type | Target Industry | Financial Loss (in Million $) |
|---|---|---|---|---|---|
| 0 | China | 2019 | Phishing | Education | 80.53 |

```
[42]: print(df['Year'].dtype)
      df['Year']= df['Year'].astype('category')
      print(df['Year'].dtype)
```

```
int64
category
```

- Standardize text formatting
- Remove duplicates
- Convert 'Year' to categorical
- Ensure numeric columns are valid

Changed Year to Categorical

# Cleaning Requirements

```
numeric_columns = ['Financial Loss (in Million $)', 'Number of Affected Users', 'Incident Resolution Time (in Hours)']
print(numeric_columns[0] + ' - Description:')
print(df[numeric_columns[0]].describe())

print(numeric_columns[1] + ' - Description:')
print(df[numeric_columns[1]].describe())

print(numeric_columns[2] + ' - Description:')
print(df[numeric_columns[2]].describe())

negative_count_0 = (df[numeric_columns[0]] < 0).sum()
print(numeric_columns[0] + ' - Negative values: ' + str(negative_count_0))

negative_count_1 = (df[numeric_columns[1]] < 0).sum()
print(numeric_columns[1] + ' - Negative values: ' + str(negative_count_1))

negative_count_2 = (df[numeric_columns[2]] < 0).sum()
print(numeric_columns[2] + ' - Negative values: ' + str(negative_count_2))
```

Ensured columns were valid using .describe() to show summary statistics

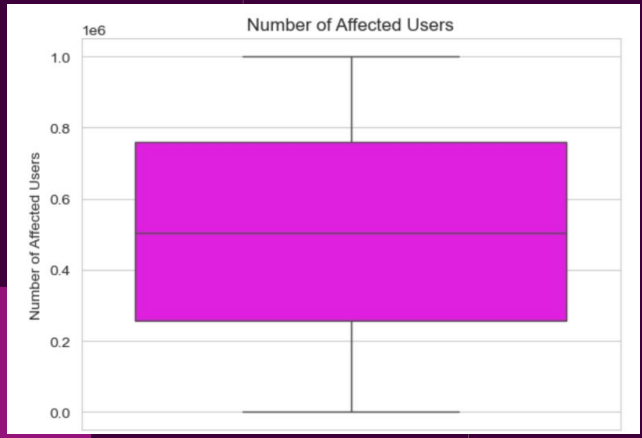
```
negative_count_1 = (df[numeric_columns[1]] < 0).sum()
print(numeric_columns[1] + ' - Negative values: ' + str(negative_count_1))

negative_count_2 = (df[numeric_columns[2]] < 0).sum()
print(numeric_columns[2] + ' - Negative values: ' + str(negative_count_2))
```

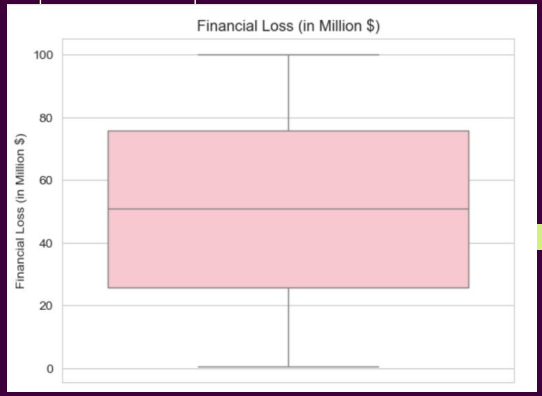
```
Financial Loss (in Million $) - Description:
count    3000.000000
mean       50.492970
std        28.791415
min         0.500000
25%        25.757500
50%        50.795000
75%        75.630000
max        99.990000
Name: Financial Loss (in Million $), dtype: float64
Number of Affected Users - Description:
count      3000.000000
mean     504684.136333
std      289944.084972
min         424.000000
25%      255805.250000
50%      504513.000000
75%      758088.500000
max      999635.000000
Name: Number of Affected Users, dtype: float64
Incident Resolution Time (in Hours) - Description:
count    3000.000000
mean       36.476000
std        20.570768
min         1.000000
25%        19.000000
50%        37.000000
75%        55.000000
max        72.000000
Name: Incident Resolution Time (in Hours), dtype: float64
Financial Loss (in Million $) - Negative values: 0
Number of Affected Users - Negative values: 0
Incident Resolution Time (in Hours) - Negative values: 0
```

```
plt.figure()
sns.boxplot(y=df['Financial Loss (in Million $)'], color = 'pink')
```

# Cleaning Requirements

```
plt.figure()
sns.boxplot(y=df['Financial Loss (in Million $)'], color = 'pink')
plt.title('Financial Loss (in Million $)')
plt.ylabel('Financial Loss (in Million $)')
plt.show()

plt.figure()
sns.boxplot(y=df['Number of Affected Users'], color = 'magenta')
plt.title('Number of Affected Users')
plt.ylabel('Number of Affected Users')
plt.show()

plt.figure()
sns.boxplot(y=df['Incident Resolution Time (in Hours)'], color = 'purple')
plt.title('Incident Resolution Time (in Hours)')
plt.ylabel('Incident Resolution Time (in Hours)')
plt.show()
```




Did not notice any significant outliers

# Cleaning Requirements

```
[46]: df['Loss Per User'] = (df['Financial Loss (in Million $)'] * 1000000) / df['Number of Affected Users']

      print(df.head())

      print(df['Loss Per User'].describe())

      import matplotlib.pyplot as plt
      import seaborn as sns

      plt.figure()
      sns.boxplot(y=df['Loss Per User'])
      plt.title('Loss Per User')
      plt.ylabel('Loss Per User ($)')
      plt.show()
```

```
   Country  Year        Attack Type     Target Industry  \
0    China  2019           Phishing           Education
1    China  2019         Ransomware              Retail
2    India  2017  Man-In-The-Middle                  It
3       Uk  2024         Ransomware  Telecommunications
4  Germany  2018  Man-In-The-Middle                  It

   Financial Loss (in Million $)  Number of Affected Users Attack Source  \
0                          80.53                    773169  Hacker Group
1                          62.19                    295961  Hacker Group
2                          38.65                    605895  Hacker Group
3                          41.44                    659320  Nation-State
4                          74.41                    810682       Insider

  Security Vulnerability Type Defense Mechanism Used  \
0          Unpatched Software                    Vpn
1          Unpatched Software               Firewall
2              Weak Passwords                    Vpn
3          Social Engineering     Ai-Based Detection
4          Social Engineering                    Vpn

   Incident Resolution Time (in Hours)  Loss Per User
0                                   63     104.155754
1                                   71     210.129037
2                                   20      63.789931
3                                    7      62.852636
4                                   68      91.786915
count      3000.000000
mean        515.184504
std        4325.834901
min           0.620994
25%          49.862220
50%         100.312440
75%         202.896344
max      141058.020478
Name: Loss Per User, dtype: float64
```


Create derived columns (e.g., Loss per User)

Divided total financial loss by affected users

# Exploratory Data

SECTION 2

# Initial Analysis

*Exploratory Data*

```
[47]: print("Attack Type - Counts:")
      print(df['Attack Type'].value_counts())

      print("\nAttack Type - Percentages:")
      print(round(df['Attack Type'].value_counts(normalize=True) * 100, 2))

      print("\nTarget Industry - Counts:")
      print(df['Target Industry'].value_counts())

      print("\nTarget Industry - Percentages:")
      print(round(df['Target Industry'].value_counts(normalize=True) * 100, 2))

      print("\nTop 10 Countries - Counts:")
      print(df['Country'].value_counts().head(10))

      print("\nTop 10 Countries - Percentages:")
      print(round(df['Country'].value_counts(normalize=True).head(10) * 100, 2))
```

```
Attack Type - Counts:
Attack Type
Ddos                531
Phishing            529
Sql Injection       503
Ransomware          493
Malware             485
Man-In-The-Middle   459
Name: count, dtype: int64

Attack Type - Percentages:
Attack Type
Ddos                17.70
Phishing            17.63
Sql Injection       16.77
Ransomware          16.43
Malware             16.17
Man-In-The-Middle   15.30
Name: proportion, dtype: float64
```

```
Target Industry - Counts:
Target Industry
It                  478
Banking             445
Healthcare          429
Retail              423
Education           419
Telecommunications  403
Government          403
Name: count, dtype: int64

Target Industry - Percentages:
Target Industry
It                  15.93
Banking             14.83
Healthcare          14.30
Retail              14.10
Education           13.97
Telecommunications  13.43
Government          13.43
Name: proportion, dtype: float64
```

```
Top 10 Countries - Counts:
Country
Uk          321
Brazil      310
India       308
France      305
Japan       305
Australia   297
Russia      295
Germany     291
Usa         287
China       281
Name: count, dtype: int64
```

```
Top 10 Countries - Percentages:
Country
Uk          10.70
Brazil      10.33
India       10.27
France      10.17
Japan       10.17
Australia    9.90
Russia       9.83
Germany      9.70
Usa          9.57
China        9.37
Name: proportion, dtype: float64
```

- Basic statistics for numeric features
- Identify the most common types of attacks and most targeted industries

# DATA VISUALISATIONS

01

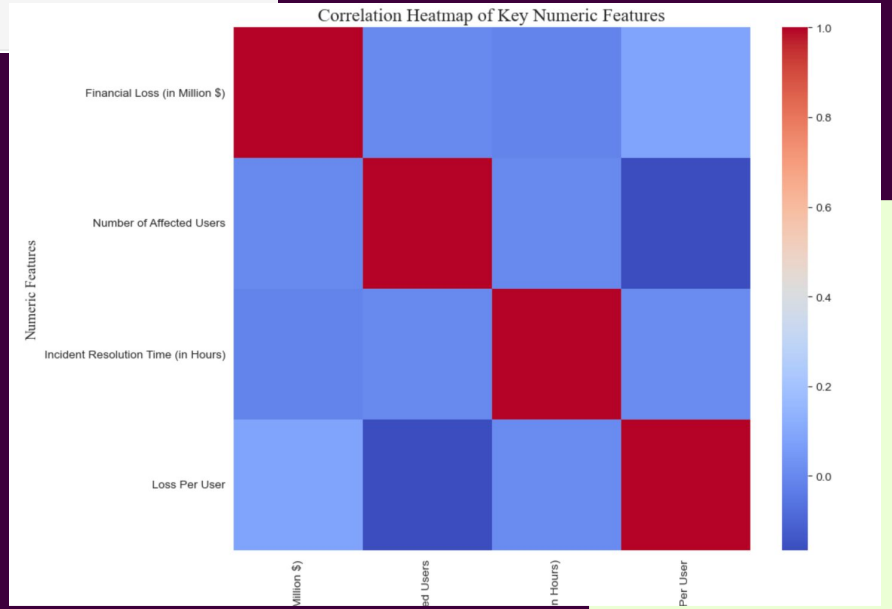
SECTION 3

```
]: import matplotlib.pyplot as plt
   import seaborn as sns

   sns.set_style("whitegrid")

   numeric_cols = ['Financial Loss (in Million $)', 'Number of Affected Users', 'Incident Resolution Time (in Hours)', 'Loss Per User']

   correlation_matrix = df[numeric_cols].corr()

   print(correlation_matrix)

   plt.figure(figsize=(10, 8))
   sns.heatmap(correlation_matrix, cmap='coolwarm', fmt=".2f")
   plt.title('Correlation Heatmap of Key Numeric Features', fontsize=14)
   plt.xlabel('Numeric Features', fontsize=12)
   plt.ylabel('Numeric Features', fontsize=12)

   plt.title('Correlation Heatmap of Key Numeric Features', fontsize=16, fontfamily='Times New Roman')
   plt.xlabel('Numeric Features', fontsize=12, fontfamily='Times New Roman')
   plt.ylabel('Numeric Features', fontsize=12, fontfamily='Times New Roman')

   plt.show()
```

# Heatmap

Most of the correlation is very low which means that there is not a strong linear relationship between variables

```
                                 Financial Loss (in Million $)  \
Financial Loss (in Million $)                         1.000000
Number of Affected Users                              0.001787
Incident Resolution Time (in Hours)                  -0.012671
Loss Per User                                         0.083258

                                 Number of Affected Users  \
Financial Loss (in Million $)                    0.001787
Number of Affected Users                         1.000000
Incident Resolution Time (in Hours)              0.005893
Loss Per User                                   -0.167098

                                 Incident Resolution Time (in Hours)
Financial Loss (in Million $)                              -0.012671
Number of Affected Users                                    0.005893
Incident Resolution Time (in Hours)                         1.000000
Loss Per User                                               0.008831

                                 Loss Per User
Financial Loss (in Million $)         0.083258
Number of Affected Users             -0.167098
Incident Resolution Time (in Hours)   0.008831
Loss Per User                         1.000000
```

# Top 10 Countries by Financial Loss

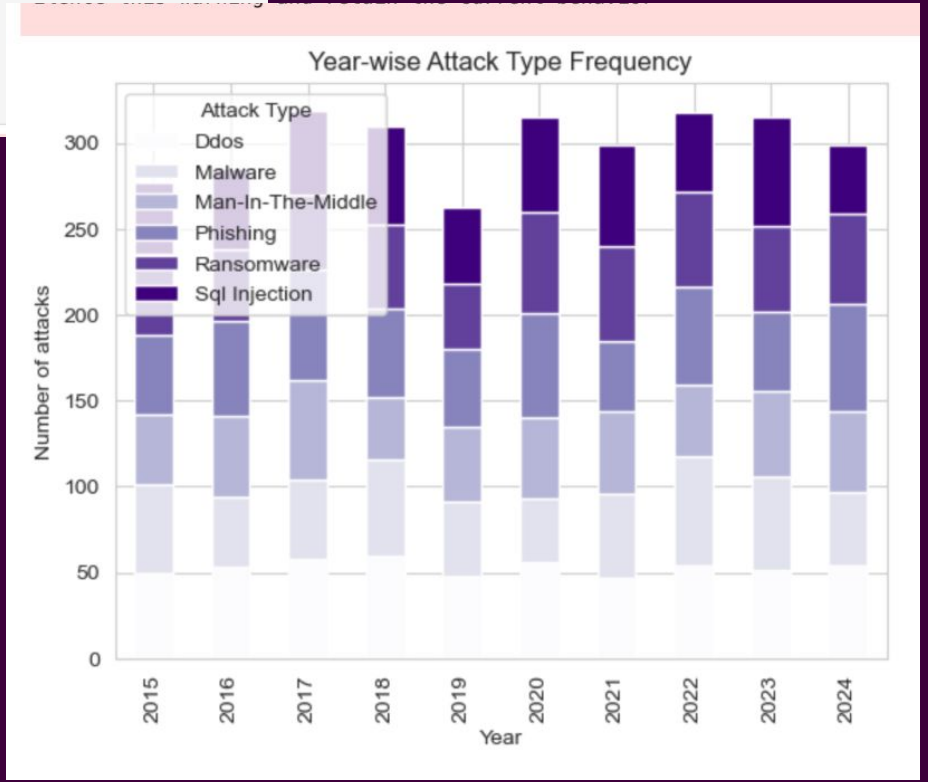
```
import matplotlib.pyplot as plt
import seaborn as sns

country_loss = df.groupby('Country')['Financial Loss (in Million $)'].sum()

top_10_countries = country_loss.sort_values(ascending=False)

print(top_10_countries)

plt.figure(figsize=(12, 7))
sns.barplot(x=top_10_countries.index, y=top_10_countries.values, palette='husl')
plt.title('Top 10 Countries by Total Financial Loss', fontsize=14)
plt.xlabel('Country', fontsize=12)
plt.ylabel('Total Financial Loss (Million $)', fontsize=12)
plt.show()
```

| Country | |
|---|---|
| Uk | 16502.99 |
| Germany | 15793.24 |
| Brazil | 15782.62 |
| Australia | 15403.00 |
| Japan | 15197.34 |
| France | 14972.28 |
| Usa | 14812.12 |
| Russia | 14734.73 |
| India | 14566.12 |
| China | 13714.47 |

Multiple Regions shown

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

pivot_table = df.pivot_table(index='Year', columns='Attack Type', aggfunc='size')

pivot_table.plot(kind='bar', stacked=True, colormap ='Purples')
plt.title('Year-wise Attack Type Frequency')
plt.xlabel('Year')
plt.ylabel('Number of attacks')
plt.show()
```

# Year-Wise Attack Type Frequency

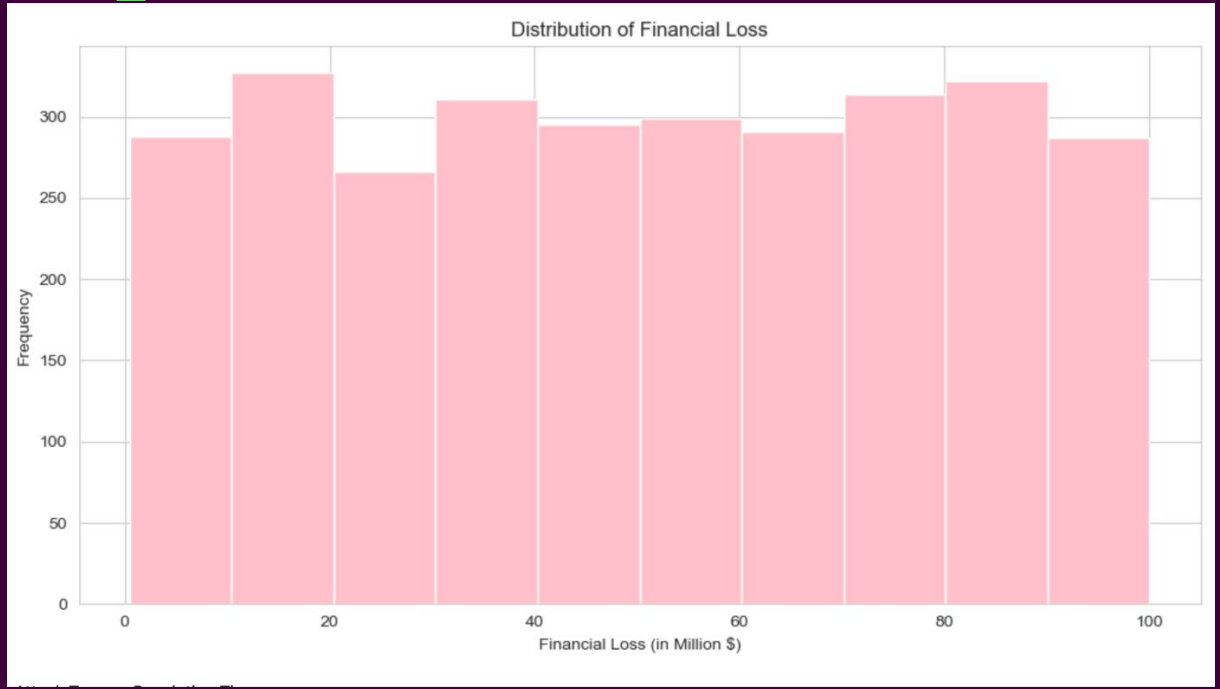



Very equally distributed, the number of attacks seems consistent

```python
import matplotlib.pyplot as plt

df['Financial Loss (in Million $)'].hist(figsize=(12,6), color='pink', edgecolor='white')
plt.title('Distribution of Financial Loss')
plt.xlabel('Financial Loss (in Million $)')
plt.ylabel('Frequency')
plt.show()
```

Wide attack severity and many different loss levels

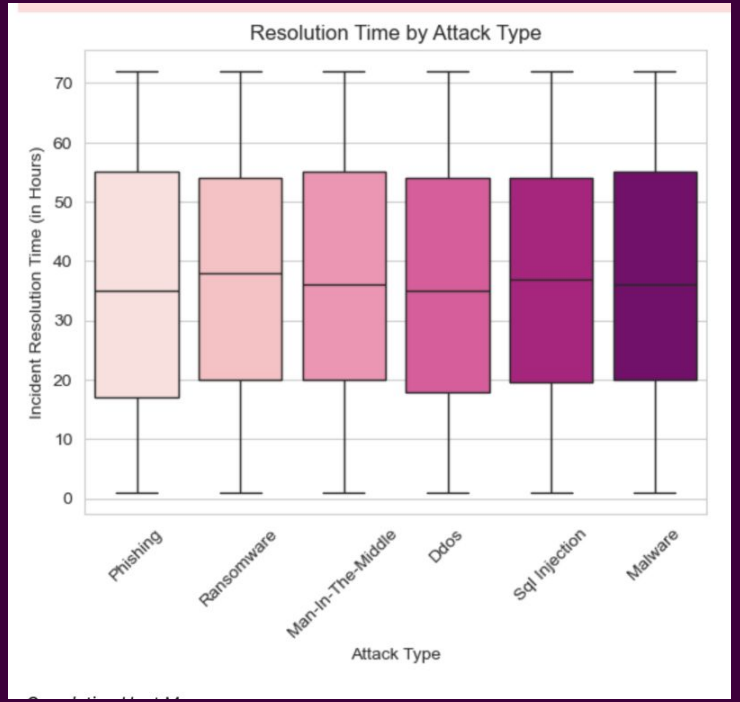
# Financial Loss Distribution



```
sns.boxplot(x='Attack Type', y='Incident Resolution Time (in Hours)', data=df, palette='RdPu')
plt.title('Resolution Time by Attack Type')
plt.xlabel('Attack Type')
plt.ylabel('Incident Resolution Time (in Hours)')
plt.xticks(rotation=45)
plt.show()
```

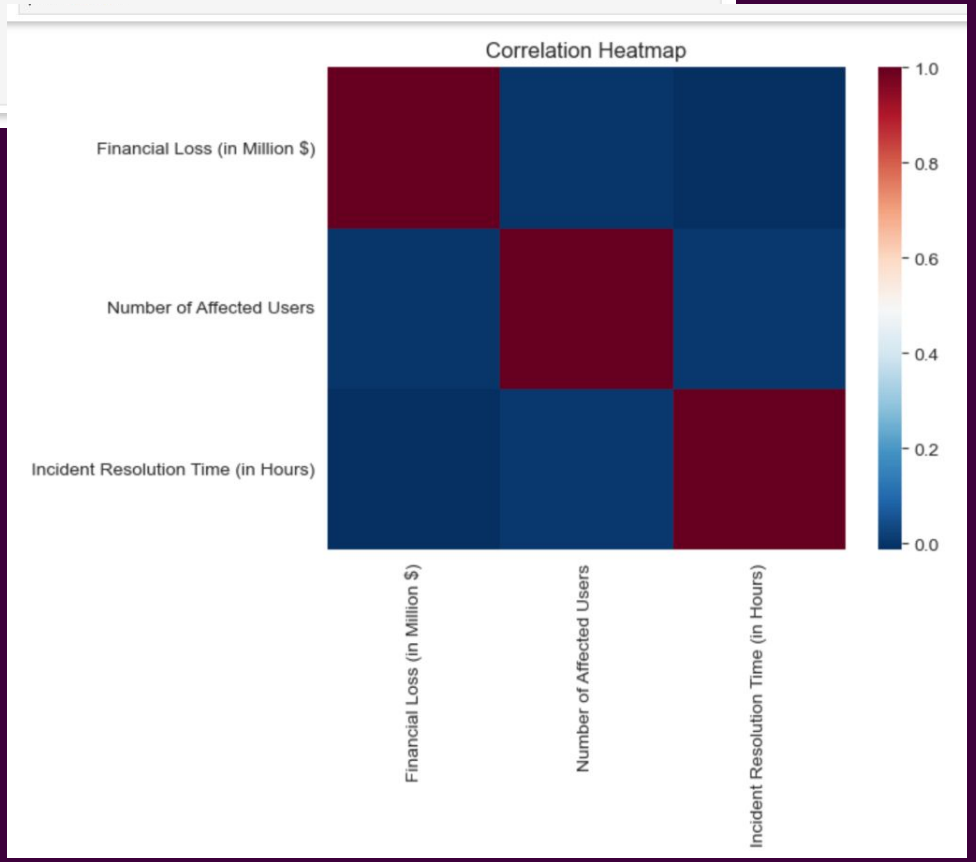
# Resolution Time by Attack Type

On average it takes about 35-40 mins to resolve most attacks



```
import matplotlib.pyplot as plt
import seaborn as sns

correlation_matrix = df[['Financial Loss (in Million $)', 'Number of Affected Users', 'Incident Resolution Time (in Hours)']].corr()

sns.heatmap(correlation_matrix, cmap='RdBu_r' )
plt.title('Correlation Heatmap')
plt.show()
```

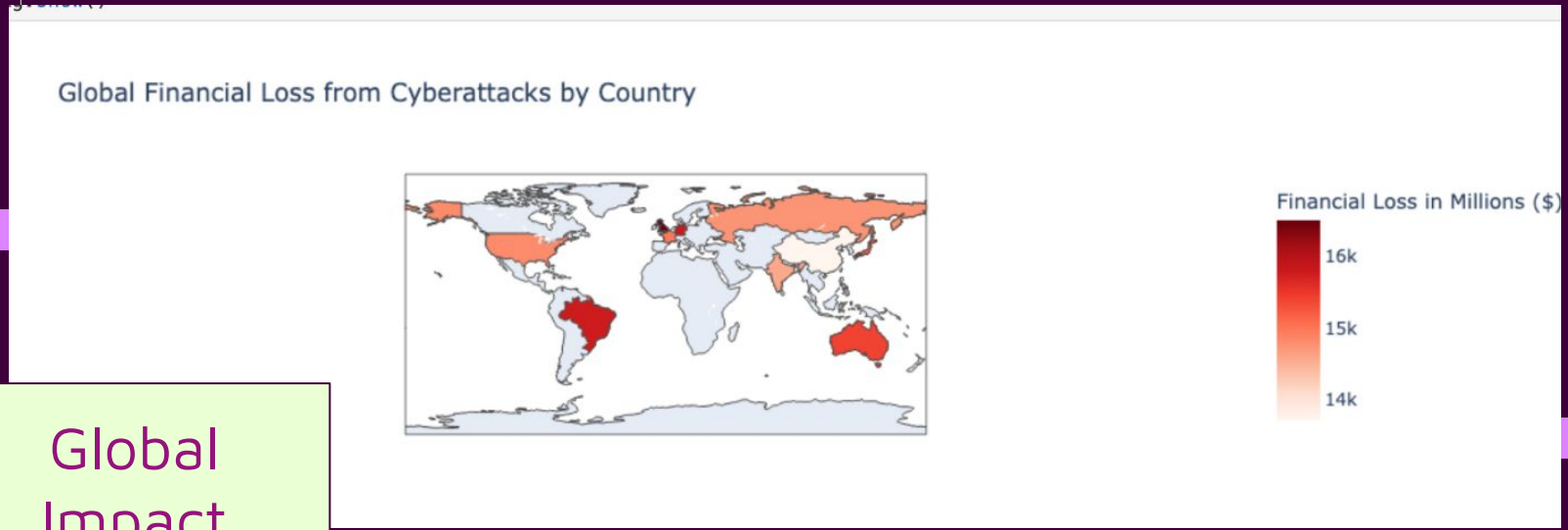
# Correlation Heat Map




No strong relationships shown

```python
import plotly.graph_objects as go

country_loss = df.groupby('Country')['Financial Loss (in Million $)'].sum().reset_index()

fig = go.Figure(go.Choropleth(
    locations=country_loss['Country'],
    locationmode='country names',
    z=country_loss['Financial Loss (in Million $)'],
    colorscale='Reds',
    colorbar_title='Financial Loss in Millions ($)'
))

fig.update_layout(title='Global Financial Loss from Cyberattacks by Country')
fig.show()
```

# Financial Loss by Country




# Global Impact

# INSIGHTS AND GENERALIZATIONS

1. Which countries and industries suffered the most?

- UK, Germany, Brazil, the US,
- IT, banking, healthcare, retail, education, telecom, and government

2. Which types of attacks caused the highest losses or affected the most users?

- Pretty equal distribution

3. Are some defense mechanisms associated with faster resolution?

- Pretty equal distribution

4. Any surprising patterns or outliers

- How equal distribution was

5. Limitations of the dataset (e.g., unreported incidents, missing values)

- Coverage and reporting bias

# Optional Questions

SECTION 4

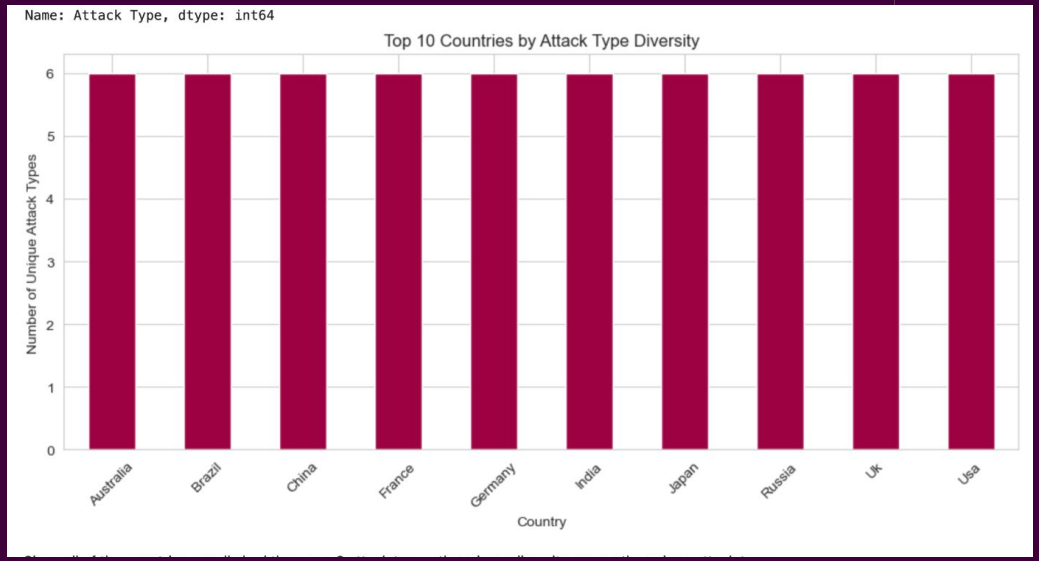
```python
import matplotlib.pyplot as plt

total_attack_diversity = df.groupby('Country')['Attack Type'].nunique()

top_10_diversity = total_attack_diversity.sort_values(ascending = False).head(10)

print(top_10_diversity)

top_10_diversity.plot(kind='bar', figsize= (12,5), colormap='Spectral')
plt.title('Top 10 Countries by Attack Type Diversity')
plt.xlabel('Country')
plt.ylabel('Number of Unique Attack Types')
plt.xticks(rotation=45)
plt.show()
```

```
Country
Australia    6
Brazil       6
China        6
France       6
Germany      6
India        6
Japan        6
Russia       6
Uk           6
Usa          6
Name: Attack Type, d
```

No strong relationships shown

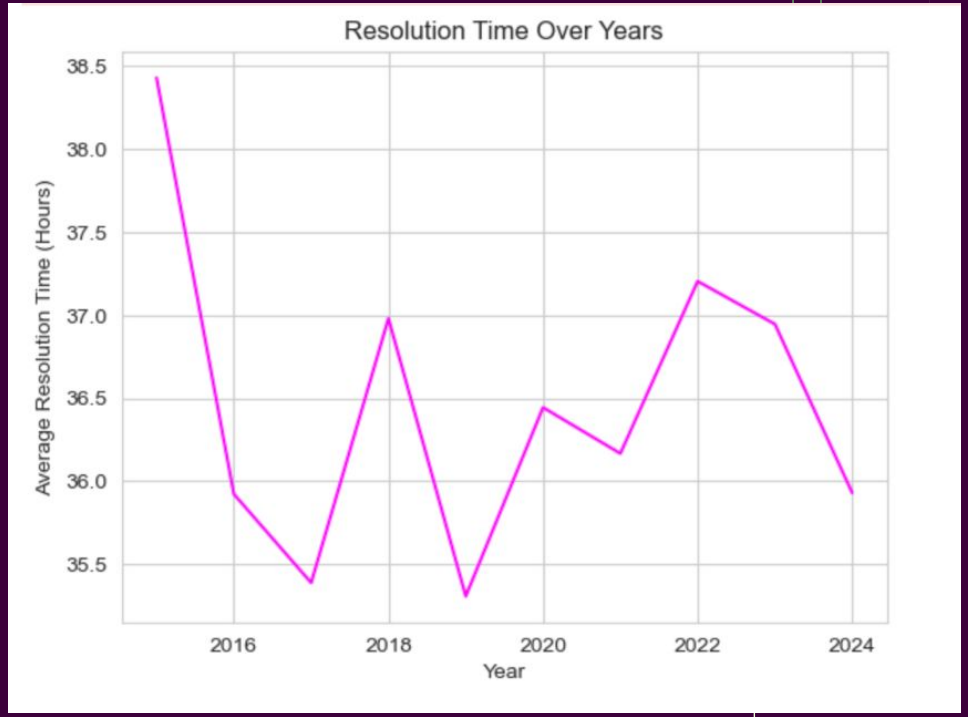
# Which countries experience the most diverse types of cyber attacks?

```
Name: Attack Type, dtype: int64
```



```
import matplotlib.pyplot as plt

yearly_average =df.groupby('Year')['Incident Resolution Time (in Hours)'].mean()
print(yearly_average)

plt.plot(yearly_average.index, yearly_average.values, color='magenta')
plt.title('Resolution Time Over Years')
plt.xlabel('Year')
plt.ylabel('Average Resolution Time (Hours)')
plt.show()
```

| Year | |
|---|---|
| 2015 | 38.429603 |
| 2016 | 35.922807 |
| 2017 | 35.388715 |
| 2018 | 36.980645 |
| 2019 | 35.307985 |
| 2020 | 36.444444 |
| 2021 | 36.167224 |
| 2022 | 37.204403 |
| 2023 | 36.946032 |
| 2024 | 35.929766 |




Is resolution time decreasing over the years?

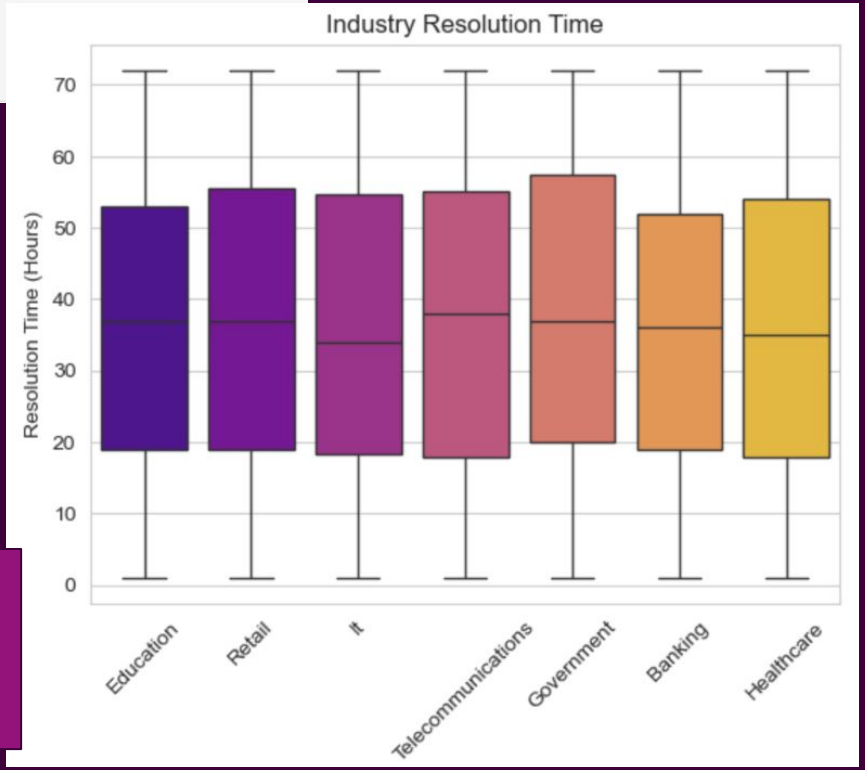
No !

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

industry_average = df.groupby('Target Industry')['Incident Resolution Time (in Hours)'].mean()
print(industry_average)

sns.boxplot(x='Target Industry', y='Incident Resolution Time (in Hours)', data=df, palette='plasma')
plt.xlabel('Industry')
plt.ylabel('Resolution Time (Hours)')
plt.xticks(rotation=45)
plt.show()
```

```
Target Industry
Banking                35.737079
Education              35.906921
Government             37.593052
Healthcare             35.806527
It                     36.169456
Retail                 37.219858
Telecommunications     37.062035
Name: Incident Resolution Time (in
```

# Do certain industries recover faster than others?

No !



# Thank you!