

```
In [1]: import pandas as pd
```

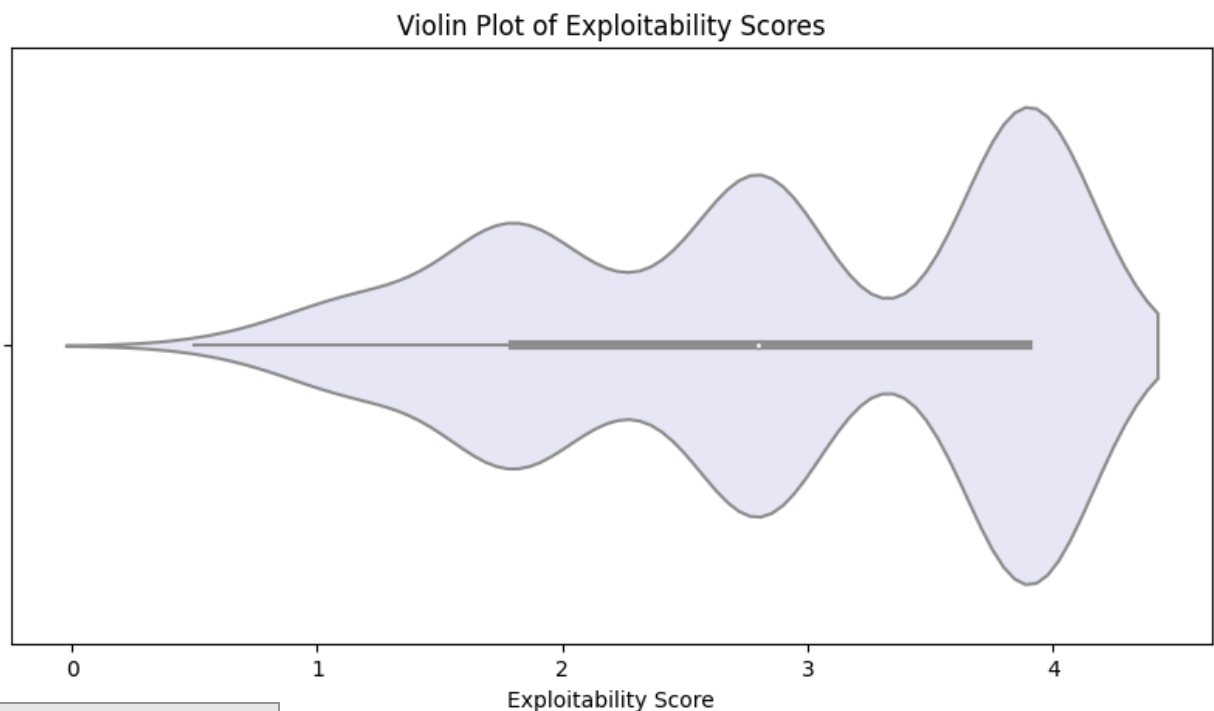
```
In [2]: import matplotlib.pyplot as plt
import seaborn as sns
```

```
In [3]: df=pd.read_pickle('C:/Ava/Raphael/vulnerability-prediction/2020_all_valid_cve_features.pkl')
df['exploited_in_wild'] = df[['clam_report', 'cisa_report', 'secureworks_report', 'graynoise_report', 'reference_hosts', 'vendor_employee_size_clusters', 'total_summary_word_count', 'clean_summary', 'rake_summary', 'vendors', 'products', 'vendor_graynoise_reports_count', 'vendor_graynoise_reports_count_cluster', 'open_source_products', 'open_source_vendors', 'open_source_vendors_products', 'exploited_in_wild']].astype(int)
wild_df=df[df['exploited_in_wild']==1]
non_wild_df=df[df['exploited_in_wild']!=1]
```

```
In [6]: wild_df.columns
```

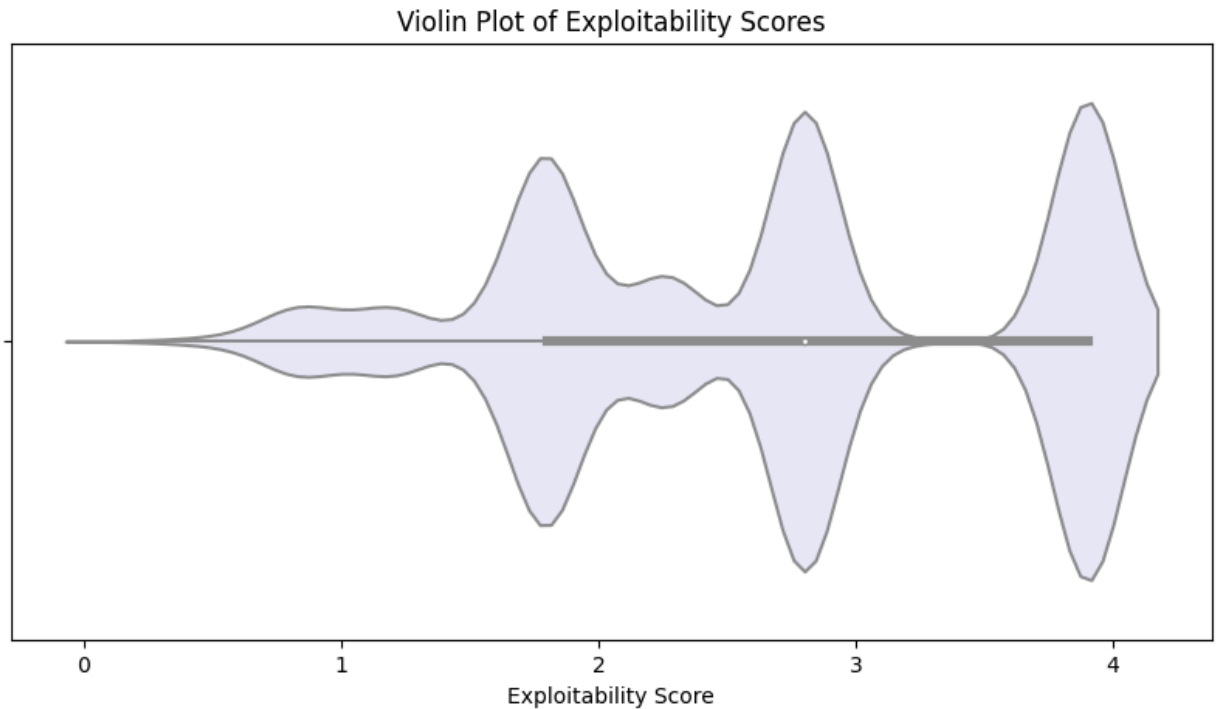
```
Out[6]: Index(['id', 'summary', 'reference_data', 'cpes', 'vector', 'attack_complexity', 'confidentiality_impact', 'integrity_impact', 'availability_impact', 'privileges_required', 'base_score', 'base_severity', 'exploitability_score', 'impact_score', 'cpes_logic', 'Source', 'lastModifiedDate', 'publishedDate', 'graynoise_report', 'cisa_report', 'clam_report', 'secureworks_report', 'reference_hosts', 'vendor_employee_size_clusters', 'total_summary_word_count', 'clean_summary', 'rake_summary', 'vendors', 'products', 'vendor_graynoise_reports_count', 'vendor_graynoise_reports_count_cluster', 'open_source_products', 'open_source_vendors', 'open_source_vendors_products', 'exploited_in_wild'],
dtype='object')
```

```
In [4]: plt.figure(figsize=(10, 5))
sns.violinplot(x=wild_df['exploitability_score'], color='lavender')
plt.xlabel('Exploitability Score')
plt.title('Violin Plot of Exploitability Scores')
plt.show()
```



File failed to load: /extensions/MathMenu.js

```
In [5]: plt.figure(figsize=(10, 5))
sns.violinplot(x=non_wild_df['exploitability_score'], color='lavender')
plt.xlabel('Exploitability Score')
plt.title('Violin Plot of Exploitability Scores')
plt.show()
```



```
In [23]: import matplotlib.pyplot as plt

# Value counts for 'privileges_required' in wild_df
value_counts_wild = wild_df['privileges_required'].value_counts()

# Value counts for 'privileges_required' in non_wild_df
value_counts_non_wild = non_wild_df['privileges_required'].value_counts()

# Define the colors and labels
colors = ['#ff9999', '#66b3ff', '#99ff99', '#ffcc99']
labels = value_counts_wild.index

# Create subplots for side-by-side pie charts
fig, axs = plt.subplots(1, 2, figsize=(12, 6))

# Pie chart for wild_df
wedges, texts, autotexts = axs[0].pie(value_counts_wild, autopct='%1.1f%%', startangle=90)
axs[0].set_title('Distribution of CVEs Privileged Required in CVEs Exploited in Wild')
axs[0].axis('equal') # Equal aspect ratio ensures the pie chart is circular.

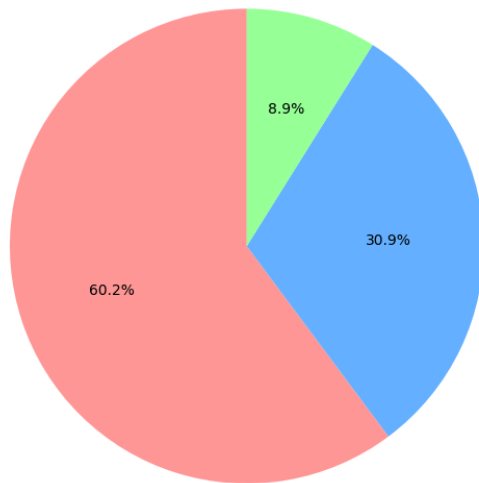
# Pie chart for non_wild_df
wedges2, texts2, autotexts2 = axs[1].pie(value_counts_non_wild, autopct='%1.1f%%', startangle=90)
axs[1].set_title('Distribution of CVEs Privileged Required in CVEs Not Exploited in Wild')
axs[1].axis('equal') # Equal aspect ratio ensures the pie chart is circular.

# Add Legend
fig.legend(wedges, labels, loc=(0.90, 0.75))
```

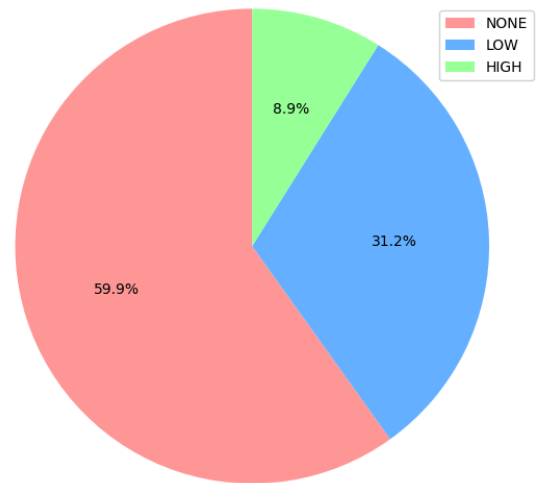
File failed to load: /extensions/MathMenu.js

```
plt.tight_layout()
plt.show()
```

Distribution of CVEs Privileged Required in CVEs Exploited in Wild



Distribution of CVEs Privileged Required in CVEs Not Exploited in Wild



```
In [25]: # Value counts for 'attack_complexity' in wild_df
value_counts_wild = wild_df['attack_complexity'].value_counts()

# Value counts for 'attack_complexity' in non_wild_df
value_counts_non_wild = non_wild_df['attack_complexity'].value_counts()

# Define the colors and labels
colors = ['#ff9999', '#66b3ff']
labels = value_counts_wild.index

# Create subplots for side-by-side pie charts
fig, axs = plt.subplots(1, 2, figsize=(12, 6))

# Pie chart for wild_df
axs[0].pie(value_counts_wild, autopct='%1.1f%%', startangle=90, colors=colors)
axs[0].set_title('Distribution of CVEs Attack Complexity in CVEs Exploited in Wild')
axs[0].axis('equal') # Equal aspect ratio ensures the pie chart is circular.

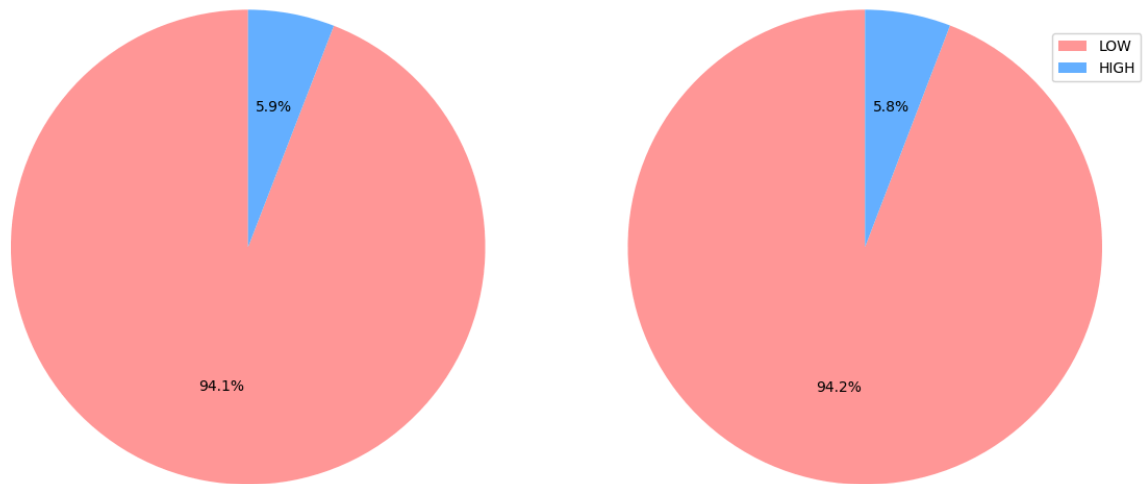
# Pie chart for non_wild_df
axs[1].pie(value_counts_non_wild, autopct='%1.1f%%', startangle=90, colors=colors)
axs[1].set_title('Distribution of CVEs Attack Complexity in CVEs not Exploited in Wild')
axs[1].axis('equal') # Equal aspect ratio ensures the pie chart is circular.

fig.legend(wedges, labels, loc=(0.90, 0.75))

# Show the plot
plt.tight_layout()
plt.show()
```

Distribution of CVEs Attack Complexity in CVEs Exploited in Wild

Distribution of CVEs Attack Complexity in CVEs not Exploited in Wild)



```
In [26]: # Value counts for 'base_severity' in wild_df
value_counts_wild = wild_df['base_severity'].value_counts()

# Value counts for 'base_severity' in non_wild_df
value_counts_non_wild = non_wild_df['base_severity'].value_counts()

# Define the colors and labels
colors = ['#ff9999', '#66b3ff', '#99ff99', '#ffcc99']
labels = value_counts_wild.index

# Create subplots for side-by-side pie charts
fig, axs = plt.subplots(1, 2, figsize=(12, 6))

# Pie chart for wild_df
axs[0].pie(value_counts_wild, autopct='%1.1f%%', startangle=90, colors=colors)
axs[0].set_title('Distribution of CVEs Base Severity in CVEs Exploited in Wild')
axs[0].axis('equal') # Equal aspect ratio ensures the pie chart is circular.

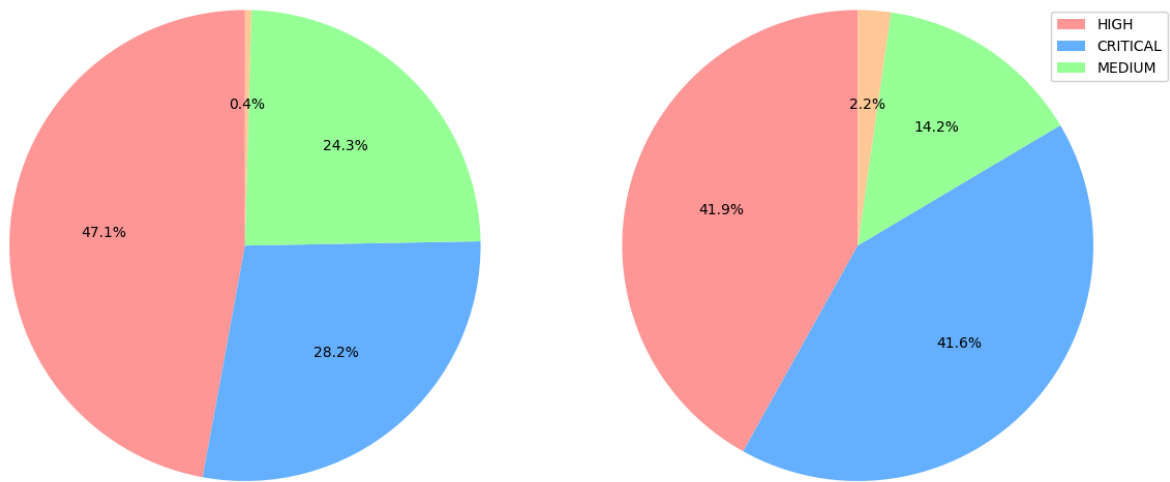
# Pie chart for non_wild_df
axs[1].pie(value_counts_non_wild, autopct='%1.1f%%', startangle=90, colors=colors)
axs[1].set_title('Distribution of CVEs Base Severity in CVEs not Exploited in Wild')
axs[1].axis('equal') # Equal aspect ratio ensures the pie chart is circular.

fig.legend(wedges, labels, loc=(0.90, 0.75))

# Show the plot
plt.tight_layout()
plt.show()
```

Distribution of CVEs Base Severity in CVEs Exploited in Wild

Distribution of CVEs Base Severity in CVEs not Exploited in Wild



In []: