Nama Dosen : Teguh Iman Hermanto, M.Kom
Mata Kuliah : Machine Learning 1
Pembahasan : Density-Based Spatial Clustering of Applications with Noise (DBSCAN)
Pokok Pemb : - Membangun Model DBSCAN
- Simulasi Algoritma DBSCAN
- Profiling Hasil Clustering DBSCAN

1. Load Library dan dataset pada file Notebook

```python
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import DBSCAN
import matplotlib.pyplot as plt
import seaborn as sns
```

```python
df = pd.read_csv('net_detection.csv')
```

```python
df.head()
```

0.0s Python

	src_ip	dst_ip	protocol	src_port	dst_port	bytes_sent	bytes_received	duration	packet_count	attack_type
0	39.170.115.188	133.204.219.238	TCP	1762	62458	3422	5989	213	131	normal
1	80.35.125.105	246.113.106.207	TCP	32718	9699	3736	989	277	96	normal
2	49.134.137.30	151.26.62.67	TCP	1225	43970	2865	5943	305	89	ddos
3	157.51.229.193	175.153.3.55	TCP	20804	303	1852	9389	552	80	normal
4	121.123.112.174	72.234.63.118	UDP	15457	17942	8318	5160	533	46	normal

2. **Exploratory data analysis**
   a. **Menghitung jumlah data yang ada pada kolom protocol**

```
# menghitung jumlah data yang ada pada kolom protocol
protocol_counts = df['protocol'].value_counts()
protocol_counts
```

```
# membuat bar plot hasil perhitungan kolom protocol
plt.figure(figsize=(10, 6))
protocol_counts.plot(kind='bar')
plt.title('Number of Data Points per Protocol')
plt.xlabel('Protocol')
plt.ylabel('Count')
plt.show()
```

Lakukan perhitungan untuk mendapatkan jumlah pada kolom “attack_type” dan buatkan bar plot nya

   b. **Membuat box plot untuk menampilkan rata-rata nilai source plot berdasarkan jenis attack type**

```
plt.figure(figsize=(12, 6))
sns.boxplot(x='attack_type', y='src_port', data=df)
plt.title('Source Port Distribution by Attack Type')
plt.xlabel('Attack Type')
plt.ylabel('Source Port')
plt.show()
```

Buatkan boxplot untuk menampilkan bytes_sent dan berdasarkan senis protocol

3. Buat model DBSCAN

```
# pilih fitur untuk clustering
features = ['src_port','dst_port','bytes_sent','bytes_received']
X = df[features]
```

```
# Standardize the data
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

```
# implementasi algoritma DBSCAN
# tentukan nilai epsilon dan min sample
dbscan = DBSCAN(eps=0.5, min_samples=5)
df['dbscan_cluster'] = dbscan.fit_predict(X_scaled)
```

```
# Plot hasil cluster
plt.figure(figsize=(10, 6))
sns.scatterplot(x='bytes_sent', y='bytes_received',
                hue='dbscan_cluster', data=df,
                palette='viridis')
plt.title('DBSCAN Clustering Results')
plt.show()
```

```
df.head()
```
✓ 0.0s

	src_ip	dst_ip	protocol	src_port	dst_port	bytes_sent	bytes_received	duration	packet_count	attack_type	dbscan_cluster
0	39.170.115.188	133.204.219.238	TCP	1762	62458	3422	5989	213	131	normal	-1
1	80.35.125.105	246.113.106.207	TCP	32718	9699	3736	989	277	96	normal	0
2	49.134.137.30	151.26.62.67	TCP	1225	43970	2865	5943	305	89	ddos	-1
3	157.51.229.193	175.153.3.55	TCP	20804	303	1852	9389	552	80	normal	-1
4	121.123.112.174	72.234.63.118	UDP	15457	17942	8318	5160	533	46	normal	0

```python
# memisahkan data anomaly yang bernilai -1
df['anomaly'] = df['dbscan_cluster'] == -1
```

```python
# Plot hasil deteksi anomaly
plt.figure(figsize=(10, 6))
sns.scatterplot(x='bytes_sent', y='bytes_received',
                hue='anomaly', data=df,
                palette=['blue', 'red'])
plt.title('Anomaly Detection')
plt.show()
```

```
df.head()
```

✓ 0.0s Python

	src_ip	dst_ip	protocol	src_port	dst_port	bytes_sent	bytes_received	duration	packet_count	attack_type	dbscan_cluster	anomaly
0	39.170.115.188	133.204.219.238	TCP	1762	62458	3422	5989	213	131	normal	-1	True
1	80.35.125.105	246.113.106.207	TCP	32718	9699	3736	989	277	96	normal	0	False
2	49.134.137.30	151.26.62.67	TCP	1225	43970	2865	5943	305	89	ddos	-1	True
3	157.51.229.193	175.153.3.55	TCP	20804	303	1852	9389	552	80	normal	-1	True
4	121.123.112.174	72.234.63.118	UDP	15457	17942	8318	5160	533	46	normal	0	False

```python
# simpan hasil cluster
df.to_csv('hasil_dbscan.csv', index=False)
```

4. Buatkan profiling hasil cluster

```
# membuat df untuk type serangan ddos
ddos_df = df[df['attack_type'] == 'ddos']
```

```
# hitung jumlah anomalynya
anomaly_counts = ddos_df['anomaly'].value_counts()
anomaly_counts
```

```
# bar plot untuk serangan DDOS
plt.figure(figsize=(8, 6))
anomaly_counts.plot(kind='bar')
plt.title('Jumlah Anomaly untu tipe DDoS')
plt.xlabel('Anomaly')
plt.ylabel('Count')
plt.show()
```

Lakukan analisis untuk deteksi anomaly pada jenis serangan lainnya seperti brute force, icmp flood, dan port scan.