Aim: Outlier detection using distance based/density based method.

Theory:

What is an Outlier?

An outlier is a data point that significantly deviates from the overall pattern of a dataset. Outliers can distort statistical analyses and machine learning models, leading to incorrect conclusions. They can be caused by measurement errors, data entry mistakes, natural variability, or anomalies in the system being analyzed.

---

1. Z-Score (Distance-Based Method)

The **Z-score method** detects outliers by measuring how far a data point is from the mean in terms of standard deviations.

Formula:

$$Z = \frac{X - \mu}{\sigma}$$

- X is the data point,
- μ is the mean,
- σ is the standard deviation.

Interpretation:

- If the Z-score of a point is **greater than a threshold** (e.g., 3 or -3), it is considered an outlier.
- Z-score works best for normally distributed data but may fail for skewed datasets.

---

2. IQR (Interquartile Range - Statistical Method)

The **Interquartile Range (IQR) method** identifies outliers by analyzing the spread of the middle 50% of the data.

Formula:

$$IQR = Q3 - Q1$$

- $Q1$ (First Quartile) = 25th percentile,
- $Q3$ (Third Quartile) = 75th percentile.

Outlier Detection:

- Any data point outside the range: $[Q1 - 1.5 \times IQR, Q3 + 1.5 \times IQR]$ is considered an outlier.
- The IQR method is robust to skewed distributions and extreme values.

3. DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

DBSCAN is a **density-based clustering algorithm** that identifies outliers as points that do not belong to any cluster.

How It Works:

- Defines **core points** (high-density regions).
- **Border points** are near core points but do not meet density requirements.
- **Noise points (outliers)** are neither core nor border points.

Key Parameters:

- **eps (ε):** The radius within which points are considered neighbors.
- **min_samples:** Minimum number of points required to form a dense region.

Advantages:

- Works well with **non-linearly separable data**.
- Does **not require specifying the number of clusters** like k-means.
- Robust to noise and varying cluster densities.

Limitations:

- Sensitive to **eps and min_samples** values.
- Struggles with datasets having **varying densities**.

Comparison of Outlier Detection Techniques

Method	Strengths	Weaknesses
Z-Score	Simple, effective for normal distributions	Fails for skewed or non-Gaussian data
IQR	Robust to skewed data and extreme values	Not effective for small datasets
DBSCAN	Works with clusters of arbitrary shape, robust to noise	Sensitive to parameter selection

Implementation:

```
!pip install numpy pandas scikit-learn matplotlib seaborn
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import LocalOutlierFactor
```

```
from sklearn.cluster import DBSCAN
# Load dataset
file_path = "/content/Accumulative_distribution.csv"
data = pd.read_csv(file_path)

# Display first few rows
print(data.head())
```

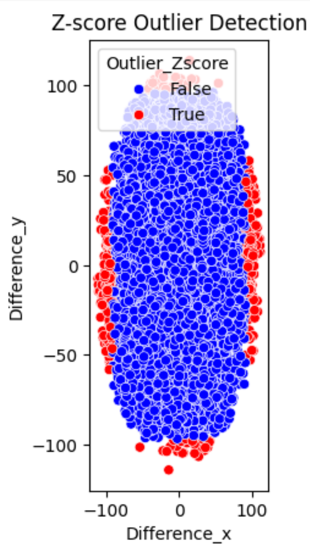
	ID	Type	Movie_number	Fly_number	Other_number	Difference_x	Difference_y	Distance
0	1	Dmelanogaster	1	1	0	-9.713147	4.991989	10.920860
1	2	Dmelanogaster	1	2	0	-23.854249	-16.237289	28.856104
2	3	Dmelanogaster	1	3	0	-59.388140	63.130531	86.674189
3	4	Dmelanogaster	1	4	0	-62.456371	-37.194611	72.692760
4	5	Dmelanogaster	1	5	0	-9.644838	9.009750	13.198428

```
num_cols = ['Difference_x', 'Difference_y', 'Distance']
scaler = StandardScaler()
data_scaled = scaler.fit_transform(data[num_cols])
plt.figure(figsize=(18, 6))
```

```
<Figure size 1800x600 with 0 Axes>
<Figure size 1800x600 with 0 Axes>
```

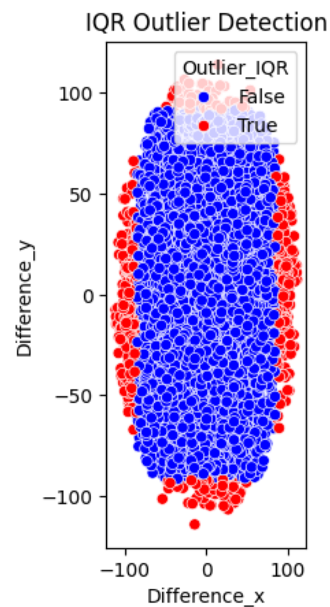
Z-Score

```
z_scores = np.abs((data[num_cols] -
data[num_cols].mean()) / data[num_cols].std())
threshold = 2.5  # Common threshold
outliers_z = (z_scores > threshold).any(axis=1)
data['Outlier_Zscore'] = outliers_z
plt.subplot(1, 3, 1)
sns.scatterplot(x=data['Difference_x'],
y=data['Difference_y'], hue=data['Outlier_Zscore'],
palette={True: 'red', False: 'blue'})
plt.title("Z-score Outlier Detection")
```



IQR

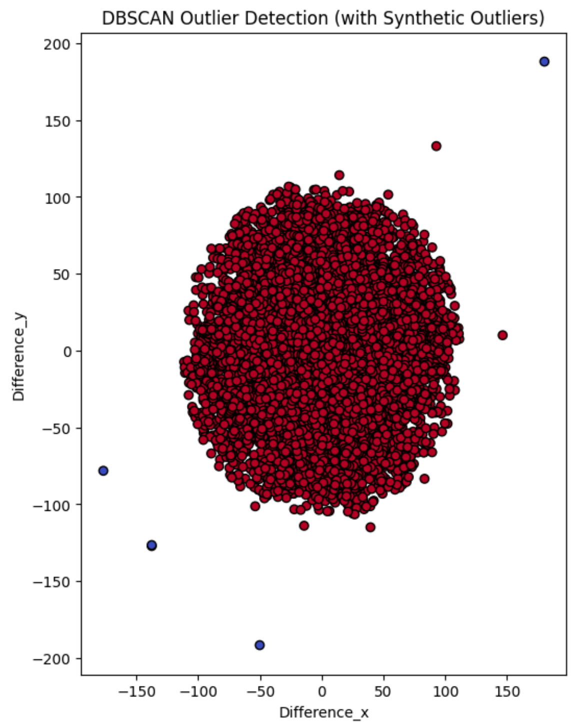
```
Q1 = data[num_cols].quantile(0.25)
Q3 = data[num_cols].quantile(0.75)
IQR = Q3 - Q1
outliers_iqr = ((data[num_cols] < (Q1 - 2 * IQR)) |
(data[num_cols] > (Q3 + 2 * IQR))).any(axis=1)
data['Outlier_IQR'] = outliers_iqr
plt.subplot(1, 3, 2)
sns.scatterplot(x=data['Difference_x'],
y=data['Difference_y'], hue=data['Outlier_IQR'],
palette={True: 'red', False: 'blue'})
plt.title("IQR Outlier Detection")
```



DBSCAN

```
dbscan = DBSCAN(eps=1.5, min_samples=3)
df_extended['Outlier_DBSCAN'] =
dbscan.fit_predict(scaled_data)
plt.figure(figsize=(6, 8))
```

```
plt.scatter(df_extended['Difference_x'],
df_extended['Difference_y'],
            c=df_extended['Outlier_DBSCAN'],
cmap='coolwarm', edgecolors='k')
plt.title("DBSCAN Outlier Detection (with Synthetic
Outliers)")
plt.xlabel("Difference_x")
plt.ylabel("Difference_y")
plt.show()
```



Conclusion:

Outlier detection helps improve data quality and model accuracy. Z-score works well for normal data, IQR handles skewed data, and DBSCAN detects anomalies in dense datasets. Choosing the right method depends on the data characteristics.