```
import pandas as pd, numpy as np, matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.neighbors import NearestNeighbors
from sklearn.cluster import DBSCAN
```

```
PATH = "/mnt/data/Weather_Station_dataset.csv"
try:
    df = pd.read_csv(PATH)
except Exception:
    from google.colab import files
    print("Upload Weather_Station_dataset.csv when prompted")
    uploaded = files.upload()
    df = pd.read_csv(next(iter(uploaded.keys())))

print("Raw shape:", df.shape)
display(df.head())
```

```
Upload Weather_Station_dataset.csv when prompted
Choose Files Weather_St...dataset.csv
Weather_Station_dataset.csv(text/csv) - 3028 bytes, last modified: 11/9/2025 - 100% done
Saving Weather_Station_dataset.csv to Weather_Station_dataset.csv
Raw shape: (91, 7)
```

	Station_ID	Location	Temperature_C	Humidity_%	Wind_Speed_kmph	Rainfall_mm	Weather_Condition
0	100	Delhi	30.5	55	16	6	Sunny
1	101	Mumbai	30.1	82	11	22	Rainy
2	102	Chennai	31.8	68	18	9	Sunny
3	103	Kolkata	29.4	74	14	15	Cloudy
4	104	Bengaluru	25.2	59	20	5	Sunny

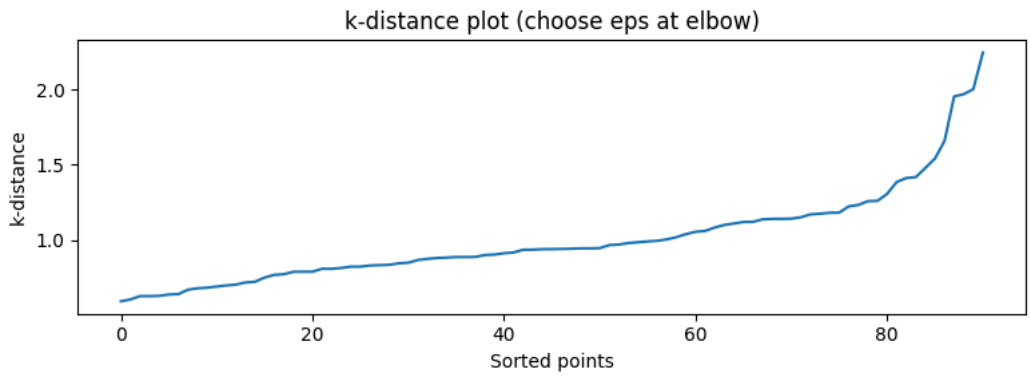
```
df = df.dropna(axis=1, how="all").drop_duplicates().reset_index(drop=True)
possible_id = [c for c in df.columns if df[c].nunique() == len(df)]
df = df.drop(columns=possible_id, errors='ignore')
```

```
num = df.select_dtypes(include=[np.number]).copy()
if num.shape[1] == 0:
    raise ValueError("No numeric columns found — please include numeric features (temp/humidity/etc).")
print("Using numeric features:", list(num.columns))
```

```
Using numeric features: ['Temperature_C', 'Humidity_%', 'Wind_Speed_kmph', 'Rainfall_mm']
```

```
num = num.fillna(num.median())

X = StandardScaler().fit_transform(num)
```

```
min_samples = max(5, int(np.sqrt(len(X))))    # rule-of-thumb
nbrs = NearestNeighbors(n_neighbors=min_samples).fit(X)
distances, _ = nbrs.kneighbors(X)
kdist = np.sort(distances[:, -1])
plt.figure(figsize=(8,3)); plt.plot(kdist); plt.title("k-distance plot (choose eps at elbow)"); plt.ylabel("k-distance"); p
```

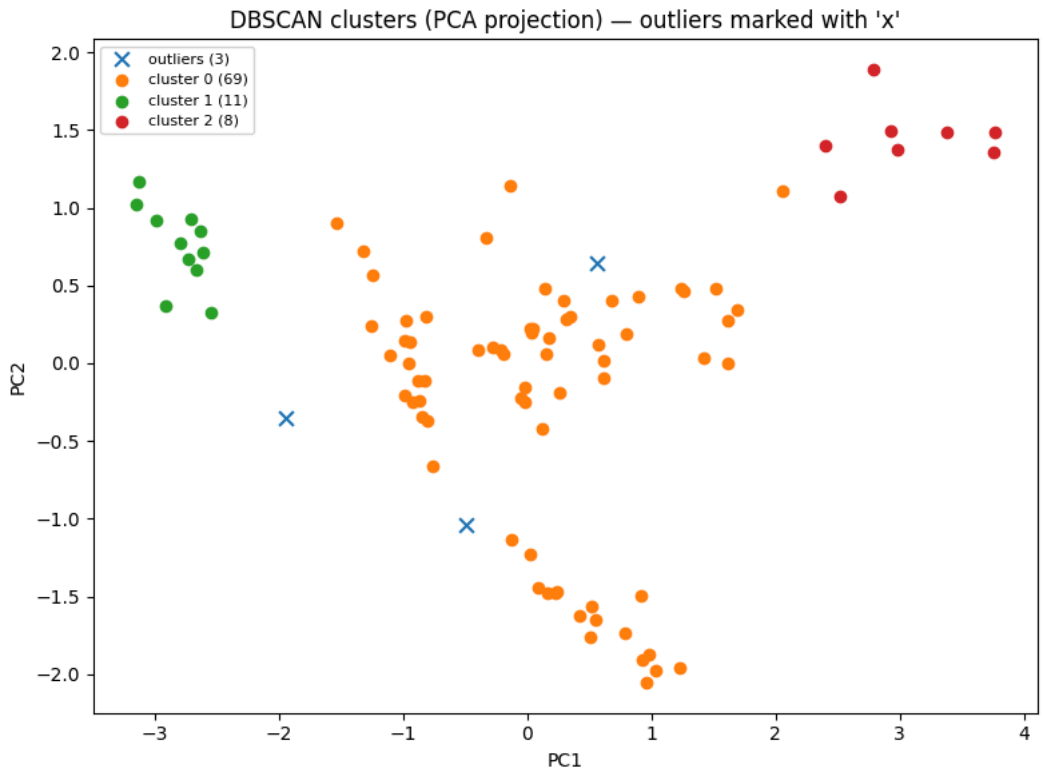



```
eps_val = float(np.percentile(kdist, 90)) * 0.8   # automatic heuristic (you can override)
print(f"min_samples = {min_samples}, eps (auto heuristic) = {eps_val:.3f}")

db = DBSCAN(eps=eps_val, min_samples=min_samples, metric="euclidean").fit(X)
labels = db.labels_
```

```
n_clusters = len(set(labels)) - (1 if -1 in labels else 0)
n_outliers = list(labels).count(-1)
print(f"DBSCAN found {n_clusters} clusters and {n_outliers} outliers (label = -1)")
```

```
min_samples = 9, eps (auto heuristic) = 1.109
DBSCAN found 3 clusters and 3 outliers (label = -1)
```

```
pca = PCA(n_components=2, random_state=0)
X2 = pca.fit_transform(X)

plt.figure(figsize=(8,6))
unique_labels = sorted(set(labels))
# plot clusters
for lab in unique_labels:
    mask = labels == lab
    if lab == -1:
        plt.scatter(X2[mask,0], X2[mask,1], marker='x', s=60, label=f"outliers ({mask.sum()})")
    else:
        plt.scatter(X2[mask,0], X2[mask,1], label=f"cluster {lab} ({mask.sum()})")
plt.title("DBSCAN clusters (PCA projection) — outliers marked with 'x'")
plt.xlabel("PC1"); plt.ylabel("PC2"); plt.legend(loc="best", fontsize=8); plt.tight_layout(); plt.show()
```




```
outlier_idx = np.where(labels == -1)[0].tolist()
print("Outlier indices (row numbers):", outlier_idx)
if outlier_idx:
    display(df.iloc[outlier_idx])
```

```
Outlier indices (row numbers): [4, 10, 23]
```

	Location	Temperature_C	Humidity_%	Wind_Speed_kmph	Rainfall_mm	Weather_Condition
4	Bengaluru	25.2	59	20	5	Sunny
10	Delhi	36.3	42	9	1	Sunny
23	Kolkata	28.7	73	19	13	Cloudy

```
df_out = df.copy()
df_out["DBSCAN_label"] = labels
df_out.to_csv("weather_dbscan_labeled.csv", index=False)
print("Saved labeled dataset as weather_dbscan_labeled.csv (download from Colab Files).")
```

```
Saved labeled dataset as weather_dbscan_labeled.csv (download from Colab Files).
```