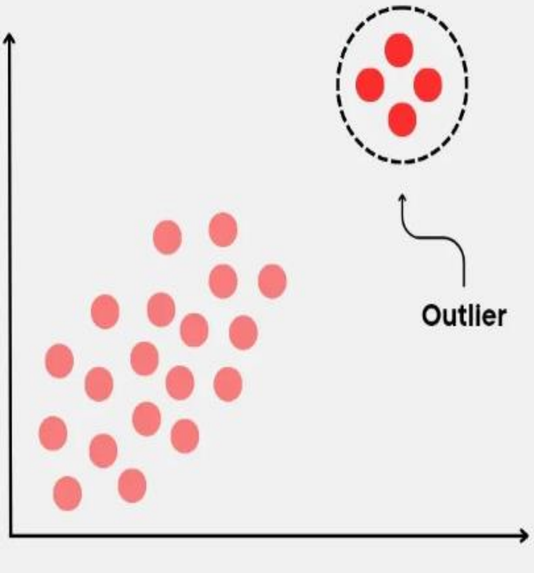
Techniques to **Detect Outliers**

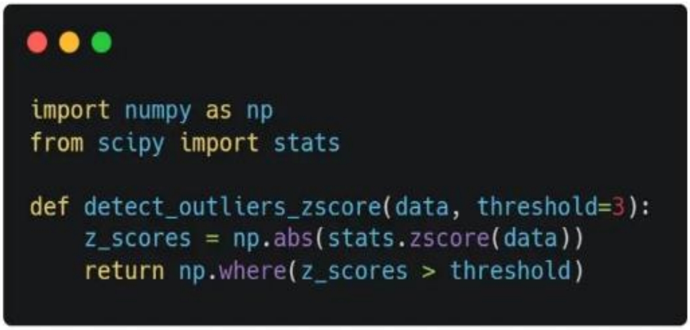
In Data Science (With Code)



Swipe
to see in
detail

Z-Score Method

The Z-Score measures how far away a particular data point is from the mean of the data in terms of standard deviations.

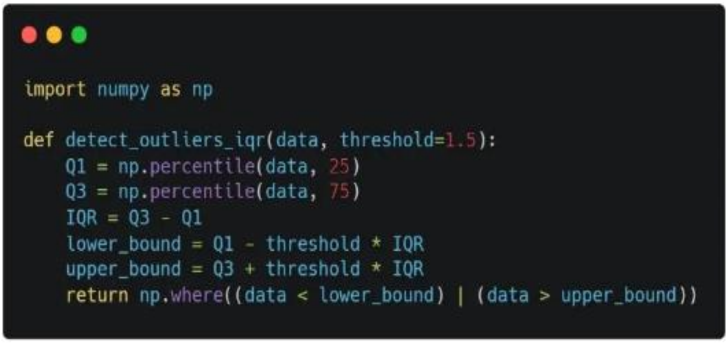


```python
import numpy as np
from scipy import stats

def detect_outliers_zscore(data, threshold=3):
    z_scores = np.abs(stats.zscore(data))
    return np.where(z_scores > threshold)
```

IQR (Interquartile Range) Method

The IQR is the range between the first quartile (Q1) and the third quartile (Q3) of the data. Outliers are identified based on this range.

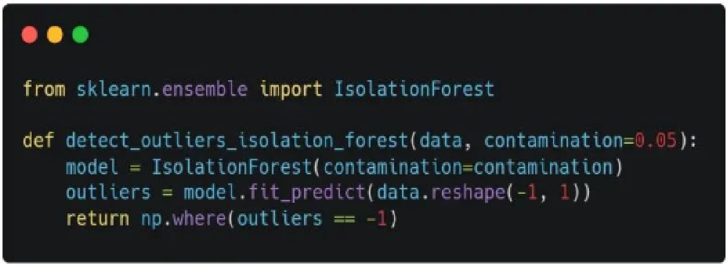


```python
import numpy as np

def detect_outliers_iqr(data, threshold=1.5):
    Q1 = np.percentile(data, 25)
    Q3 = np.percentile(data, 75)
    IQR = Q3 - Q1
    lower_bound = Q1 - threshold * IQR
    upper_bound = Q3 + threshold * IQR
    return np.where((data < lower_bound) | (data > upper_bound))
```

Isolation Forest

Isolation Forest isolates outliers by randomly selecting a feature and then randomly selecting a split value between the maximum and minimum values of the selected feature.

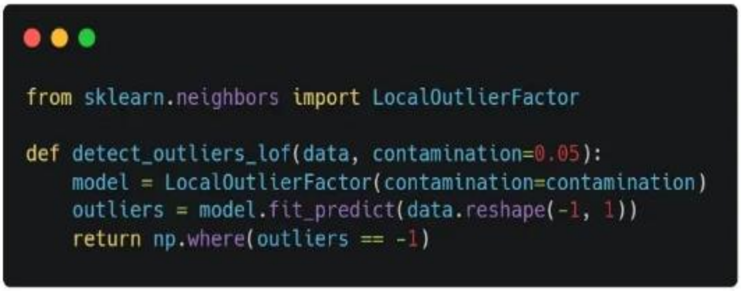


```python
from sklearn.ensemble import IsolationForest

def detect_outliers_isolation_forest(data, contamination=0.05):
    model = IsolationForest(contamination=contamination)
    outliers = model.fit_predict(data.reshape(-1, 1))
    return np.where(outliers == -1)
```

Local Outlier Factor (LOF)

LOF is a density-based method that compares the density of data points around an instance to the density of its neighbors to identify outliers.

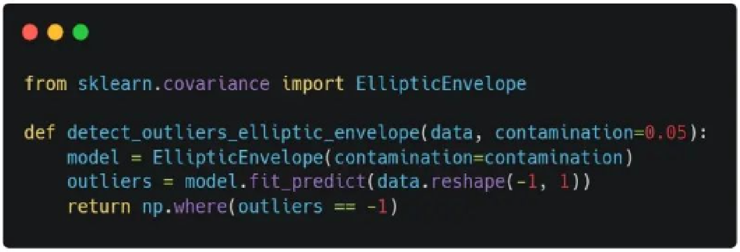


```python
from sklearn.neighbors import LocalOutlierFactor

def detect_outliers_lof(data, contamination=0.05):
    model = LocalOutlierFactor(contamination=contamination)
    outliers = model.fit_predict(data.reshape(-1, 1))
    return np.where(outliers == -1)
```

Elliptic Envelope

Elliptic Envelope fits a robust covariance estimate to the data, considering the presence of outliers. It then classifies points as outliers based on their Mahalanobis distance.

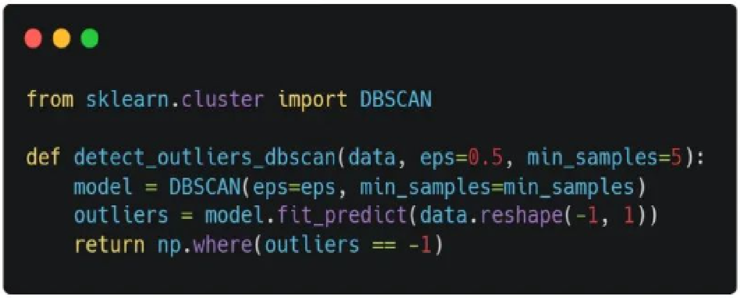


```python
from sklearn.covariance import EllipticEnvelope

def detect_outliers_elliptic_envelope(data, contamination=0.05):
    model = EllipticEnvelope(contamination=contamination)
    outliers = model.fit_predict(data.reshape(-1, 1))
    return np.where(outliers == -1)
```

DBSCAN

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) groups together data points that are close to each other and marks as outliers the points that are in low-density regions.



```python
from sklearn.cluster import DBSCAN

def detect_outliers_dbscan(data, eps=0.5, min_samples=5):
    model = DBSCAN(eps=eps, min_samples=min_samples)
    outliers = model.fit_predict(data.reshape(-1, 1))
    return np.where(outliers == -1)
```