Experiment 10

Name: Umang Kirit Lodaya **SAP ID: 60009200032**

Batch: D11

Aim: Compare and analyze the Covariance Estimation for robust estimates

Objective:

- Understand the concept of robust covariance estimation.
- Compare and contrast different robust covariance estimators.
- Apply robust covariance estimation to real data sets.

Theory:

Covariance estimation is a fundamental statistical technique used to measure the relationship between variables in a dataset. However, standard covariance estimators can be highly sensitive to outliers and non-Gaussian data, leading to unreliable and biased results. In practical scenarios where data may be contaminated by noise or extreme observations, robust covariance estimation techniques are essential to obtain accurate and robust estimates.

Traditional covariance estimation methods, such as the sample covariance matrix, assume that the underlying data is normally distributed and free from outliers. However, real-world datasets often violate these assumptions, making the conventional estimators highly susceptible to influential observations. Outliers can substantially distort the covariance matrix, leading to unreliable parameter estimates, increased Type I errors, and reduced statistical power.

Robust covariance estimation addresses these issues by developing techniques that are resilient to the presence of outliers and deviations from normality. By downweighting or ignoring extreme observations, these methods provide more stable and accurate estimates of the covariance matrix.

Key Robust Covariance Estimation Techniques:

1. M-estimators: M-estimators are a class of robust estimators that minimize a certain objective function to estimate the covariance matrix. These estimators can be designed to be less influenced by extreme observations by incorporating robust weight functions. Examples of M-estimators for covariance estimation include the Minimum Covariance Determinant (MCD) and Tyler's M-estimator.

2. Minimum Volume Ellipsoid (MVE): The MVE method finds the smallest volume ellipsoid that contains a certain proportion of the data points. This approach is particularly useful for multivariate data with a significant number of outliers.

3. S-estimators: S-estimators are robust covariance estimators based on robust estimates of location and scale, such as the M-estimators of location and the median absolute deviation (MAD) of scale. The S-estimators provide an efficient way to compute robust covariance matrices in high-dimensional data settings.

4. Shrinkage Estimators: Shrinkage estimators strike a balance between the sample covariance matrix and a target matrix (e.g., diagonal matrix) by applying a shrinkage parameter. This technique improves the performance of covariance estimation when the sample size is relatively small compared to the number of variables.

Robust covariance estimation plays a vital role in modern statistical analysis, providing reliable and accurate estimates of the covariance matrix, even in the presence of outliers and non-Gaussian data. These techniques are essential in various fields where data may be contaminated or influenced by extreme observations, contributing to more robust and trustworthy statistical inference and decision-making processes. Researchers and practitioners are encouraged to use robust covariance estimation methods to enhance the robustness and validity of their statistical analyses.

Lab Experiment to be done by students:

1. Generate a data set with outliers.
2. Estimate the covariance matrix of the data set using the sample covariance matrix and the MCD estimator.
3. Compare the two covariance matrices.
4. Repeat steps 1-3 for different data sets with different levels of outliers.

NAME: UMANG KIRIT LODAYA

SAP ID: 60009200032

BATCH: D11

In [1]:
```
import numpy as np
from sklearn.covariance import MinCovDet
```

In [2]:
```
# Generate a random dataset with outliers
def generate_outlier_dataset(num_samples, num_features, outlie
r_fraction):
    inliers = np.random.randn(int((1 - outlier_fraction) * num
_samples), num_features)
    outliers = 10.0 * np.random.randn(int(outlier_fraction * n
um_samples), num_features)
    dataset = np.vstack([inliers, outliers])
    return dataset
```

In [3]:
```
# Estimate covariance matrix using the sample covariance and
CD estimator
def estimate_covariance_matrix(dataset):
    sample_cov = np.cov(dataset, rowvar=False)
    mcd = MinCovDet().fit(dataset)
    mcd_cov = mcd.covariance_
    return sample_cov, mcd_cov
```

In [4]:
```
def compare_covariance_matrices(sample_cov, mcd_cov):
    print("Sample Covariance Matrix:")
    print(sample_cov)
    print("\nMCD Covariance Matrix:")
    print(mcd_cov)
    print("\nDifference between the matrices:")
    print(sample_cov - mcd_cov)
```

In [5]:

```
np.random.seed(42)
num_samples = 100
num_features = 3
outlier_fractions = [0.1, 0.2, 0.3]
for outlier_fraction in outlier_fractions:
    print(f"\nGenerating dataset with {int(outlier_fraction *
100)}% outliers:")
    dataset = generate_outlier_dataset(num_samples, num_featur
es, outlier_fraction)
    sample_cov, mcd_cov = estimate_covariance_matrix(dataset)
    print("\nComparing covariance matrices:")
    compare_covariance_matrices(sample_cov, mcd_cov)
```

```
Generating dataset with 10% outliers:

Comparing covariance matrices:
Sample Covariance Matrix:
[[ 9.77826035 -3.70391397  3.45120391]
 [-3.70391397  7.37128737 -4.96073006]
 [ 3.45120391 -4.96073006  9.5865595 ]]

MCD Covariance Matrix:
[[ 0.54165739 -0.04922735 -0.10961889]
 [-0.04922735  0.91676002 -0.15491047]
 [-0.10961889 -0.15491047  0.94779386]]

Difference between the matrices:
[[ 9.23660296 -3.65468662  3.5608228 ]
 [-3.65468662  6.45452735 -4.8058196 ]
 [ 3.5608228  -4.8058196   8.63876564]]

Generating dataset with 20% outliers:

Comparing covariance matrices:
Sample Covariance Matrix:
[[23.51572186 -7.1339849   0.84439369]
 [-7.1339849  22.56958587 -2.88239039]
 [ 0.84439369 -2.88239039 17.96732802]]

MCD Covariance Matrix:
[[ 0.84914283 -0.0393283  -0.06345819]
 [-0.0393283   0.89745006  0.03037772]
 [-0.06345819  0.03037772  0.97531991]]

Difference between the matrices:
[[22.66657903 -7.0946566   0.90785188]
 [-7.0946566  21.67213581 -2.91276812]
 [ 0.90785188 -2.91276812 16.99200811]]

Generating dataset with 30% outliers:

Comparing covariance matrices:
Sample Covariance Matrix:
[[17.12415013  6.55201312 -2.09108002]
 [ 6.55201312 32.92189762  9.22710994]
 [-2.09108002  9.22710994 22.40710351]]

MCD Covariance Matrix:
[[ 0.711491   -0.08264049  0.06078279]
 [-0.08264049  1.0002048  -0.09073419]
 [ 0.06078279 -0.09073419  0.967777  ]]

Difference between the matrices:
[[16.41265913  6.63465361 -2.15186281]
 [ 6.63465361 31.92169282  9.31784413]
 [-2.15186281  9.31784413 21.43932651]]
```