Robust Covariance or Elliptic Envelope is a technique used for anomaly detection. It is a type of unsupervised learning that identifies observations or data points that do not conform to the expected pattern or behavior of the majority of the data. The technique is particularly useful in situations where the data may be contaminated with outliers, which can skew statistical estimates and models.

Elliptic Envelope uses the concept of a robust estimator to fit an elliptical envelope to the majority of the data points. The robust estimator is less sensitive to the presence of outliers than traditional estimators such as the mean or variance. The envelope is defined by a covariance matrix that describes the shape and orientation of the ellipse. Data points that fall outside the envelope are considered anomalies.

To implement the Elliptic Envelope method in Python, you can use the scikit-learn library. Here is an example code:

from sklearn.covariance import EllipticEnvelope

import numpy as np

# Generate some random data with outliers

rng = np.random.RandomState(42)

X = 0.3 \* rng.randn(100, 2)

X\_outliers = rng.uniform(low=-4, high=4, size=(20, 2))

X = np.vstack([X, X\_outliers])

# Fit the Elliptic Envelope model

clf = EllipticEnvelope(contamination=0.1)

clf.fit(X)

# Predict the anomalies

y\_pred = clf.predict(X)

# Print the results

print(y\_pred)

To compute the Robust Covariance (Elliptic Envelope) in MATLAB, you can follow these steps:

1. Load your data into MATLAB. You can do this by importing a CSV file, or by creating a matrix directly in MATLAB using the **zeros()** or **ones()** function.
2. Use the **cov()** function to compute the covariance matrix of your data. This computes the sample covariance matrix, which is an estimate of the true covariance matrix of the population from which the data was drawn.

data = csvread('mydata.csv');

covariance = cov(data);

1. Use the **robustcov()** function to compute the robust covariance matrix using the Minimum Covariance Determinant (MCD) estimator. This estimator is less sensitive to the presence of outliers than the sample covariance matrix.
2. Compute the Mahalanobis distance for each data point with respect to the robust covariance matrix. The Mahalanobis distance is a measure of how far a point is from the center of the distribution in units of the standard deviation. Points with large Mahalanobis distances are more likely to be outliers.

distances = mahal(data, data);

1. Determine the threshold for identifying outliers. This can be done using a quantile of the distribution of Mahalanobis distances. For example, you might choose the 95th percentile as the threshold.

threshold = quantile(distances, 0.95);

1. Identify the outliers by finding the data points with Mahalanobis distances greater than the threshold.

outliers = find(distances > threshold);

1. You can now visualize the results by plotting the data and highlighting the outliers using different colors or symbols.

plot(data(:, 1), data(:, 2), 'bo')

hold on

plot(data(outliers, 1), data(outliers, 2), 'rx')