MahalanobisDistance

February 8, 2025

1 Mahalanobis Distance in Detail with Hand Computation Example

The Mahalanobis distance is a multivariate measure that computes the distance between a point x and a distribution characterized by its mean μ and covariance matrix S. Unlike the Euclidean distance, it accounts for the correlations between variables and the scale of the data. The Mahalanobis distance is defined as:

$$D_M(x) = \sqrt{(x-\mu)^T S^{-1}(x-\mu)}.$$

1.1 Mathematical Breakdown

1. **Mean (μ)**: The average of the dataset. It serves as the central point of the distribution.
2. **Covariance Matrix (S)**: Measures the variability and the correlation between different features in the data.
3. **Inverse Covariance Matrix (S^{-1})**: Used to scale the difference vector $x-\mu$ by accounting for the variance and correlation of the features.

The computed Mahalanobis distance tells us how many "standard deviations" away the point x is from the mean μ considering the underlying data structure.

[2]:
```
# Hand Computation Example
import numpy as np
import math

# Define a simple dataset with 3 two-dimensional points
data = np.array([
    [2, 3],
    [3, 5],
    [4, 2]
])

# Step 1: Compute the mean (\u03BC) of the dataset
mean = np.mean(data, axis=0)
print('Mean (\u03BC):', mean)

# Step 2: Compute the sample covariance matrix (using n-1 in the denominator)
cov_matrix = np.cov(data, rowvar=False)
```

```python
print('\nCovariance Matrix (S):\n', cov_matrix)

# Step 3: Compute the inverse of the covariance matrix
inv_cov_matrix = np.linalg.inv(cov_matrix)
print('\nInverse Covariance Matrix (S^{-1}):\n', inv_cov_matrix)

# Choose a point for which to compute the Mahalanobis distance
x = np.array([3, 4])
print('\nPoint x:', x)

# Step 4: Compute the difference vector (x - mean)
diff = x - mean
print('\nDifference vector (x - \u03BC):', diff)

# Step 5: Compute the squared Mahalanobis distance
# D_M(x)^2 = (x - mu)^T * S^{-1} * (x - mu)
md_squared = np.dot(np.dot(diff.T, inv_cov_matrix), diff)
print('\nSquared Mahalanobis Distance:', md_squared)

# Step 6: Take the square root to get the Mahalanobis distance
md = math.sqrt(md_squared)
print('Mahalanobis Distance:', md)

# For clarity, here is the step-by-step breakdown printed out:
print('\n--- Step-by-Step Manual Computation ---')
print('1. Mean (\u03BC):', mean)
print('2. Covariance Matrix (S):\n', cov_matrix)
print('3. Inverse Covariance Matrix (S^{-1}):\n', inv_cov_matrix)
print('4. Difference vector (x - \u03BC):', diff)
print('5. Squared Mahalanobis Distance: ', md_squared)
print('6. Mahalanobis Distance: ', md)
```

```
Mean ( ): [3.         3.33333333]

Covariance Matrix (S):
 [[ 1.         -0.5       ]
 [-0.5         2.33333333]]

Inverse Covariance Matrix (S^{-1}):
 [[1.12 0.24]
 [0.24 0.48]]

Point x: [3 4]

Difference vector (x -  ): [0.         0.66666667]

Squared Mahalanobis Distance: 0.2133333333333332
```

```
Mahalanobis Distance: 0.4618802153517005

--- Step-by-Step Manual Computation ---
1. Mean ( ): [3.         3.33333333]
2. Covariance Matrix (S):
 [[ 1.         -0.5       ]
 [-0.5         2.33333333]]
3. Inverse Covariance Matrix (S^{-1}):
 [[1.12 0.24]
 [0.24 0.48]]
4. Difference vector (x -  ): [0.         0.66666667]
5. Squared Mahalanobis Distance:  0.21333333333333332
6. Mahalanobis Distance:  0.4618802153517005
```

1.2 Interpretation

In the above example, we used a small dataset with three two-dimensional points:

- $ (2, 3) $
- $ (3, 5) $
- $ (4, 2) $

The mean μ of these points is computed, and the covariance matrix S captures the variability and correlation in the data. By calculating the inverse of the covariance matrix S^{-1} and the difference vector $x-\mu$ for a chosen point $x = [3, 4]$, we then compute the Mahalanobis distance. This distance quantifies how far x is from the distribution defined by the dataset, considering the data's spread and correlation. A higher Mahalanobis distance would indicate that the point x is more of an outlier. This manual computation illustrates each step clearly, and similar methods can be applied to larger, higher-dimensional datasets.

1.3 Additional Considerations

- **Normalization:** The Mahalanobis distance is invariant under linear transformations because it normalizes the scale of each feature using the covariance matrix.
- **Applications:** It is widely used in multivariate anomaly detection, clustering, and classification tasks, where the correlation among features plays a significant role.
- **Robustness:** The reliability of the Mahalanobis distance depends on an accurate estimation of the covariance matrix. For small or non-representative datasets, this estimation may be less robust.

[ ]:

[ ]:

[ ]:

[ ]:

[ ]: