“Unlocking the Power of Mahalanobis Distance: Exploring Multivariate Data Analysis with Python”

Vishal Sharma · Follow

5 min read · Mar 6

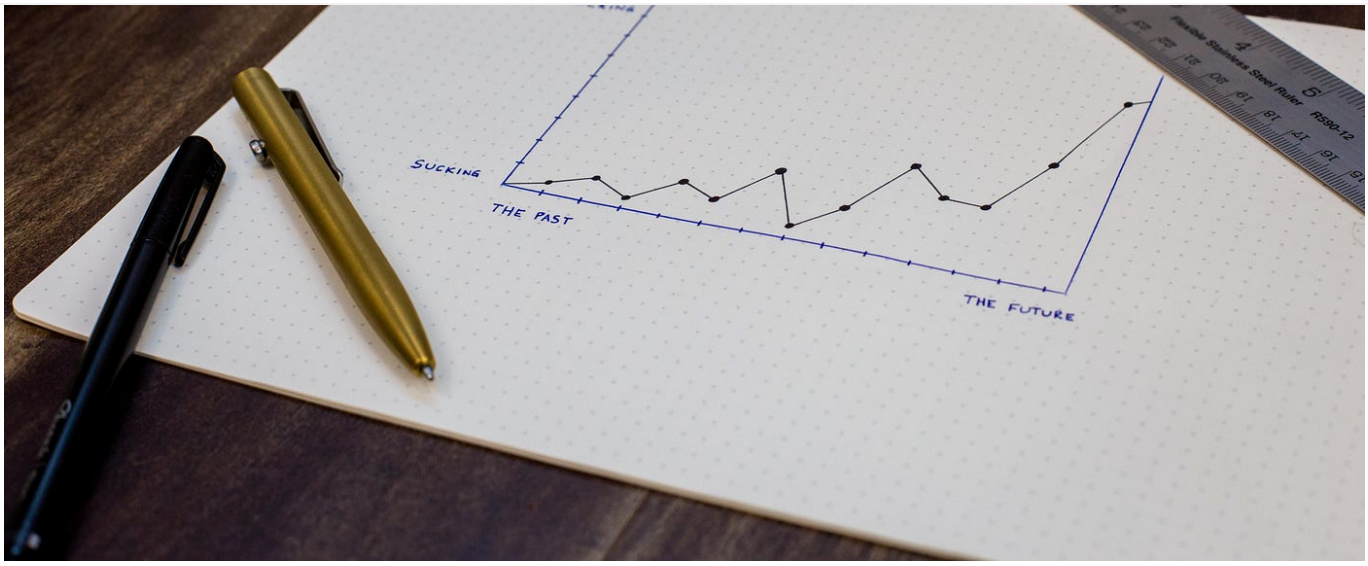


Photo by Isaac Smith on Unsplash

Introduction

Mahalanobis Distance is a statistical tool used to measure the distance between a point and a distribution. It is a powerful technique that considers the correlations between variables in a dataset, making it a valuable tool in various applications such as outlier detection, clustering, and classification.

For instance, let's consider a scenario where a company wants to identify potential fraud in credit card transactions. The company collects data on various variables such as the transaction amount, location, time, and other credit card transaction details. It then uses Mahalanobis Distance to measure the distance between each transaction and the distribution of all trades. By doing this, it can identify transactions that are significantly different from the rest and may indicate fraudulent activity.

Mahalanobis Distance measures the distance between a point and a distribution, considering the correlations between variables in the data. It is the distance between a point x and a distribution with mean vector μ and covariance matrix Σ. The formula for Mahalanobis Distance is given as:

> $D^2 = (x-\mu)^T\Sigma^{-1}(x-\mu)$

Where D^2 is the squared Mahalanobis Distance, x is the point in question, μ is the mean vector of the distribution, Σ is the covariance matrix of the distribution, and ^{T} denotes the transpose of a matrix.

To better understand this formula, let's take an example. Suppose we have a dataset containing two variables, X and Y, and we want to measure the distance between a point (2, 3) and the distribution of all points in the dataset. We calculate the mean vector and covariance matrix of the dataset as follows:

μ = [mean(X), mean(Y)] = [3, 4] Σ = [[var(X), cov(X,Y)], [cov(X,Y), var(Y)]] = [[2, -1], [-1, 2]]

We can now use the Mahalanobis Distance formula to calculate the distance between the point (2, 3) and the distribution:

$D^2 = ([2, 3]-[3, 4])^T[[2, -1], [-1, 2]]^{-1}([2, 3]-[3, 4])$

$= [-1, -1]^T[[2, -1], [-1, 2]]^{-1}[-1, -1]$

$= [2, -2]^T[[2/3, 1/3], [1/3, 2/3]][2, -2]$

$= [2/3, -2/3]^T[2, -2]$

= 4/3.

Therefore, the squared Mahalanobis Distance between the point (2, 3) and the distribution is 4/3. By calculating the Mahalanobis Distance, we can determine how far the point is from the distribution, considering the correlations between the X and Y variables.

```
import numpy as np
from scipy.spatial.distance import mahalanobis
from sklearn.datasets import make_blobs

# Create a dataset with 2 clusters
X, y = make_blobs(n_samples=100, centers=2, random_state=42)

# Calculate the mean vector and covariance matrix of the dataset
mu = np.mean(X, axis=0)
sigma = np.cov(X.T)

# Calculate the Mahalanobis Distance between two points
x1 = [2, 2]
x2 = [-2, -2]
dist_x1 = mahalanobis(x1, mu, np.linalg.inv(sigma))
dist_x2 = mahalanobis(x2, mu, np.linalg.inv(sigma))

# Print the distances
print("Distance between point x1 and the distribution:", dist_x1)
print("Distance between point x2 and the distribution:", dist_x2)

#OUTPUT

Distance between point x1 and the distribution: 2.099478227196236
Distance between point x2 and the distribution: 8.065203145117373
```

Here are some examples of how Mahalanobis distance can be used:

1. **Outlier detection:** Mahalanobis distance can detect outliers in a dataset. Outliers are data points significantly different from the rest of the

dataset. By calculating the Mahalanobis distance between each data point and the mean of the dataset, we can identify data points far away from the mean. These data points can be considered outliers and may need to be removed or investigated further.

2. **Clustering:** Mahalanobis distance can also be used for clustering data points. Clustering is the process of grouping similar data points together. By calculating the Mahalanobis distance between each data point and the mean of each cluster, we can determine which cluster a data point belongs to. This method is useful for clustering data points with different variances or covariances.
3. **Image classification:** Mahalanobis distance can be used in image classification tasks. This application uses Mahalanobis distance to measure the similarity between a test image and a set of training images. By calculating the Mahalanobis distance between the test image and each training image, we can determine which training image is most similar to the test image. This method is useful for tasks such as face recognition and object detection.
4. **Fraud detection:** Mahalanobis distance can be used for fraud detection in financial transactions. By calculating the Mahalanobis distance between a transaction and a set of historical transactions, we can determine if the transaction is unusual or suspicious. This method is useful for detecting fraudulent transactions that may otherwise go unnoticed.

Here is an example of how Mahalanobis distance can be used to create beautiful plots for a real-life dataset:

For this example, let's use the famous Iris dataset, which contains measurements for 150 Iris flowers. We will use the sepal length, width, and

petal length as our features.

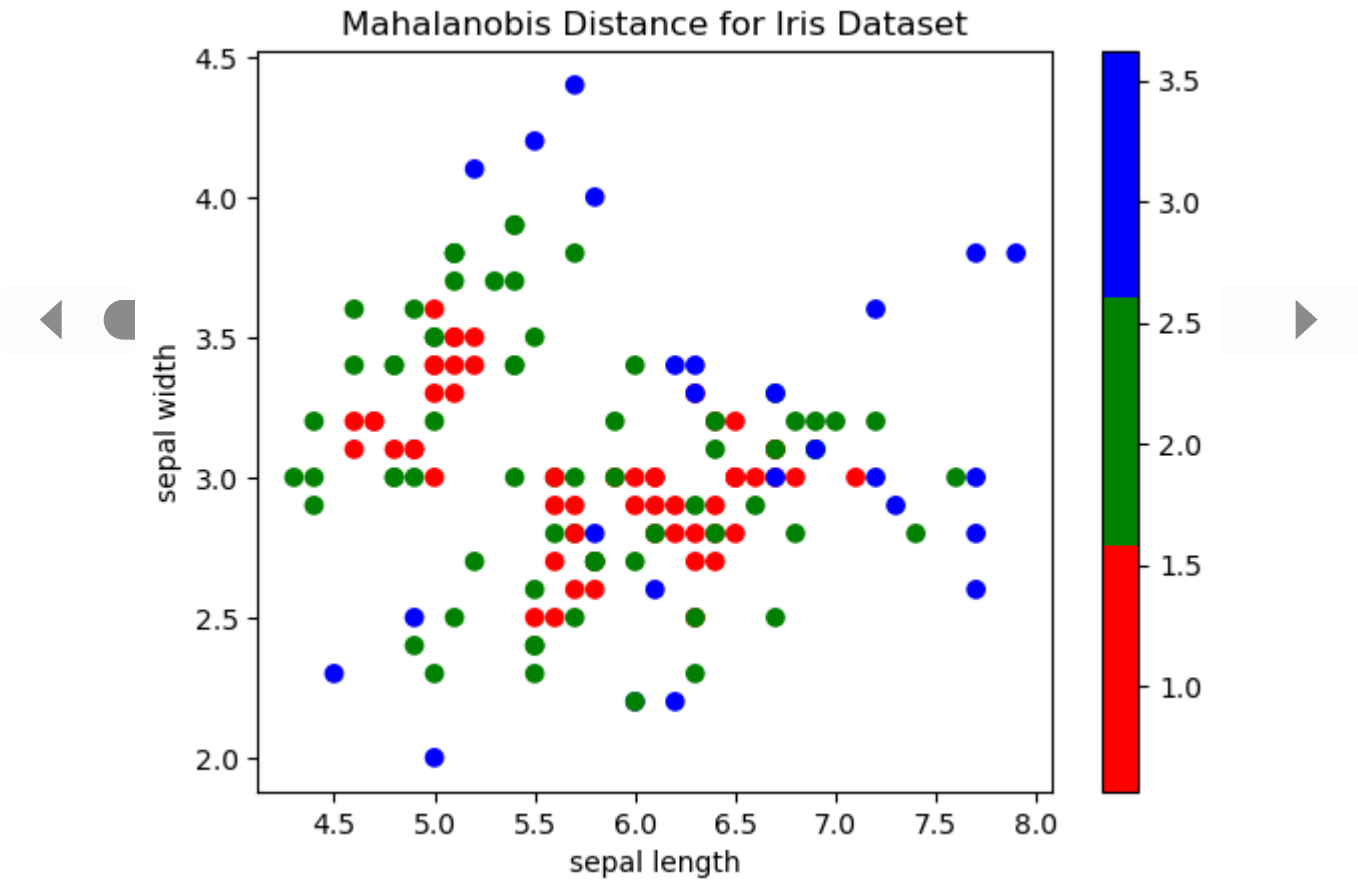
First, we will calculate the Mahalanobis distance for each data point in the dataset. We can do this using the following code in Python:

```
import numpy as np
from scipy.spatial.distance import mahalanobis

# load the iris dataset
from sklearn.datasets import load_iris
iris = load_iris()

# calculate the mean and covariance matrix of the dataset
mean = np.mean(iris.data, axis=0)
cov = np.cov(iris.data.T)

# calculate the Mahalanobis distance for each data point
mahalanobis_dist = [mahalanobis(x, mean, np.linalg.inv(cov)) for x in iris.data]
```

Next, we can create a scatter plot of the iris dataset using the first two features (sepal length and sepal width) and color each data point based on its Mahalanobis distance. We can use a color map to map the Mahalanobis distances to a color scale. Here's the code for the plot:

```
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap

# create a color map for the Mahalanobis distances
cmap = ListedColormap(['r', 'g', 'b'])
norm = plt.Normalize(min(mahalanobis_dist), max(mahalanobis_dist))

# create a scatter plot of the iris dataset
plt.scatter(iris.data[:, 0], iris.data[:, 1], c=mahalanobis_dist, cmap=cmap, nor
```

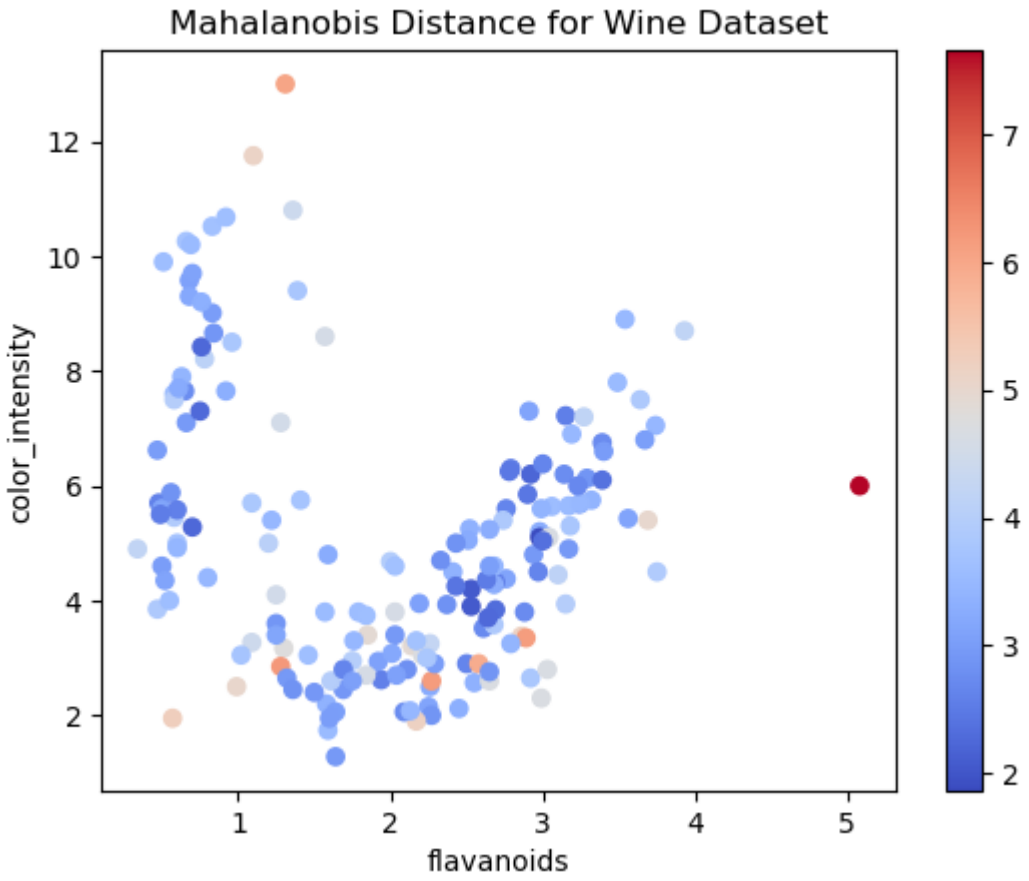
```
# add a color bar
plt.colorbar()
plt.xlabel('sepal length')
plt.ylabel('sepal width')
plt.title('Mahalanobis Distance for Iris Dataset')
plt.show()
```



The resulting plot shows the Iris dataset with each data point colored based on its Mahalanobis distance. Data points that are far away from the mean (i.e., outliers) are colored in red, while data points that are closer to the mean are colored in green and blue.

Here is another example using the famous Wine dataset that comes with the sci-kit-learn library:

```
import numpy as np
from scipy.spatial.distance import mahalanobis
import pandas as pd
from sklearn.datasets import load_wine
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap

# load the wine dataset
wine = load_wine()
wine_df = pd.DataFrame(wine.data, columns=wine.feature_names)

# calculate the mean and covariance matrix of the dataset
mean = np.mean(wine_df, axis=0)
cov = np.cov(wine_df.T)

# calculate the Mahalanobis distance for each data point
mahalanobis_dist = [mahalanobis(x, mean, np.linalg.inv(cov)) for x in wine_df.va

# create a scatter plot of the wine dataset using two highly correlated features
plt.scatter(wine_df['flavanoids'], wine_df['color_intensity'], c=mahalanobis_dis

# add a color bar
plt.colorbar()
plt.xlabel('flavanoids')
plt.ylabel('color_intensity')
plt.title('Mahalanobis Distance for Wine Dataset')
plt.show()
```



A scatter plot of two highly correlated features (flavonoids and color_intensity) with each data point colored based on its Mahalanobis distance.

Machine Learning | Data Science | Python | Statistics | Data Visualization

Vishal Sharma