République Tunisienne
Ministère de l'enseignement supérieur et de la recherche scientifique et de la technologie

Université de Sousse
Institut Supérieur de l'Informatique et de Technologies de Communication

Mahalanobis Distance

Élaboré par : Nourhene YAZIDI

3DNI1

Année Universitaire : 2021-2022

Introduction

Mahalanobis distance is an effective multivariate distance metric that measures the distance between a point (vector) and a distribution.

It has excellent applications in multivariate anomaly detection, classification on highly imbalanced datasets and one-class classification and more untapped use cases.

Considering its extremely useful applications, this metric is seldom discussed or used in stats or ML workflows. This post explains the why and the when to use Mahalanobis distance and then explains the intuition and the math with useful applications.

What's wrong with using Euclidean Distance for Multivariate data?

Let's start with the basics. Euclidean distance is the commonly used straight line distance between two points.

If the two points are in a two-dimensional plane (meaning, you have two numeric columns (p) and (q)) in your dataset), then the Euclidean distance between the two points (p1, q1) and (p2, q2) is:

$$d(p,q) = \sqrt{(p_1 - q_1)2 + (p_2 - q_2)2}$$

This formula may be extended to as many dimensions you want:

$$d(p,q) = \sqrt{(p_1 - q_1)2 + (p_2 - q_2)2 \ + \cdot \cdot + (p_n - q_n)2}$$

Well, Euclidean distance will work fine as long as the dimensions are equally weighted and are independent of each other.

What do I mean by that?

Let's consider the following tables:

Area (sq.ft)	Price ($ 1000's)
2400	156000
1950	126750
2100	105000
1200	78000
2000	130000
900	54000

Area (acre)	Price ($M)
0.0550944	156
0.0447642	126.75
0.0482076	105
0.0275472	78
0.045912	130
0.0206604	54

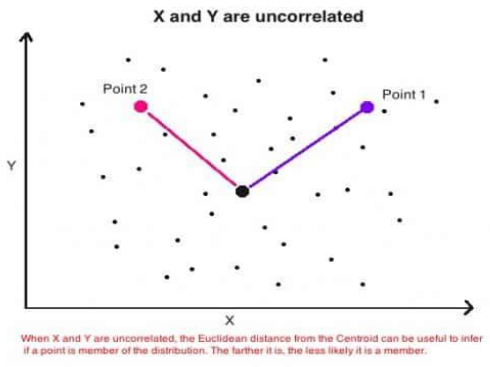
The two tables above show the 'area' and 'price' of the same objects. Only the units of the variables change.

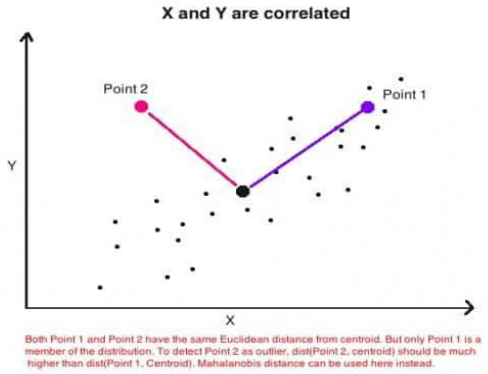
Since both tables represent the same entities, the distance between any two rows, point A and point B should be the same. But Euclidean distance gives a different value even though the distances are technically the same in physical space.

This can technically be overcome by scaling the variables, by computing the z-score (ex: (x – mean) / std) or make it vary within a particular range like between 0 and 1.

But there is another major drawback.

That is, if the dimensions (columns in your dataset) are correlated to one another, which is typically the case in real-world datasets, the Euclidean distance between a point and the center of the points (distribution) can give little or misleading information about how close a point really is to the cluster.





The above image (on the right) is a simple scatterplot of two variables that are positively correlated with each other. That is, as the value of one variable (x-axis) increases, so does the value of the other variable (y-axis).

The two points above are equally distant (Euclidean) from the center. But only one of them (blue) is actually more close to the cluster, even though, technically the Euclidean distance between the two points are equal.

This is because, Euclidean distance is a distance between two points only. It does not consider how the rest of the points in the dataset vary. So, it cannot be used to really judge how close a point actually is to a distribution of points.

What we need here is a more robust distance metric that is an accurate representation of how distant a point is from a *distribution*.

What is Mahalanobis Distance?

Mahalonobis distance is the distance between a point and a distribution. And not between two distinct points. It is effectively a multivariate equivalent of the Euclidean distance.

It was introduced by Prof. P. C. Mahalanobis in 1936 and has been used in various statistical applications ever since. However, it's not so well known or used in the machine learning practice. Well, let's get into it.

So computationally, how is Mahalanobis distance different from Euclidean distance?

1. It transforms the columns into uncorrelated variables
2. Scale the columns to make their variance equal to 1
3. Finally, it calculates the Euclidean distance.

The above three steps are meant to address the problems with Euclidean distance we just talked about. But how?

Let's look at the formula and try to understand its components.

The math and intuition behind Mahalanobis Distance

The formula to compute Mahalanobis distance is as follows:

$$D^2 = (x - m)^T \cdot C^{-1} \cdot (x - m)$$

where,

- D^2 is the square of the Mahalanobis distance.

- x is the vector of the observation (row in a dataset),

- m is the vector of mean values of independent variables (mean of each column),

- C^(-1) is the inverse covariance matrix of independent

variables.

So, how to understand the above formula?

Let's take the (x – m)^T . C^(-1) term.

(x – m) is essentially the distance of the vector from the mean. We then divide this by the covariance matrix (or multiply by the inverse of the covariance matrix).

If you think about it, this is essentially a multivariate equivalent of the regular standardization (z = (x – mu)/sigma). That is, z = (x vector) – (mean vector) / (covariance matrix).

So, What is the effect of dividing by the covariance?

If the variables in your dataset are strongly correlated, then, the covariance will be high. Dividing by a large covariance will effectively reduce the distance.

Likewise, if the X’s are not correlated, then the covariance is not high and the distance is not reduced much.

So effectively, it addresses both the problems of scale as well as the correlation of the variables that we talked about in the introduction.