Distance Measurement Metrics.

For the algorithm to work best on a particular dataset we need to choose the most appropriate distance metric accordingly. There are a lot of different distance metrics available, but we are only going to talk about a few widely used ones. Euclidean distance function is the most popular one among all of them as it is set default in the SKlearn KNN classifier library in python.

1. Minkowsi Distance.

- It is a metric intended for real-valued vector spaces. We can calculate Minkowski distance only in a normed vector space, which means in a space where distances can be represented as a vector that has a length and the lengths cannot be negative.
- There are a few conditions that the distance metric must satisfy:

- Non-negativity: `d(x, y) >= 0`
- Identity: `d(x, y) = 0 if and only if x == y`
- Symmetry: `d(x, y) = d(y, x)`
- Triangle Inequality: `d(x, y) + d(y, z) >= d(x, z)`

$$\left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{1/p}$$

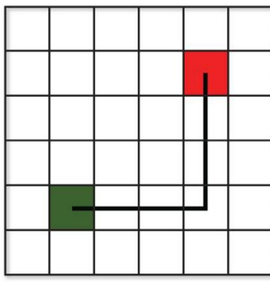
This above formula for Minkowski distance is in generalized form and we can manipulate it to get different distance metrices.

The p value in the formula can be manipulated to give us different distances like:

- p = 1, when p is set to 1 we get Manhattan distance
- p = 2, when p is set to 2 we get Euclidean distance
- Here, p represents the order of the norm. Let's calculate the Minkowski Distance of the order 3:

2. Manhattan Distance

This distance is also known as taxicab distance or city block distance, that is because the way this distance is calculated. The distance between two points is the sum of the absolute differences of their Cartesian coordinates.



Manhattan Distance

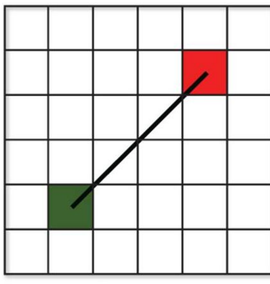
As we know we get the formula for Manhattan distance by substituting p=1 in the Minkowski distance formula.

$$d = \sum_{i=1}^{n} |x_i - y_i|$$

- Suppose we have two points as shown in the image the red(4,4) and the green(1,1).
- And now we have to calculate the distance using Manhattan distance metric.
- We will get,
- `d = |4-1| + |4-1| = 6`
- This distance is preferred over Euclidean distance when we have a case of high dimensionality.

3. Euclidean Distance

This distance is the most widely used one as it is the default metric that SKlearn library of Python uses for K-Nearest Neighbour. It is a measure of the true straight line distance between two points in Euclidean space.



Euclidean Distance

- It can be used by setting the value of p equal to 2 in Minkowski distance metric.

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

- Now suppose we have two point the red (4,4) and the green (1,1).
- And now we have to calculate the distance using Euclidean distance metric.
- We will get,
- 4.24
- It is used pythogorous thereom in order to calculate the digonal distance.

4. Hamming Distance

So far, we have covered the distance metrics that are used when we are dealing with continuous or numerical variables. But what if we have categorical variables? How can we decide the similarity between categorical variables? This is where we can make use of another distance metric called Hamming Distance.

Hamming Distance measures the similarity between two strings of the same length. The Hamming Distance between two strings of the same length is the number of positions at which the corresponding characters are different.

Let's understand the concept using an example. Let's say we have two strings:

- **"euclidean" and "manhattan"**
- Since the length of these strings is equal, we can calculate the Hamming Distance. We will go character by character and match the strings. The first character of both the strings (e and m respectively) is different. Similarly, the second character of both the strings (u and a) is different. and so on.
- Look carefully – seven characters are different whereas two characters (the last two characters) are similar:
- distance metrics
- Hence, the Hamming Distance here will be 7. Note that larger the Hamming Distance between two strings, more dissimilar will be those strings (and vice versa).
- As we saw in the example above, the Hamming Distance between "euclidean" and "manhattan" is 7. We also saw that Hamming Distance only works when we have strings of the same length.

5. Cosine Distance

- This distance metric is used mainly to calculate similarity between two vectors. It is measured by the cosine of the angle between two vectors and determines whether two vectors are pointing in the same direction. It is often used to measure document similarity in text analysis. When used with KNN this distance gives us a new perspective to a business problem and lets us find some hidden information in the data which we didn't see using the above two distance matrices.
- It is also used in text analytics to find similarities between two documents by the number of times a particular set of words appear in it.
- Formula for cosine distance is:

$$\cos\theta = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \cdot \|\vec{b}\|}$$

- Using this formula we will get a value which tells us about the similarity between the two vectors and `1-cosθ` will give us their cosine distance.
- Using this distance we get values between `0 and 1`, where `0` means the vectors are `100%` similar to each other and `1` means they are not similar at all.

5. Jaccard Distance

- The Jaccard coefficient is a similar method of comparison to the Cosine Similarity due to how both methods compare one type of attribute distributed among all data. The Jaccard approach looks at the two data sets and finds the incident where both values are equal to 1. So the resulting value reflects how many 1 to 1 matches occur in comparison to the total number of data points. This is also known as the frequency that 1 to 1 match, which is what the Cosine Similarity looks for, how frequent a certain attribute occurs.
- It is extremely sensitive to small samples sizes and may give erroneous results, especially with very small data sets with missing observations.
- The formula for Jaccard index is:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

- Jaccard distance is the complement of the Jaccard index and can be found by subtracting the Jaccard Index from 100%, thus the formula for Jaccard distance is:
- `D(A,B) = 1 – J(A,B)`

Thank You !!