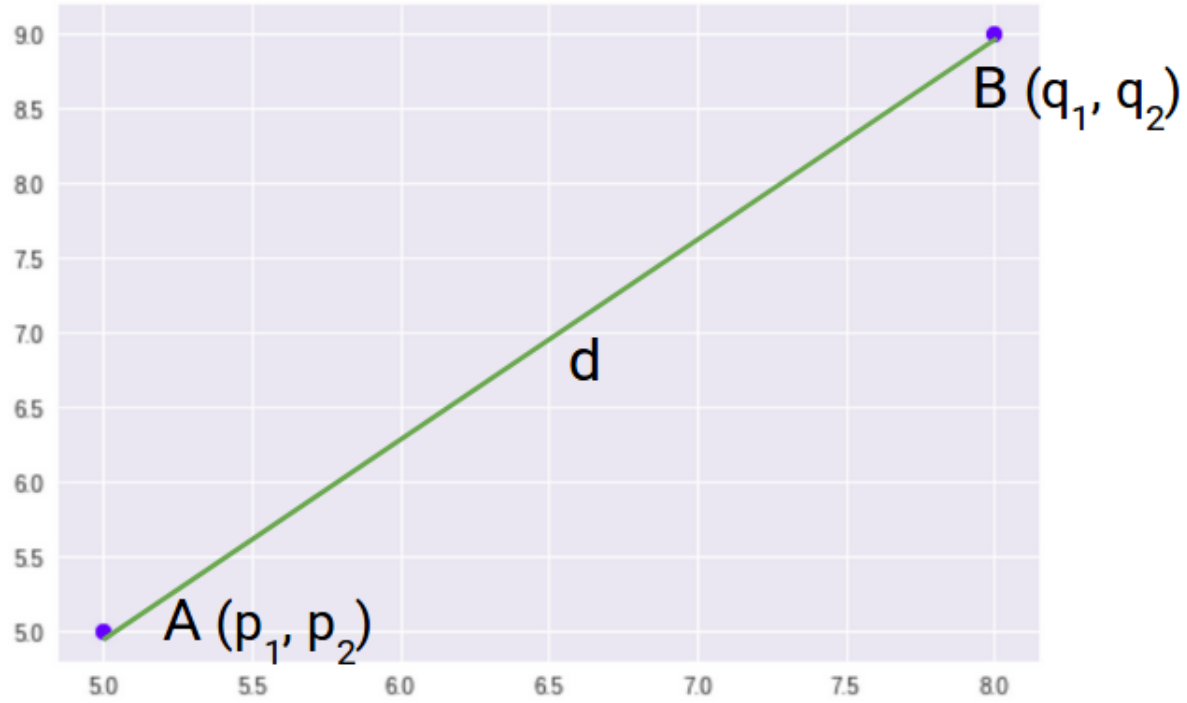
EUCLIDEAN

Euclidean distance calculates the distance between two real-valued vectors. Euclidean Distance represents the shortest distance between two points.



$$d = ((p_1 - q_1)^2 + (p_2 - q_2)^2)^{1/2}$$

The generalized formula for an n-dimensional space is given as

$$D_e = \left(\sum_{i=1}^{n} (p_i - q_i)^2 \right)^{1/2}$$

n = number of dimensions

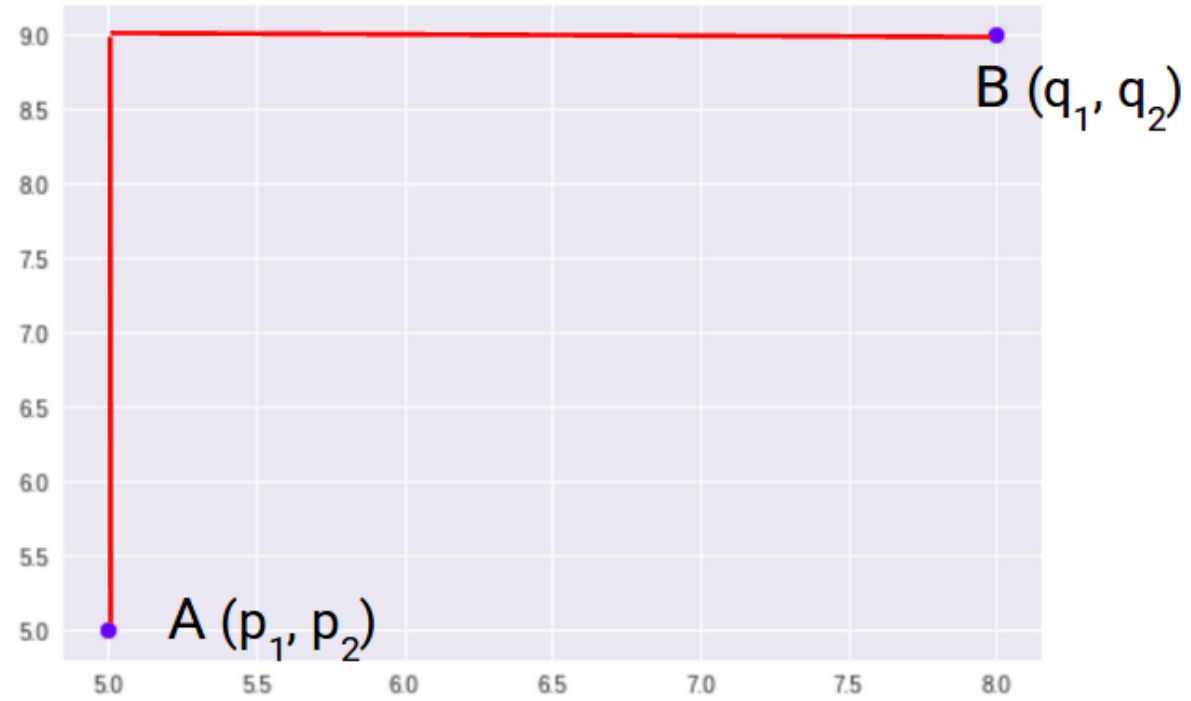
pi, qi = data points

```
Euclidean Distance b/w (1, 2, 3) and (4, 5, 6) is:  5.196152422706632
```

MANHATTAN

The Manhattan distance, also called the Taxicab distance or the City Block distance calculates the distance between two real-valued vectors.

Manhattan Distance is the sum of absolute differences between points across all the dimensions.



$$d = |p_1 - q_1| + |p_2 - q_2|$$

The generalized formula for an n-dimensional space is given as

$$D_m = \sum_{i=1}^{n} |p_i - q_i|$$

n = number of dimensions

pi, qi = data points

```
Manhattan Distance b/w (1, 2, 3) and (4, 5, 6) is:  9
```

MINKOWSKI

Minkowski distance calculates the distance between two real-valued vectors.

It is a generalization of the Euclidean and Manhattan distance measures and adds a parameter, called the order that allows different distance measures to be calculated.

$$D = \left(\sum_{i=1}^{n} |p_i - q_i|^p \right)^{1/p}$$

```
Minkowski Distance b/w (1, 2, 3) and (4, 5, 6) is:  4.3267487109222245
```

HAMMING

Hamming distance calculates the distance between two binary vectors (binary strings or bit strings)

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.

For a one-hot encoded string, it might make more sense to summarize to the sum of the bit differences between the strings which will always be a 0 or 1.

red = [1, 0, 0]

green = [0, 1, 0]

blue = [0, 0, 1]

Hamming Distance = sum for i to N abs(v1[i] – v2[i])

For bit strings that may have many 1 bit, it is more common to calculate the average number of bit differences to give a hamming distance score between 0 (identical) and 1 (all different).

row1 = [0, 0, 0, 0, 0, 1]

row2 = [0, 0, 0, 0, 1, 0]

Hamming Distance = (sum for i to N abs(v1[i] – v2[i])) / N) = 1/3