

---

# L7: Distances

---

We have mainly been focusing on similarities so far, since it is easiest to explain locality sensitive hashing that way, and in particular the Jaccard similarity is easy to define in regards to the  $k$ -shingles of text documents. In this lecture we will define a *metric* and then enumerate several important distances and their properties.

In general, choosing which distance to use is an important, but often ignored modeling problem. The  $L_2$  distance is most likely overused. This is likely because in many situations (but not all) it is very easy to use. Unfortunately in many situations the  $L_1$  distance is more robust and makes more sense.

## 7.1 Metrics

So what makes a good distance? There are two aspects to the answer to this question. The first is that it captures the “right” properties of the data, but this is a sometimes ambiguous modeling problem. The second is more well-defined; it is the properties which makes a distance a metric.

A distance  $\mathbf{d} : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}^+$  is a bivariate operator (it takes in two arguments, say  $a \in \mathcal{X}$  and  $b \in \mathcal{X}$ ) that maps to  $\mathbb{R}^+ = [0, \infty)$ . It is a *metric* if

- (M1)  $\mathbf{d}(a, b) \geq 0$  (non-negativity)
- (M2)  $\mathbf{d}(a, b) = 0$  if and only if  $a = b$  (identity)
- (M3)  $\mathbf{d}(a, b) = \mathbf{d}(b, a)$  (symmetry)
- (M4)  $\mathbf{d}(a, b) \leq \mathbf{d}(a, c) + \mathbf{d}(c, b)$  (triangle inequality)

A distance that satisfies (M1), (M3), and (M4) (but not necessarily (M2)) is called a *pseudometric*.

A distance that satisfies (M1), (M2), and (M4) (but not necessarily (M3)) is called a *quasimetric*.

## 7.2 Distances

We now enumerate a series of common distances.

### 7.2.1 $L_p$ Distances

Consider two vectors  $a = (a_1, a_2, \dots, a_d)$  and  $b = (b_1, b_2, \dots, b_d)$  in  $\mathbb{R}^d$ . Now an  $L_p$  distances is defined as

$$\mathbf{d}_p(a, b) = \|a - b\|_p = \left( \sum_{i=1}^d (|a_i - b_i|)^p \right)^{1/p}.$$

1. The most common is the  $L_2$  distance

$$\mathbf{d}_2(a, b) = \|a - b\| = \|a - b\|_2 = \sqrt{\sum_{i=1}^d (a_i - b_i)^2}.$$

It easy interpreted as the *Euclidean* or “straight-line” distance between two points or vectors, since if you draw a line between two points, its length measures the Euclidean distance.

It is also the only  $L_p$  distance that is invariant to the rotation of the coordinate system (which will be often be useful, but sometimes restrictive).

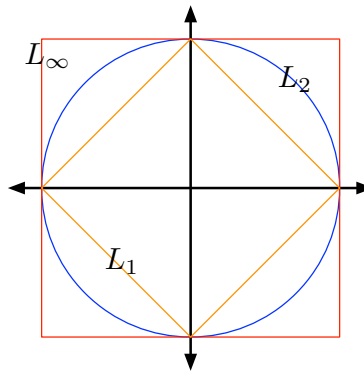


Figure 7.1: Unit balls in  $\mathbb{R}^2$  for the  $L_1$ ,  $L_2$ , and  $L_\infty$  distance.

- Another common distance is the  $L_1$  distance

$$\mathbf{d}_1(a, b) = \|a - b\|_1 = \sum_{i=1} |a_i - b_i|.$$

This is also known as the “Manhattan” distance since it is the sum of lengths on each coordinate axis; the distance you would need to walk in a city like Manhattan since must stay on the streets and can’t cut through buildings.

It is also amenable to LSH through 1-stable distributions (using Cauchy distribution  $\frac{1}{\pi} \frac{1}{1+x^2}$  in place of the Gaussian distribution).

- A common modeling goal is the  $L_0$  distance

$$\mathbf{d}_0(a, b) = \|a - b\|_0 = \sum_{i=1}^d \mathbb{1}(a_i \neq b_i),$$

where  $\mathbb{1}(a = b) = \begin{cases} 1 & \text{if } a = b \\ 0 & \text{if } a \neq b. \end{cases}$  Unfortunately,  $\mathbf{d}_0$  is not convex.

When each coordinate  $a_i$  is either 0 or 1, then this is known as the *Hamming distance*.

There is no associated  $p$ -stable distribution, but can be approximated by a 0.001-stable distribution (but this is quite inefficient). However, this is amenable to LSH using min-hashing.

- Finally, another useful variation is the  $L_\infty$  distance

$$\mathbf{d}_\infty(a, b) = \|a - b\|_\infty = \max_{i=1}^d |a_i - b_i|.$$

It is the maximum deviation along any one coordinate. Geometrically, it is a rotation of the  $L_1$  distance, so many algorithms designed for  $L_1$  can be adapted to  $L_\infty$ .

All of these distances are metrics. (M1) and (M2) hold since the distances are basically a sum of non-negative terms, and are only all 0 if all coordinates are identical. (M3) holds since  $|a_i - b_i| = |b_i - a_i|$ . (M4) is a bit trickier to show, but follows by drawing a picture  $\smile$ .

Figure 7.2.1 illustrates the unit balls for  $L_1$ ,  $L_2$ , and  $L_\infty$ . Note that the smaller the  $p$  value, the smaller the unit ball, and all touch the points a distance 1 from the origin along each axis. The  $L_0$  ball is inside the  $L_1$  ball, and in particular, for any  $p < 1$ , the  $L_p$  ball is *not* convex.

## 7.2.2 Jaccard Distance

The Jaccard distance between two sets  $A$  and  $B$  is defined

$$\mathbf{d}_J(A, B) = 1 - \text{JS}(A, B) = 1 - \frac{|A \cap B|}{|A \cup B|}.$$

We can see it is a metric. (M1) holds since the intersection size cannot exceed the union size. (M2) holds since  $A \cap A = A \cup A = A$ , and if  $A \neq B$ , then  $A \cap B \subset A \cup B$ . (M3) since  $\cap$  and  $\cup$  operations are symmetric. (M4) requires a bit more effort to show  $\mathbf{d}_J(A, B) \leq D_J(A, C) + D_J(C, B)$ . But it is true.

## 7.2.3 Cosine Distance

This measures the cosine of the “angle” between vectors  $a = (a_1, a_2, \dots, a_d)$  and  $b = (b_1, b_2, \dots, b_d)$  in  $\mathbb{R}^d$

$$\mathbf{d}_{\cos}(a, b) = 1 - \frac{\langle a, b \rangle}{\|a\| \|b\|} = 1 - \frac{\sum_{i=1}^d a_i b_i}{\|a\| \|b\|}.$$

Note that  $\mathbf{d}(A, B) \in [0, \pi]$  and it does not depend on the magnitude  $\|a\|$  of the vectors since this is normalized out. It only cares about their directions. This is useful when a vector of objects represent data sets of different sizes and we want to compare how similar are those distributions, but not their size. This makes it a pseudometric since for two vectors  $a$  and  $a' = (2a_1, 2a_2, \dots, 2a_d)$  where  $\|a'\| = 2\|a\|$  have  $\mathbf{d}_{\cos}(a, a') = 0$ , but they are not equal.

(M1) and (M3) holds by definition. (M4) can be seen by considering the mapping of any vector  $a \in \mathbb{R}^d$  to the  $(d - 1)$ -dimensional sphere  $\mathbb{S}^{d-1}$  as  $a/\|a\|$ . Then the cos distance describes the shortest geodesic distance on this sphere (or the shortest rotation from one to the other).

We can also develop an LSH function  $h$  for  $\mathbf{d}_{\cos}$  as follows. Choose a random vector  $v \in \mathbb{R}^d$ . Then let

$$h_v(a) = \begin{cases} +1 & \text{if } \langle v, a \rangle > 0 \\ -1 & \text{otherwise} \end{cases}.$$

Is sufficient to make  $v \in \{-1, +1\}^d$ . The analysis is similar to for JS but in  $[0, \pi]$  instead of  $[0, 1]$ . It is  $(\gamma, \phi, (\pi - \gamma)/\pi, \phi/\pi)$ -sensitive, for any  $\gamma < \phi \in [0, \pi]$ .

## 7.2.4 Edit Distance

The edit distance considers two strings  $a, b \in \Sigma^d$ , and

$$\mathbf{d}_{\text{ed}}(a, b) = \# \text{ operations to make } a = b,$$

where an operation can **delete** a letter or **insert** a letter. Often  $\Sigma$  is the *alphabet* =  $\{a, b, \dots, z\}$ .

Lets see an example with  $a = \text{mines}$  and  $b = \text{smiles}$ . Here  $\mathbf{d}_{\text{ed}}(a, b) = 3$ .

```
mines
1 : smines insert s
2 : smies delete n
3 : smiles insert l
```

There are many alternative variations of operations. **insert** may cost more than **delete**. Or we could have a **replace** operation.

It is a metric. (M1) holds since the number of edits is always non-negative. (M2) There are no edits only if they are the same. (M3) the operations can be reversed. (M4) If  $c$  is an intermediate “word” then the  $\mathbf{d}_{\text{ed}}(a, c) + \mathbf{d}_{\text{ed}}(c, b) = \mathbf{d}_{\text{ed}}(a, b)$ , otherwise it requires more edits.

Is this good for large text documents? Not really. It is slow. And removing one sentence can cause a large edit distance without changing meaning. But this *is* good for small strings. Some version used in most spelling recommendation systems (e.g. Google’s auto-correct). Its a good guide that usually  $\mathbf{d}_{\text{ed}}(a, b) > 3$  is pretty large since, e.g.,  $\mathbf{d}_{\text{ed}}(\text{cart}, \text{score}) = 4$ .

There is a lot of work to approximate  $\mathbf{d}_{\text{ed}}$  by some sort of  $L_1$  distance so that it can be used in an LSH scheme. But as of now, there is not a good approximation, and this is hard to use with LSH (so its hard to find all close pairs quickly).

## 7.2.5 Graph Distance

Another important type of distance is the hop distance on a graph. A graph is a structure we will visit in more details later on. Consider a series of vertices  $V = \{v_1, v_2, \dots, v_n\}$  and a series of edges  $E = \{e_1, e_2, \dots, e_m\}$ . Each edge  $e = \{v_i, v_j\}$  is a pair of vertices. Here consider only unordered pairs (so the graph is not directed). The set  $G = (V, E)$  defines a graph.

Now the distance  $\mathbf{d}_G$  between two vertices  $v_i, v_j \in V$  in a graph  $G$ , is the fewest number of edges needed so there is a path  $\langle v_i, v_1, v_2, \dots, v_{k-1}, v_j \rangle$  so every consecutive pair  $\{v_\ell, v_{\ell+1}\} \in E$ , where  $v_i$  corresponds with  $v_\ell$  with  $\ell = 0$  and  $v_j$  corresponds with  $v_\ell$  where  $\ell = k$ . So here the length of the path is  $k$ , and if this is the shortest such path, then the length  $\mathbf{d}_G(v_i, v_j) = k$ .

The hop distance in a graph is a metric. Its clearly non-negative (M1), is only 0 if  $v_i = v_j$  (M2), and can be reversed (M3). To see the triangle inequality, assume that otherwise there is a node  $c \in V$  such that  $\mathbf{d}_G(v_i, c) + \mathbf{d}_G(c, v_j) < \mathbf{d}_G(v_i, v_j)$ , then we could instead create a path from  $v_i$  to  $v_j$  that went though  $c$ , and by transitivity, in the above equation the left-hand-side must be at least as large as the right-hand-side.