Locality Sensitive hashing (LSH)

Classical hashing

- if $x = y$ then $h(x) = h(y)$
- if $x \neq y$ then $h(x) \neq h(y)$

U : Universe. T : hash table

$|T| \ll |U| \qquad h : U \to [0, |T|-1]$

$$\mathcal{H} = \{ h : U \to [0, |T|-1] \}$$

Universal hashing

Collisions are rare as possible

$\forall x, y \in U, \; x \neq y,$

$$\Pr_{h \in \mathcal{H}} [h(x) = h(y)] = \frac{1}{|T|}$$

Locality Sensitive hashing

→ Collision between similar elements

→ $\forall\, x, y \in \mathcal{U}$

$$Ed(x,y) \leq d_1 \Rightarrow \Pr_{h \in \mathcal{H}} [h(x) = h(y)] \geq P_1$$

$$Ed(x,y) \geq d_2 \Rightarrow \Pr_{h \in \mathcal{H}} [h(x) = h(y)] \leq P_2$$

→ A family of hash functions where similar elements are more likely to have the same value than distant elements.

→ Low distance ⟺ high collisions
high distance ⟺ low collisions

In between d_1 d_2 ⇒ No guarantee

☆ Probability over choice of $h \in \mathcal{H}$ not over the elements x, y

→ if $d_1 = d_2$ → Ungapped LSH

Notion of similarity: Jaccard

let's take two sets S_1 & S_2

$$J(S_1, S_2) = \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|}$$

$$= \frac{\text{cardinality of intersection}}{\text{cardinality of union}}$$

Example:-

$S_1 = \{3, 10, 15, 19\}$ $\quad$ $S_2 = \{4, 10, 15\}$

$$J(S_1, S_2) = \frac{\{10, 15\}}{\{3, 4, 10, 15, 19\}}$$

$$= \frac{2}{5}$$

$$\boxed{\text{Weighted Jaccard :- } \frac{\sum_K \min(S_1^i, S_2^i)}{\sum_K \max(S_1^i, S_2^i)}}$$

Minwise Hashing: Andrei Broder

Let us take a random universal hash f^{tn}.

U_i : strings $\longrightarrow N$

→ We take the minimum hash value.
→ Everytime we want to generate a new minhash value, we'll generate a new universal hash f^{tn}. & compute minimum.

→ **Properties:**

For any set of objects S_1 & S_2. Probability of hash collision is exactly equal to Jaccard index.

$$Pr(\text{minhash}(S_1) = \text{Minhash}(S_2)) = \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|}$$

→ If similarity between the two sets increases then the probability of collision also increases

→ Unbiased estimator → Ungapped LSH.

Intuition:-

- Take two sets S_1 & S_2.
- Hash every element of $S_1 \cup S_2$ and find the minimum hash value.
- The chance of having the same minimum value after this permutation is equal to the number of common element proportional to the union.

- If the minimum belongs to $S_1 \cap S_2$, then minhash of both the sets will be the same.

$$S_1 = \{3, 10, 15, 19\} \qquad S_2 = \{4, 10, 15\}$$

$$J(S_1, S_2) = \frac{\{10, 15\}}{\{3, 4, 10, 15, 19\}}$$

$$= \frac{2}{5}$$

Q: how can we estimate Jaccard if we have 50 minhashes to S_1 & S_2?

→ Instead of dealing with large sets, which requires a lot of computing time & memory, MinHash can provide a sketch to approximate this measure in a scalable way by computing a small fixed sized sketch which represents each large set.

Example:-

Using the sketching technique, to estimate how many collisions out of 50 there are:

$$\text{Variance} = \frac{J(1-J)}{\text{size of sketch}}$$

from Beronoulli distribution

if $J = 0.8$, $|\text{sketch}| = 50$ Error $= \sqrt{\frac{0.16}{50}} \approx 0.05$

Parity of Minhash :-

We only store the parity of minhash

$$P(\text{Parity}(M(S_1)) = \text{Parity}(M(S_2)))$$

$$= \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|} + \left(1 - \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|}\right) \times 0.5$$

$$= 0.5 \times \left(1 + \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|}\right)$$

$$\text{Variance}' = \frac{P'(1-P')}{\text{Size of sketch}}$$

$$= \frac{0.5(1+J)(1 - 0.5(1+J))}{\text{size of sketch}}$$

Extension of Minwise hashing.

One-Permutation hashing. Li et al 2012

π_i : strings $\rightarrow N$

→ divide the space $[0, N]$ into K bins and take the minimon of each

→ This methodology allows sampling with $1/K$ as much pre-processing cost as the original min-wise hashing.

→ What if a bin is empty?
Shrivastav & Li 2014

→ introduce a "rotation" scheme to assign values to empty bins

→ borrows value from the closest non-empty bin in clockwise direction added with offset C.