More LSH

Application: Entity Resolution
Application: Similar News Articles
Distance Measures
LSH for Cosine Distance
LS Families of Hash Functions

Jeffrey D. Ullman
Stanford University

Entity Resolution

Similarity of Records
A Simple Bucketing Process
Validating the Results

Entity Resolution

- The *entity-resolution* problem is to examine a collection of records and determine which refer to the same entity.
    - *Entities* could be people, events, etc.
- Typically, we want to merge records if their values in corresponding fields are similar.

Matching Customer Records

- I once took a consulting job solving the following problem:

  - Company A agreed to solicit customers for Company B, for a fee.

  - They then argued over how many customers.

  - Neither recorded exactly which customers were involved.

Customer Records – (2)

- Each company had about 1 million records describing customers that might have been sent from A to B.
- Records had name, address, and phone, but for various reasons, they could be different for the same person.
  - E.g., misspellings, but there are many sources of error.

Customer Records – (3)

- **Problem**: (1 million)^2 is too many pairs of records to score.
- **Solution**: A simple LSH.
  - Three hash functions: exact values of name, address, phone.
    - Compare iff records are identical in at least one.
  - Misses similar records with a small differences in all three fields.

Customer Records – (4)

- Design a measure ("*score* ") of how similar records are:
  - E.g., deduct points for small misspellings ("Jeffrey" vs. "Jeffery") or same phone with different area code.
- Score all pairs of records that the LSH scheme identified as candidates; report high scores as matches.

Aside: Hashing Names, Etc.

- Question for Thought: How do we hash strings such as names so there is one bucket for each string?
- Answer: Sort the strings instead.

Aside: Validation of Results

- We were able to tell what values of the scoring function were reliable in an interesting way.
- Identical records had an average creation-date difference of 10 days.
- We only looked for records created within 90 days of each other, so bogus matches had a 45-day average difference in creation dates.

Validation – (2)

- By looking at the pool of matches with a fixed score, we could compute the average time-difference, say x, and deduce that fraction (45-x)/35 of them were valid matches.
- Alas, the lawyers didn't think the jury would understand.

Validation – Generalized

- Any field not used in the LSH could have been used to validate, provided corresponding values were closer for true matches than false.
- Example: if records had a height field, we would expect true matches to be close, false matches to have the average difference for random people.

Similar News Articles

A New Way of Shingling
Bucketing by Length

Application: Same News Article

- The Political-Science Dept. at Stanford asked a team from CS to help them with the problem of identifying duplicate, on-line news articles.
- Problem: the same article, say from the Associated Press, appears on the Web site of many newspapers, but looks quite different.

News Articles – (2)

- Each newspaper surrounds the text of the article with:
  - It's own logo and text.
  - Ads.
  - Perhaps links to other articles.
- A newspaper may also "crop" the article (delete parts).

News Articles – (3)

- The team came up with its own solution, that included shingling, but not minhashing or LSH.

  - A special way of shingling that appears quite good for this application.

  - LSH substitute: candidates are articles of similar length.

Enter LSH

- I told them the story of minhashing + LSH.
- They implemented it and found it faster for similarities below 80%.
  - Aside: That's no surprise. When the similarity threshold is high, there are better methods – see Sect. 3.9 of MMDS.

Enter LSH – (2)

- Their first attempt at minhashing was very inefficient.
- They were unaware of the importance of doing the minhashing row-by-row.
- Since their data was column-by-column, they needed to sort once before minhashing.

Specialized Shingling Technique

- The team observed that news articles have a lot of *stop words*, while ads do not.
  - "Buy Sudzo" vs. "I recommend that you buy Sudzo for your laundry."
- They defined a *shingle* to be a stop word and the next two following words.

Why it Works

- By requiring each shingle to have a stop word, they biased the mapping from documents to shingles so it picked more shingles from the article than from the ads.
- Pages with the same article, but different ads, have higher Jaccard similarity than those with the same ads, different articles.

Distance Measures

Triangle Inequality
Euclidean Distance
Cosine Distance
Jaccard Distance
Edit Distance

Distance Measures

- Generalized LSH is based on some kind of "distance" between points.

    - Similar points are "close."

- Example: Jaccard similarity is not a distance; 1 minus Jaccard similarity is.

Axioms of a Distance Measure

- *d* is a *distance measure* if it is a function from pairs of points to real numbers such that:

1. $d(x,y) \geq 0$.
2. $d(x,y) = 0$ iff $x = y$.
3. $d(x,y) = d(y,x)$.
4. $d(x,y) \leq d(x,z) + d(z,y)$ (*triangle inequality*).

Some Euclidean Distances

- L_2 *norm*: d(x,y) = square root of the sum of the squares of the differences between *x* and *y* in each dimension.
  - The most common notion of "distance."
- L_1 *norm*: sum of the differences in each dimension.
  - *Manhattan distance* = distance if you had to travel along coordinates only.

b = (9,8)

L_2-norm:
dist(a,b) =
$\sqrt{(4^2+3^2)}$
= 5

5   3

4

a = (5,5)

L_1-norm:
dist(a,b) =
4+3 = 7
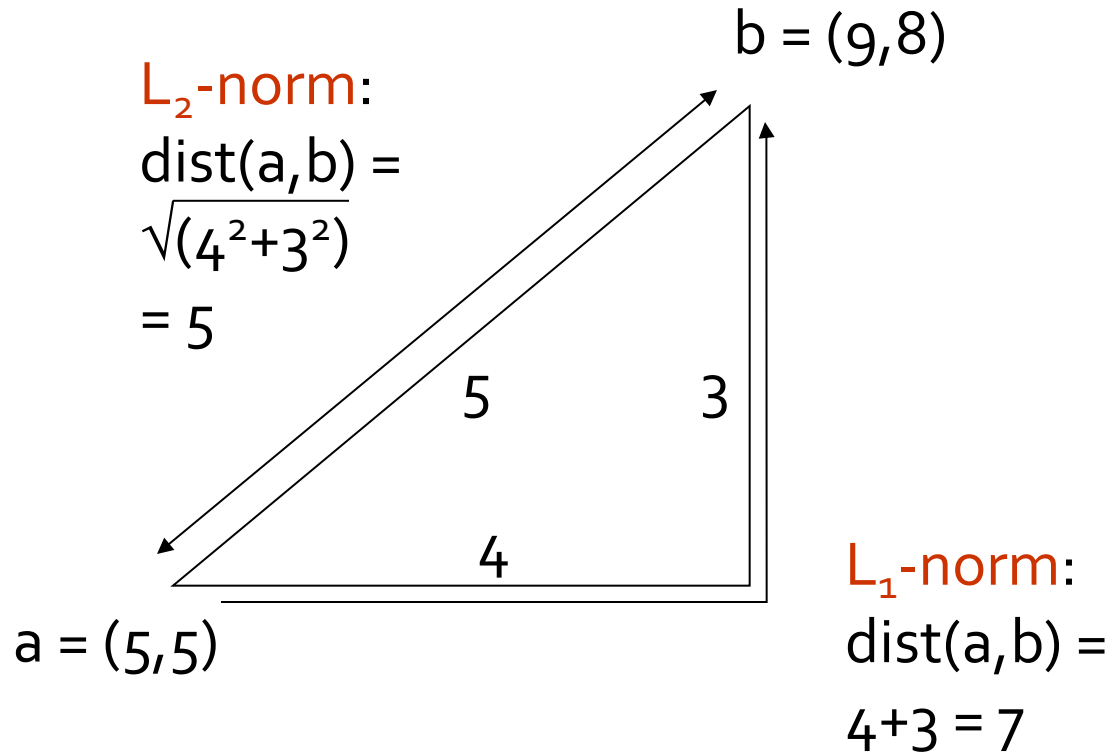
Question For Thought

- People have defined L_r norms for any r, even fractional r.

  - *L_r norm*: d(x,y) = r-th root of the sum of the r-th powers of the differences between *x* and *y* in each dimension.

- What do these norms look like as r gets larger?
- What if r approaches 0?

Some Non-Euclidean Distances

- *Jaccard distance* for sets = 1 minus Jaccard similarity.
- *Cosine distance* for vectors = angle between the vectors.
- *Edit distance* for strings = number of inserts and deletes to change one string into another.

Example: Jaccard Distance

- Consider x = {1,2,3,4} and y = {1,3,5}
- Size of intersection = 2; size of union = 5, Jaccard similarity (not distance) = 2/5.
- d(x,y) = 1 − (Jaccard similarity) = 3/5.

Why J.D. Is a Distance Measure

- d(x,y) $\geq$ 0 because $|x \cap y| \leq |x \cup y|$.
  - Thus, similarity $\leq$ 1 and distance = 1 − similarity $\geq$ 0.
- d(x,x) = 0 because $x \cap x = x \cup x$.
- And if x $\neq$ y, then $|x \cap y|$ is strictly less than $|x \cup y|$, so sim(x,y) < 1; thus d(x,y) > 0.
- d(x,y) = d(y,x) because union and intersection are symmetric.
- d(x,y) $\leq$ d(x,z) + d(z,y) trickier − next slide.

Triangle Inequality for J.D.

$$d(x,z) \qquad d(z,y) \qquad d(x,y)$$

$$1 - \frac{|x \cap z|}{|x \cup z|} + 1 - \frac{|y \cap z|}{|y \cup z|} \geq 1 - \frac{|x \cap y|}{|x \cup y|}$$

- Remember: $|a \cap b| / |a \cup b|$ = probability that minhash(a) = minhash(b).

- Thus, $1 - |a \cap b| / |a \cup b|$ = probability that minhash(a) $\neq$ minhash(b).

- Need to show: prob[minhash(x) $\neq$ minhash(y)] $\leq$ prob[minhash(x) $\neq$ minhash(z)] + prob[minhash(z) $\neq$ minhash(y)]

Proof

- Whenever minhash(x) ≠ minhash(y), at least one of minhash(x) ≠ minhash(z) and minhash(z) ≠ minhash(y) must be true.

minhash(x) ≠ minhash(y

minhash(x) ≠ minhash(z)   minhash(z) ≠ minhash(y)
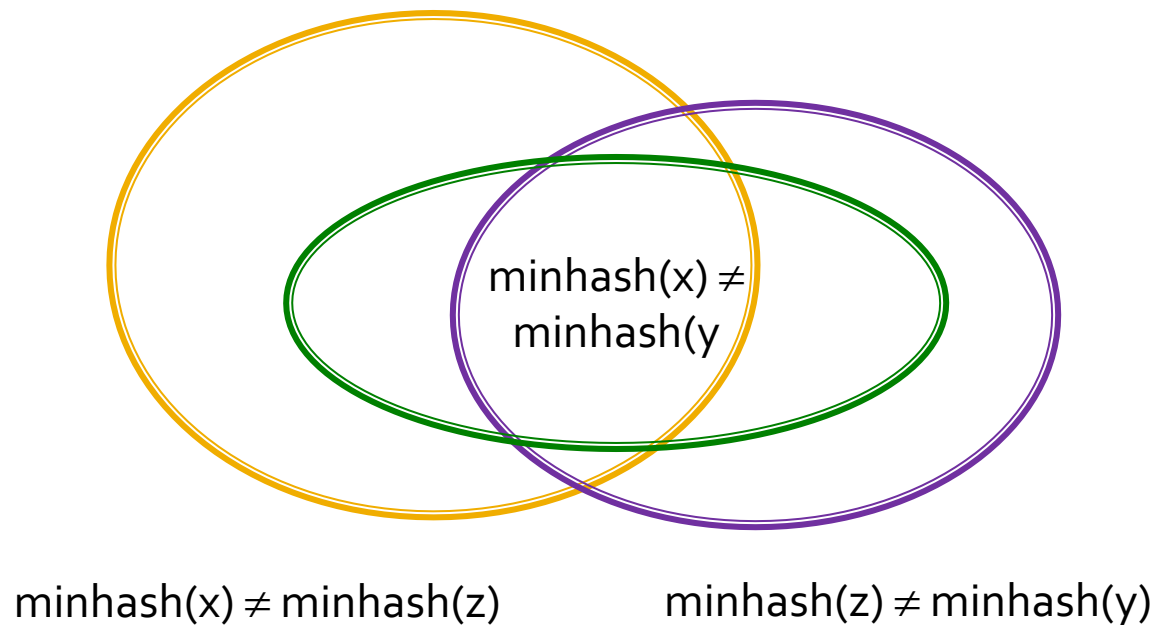
Cosine Distance

- Think of a point as a vector from the origin [0,0,…,0] to its location.
- Two points' vectors make an angle, whose cosine is the normalized dot-product of the vectors: $p_1.p_2/|p_2||p_1|$.
  - Example: $p_1 = [1,0,2,-2,0]$; $p_2 = [0,0,3,0,0]$.
  - $p_1.p_2 = 6$; $|p_1| = |p_2| = \sqrt{9} = 3$.
  - $\cos(\theta) = 6/9$; θ is about 48 degrees.

Edit Distance

- The *edit distance* of two strings is the number of inserts and deletes of characters needed to turn one into the other.
- An equivalent definition: d(x,y) = |x| + |y| - 2|LCS(x,y)|.
  - LCS = *longest common subsequence* = any longest string obtained both by deleting from *x* and deleting from *y*.

Example: Edit Distance

- *x = abcde* ; *y = bcduve*.
- Turn *x* into *y* by deleting *a*, then inserting *u* and *v* after *d*.
  - Edit distance = 3.
- Or, computing edit distance through the LCS, note that LCS(x,y) = *bcde*.
- Then: $|x| + |y| - 2|LCS(x,y)| = 5 + 6 - 2*4 = 3$ = edit distance.
- Question for thought: An example of two strings with two different LCS's?
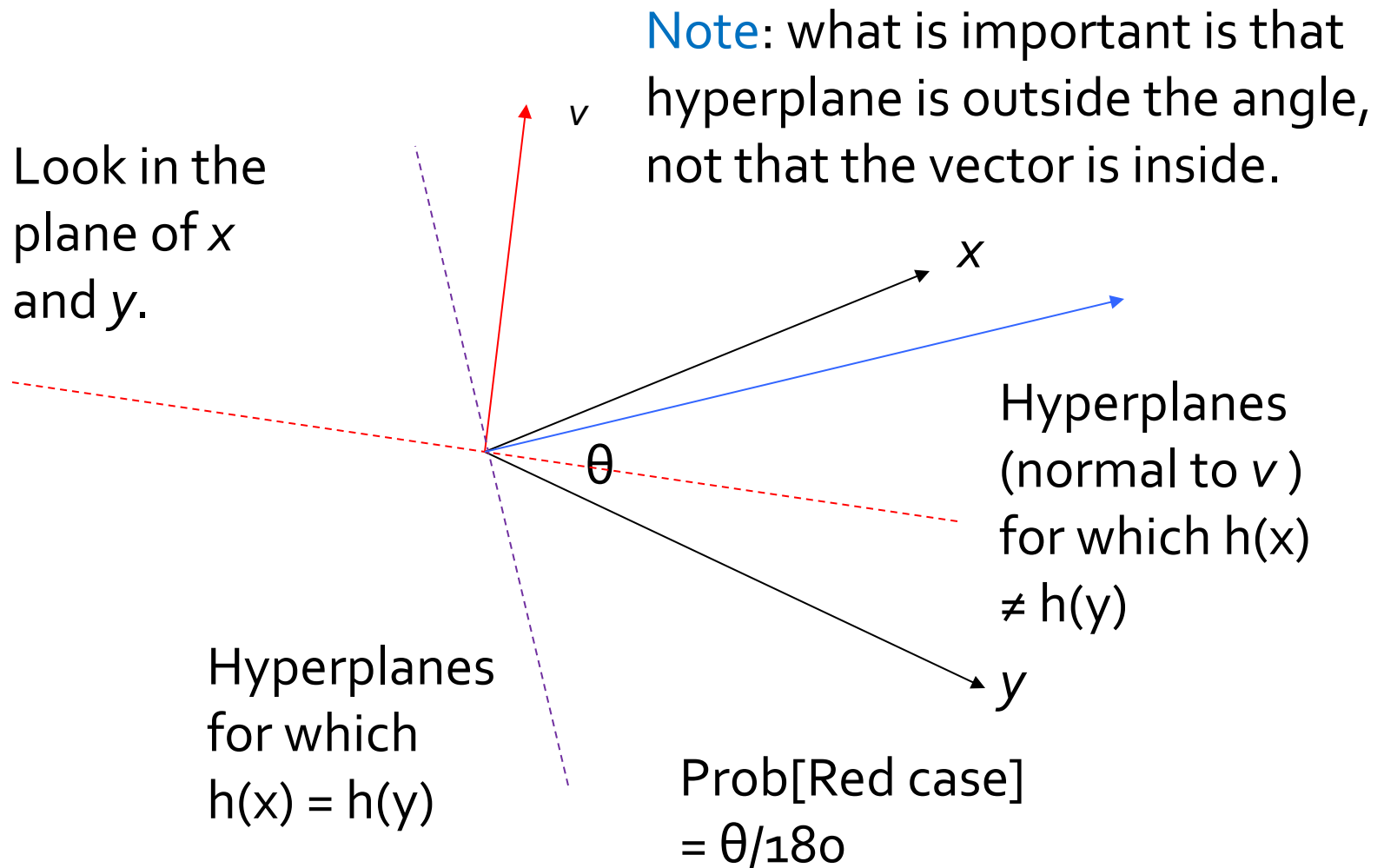 - Hint: let one string be ab.

33

An LSH Family for Cosine Distance

Random Hyperplanes
Sketches (Signatures)

Random Hyperplanes

- For cosine distance, there is a technique analogous to minhashing for generating signatures, called *random hyperplanes*.
- Each vector *v* determines a hash function h_v with two buckets.
- $h_v(x)$ = +1 if v.x > 0; $h_v(x)$ = -1 if v.x < 0.
- Instead of minhash functions, use the set of all functions derived from any vector v.
- Claim: Prob[h(x)=h(y)] = 1 − (angle between *x* and *y* divided by 180).

Proof of Claim

Note: what is important is that hyperplane is outside the angle, not that the vector is inside.

Look in the plane of *x* and *y*.

v

x

Hyperplanes (normal to *v*) for which h(x) ≠ h(y)

θ

y

Hyperplanes for which h(x) = h(y)

Prob[Red case] = θ/180

Signatures for Cosine Distance

- Pick some number of vectors, and hash your data for each vector.
- The result is a signature (*sketch*) of +1's and −1's that can be used for LSH like the minhash signatures for Jaccard distance.

Simplification

- We need not pick from among all possible vectors *v* to form a component of a sketch.
- It suffices to consider only vectors *v* consisting of +1 and −1 components.

LSH Families of Hash Functions

Definition

**Combining hash functions**

Making steep S-Curves

Hash Functions Decide Equality

- There is a subtlety about what a "hash function" is, in the context of LSH families.
- A hash function h really takes two elements x and y, and returns a decision whether x and y are candidates for comparison.
- Example: The family of minhash functions. Each minhash function computes minhash values and says "yes" for two sets/columns iff those values are the same.
- Shorthand: "$h(x) = h(y)$" means h says "yes" for pair of elements x and y.

LSH Families Defined

- Suppose we have a space S of points with a distance measure d.

- A family **H** of hash functions is said to be (d_1,d_2,p_1,p_2)-*sensitive* if for any x and y in S:

  1. If $d(x,y) \leq d_1$, then the probability over all h in **H**, that $h(x) = h(y)$ is at least p_1.

  2. If $d(x,y) \geq d_2$, then the probability over all h in **H**, that $h(x) = h(y)$ is at most p_2.

LS Families: Illustration

p_1

High
probability;
at least p_1

???

Low
probability;
at most p_2

p_2

d_1 d_2
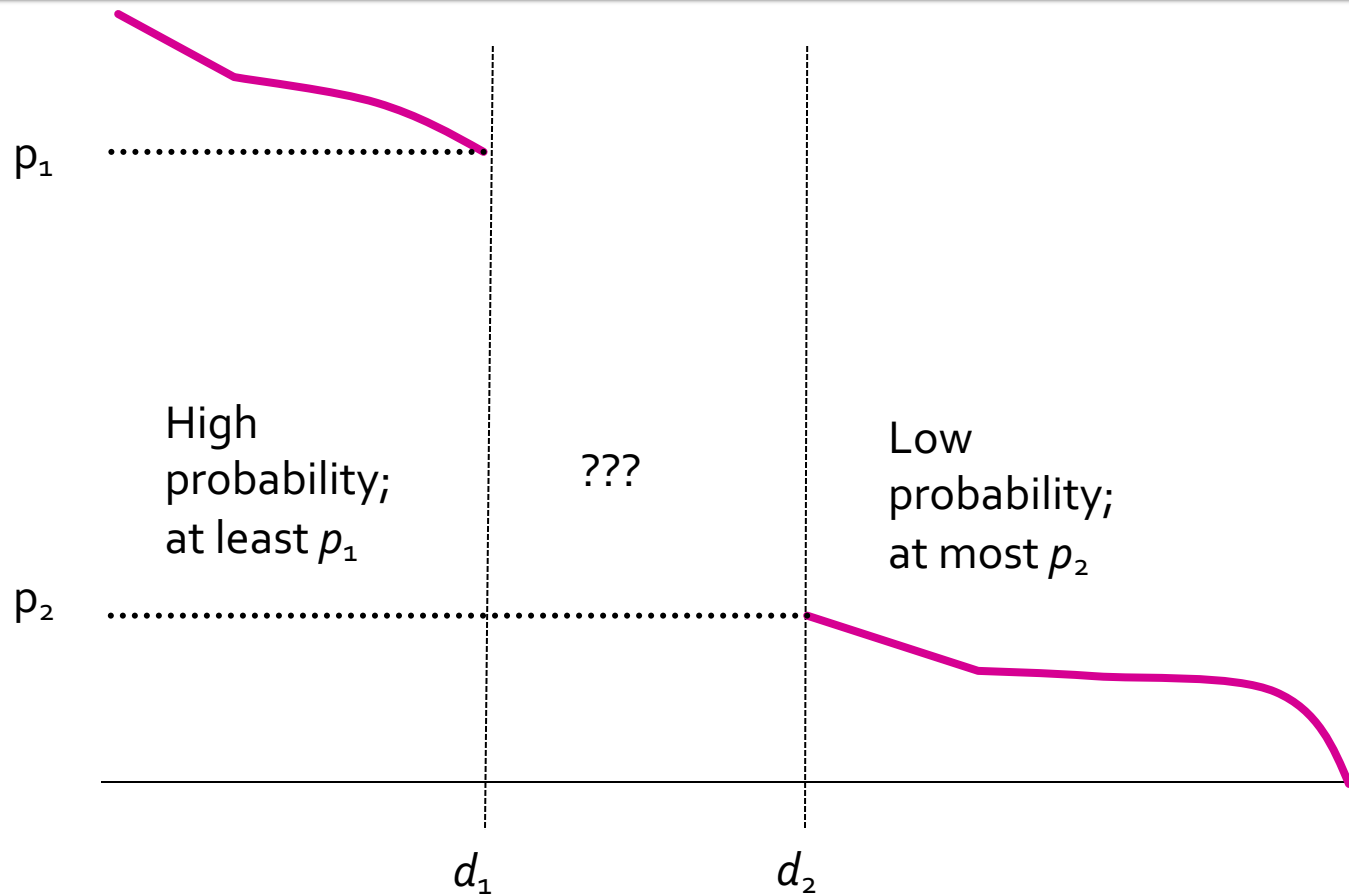
Example: LS Family

- Let:
  - *S* = subsets of some universal set,
  - *d* = Jaccard distance,
  - **H** formed from the minhash functions for all permutations of the universal set.
- Then Prob[h(x)=h(y)] = 1-d(x,y).
  - Restates theorem about Jaccard similarity and minhashing in terms of Jaccard distance.

Example: LS Family – (2)

- Claim: **H** is a (1/3, 3/4, 2/3, 1/4)-sensitive family for *S* and *d*.

If distance $\geq$ 3/4
(so similarity $\leq$ 1/4)

Then probability
that minhash values
agree is $\leq$ 1/4

If distance $\leq$ 1/3
(so similarity $\geq$ 2/3)

Then probability
that minhash values
agree is $\geq$ 2/3

For Jaccard similarity, minhashing gives us a
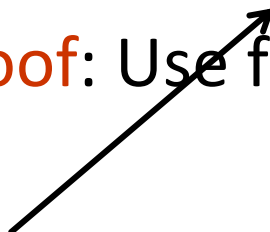$(d_1, d_2, (1-d_1), (1-d_2))$-sensitive family for any $d_1 < d_2$.

Amplifying an LSH-Family

- The "bands" technique we learned for signature matrices carries over to this more general setting.
  - Goal: the "S-curve" effect seen there.
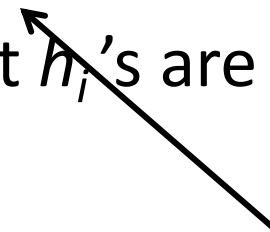- AND construction like "rows in a band."
- OR construction like "many bands."

AND of Hash Functions

- Given family **H**, construct family **H'** whose members each consist of *r* functions from **H**.
- For $h = \{h_1,...,h_r\}$ in **H'**, h(x)=h(y) if and only if $h_i(x)=h_i(y)$ for all *i*.
- Theorem: If **H** is (d_1,d_2,p_1,p_2)-sensitive, then **H'** is $(d_1,d_2,(p_1)^r,(p_2)^r)$-sensitive.
  - Proof: Use fact that h_i's are independent.

Also lowers probability for small distances (Bad)

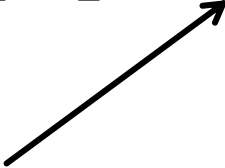Lowers probability for large distances (Good)

OR of Hash Functions

- Given family **H**, construct family **H'** whose members each consist of b functions from **H**.
- For $h = \{h_1, \ldots, h_b\}$ in **H'**, h(x)=h(y) if and only if $h_i(x)=h_i(y)$ for some i.
- Theorem: If **H** is (d_1, d_2, p_1, p_2)-sensitive, then **H'** is $(d_1, d_2, 1-(1-p_1)^b, 1-(1-p_2)^b)$-sensitive.

Raises probability for small distances (Good)

Raises probability for large distances (Bad)

Combine AND and OR Constructions

- By choosing *b* and *r* correctly, we can make the lower probability approach 0 while the higher approaches 1.
- As for the signature matrix, we can use the AND construction followed by the OR construction.
  - Or vice-versa.
  - Or any sequence of AND's and OR's alternating.

AND-OR Composition

- Each of the two probabilities p is transformed into $1-(1-p^r)^b$.
  - The "S-curve" studied before.
- Example: Take **H** and construct **H'** by the AND construction with $r = 4$. Then, from **H'**, construct **H''** by the OR construction with $b = 4$.

Table for Function $1-(1-p^4)^4$

p	$1-(1-p^4)^4$
.2	.0064
.3	.0320
.4	.0985
.5	.2275
.6	.4260
.7	.6666
.8	.8785
.9	.9860

Example: Transforms a (.2,.8,.8,.2)-sensitive family into a (.2,.8,.8785,.0064)-sensitive family.

OR-AND Composition

- Each of the two probabilities *p* is transformed into $(1-(1-p)^b)^r$.

  - The same S-curve, mirrored horizontally and vertically.

- Example: Take **H** and construct **H'** by the OR construction with *b* = 4. Then, from **H'**, construct **H''** by the AND construction with *r* = 4.

Table for Function $(1-(1-p)^4)^4$

p	$(1-(1-p)^4)^4$
.1	.0140
.2	.1215
.3	.3334
.4	.5740
.5	.7725
.6	.9015
.7	.9680
.8	.9936

Example: Transforms a (.2,.8,.8,.2)-sensitive family into a (.2,.8,.9936,.1215)-sensitive family.

Cascading Constructions

- Example: Apply the (4,4) OR-AND construction followed by the (4,4) AND-OR construction.
- Transforms a (.2,.8,.8,.2)-sensitive family into a (.2,.8,.9999996,.0008715)-sensitive family.
- Requires signatures of length 256.

General Use of S-Curves

- For each AND-OR S-curve $1-(1-p^r)^b$, there is a *threshold* t, for which $1-(1-t^r)^b = t$.
- Above t, high probabilities are increased; below t, low probabilities are decreased.
- You improve the sensitivity as long as the low probability is less than t, and the high probability is greater than t.
  - Iterate as you like.
- Similar observation for the OR-AND type of S-curve: $(1-(1-p)^b)^r$.

Visualization of Threshold

Probability
Is raised

Threshold
t

Probability
Is lowered

p ⟶   t