Who is your neighbor?

--A brief introduction of Local Sensitive Hash and MinHash

-- Xinzuo Wang

An Outline:

- Near-optimal hashing algorithm for approximate near(est) neighbor problem

  --Local sensitive hash

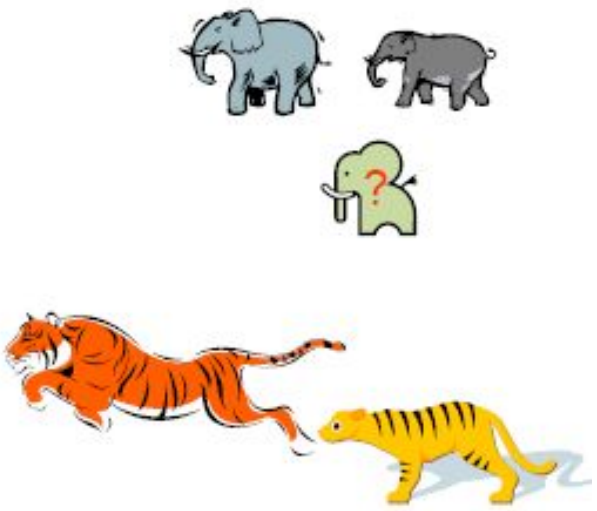
- Finding similarity items from high-dimensional objects

  -- MinHash

Background

Nearest Neighbor: Motivation

- Learning: nearest neighbor rule
- Database retrieval
- Vector quantization,
  also known as compression



Background

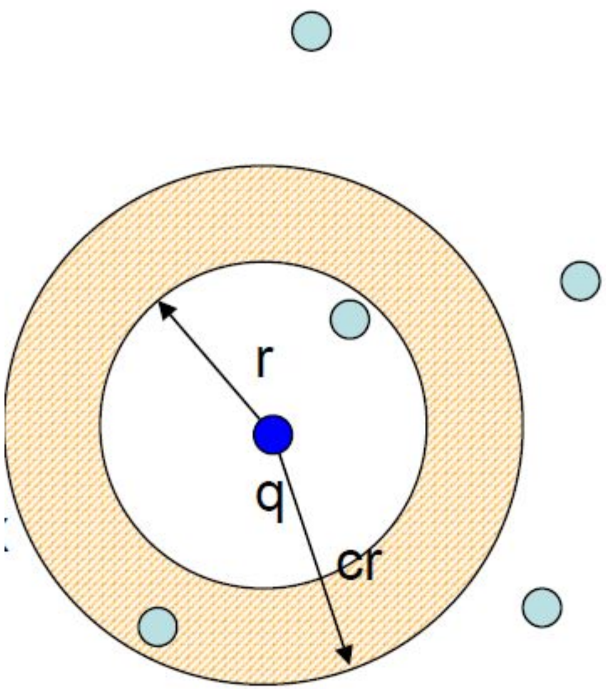
Challenge : Curse of dimensionality

-- Difficult to compute large data set with large dimensions using original methods.

Approximate Near Neighbor

Approximate Near Neighbor :

- c-Approximate r-Near Neighbor: build data structure which, for any query q:
  - If there is a point $p \in P$, $\|p-q\| \leq r$
  - it returns $p' \in P$, $\| p' - q\| \leq cr$

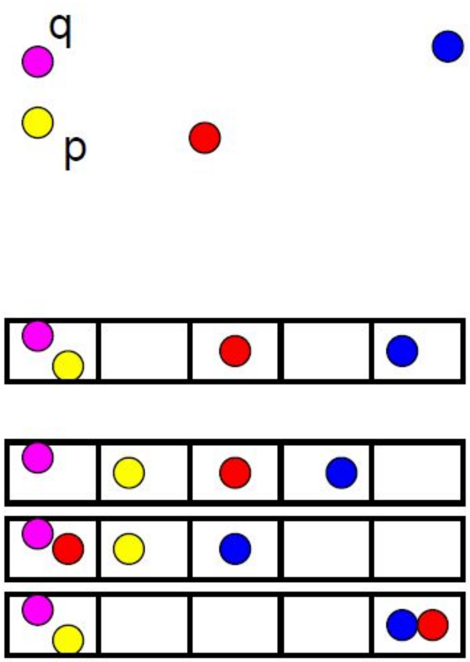


Here, ***Local Sensitive Hash*** is employed.

Local Sensitive Hash :

-- *Transfer high dimensional data onto lower dimensions and preserve their 'distance'.*

- Idea: construct hash functions g: Rd → U such that for any points p,q:
  - If $||p\text{-}q|| \le r$, then P[g(p)=g(q)] is "high"
  - If $||p\text{-}q|| > cr$, then P[g(p)=g(q)] is "small"
- LSH based on hamming norm.
- LSH based on projection.



LSH on Hamming Norm:

Problem define :

Given a set P, return a point p $\in$ P, for any query point q that $d(p, q) < (1+\varepsilon)d'(p, q)$, where $d'(p, q)$ is the smallest distance from q to p

LSH on Hamming Norm:

Hash functions:

h(p)= p_i, i.e., the i-th bit of p

For example :

p = 101101010101

if we define i=3, h(p) = 1

LSH on Hamming Norm:

The algorithm :

We use functions of the form $g(p)=<h1(p), h2(p),\ldots, hk(p)>$

- Preprocessing:
  - Select g1…gL
  - For all $p \in P$, hash p to buckets g1(p)…gL(p)

- Query:
  - Retrieve the points from buckets g1(q), g2(q), … , until

    Either the points from all L buckets have been retrieved, or

    Total number of points retrieved exceeds 2L
  - Answer the query based on the retrieved points

LSH on Hamming Norm:

- Some detailed analysis :
  - A crucial point : How to choose k and L ?

Intuitively, when k decreases, the precise of hash will increase;

when L decreases, some points may be ‘unfortunately’ missed.

LSH on Hamming Norm:

– If $\|p-q\| \leq r1$, then P{g(p)=g(q)} >= p1
– If $\|p-q\| > r2$, then P{g(p)=g(q)} <= p2 where (p1>p2)

Then we call the hash function is is (r1, r2, p1, p2) sensitive

$\Rightarrow$ h(p) is $(r, (1+\varepsilon)r, 1-\frac{r}{d}, \frac{r(1+\varepsilon)}{d})$ sensitive

Thus, by correctly choose k and L, we can hash the point in $(1+\varepsilon)d'(p, q)$ to the same bucket with q with the probability more than p1

It can be proved that the optimal value of k is $log_{\frac{1}{p_2}}(\frac{n}{B})$ and L is $(\frac{n}{B})^{\rho}$ where $\rho = \frac{\ln(1/p_1)}{\ln(1/p_2)}$ and B is the max number of buckets.

LSH on p-Norm – Based on projection

- Stable distribution :

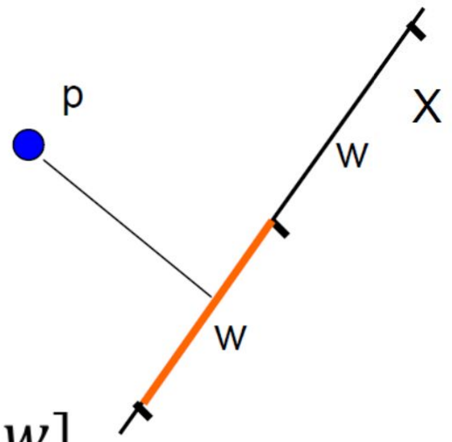
A stable distribution is called s-stable, if there exists p ≥ 0 such that for any real numbers $v_1, v_2, v_3 \dots v_n$ and i.d.d. variables $X_1, X_2, X_3 \dots X_n$ with distribution D, the random variable $(\sum_i |v_i|^p)^{1/p} X$, where X is a random variable with distribution D.

➡ One of the 2-stable distribution is the Normal distribution.

LSH on p-Norm – Based on projection

- Hash function :

$$h_{\bar{a},b}(\bar{v}) = \left\lfloor \frac{\vec{a} * \bar{v} + b}{w} \right\rfloor \quad (R^d \rightarrow N)$$



b is a real number chosen randomly from [0, w],
$\vec{a}$ is a d-dimension vector with entries chosen independently from a s-stable distribution.

MinHash :

-- Find similar items from high-dimensional objects.

Main idea:

- Use ***Min-Hash Signature*** to preserve the ‘distance’ or ‘similarity’ between objects.
- Use ***LSH*** to find approximate nearest neighbors of each objects from ***Min-Hash Signature***

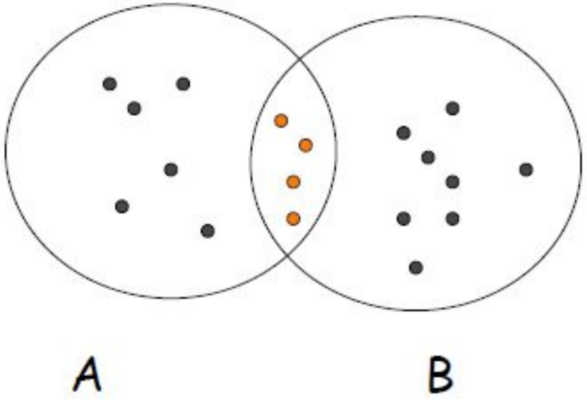
MinHash :

- Define similarity :

Similarity metric, “distance”, for sets

Jaccard similarity:

$$\mathrm{SIM}(A, B) := \frac{A \cap B}{A \cup B}$$



4 common
18 total

SIM(A,B) = 4/18
= 2/9

MinHash :

- Hash functions :
- Minhash(π) of a set is the number of the row (element) with first non-zero in the permuted order π.

Element num	Set1	Set2	Set3	Set4
1	0	0	1	0
4	0	0	1	0
0	1	0	0	1
3	1	0	1	1
2	0	1	0	1
…				

$\Pi=(1,4,0,3,2)$

MinHash -- Min-Hash Signature

SIM(Q, T) = P{h(Q) = h(T)}

How to compute P{h(Q) = h(T)} ?

- MinHash Signature :
  - Let h1, h2, …, hn be different MinHash functions.
  - Then signature for set S is: SIG(S) = [h1(S), h2(S),.., hn(S)]

SIM(Q, T) ≈ the ratio of the ratio of equal elements of SIG(Q) and SIG(T)

An Outline:

- For example : A matrix of four sets with n elements

Element num	Set1	Set2	Set3	Set4	x + 1 mod 5	3x +1 mod 5
0	1	0	0	1	1	1
1	0	0	1	0	2	4
2	0	1	0	1	3	2
3	1	0	1	1	4	0
4	0	0	1	0	0	3
...						

MinHash signature :

	set1	set2	set3	set4
1	1	3	2	1
2	1	3	2	1
3	1	2	4	1

LSH based on MinHash

Mean idea:

- Divide the signature matrix rows into **b** bands of **r** rows
- Hash the columns in each band with a basic hash-function, each band divided to buckets.

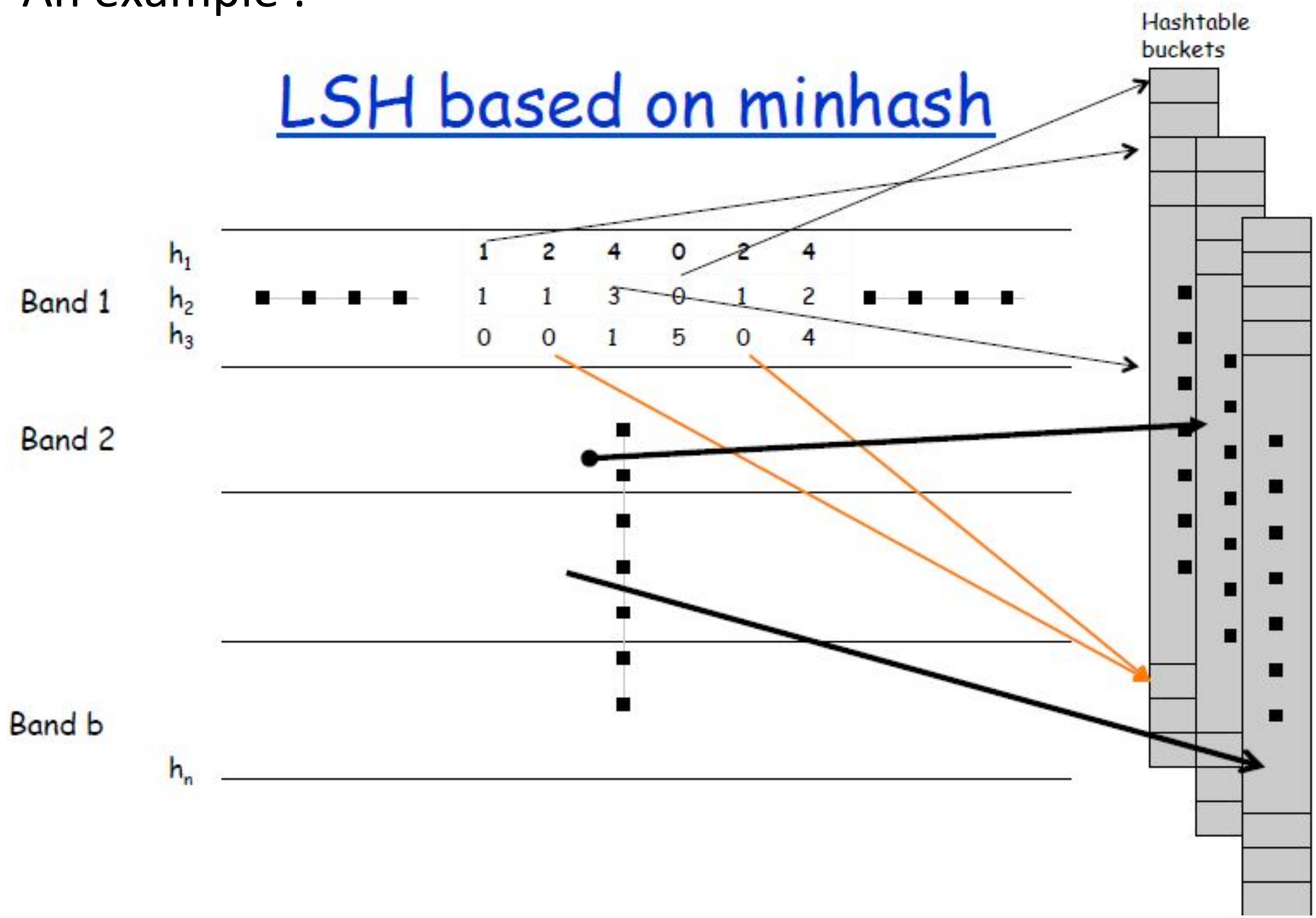
If sets S and T have same values in a band, they will be hashed

into the same bucket in that band.

- For nearest-neighbor, the candidates are the items in the same bucket as query item, in each band.

LSH based on MinHash

- An example :



LSH based on MinHash

- Analysis :

-- What is the probability the S will be chosen in the same bucket with T when given SIM(S, T) = s?

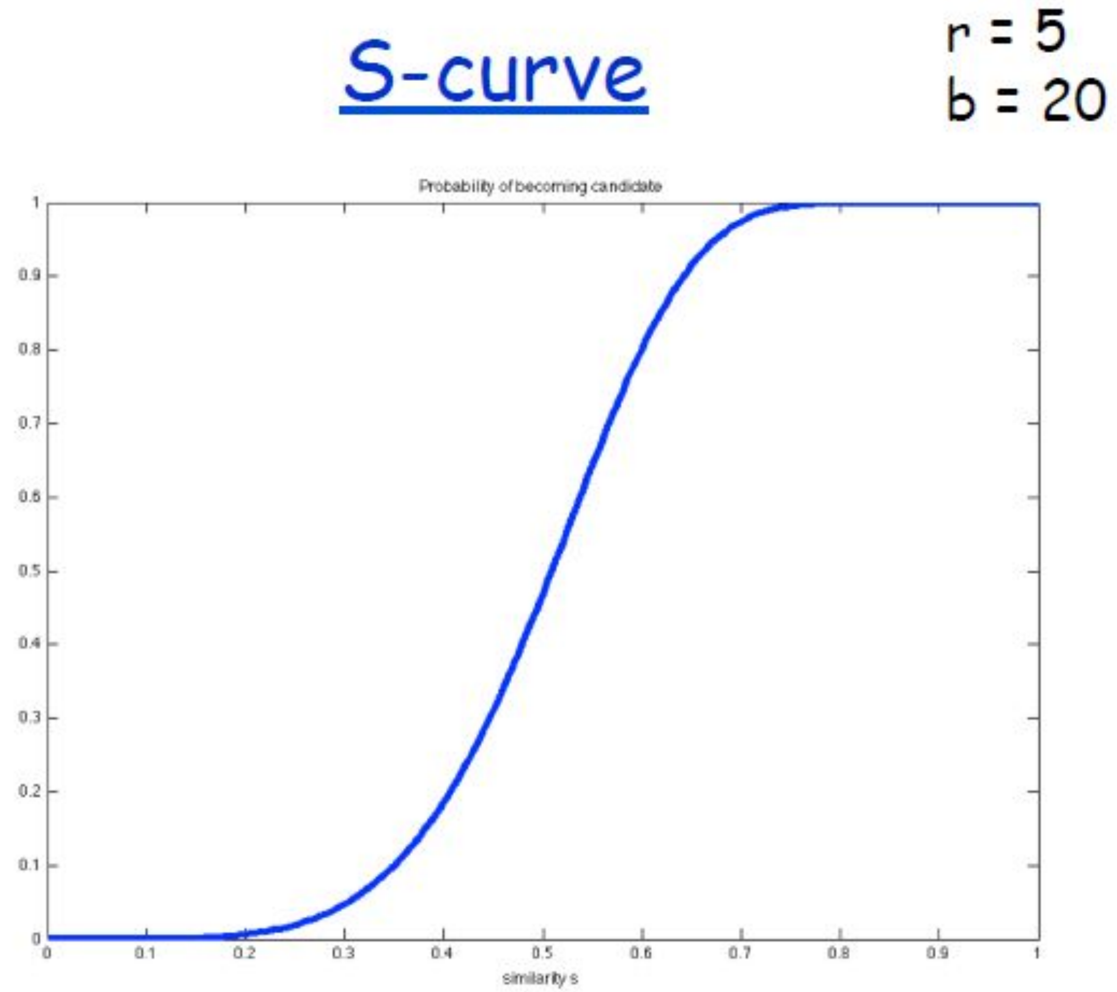
- P{Q and T agree on all rows in a band} $= s^r$
- P{S will be chosen as the candidate}

= P{S and T agree at least on one band}

$= 1 - (1 - s^r)^b$

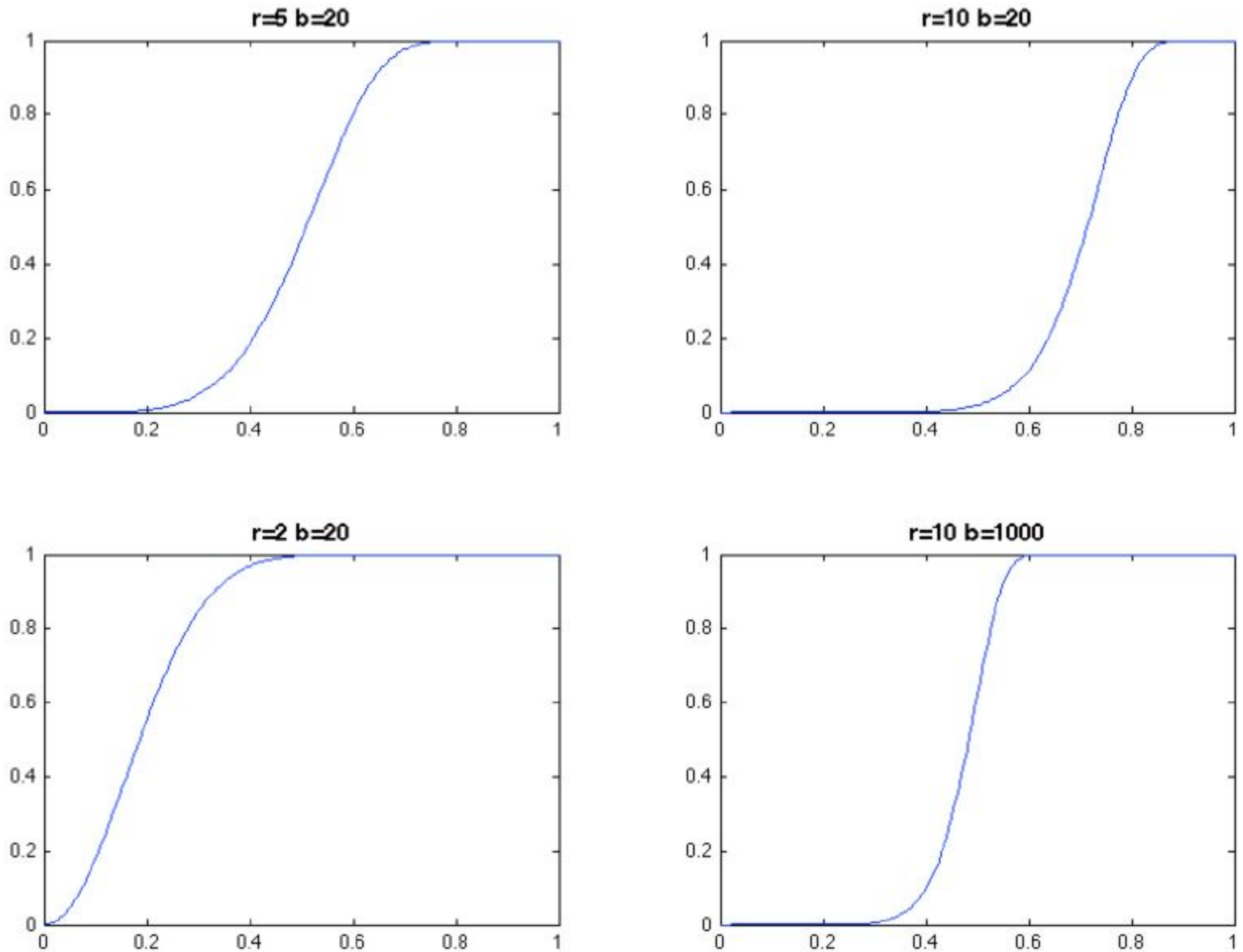
LSH based on MinHash

- Relationship between SIM(S, T) and the probability of becoming candidate with the value of r and b.



LSH based on MinHash

- When choosing different r and b :



Summary :

- Local sensitive hash -- Find nearest neighbors
  - Transform high-dimensional data onto lower dimensions with similarity preserved(similar points are more likely to be hashed in the same bucket).

- MinHash – Find similarity items
  - Use MinHash Signature to approximately preserve similarities between each other (aims to reduce data size).
  - Use LHS to find neighbors of each node.

References :

- Website [1] lists a series of algorithm of LSH and the implement of Euclid Norm.
- [2] is a brief introduction of LSH.
- Chapter 3.3 in [3] introduced MinHash algorithm.
- Monograph [4] systematically introduced Nearest-neighbor methods.

[1] http://www.mit.edu/~andoni/LSH/

[2] http://web.iitd.ac.in/~sumeet/Slaney2008-LSHTutorial.pdf

[3] Rajaraman A, Ullman J D. Mining of massive datasets[M]. Cambridge University Press, 2011.

[4] Shakhnarovich G, Indyk P, Darrell T. Nearest-neighbor methods in learning and vision: theory and practice[M]. 2006.

Thanks !!