ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA TOÁN - TIN HỌC

———————oOo———————

Báo Cáo Tiểu Luận Cuối Kỳ

Môn Học Xử Lý Da Chiều

Đề Tài
Locally Sensitive Hashing functions and Its specific example like MinHash

Nhóm thực hiện :
Lê Phú Trường - 20110344
Hồ Đắc Lực - 20110233
Đỗ Tấn Phát - 20110270 .

THÀNH PHỐ HỒ CHÍ MINH

6 - 2023

Mục lục

Chương 1

Shingling of Documents

Để biểu diễn tài liệu dưới dạng tập hợp với mục đích xác định các tài liệu tương tự về mặt từ vựng là xây dựng từ tài liệu tập hợp của các chuỗi ngắn xuất hiện bên trong nó.
Làm như vậy thì các tài liệu mà có các câu hoặc cụm sẽ có sự giống nhau (common elements) trong tập hợp của chúng. Kể cả khi chúng xuất hiện không theo thứ tự giống nhau trong 2 tài liệu khác nhau.

1.1 k-Shingles

Ta thấy với mỗi 1 tài liệu là 1 chuỗi các ký tự, định nghĩa k-Shingles cho 1 document là bất kỳ chuỗi con nào có độ dài là k được tìm thấy trong tài liệu, từ những tập hợp k-Shingles đó xuất hiện 1 hay nhiều lần trong tài liệu . Thì từ đó ta có thể tạo ra sự liên kếtkết.

Ví dụ :
Cho 1 tài liệu có dạng là chuỗi D = " abcdabd ".
Ta chọn $k = 2 \Rightarrow$ Ta sẽ có tập 2-shingles là $\{ab, bc, cd, da, bd\}$.

Lưu Ý : Chuỗi con ab xuất hiện hai lần trong chuỗi D , nhưng chỉ xuất hiện một lần trong document set of 2-shingles là vì document set of 2-shingles là một tập hợp , mà mỗi phần tử trong tập hợp thì chỉ xuất hiện đúng 1 lần . Một biến thể khác của shingling là tạo thành 1 cái túi thay vì một tập hợp , nó sẽ tốt hợp tập hợp bởi vì mỗi shingling sẽ xuất hiện nhiều lần trong kết quả giống như là nó xuất hiện trong document.

Ví dụ : xét 2 câu "The plane was ready for touch down" và "The quarterback scored a touchdown", nếu ta chọn k = 9 thì khi đó tập hợp 9- shingles của câu đầu tiên có chứa cụm "touch dow" hoặc "ouch down", trong khi tập hợp 9- shingles của câu thứ 2 sẽ chứa "touchdown". Tuy nhiên nếu như ta loại bỏ các khoảng trống đi thì tập hợp 9- shingles của 2 câu trên sẽ xuất hiện sự trùng lặp đó là "touchdown", từ đây ta cũng có thể thấy rằng việc loại bỏ khoảng trắng hay giữ khoảng trắng thì nó cũng sẽ ảnh hưởng ít nhiều đến Jaccard similarty.

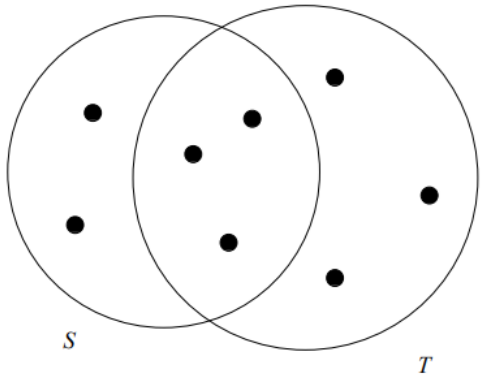
1.2 Choosing the Shingle Size

Có thể chọn k là bất kỳ hằng số nguyên dương bất kỳ . Tuy nhiên, chọn k quá nhỏ thì khi đó các phần tử thuộc tập document set of 2 shingles sẽ xuất hiện hầu khắp trong document , nếu vậy khi đó tập shingle set của 2 hay nhiều document sẽ có Jaccard Similarty cao , trong khi

các tập đó không có câu hay cụm từ nào giống nhau .

Có thể chọn các tài liệu có shingle-sets có Jaccard Similarty cao, nhưng các tài liệu không có câu hoặc thậm chí cụm từ nào giống nhau.

Jaccard similirty của các tập S và T là tỷ số $|S \cap T|/|S \cup T|$, ký hiệu là $SIM(S,T)$.



Xét hai tập hợp S và T có ba điểm chung trong giao điểm và hợp của S và T có tám phần tử .

Ta có $SIM(S,T) = \dfrac{|S \cap T|}{|S \cup T|} = \dfrac{3}{8}$.

Quay lại với cách chọn size cho shingle :

Một ví dụ tiêu biểu là chúng ta chọn k =1 .

Nếu sử dụng k = 1, các trang Web sẽ có nhiều các ký tự giống nhau và số ítít ký tự khác nhau, vì vậy các trang Web sẽ có độ tương đồng cao.

K nên lớn bao nhiêu tùy thuộc vào độ dài của các loại tài liệu và độ lớn của tập hợp chứa các ký tự của tài liệu đó .

K nên được chọn lớn vừa đủ để xác suất của bất kỳ shingle thuộc vào document set of k-shingles , xuất hiện giữa các documnet khác nhau là thấp.

Đối với email , chọn k = 5 sẽ ổn. Để hiểu tại sao, giả sử rằng chỉ có các chữ cái và ký tự khoảng trắng xuất hiện trong email (trong thực tế, các ký tự ASCII đều có khả năng xuất hiện). Nếu vậy, thì sẽ có $27^5 = 14.348.907$ shingle được tạo ra. Vì email thông thường có độ lớn nhỏ 14 triệu ký tự, ta kỳ vọng k = 5 sẽ làm tốt .

Tuy nhiên, hơn 27 ký tự xuất hiện trong email là bởi vì ngoài bảng chữ cái có 26 ký tự và khoảng trắng ra còn có các ký tự đặc biệt khác như @ ,#,.. , tất cả các ký tự đều xuất hiện với xác suất khác nhau. Các chữ cái phổ biến và khoảng trắng sẽ xuất hiện phần lớn trong các document . Do đó, những email có set of 5 shingles bao gồm các chữ cái phổ biến và xác suất các email không liên quan có các shingles giống nhau sẽ lớn hơn so với tính toán ở trên. Một nguyên tắc nhỏ là sẽ có 20 ký tự thường xuyên xuất hiện trong các văn bản, thay vì dùng toàn bộ 27 kí tự như ở trên thì ta chỉ dùng 20 kí tự và ước tính số lượng k-shingles là 20^k . Với các tài liệu lớn, chẳng hạn như các bài báo nghiên cứu, chọn $k = 9$ được coi là an toàn.

1.3 Hashing Shingles

Thay vì sử dụng các chuỗi con trực tiếp dưới dạng shingle, chúng ta có thể chọn một hash function ánh xạ các chuỗi có độ dài k tới các buckets và tạo thành 1 number bucket đại diện cho shingle. Mỗi một document sẽ được đại diện bởi một tập hợp , khi đó tập hợp các số nguyên chứa số bucket của hay nhiều k-shinggles xuất hiện trong tài liệu .

e.g: Chúng ta có thể xây dựng một set of 9 shingles cho một document và sau đó ánh xạ từng 9 shingles đó tới 1 bucket tạo thành 1 bucket number có giá trị nằm trong khoảng từ 2^{32}˘1. Do đó , mỗi shingle được biểu thị bằng 4 byte thay vì 9 .

Lưu ý rằng chúng ta có thể phân biệt các tài liệu tốt hơn nếu chúng ta sử dụng 9-shingles và băm chúng xuống còn bốn byte so với việc sử dụng 4-shingles, mặc dù không gian đã sử dụng để biểu diễn cho một shingle là như nhau. Nếu chúng ta sử dụng 4-shingles, hầu hết các chuỗi bốn byte sẽ không thể hoặc không thể tìm trong các tài liệu điển hình.

Chúng ta biết rằng chỉ có khoảng 20 ký tự được thường xuyên được sử dụng trong văn bản tiếng Anh, số lượng các 4-shingles có khả năng xảy ra là $20^4 = 160.000$. Tuy nhiên, nếu chúng ta sử dụng 9-shingles, có nhiều hơn 2^{32} shingle. Khi chúng ta băm chúng xuống còn 4 byte, với kỳ vọng rằng mọi chuỗi 4 byte đều có thể thực hiện được.

1.4 Shingles Built from Words

Trong thực tế thì các bài báo, và các bài văn xuôi, có nhiều stop words , những từ stop word phổ biến nhất như “and”, “you”, “to”, v.v. Trong nhiều ứng dụng, muốn bỏ qua các stop words trùng lặp vì nó không cho chúng ta biết thông tin gì hữu ích về nội dung bài viết, chẳng hạn như chủ đề của nó. Tuy nhiên, đối với vấn đề tìm kiếm các similar news articles, người ta nhận thấy rằng việc xác định một shingle là một stop word theo sau là hai từ tiếp theo, bất kể chúng có phải là các stop word hay không, việc loại bỏ những từ vô dụng ấy, tạo thành một tập hợp shingles hữu ích.

Chú Ý : Stop word là những từ được sử dụng phổ biến nhất trong tiếng anh ví dụ như là “I”, “you”, “we” , ... những từ này thường được loại bỏ trước khi để phân loại tài liệu.

Ưu điểm của phương pháp này là các news articles sẽ đóng góp các shingles hữu ích vào việc xây dựng set representing,để dễ dàng tìm kiếm trên Web page .Chúng ta sẽ ưu tiên set of shingles có các phần tử liên quan mật thiết đến article, từ đó ta dễ dàng tìm được những trang có chứa các article giống nhau , hoặc các article khác có Jaccard similarity cao , và loại bỏ 1 phần các trang có chứa nội dung khác và Jaccard similarity thấp với articcle mà ta cần tìm.

1.5 Similarity-Preserving Summaries of Sets

Set of shingles có kích thước rất lớn .Ngay cả khi băm nó thành bốn byte , dung lượng cần thiết để lưu trữ một sets vẫn gấp khoảng bốn lần dung lượng mà tài liệu sử dụng. Nếu chúng ta có hàng triệu tài liệu, nên không thể lưu trữ được tất cả shingles-set trong bộ nhớ . Mục tiêu trong phần này là thay thế các tập hợp lớn bằng các biểu diễn nhỏ hơn là tập con của Set of k-shingles được gọi là “signatures” . Tính chất quan trọng cần cho signatures là có thể so sánh

signatures của hai tập hợp và ước tính Jaccard similarty của các tập hợp cơ bản từ signatures. Các signatures chỉ biểu diễn một phần của tập hợp mà nó đại diện nên không thể nào yêu cầu độ chính xác tuyện đối so với tập hợp ban đầu, nhưng các ước tính mà nó cung cấp là gần với kết quả cần tìm ,signatures có kích thước càng lớn thì càng chính xác.

1.6 Matrix Representation of Sets

Matrix Representation of Sets có thể xây dựng dựa trên small signature từ large set .Sẽ rất hữu ích khi hình dung một tập hợp các tập hợp được biểu diễn dưới dạng ma trận đặc trưng. Các cột của ma trận tương ứng với các tập hợp và các hàng tương ứng với các phần tử được lấy ra từ universal set .Giả sử Matrix Representation of Sets là $A = (a_{ij})_{(1 \leqslant i \leqslant n, 1 \leqslant j \leqslant m)}$, nếu phần tử ở hàng r và nó thuộc vào tập hợp ở cột c thì $a_{rc} = 1$, còn ngược lại thì $a_{rc} = 0$.

e.g :

Element	S_1	S_2	S_3	S_4
a	1	0	0	1
b	0	0	1	0
c	0	1	0	1
d	1	0	1	1
e	0	0	1	0

Figure 3.2: A matrix representing four sets

Trong Hình 3.2 là một ví dụ về ma trận biểu diễn các tập hợp được chọn từ universal set $\{a, b, c, d, e\}$. Ở đây, $S_1 = \{a, b\}$,$S_2 = \{c\}$, $S_3 = \{b, d, e\}$ và $S_4 = \{a, b\}$.

1.7 Minhashing

Để xây dựng các signatures cho các tập hợp thì phải thực hiện lượng lớn phép tính, chẳng hạn vài trăm phép tính, mỗi phép tính là 1 "minhash" của Matrix Representation of Sets. Giá trị minhash của một cột bất kỳ là phần tử đầu tiên trong hàng mà cột đó có giá trị là 1.

Ví dụ : Trong ví dụ này các thứ tự các hàng cho characteristic matrix là *beadc*. Permutation được định nghĩa là a minhash function h ánh xạ từ sets đến hàng. Để tính minhash value của S_1 thông qua h, ở cột S_1 khi dò từ trên xuống ta thấy vị trí đầu tiên mà S_1 nhận giá trị 1 là ở hàng a do đó $h(S_1) = a$.

Element	S_1	S_2	S_3	S_4
b	0	0	1	0
e	0	0	1	0
a	1	0	0	1
d	1	0	1	1
c	0	1	0	1

Figure 3.3: A permutation of the rows of Fig. 3.2

Về mặt lý thuyết số lượng các permute của characteristic matrices là rất lớn, ta có thể biến ma trận ở hình 3.2 thành ma trận ở hình 3.3 hoàn toàn bằng minhash function h thông qua việc hoán vị các hàng với nhau. Dễ thấy các giá trị còn lại là $h(S_2) = c$,$h(S_3) = b$,và $h(S_4) = a$.

1.8 Minhashing and Jaccard Similarity

Phần này sẽ tìm hiểu mối liên hệ giữa Minhash và Jaccard Similarity.

Ta xem mỗi tập hợp trong characteristic matrix là 1 vector cột, so sánh 2 tập hợp khi xét cùng 1 hàng sẽ có 3 trường hợp xảy ra :

1. Type X cả hai đều nhận giá trị là 1.
2. Type Y có vị trí nhận giá trị là 1, vị trí còn lại nhận giá trị 0.
2. Type Z cả hai đều nhận giá trị là 0.

Vì characteristic matrix là 1 sparse matrix, nên phần lớn các hàng của characteristic matrix đều thuộc vào Type Z. Xét các hàng thuộc vào Type X và Type Y của 2 set là S_1 và S_2, gọi x là số hàng thuộc vày Type X và y là sống hàng thuộc vào Type Y, khi đó $SIM(S_1, S_2) = \dfrac{x}{x+y}$, còn có thể hiểu x là lực lượng của tập $S_1 \cap S_2$ và $x + y$ là lực lượng của $S_1 \cup S_2$.

Nhắc Lại minhash function ký hiệu là h là 1 ánh xạ mà nó hoán vị các dòng trong characteristic matrix, Xác xuất để $h(S_1) = h(S_2)$ thì nó sẽ bằng với Jaccard Similarity của S_1 và S_2, tức là $P(h(S_1) = h(S_2)) = SIM(S_1, S_2) = \dfrac{x}{x+y}$.

Mặt khác khi ta bỏ đi các dòng thuộc vào Type Z của 2 tập hợp S_1 và S_2 ,xét từ trên xuống nếu hàng đầu tiên thuộc vào Type X thì $h(S_1) = h(S_2)$, nếu hàng đầu tiên thuộc vào Type Y thì $h(S_1) \neq h(S_2)$.

1.9 Minhash Signatures

Ta có tập dữ liệu là S và characteristic matrix của S là M, ta chọn ngẫu nhiên n minhash function là $h_1, h_2, ..., h_n$, xây dựng các minhash signature cho tập S nó là các vector $[h_1(S), h_2(S), h_3(S), ..., h_n(S)]$, từ các Minhash Signatures này ta xây dựng thành 1 signature matrix , trong đó cột thứ i của ma trận M sẽ được thay thế bằng minhash signature của cột thứ i .

1.10 Computing Minhash Signatures

Như đã trình bày ở trên thì số hoán vị của một characteristic matrix rất lớn , nên trong thực tế sẽ sử dụng các random hash function để ánh xạ từ các dòng đến các bucket, cụ thể hơn là sẽ ánh xạ từ $0, 1, 2, ..., k-1$ đến các bucket number có giá trị từ $0, 1, 2, ..., k-1$,với giả thuyết là sẽ hoán vị dòng r đến vị trí $h(r)$. Ta chọn ngẫu nhiên n hash functions $h_1, h_2, ..., h_n$, đặt $SIG(i, c)$ là phần tử của signature matrix, với hash function thứ i và cột c . Ban đầu sẽ khởi tạo tập $SIG(i, c) = \infty$ cho tất cả các vị trí i và c .

Cụ thể ta sẽ xử lý hàng r qua các bước như sau :

1. Tính $h_1(r), h_2(r), ..., h_n(r)$.
2. Với mỗi cột c:
   a. Ở dòng c tất cả các giá trị đều bằng 0 thì không làm gì.
   b. Nếu ở hàng c có ít nhất 1 giá trị là 1 , thì tính $h_i(r)$ với $i = 1, 2, ..., n$.

Ví dụ: Xét ma trận ở hình 3.2 và sử dụng lại dữ liệu ở hình 3.4 , thay thế a, b, c, d, e thành các giá trị từ 0 đến 4 , chọn 2 hàm hash function $h_1(x) = x + 1 mod 5$ và $h_2(x) = 3x + 1 mod 5$, giá trị của các hàm được cho trong bảng bên dưới.

Row	S_1	S_2	S_3	S_4	$x+1 \mod 5$	$3x+1 \mod 5$
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

Figure 3.4: Hash functions computed for the matrix of Fig. 3.2

Minh họa thuật toán tính signature matrix:

Khởi tạo $SIG(i, c) = \infty$ với $1 \leq i \leq 2$ và $c \in \{S_1, S_2, S_3, S_4\}$.

	S_1	S_2	S_3	S_4
h_1	∞	∞	∞	∞
h_2	∞	∞	∞	∞

Tính $h_1(0) = 1, h_2(0) = 1$, quan sát hình 3.4 ở dòng 0 thì chỉ có tập S_1, S_2 là có giá trị 1, nên $SIG(1,1) = 1$,$SIG(2,1) = 1$,$SIG(1,4) = 1$ và $SIG(2,4) = 1$.

	S_1	S_2	S_3	S_4
h_1	1	∞	∞	1
h_2	1	∞	∞	1

Ở dòng 1 thì chỉ có S_3 là nhận giá trị 1 , $h_1(1) = 2, h_2(1) = 4$ nên $SIG(1,3) = 2$ và $SIG(2,3) = 4$, cập nhật signature matrix :

	S_1	S_2	S_3	S_4
h_1	1	∞	2	1
h_2	1	∞	4	1

Dòng 2 thì có S_2 và S_4 nhận giá trị là 1, $h_1(2) = 3$ và $h_2(2) = 2$,đến đây ta có thể thay đổi giá trị signature, nhưng giá trị cột S_3 trong signature matrix là [1,1] nhỏ hơn [3,2] nên ta không cập nhật giá trị tại cột S_3, mà chỉ cập nhật tại cột S_2 .

	S_1	S_2	S_3	S_4
h_1	1	3	2	1
h_2	1	2	4	1

Dòng 3 thì tất cả các cột đều nhận giá trị 1 ngoại trừ cột S_2 , Tính $h_1(3) = 4$ và $h_2(3) = 0$, vì $h_1(3) = 3$ là giá trị lớn hoặc bằng các giá trị trong signature matrix nên ta không cập nhật, nhưng giá trị $h_2(3) = 0$ nhỏ hơn các giá trị tương ở các cột S_1, S_2, S_3 tiến hành cập nhật, $SIG(2,1) = SIG(2,3) = SIG(2,4) = 0$.

	S_1	S_2	S_3	S_4
h_1	1	3	2	1
h_2	0	2	0	0

Dòng 4 thì chỉ có cột S_3 nhận giá trị là 1 , tính $h_1(4) = 0$ và $h_2(4) = 3$, so sánh các giá trị ở cột S_3 , $SIG(1,3) = 0 < 2$,$SIG(2,3) = 3 > 0$, do đó ta chỉ cập nhật giá trị ở vị trí $SIG(1,3) = 0$, ta được signature matrix cuối cùng .

	S_1	S_2	S_3	S_4
h_1	1	3	0	1
h_2	0	2	0	0

Từ signature matrix thu được ở trên, ta thấy rằng cột S_1 và S_4 giống nhau nên dự đoán

$SIM(S_1, S_2) = 1$ nhưng theo dữ liệu đúng ở hình 3.4 thì $SIM(S_1, S_2) = \frac{2}{3}$, cột S_1 và S_3 dự đoán là $SIM(S_1, S_2) = \frac{1}{2}$ so với $SIM(S_1, S_2) = \frac{1}{4}$ với dữ liệu đúng ban đầu, trong khi $SIM(S_1, S_2) = 0$, như vậy Jaccard Similarity được tính từ signature matrix nó nằm trong 1 lân cận của Jaccard Similarity được tính bởi data ban đầu.

Chương 2

Locality-Sensitive Hashing

Mục tiêu của LSH : tìm 1 hash function để bảo toàn khoảng xấp xỉ của các object bao gồm 2 bước :

Bước 1. là tiến hành embed-project-hash paradigm , nó là chìa khoá để hashing function sử dụng trong bước 2 để bảo toàn khoảng cách hoặc tối thiểu sai số .

Gọi $\mathcal{X}$ là data set .

Ta sẽ : Nhúng các data point từ $\mathbb{R}^d$ vào Hamming space.

Hamming space là không gian vector mà mỗi vector của nó là các binary string .
$(\mathcal{X} \subset \mathbb{R}^d, L_p) \longrightarrow (\mathcal{X}, \subset Hammingspace, L_1)$.

Các giá trị c_i của mỗi feature f_i được ánh xã tạo thành 1 vector vi ở trong Hamming space , cho giá trị c_i nằm trong đoạn [0,t] , t là số nguyên $\geqslant$ giá trị lớn nhất của tất cả giá trị mà feature nhận .

v_i được đại diện bởi chuỗi của c_i số 1 sau đó là t-c_i số 0. Ví dụ feature có giá trị là 5 trong miền [0,t] ánh xạ vào Hamming space , ta nhận được 1 bit string 1111100 , tương ứng với 5 số 1 và 2 số 00, các vector v_i sau đó được nối với nhau để có 1 dạng shingle bit string v , chuẩn L_1 là chuẩn được bảo toàn khoảng cách .

Bước 2. Được thực hiện bằng cách chiếu bit string v thu được ở bước 1 lên trên m-dimensimal subspace của Hamming space , cái subspace này được gọi là không gian hình chiếu (projection space).

Hình chiếu này có được bằng cách rút m bit ở các vị trí ngẫu nhiên, cả 2 quá trình trên sửu dụng đến 2 hashing functions .

Hashing function thứ 1 ký hiệu là h_1 , h_2 dùng để ánh xạ tập dữ liệu ban đầu sang Hamming space.

Hashing function thứ 2 ký là h_2 , h_2 dùng để chiếu dữ liệu ở Hamming space sang projection space .

Ta dùng hashing function thứ 3 ,là ánh xạ hợp nối giữa h_1 và h_2, $h_3 = h_1 \circ h_2$, để truy vấn trực tiếp từ dữ liệu ban đầu sang projection space.

Ví dụ cụ thể của LSH , ta xét 1 tập hợp ghi lại dữ liệu của 8 thành phố của Hoa Kỳ bao gồm các đặc trưng x,y và các giá trị tương ứng , ta sẽ chuẩn hoá các giá trị của x và y để

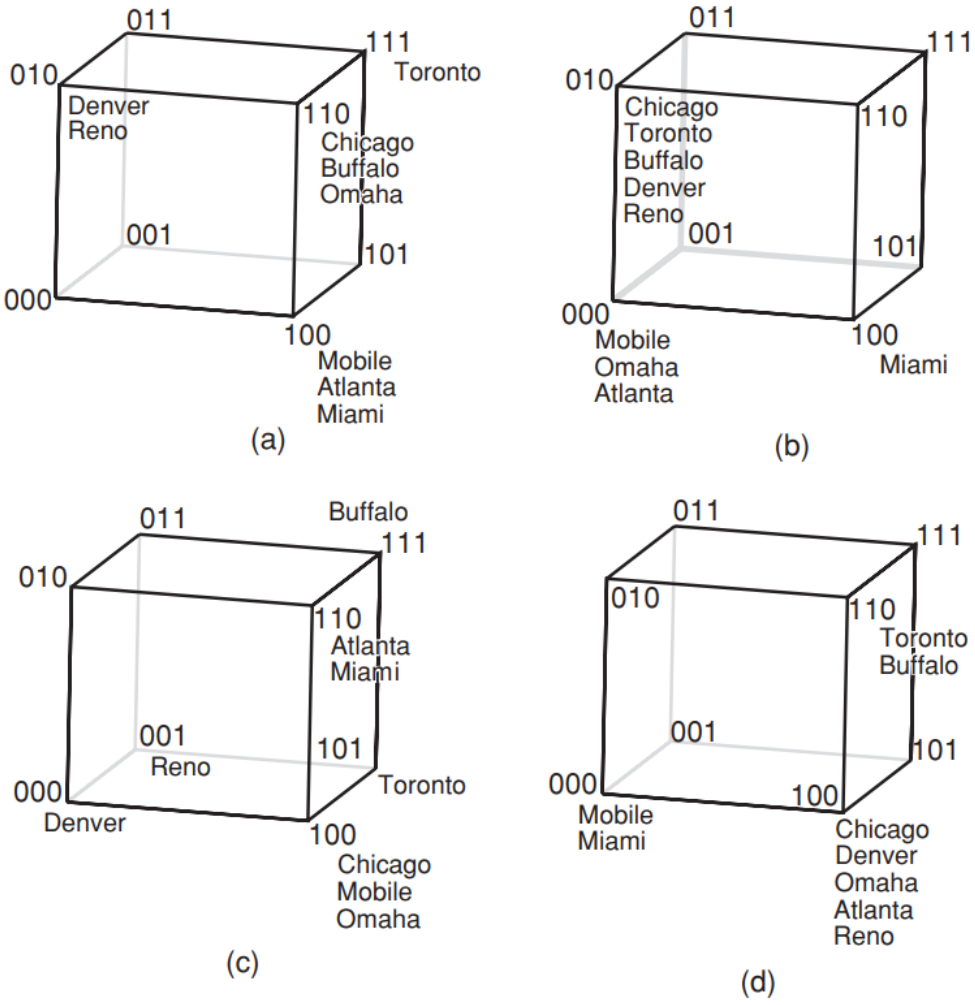
nằm trong miền [0,t] bằng cách dùng ánh xạ , $f(x) = \frac{x}{12.5}$ và $f(y) = \frac{y}{12.5}$.

Name	Attribute i		$f(i) = i \div 12.5$		HS = Unary(X) & Unary(Y)		Projection Instances			
	X	Y	f(X)	f(Y)	Unary(X)	Unary(Y)	{2,9,13}	{7,10,14}	{1,5,11}	{8,12,14}
Chicago	35	42	2	3	1100000	1110000	110	010	100	100
Mobile	52	10	4	0	1111000	0000000	100	000	100	000
Toronto	62	77	4	6	1111000	1111110	111	010	101	110
Buffalo	82	65	6	5	1111110	1111100	110	010	111	110
Denver	5	45	0	3	0000000	1110000	010	010	000	100
Omaha	27	35	2	2	1100000	1100000	110	000	100	100
Atlanta	85	15	6	1	1111110	1000000	100	000	110	100
Miami	90	5	7	0	1111111	0000000	100	100	110	000
q = Reno	5	55	0	4	0000000	1111000	010	010	001	100

Figure 4.99

Trong hình 4.99 , ta có thể thấy ta chọn số chiều của không gian hình chiếu $m = 3$, từ hình trên có thể thấy nhanh được rằng có 1 vài điểm trùng khớp , như là chicago và omaha , với $\{2, 9, 13\}$ projection, trong trường hợp này các vị trí mà ta chọn $\{2, 9, 13\}$ có characterifed as " good " conclusions bởi vì ở trong không gian metric ban đầu thì chicago = (2,3) và omaha = (2,2) nằm gần nhau, khi chiếu xuống $\{2, 9, 13\}$ projection space thì chicago= (110) và omaha =(110) , mặt khác giữa Atlanta và Denver với $\{8, 12, 14\}$ projection là character as "bad" conclusions bởi vì ở không gian metric ban đầu thì Atlanta =(6,1) và Denver = (0,3) nằm xa nhau nhưng khi chiếu xuống $\{8, 12, 14\}$ projection space thì nó lại nằm gần nhau Atlanta= Denver= (100) lại nằm gần nhau . Rõ ràng động lực cho công thức h_3 là giảm xác xuất bad conclusions xuống thấp hơn good conclusions.

Bất kỳ 1 hashing function nào có tính chất locality preserving thì hashing function đó được gọi là Locality sensitive hashing functions – LSH. Để làm giảm xác xuất bad conclusions bằng cách tăng số lượng hình chiếu khác nhau .



Đây là các kết quả khi ta chiếu lên hình lập phương đơn vị , m càng lớn thì càng chính xác nhưng cùng với sẽ là tính toán cồng kềnh, nên ta sẽ nỗ lực giảm m xuống nhưng vẫn đảm bảo độ chính xác trong ngưỡng cho phép.

2.1 Locality-Sensitive Hashing for Documents

Trong phần này, chúng ta sẽ xem xét một dạng cụ thể của LSH, được thiết kế để giải quyết vấn đề về documents được biểu diễn bằng shingle-sets sau đó được minhashed thành các short signatures.

2.2 LSH for Minhash Signatures

Một hướng tiếp cận chung cho LSH là " Băm"(hash) các item nhiều lần sao cho các items giống nhau được băm vào cùng 1 nhóm .

Bất kể cặp nào trong cùng 1 nhóm được xem là 1 cặp ứng viên. Những cặp nào khác nhau mà được băm (hash) vào cùng 1 nhóm được xem là false positives .Với mong đợi những cặp này chỉ

chiếm 1 thiểu số trong tất cả các cặp . Với Kỳ vọng rằng hầu hết các cặp thực sự giống nhau sẽ được băm vào cùng 1 bucket với ít nhất 1 trong các hash function .Những cặp giống nhau mà được băm vào những nhóm khác nhau được xem là false negatives. Hy vọng chúng chỉ chiếm 1 phần nhỏ .

Nếu ta có minhash signatures cho các items 1 cách hiệu quả để chọn các hàm băm là chia ma trận signature thành b bands, mỗi band gồm r dòng .Với mỗi band có 1 hàm băm nhận các vectors gồm r số nguyên (1 phần của 1 cột trong band đó) và băm chúng vào nhiều nhóm . Ta có thể dùng cùng 1 hàm băm cho tất cả các bands, nhưng ta phải dùng một mảng bucket riêng biệt cho mỗi band , do đó các cột với cùng 1 vector ở các bands khác nhau sẽ không băm vào cùng 1 bucket.

Ví dụ :

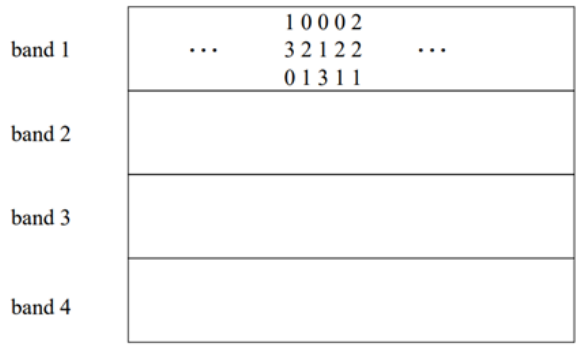


Figure 3.6: Dividing a signature matrix into four bands of three rows per band

2.3 Analysis of the Banding Technique.

Giả sử ta sử dụng b bands của mỗi r dòng và 1 cặp tài liệu cụ thể có Jaccard similarity là s. Theo phần trên , xác xuất mà minhash signatures của các document này tuân theo bất kỳ 1 hàng cụ thể của ma trận signature là s.

Ta có thể tính xác xuất mà những tài liệu này (hoặc các signatures của chúng) trở thành 1 cặp ứng viên như sau

1. Xác xuất mà các signatures nằm trên tất cả các dòng của 1 band cụ thể là s^r.
2. Xác xuất mà các signatures không nằm trên ít nhật 1 dòng của band cụ thể là $1 - s^r$.
3. Xác xuất mà các signatures không nằm trên ít nhất 1 dòng của mỗi band là $1 - s^{rb}$.
4. Xác xuất mà các signatures nằm trên tất cả các dòng của ít nhất 1 band , và là 1 cặp ứng viên là $1 - (1 - s^r)^b$

Về việc các hằng số b và r được chọn , hàm này có dạng là 1 s-curve như hình 3.7 .

Ngưỡng mà giá trị của độ tương đồng s mà từ đó xác suất của việc trở thành 1 cặp ứng viên là

$\frac{1}{2}$, là hàm của b và r. 1 xấp xỉ của ngưỡng là $(\frac{1}{b})^{\frac{1}{r}}$.

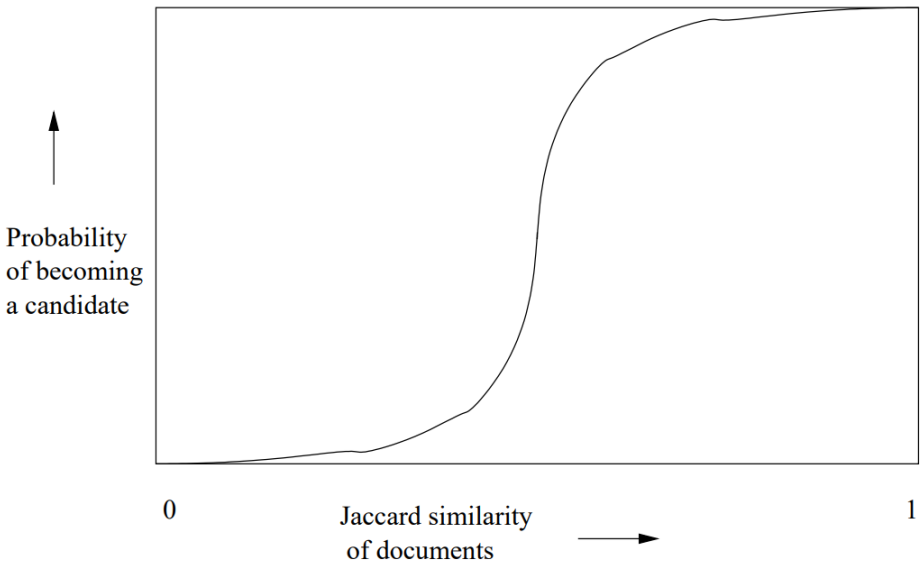


Figure 3.7: The S-curve

Ví Dụ: nếu $b = 16$ và $r = 4$ thì ngưỡng được xấp xỉ tại $s = \frac{1}{2}$ và căn bậc 4 của $\frac{1}{16}$ là $\frac{1}{2}$.

2.4 Combining the TechniquesTechniques.

Một cách tiếp cận để tìm tập hợp những cặp ứng viên cho các tài liệu tương đồng và sau đó tìm kiếm các tài liệu giống nhau . Cần phải nhấn mạnh rằng phương pháp này có thể dẫn đến false negatives,các cặp tương đồng có thể không giống nhau, vì chúng chưa bao giờ trở thành 1 cặp ứng viên. Chúng cũng có thể là false positives - những cặp ứng viên đã được dánh giá , nhưng chưa đủ độ tương đồng.

1. Chọn 1 giá trị k và xây dựng tập k-shingles từ mỗi tài liệu
2. Sắp xếp các cặp document-shingles theo shingle
3. Chọn độ dài n cho các minhash signatures . Đưa mảng đã sắp xếp vào thuật toán ở phần trên để tính minhash signatures cho tất cả tài liệu .
4. Chọn 1 ngưỡng t để xác định các tài liệu nên tương đồng như thế nào để được xem là 1 cặp tương đồng mong muốn.Chọn 1 vài bands b và 1 vài dòng r sao cho $br = n$ và ngưỡng t được xấp xỉ bằng $(\frac{1}{b})^{\frac{1}{r}}$ .Việc hạn chế false negatives rất quan trọng , ta có thể muốn chọn b và r để tạo ra 1 ngưỡng thấp hơn t .Nếu tốc độ là quan trọng, ta có thể muốn giới hạn false positives, chọn b và r để tạo ra 1 ngưỡng cao hơn.
5. Thành lập tập các cặp ứng viên bằng việc áp dụng kĩ thuật LSH ở phần trên.

6. Kiểm tra các signatures của mỗi cặp ứng viên và xác định xem tỉ số của các thành phần mà chúng thuộc về ít nhất là t.

7. (Tùy chọn) Nếu các signatures đã đủ tương đồng, thì kiểm tra các tài liệu xem chúng có thuộc sự giống nhau hay không, hơn là chỉ giống nhau vì mong muốn.

Tài liệu tham khảo

[1] Anand Rajaraman, Jure Leskovec and Jeffrey D. Ullman, Mining of Massive Datasets , Chapter 3. Finding Similar Items.

[2] Hanan Samet ,Foundations of Multidimensional and Metric Data Structures, 4.7.4 Locality Sensitive Hashing.

Math. **58**, 339–378 (2001)