

Similarity Search

CSE545 - Spring 2023
Stony Brook University

H. Andrew Schwartz

$A \cap B$

Big Data Analytics, The Class

Goal: Generalizations
A model or summarization of the data.

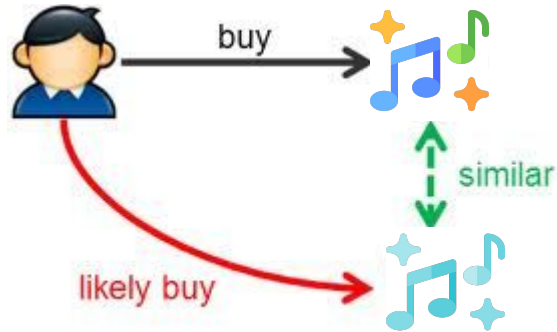
Data Workflow Frameworks

Analytics and Algorithms

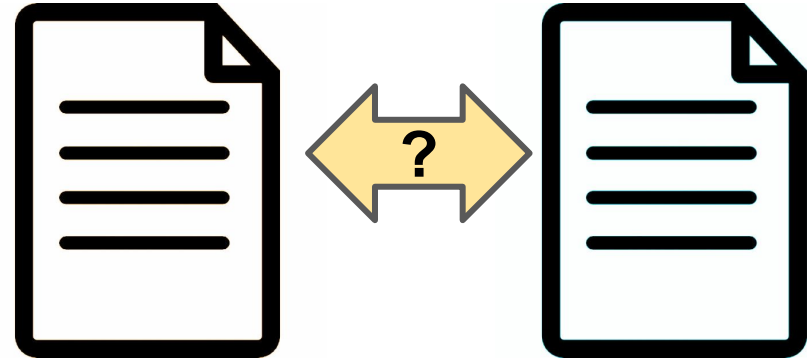
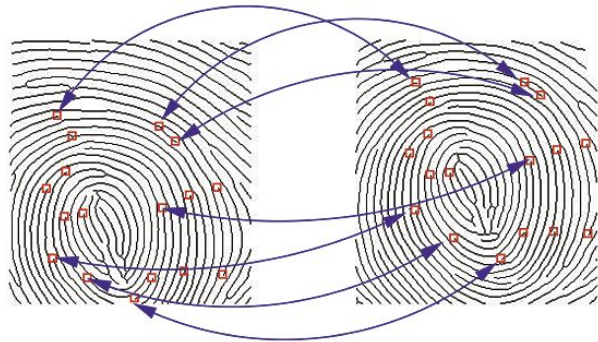
Hadoop File System ✓
MapReduce ✓
Streaming ✓
Deep Learning Frameworks ✓
Spark ✓

Similarity Search
Hypothesis Testing
Transformers/Self-Supervision
Recommendation Systems
Link Analysis

Finding Similar Items



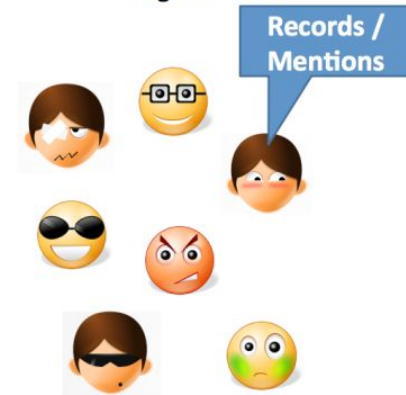
(<http://blog.soton.ac.uk/hive/2012/05/10/recommendation-system-of-hive/>)



Real World



Digital World



(<http://www.datacommunitydc.org/blog/2013/08/entity-resolution-for-big-data>)

Finding Similar Items: Topics

- Shingling
- Minhashing
- Locality-sensitive hashing
- Distance Metrics

Finding Similar Items: Topics

Challenge: How to represent the document in a way that can be efficiently encoded and compared?

- Shingling
- Minhashing
- Locality-sensitive hashing
- Distance Metrics

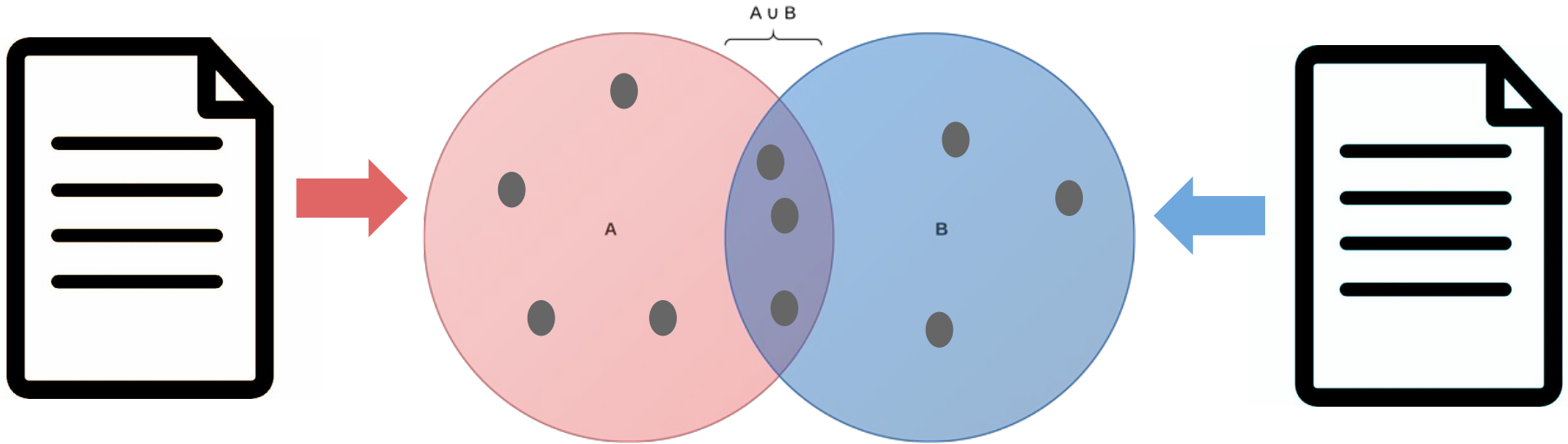
Finding Similar Items: Topics

Challenge: How to represent the document in a way that can be efficiently encoded and compared?

- **Shingling** *(sparse, high-dimensional representation)*
- **Minhashing** *(low dimensional representation)*
- **Locality-sensitive hashing**
(fast search of similar items)
- **Distance Metrics** *(quantify similarity)*

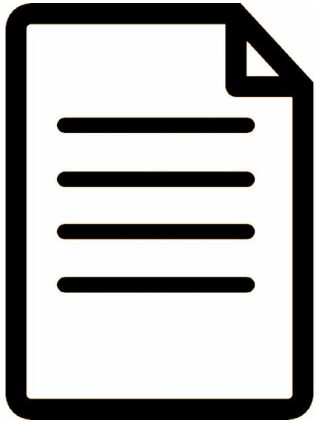
Shingles

Goal: Convert documents to sets



Shingles

Goal: Convert documents to sets



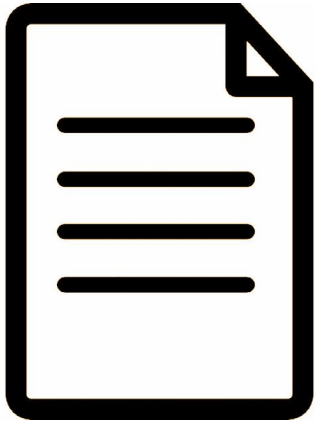
k-shingles (aka “character n-grams”)
- sequence of k characters



E.g. $k=2$ doc=“abcdabd”
 $\text{singles}(\text{doc}, 2) = \{\text{ab}, \text{bc}, \text{cd}, \text{da}, \text{bd}\}$

Shingles

Goal: Convert documents to sets



k-shingles (aka “character n-grams”)
- sequence of k characters



E.g. $k=2$ doc="abcdabd"
 $\text{singles}(\text{doc}, 2) = \{\text{ab}, \text{bc}, \text{cd}, \text{da}, \text{bd}\}$

- Similar documents have many common shingles
- Changing words or order has minimal effect.
- In practice use $5 < k < 10$

Shingles

Goal: Convert documents to sets



Large enough that any given shingle appearing a document is highly unlikely (e.g. $< .1\%$ chance)

Can hash large shingles to smaller (e.g. 9-shingles into 4 bytes)

Can also use words (aka n-grams).

- Similar documents have many common shingles
- Changing words or order has minimal effect.
- **In practice use $5 < k < 10$**

Shingles

Problem: Even if hashing, sets of shingles are large
(e.g. 4 bytes \Rightarrow 4x the size of the document).

Minhashing

Goal: Convert sets to shorter ids, signatures

Minhashing

Goal: Convert sets to shorter ids, “signatures”

Characteristic Matrix, X :

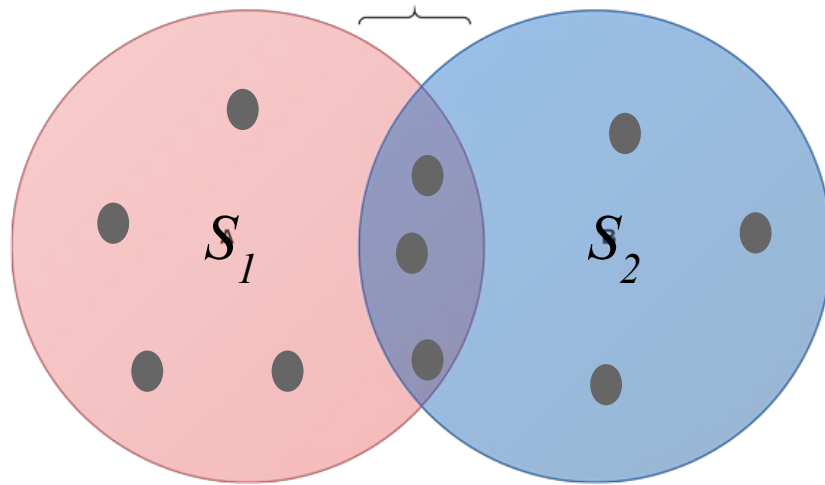
<i>Element</i>	S_1	S_2	S_3	S_4
a	1	0	0	1	
b	0	0	1	0	
c	0	1	0	1	
d	1	0	1	1	
e	0	0	1	0	

(Leskovec et al., 2014; <http://www.mmms.org/>)

often very sparse! (lots of zeros)

Jaccard Similarity:

$$\text{sim}(S_1, S_2) = \frac{S_1 \cap S_2}{S_1 \cup S_2}$$



Minhashing

Characteristic Matrix:

	S_1	S_2
ab	1	1
bc	0	1
de	1	0
ah	1	1
ha	0	0
ed	1	1
ca	0	1

Jaccard Similarity:

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

Minhashing

Characteristic Matrix:

	S_1	S_2	
ab	1	1	**
bc	0	1	*
de	1	0	*
ah	1	1	**
ha	0	0	
ed	1	1	**
ca	0	1	*

Jaccard Similarity:

$$\text{sim}(S_1, S_2) = \frac{S_1 \cap S_2}{S_1 \cup S_2}$$

Minhashing

Characteristic Matrix:

	S_1	S_2	
ab	1	1	**
bc	0	1	*
de	1	0	*
ah	1	1	**
ha	0	0	
ed	1	1	**
ca	0	1	*

Jaccard Similarity:

$$\text{sim}(S_1, S_2) = \frac{S_1 \cap S_2}{S_1 \cup S_2}$$

$$\text{sim}(S_1, S_2) = 3 / 6$$

both have / # at least one has

Minhashing

Problem: Even if hashing shingle contents,
sets of shingles are large

e.g. 4 byte integer per shingle: assume all unique shingles,
=> 4x the size of the document

(since there are as many shingles as characters and 1byte per char).

Minhashing

Goal: Convert sets to shorter ids, “signatures”

Characteristic Matrix: X

	S_1	S_2	S_3	S_4
ab	1	0	1	0
bc	1	0	0	1
de	0	1	0	1
ah	0	1	0	1
ha	0	1	0	1
ed	1	0	1	0
ca	1	0	1	0

(Leskovec et al., 2014; <http://www.mmms.org/>)

Minhashing

Goal: Convert sets to shorter ids, “signatures”

Characteristic Matrix: X

	S_1	S_2	S_3	S_4
ab	1	0	1	0
bc	1	0	0	1
de	0	1	0	1
ah	0	1	0	1
ha	0	1	0	1
ed	1	0	1	0
ca	1	0	1	0


Approximate Approach:

- 1) Instead of keeping whole characteristic matrix, just keep first row where 1 is encountered.
- 2) Shuffle and repeat to get a “signature” for each set.

Minhashing

Goal: Convert sets to shorter ids, “signatures”

Characteristic Matrix: X



	S_1	S_2	S_3	S_4
ab	1	0	1	0
bc	1	0	0	1
de	0	1	0	1
ah	0	1	0	1
ha	0	1	0	1
ed	1	0	1	0
ca	1	0	1	0

Approximate Approach:

1) Instead of keeping whole characteristic matrix, just keep first row where 1 is encountered.

2) Shuffle and repeat to get a “signature” for each set.

Minhashing

Goal: Convert sets to shorter ids, “signatures”

Characteristic Matrix: X

	S_1	S_2	S_3	S_4
ab	1	0	1	0
bc	1	0	0	1
de	0	1	0	1
ah	0	1	0	1
ha	0	1	0	1
ed	1	0	1	0
ca	1	0	1	0

1 3 1 2

Approximate Approach:

1) Instead of keeping whole characteristic matrix, just keep first row where 1 is encountered.

2) Shuffle and repeat to get a “signature”.

	S_1	S_2	S_3	S_4
ah	0	1	0	1
ca	1	0	1	0
ed	1	0	1	0
de	0	1	0	1
ab	1	0	1	0
bc	1	0	0	1

2 1 2 1

...

Minhashing

Goal: Convert sets to shorter ids, “signatures”

Characteristic Matrix: X

	S_1	S_2	S_3	S_4
ab	1	0	1	0
bc	1	0	0	1
de	0	1	0	1
ah	0	1	0	1
ha	0	1	0	1
ed	1	0	1	0
ca	1	0	1	0

1 3 1 2

Approximate Approach:

1) Instead of keeping whole characteristic matrix, just keep first row where 1 is encountered.

2) Shuffle and repeat to get a “signature”.

	S_1	S_2	S_3	S_4
ah	0	1	0	1
ca	1	0	1	0
ed	1	0	1	0
de	0	1	0	1
ab	1	0	1	0
bc	1	0	0	1

2 1 2 1

signatures

S_1	S_2	S_3	S_4
1	3	1	2
2	1	2	1
...

...

Minhashing

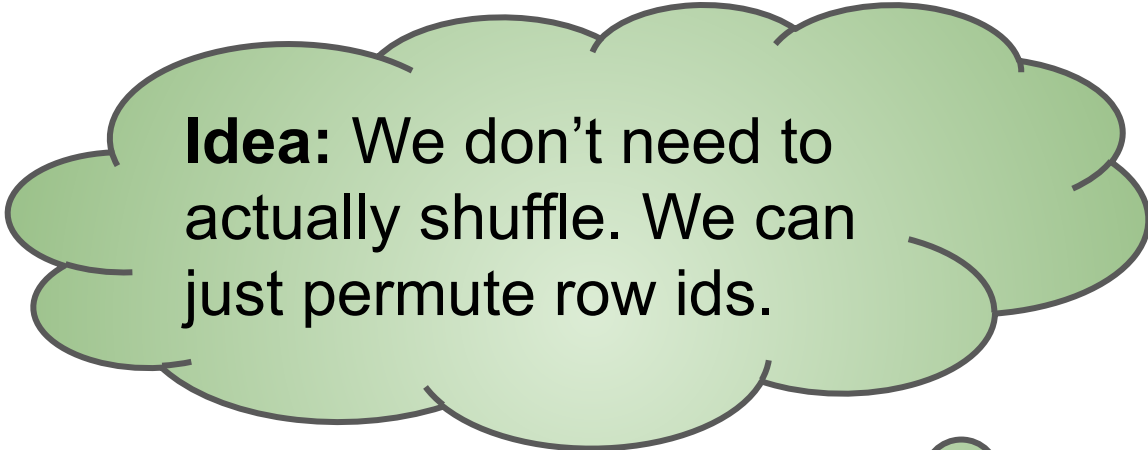
Goal: Convert sets to shorter ids, “signatures”

Characteristic Matrix: X

	S_1	S_2	S_3	S_4
ab	1	0	1	0
bc	1	0	0	1
de	0	1	0	1
ah	0	1	0	1
ha	0	1	0	1
ed	1	0	1	0
ca	1	0	1	0

Approximate Approach:

- 1) Instead of keeping whole characteristic matrix, just keep first row where 1 is encountered.
- 2) Shuffle and repeat to get a “signature” for each set.



Idea: We don't need to actually shuffle. We can just permute row ids.

Minhashing

Characteristic Matrix:

	S_1	S_2	S_3	S_4
ab	1	0	1	0
bc	1	0	0	1
de	0	1	0	1
ah	0	1	0	1
ha	0	1	0	1
ed	1	0	1	0
ca	1	0	1	0

Minhash function: h

- Based on permutation of rows in the characteristic matrix, h maps sets to first row where set appears.

Minhashing

Characteristic Matrix:

	S_1	S_2	S_3	S_4
ab	1	0	1	0
bc	1	0	0	1
de	0	1	0	1
ah	0	1	0	1
ha	0	1	0	1
ed	1	0	1	0
ca	1	0	1	0

permuted order
1 ha
2 ed
3 ab
4 bc
5 ca
6 ah
7 de

Minhash function: h

- Based on permutation of rows in the characteristic matrix, h maps sets to first row where set appears.

Minhashing

Characteristic Matrix:

		S_1	S_2	S_3	S_4
3	ab	1	0	1	0
4	bc	1	0	0	1
7	de	0	1	0	1
6	ah	0	1	0	1
1	ha	0	1	0	1
2	ed	1	0	1	0
5	ca	1	0	1	0

permuted order
1 ha
2 ed
3 ab
4 bc
5 ca
6 ah
7 de

Minhash function: h

- Based on permutation of rows in the characteristic matrix, h maps sets to first row where set appears.

Minhashing

Characteristic Matrix:

		S_1	S_2	S_3	S_4
3	ab	1	0	1	0
4	bc	1	0	0	1
7	de	0	1	0	1
6	ah	0	1	0	1
1	ha	0	1	0	1
2	ed	1	0	1	0
5	ca	1	0	1	0

permuted order
1 ha
2 ed
3 ab
4 bc
5 ca
6 ah
7 de

Minhash function: h

- Based on permutation of rows in the characteristic matrix, h maps sets to first row where set appears.

$$h(S_1) = ed \text{ \#permuted row 2}$$

$$h(S_2) = ha \text{ \#permuted row 1}$$

$$h(S_3) =$$

Minhashing

Characteristic Matrix:

		S_1	S_2	S_3	S_4
3	ab	1	0	1	0
4	bc	1	0	0	1
7	de	0	1	0	1
6	ah	0	1	0	1
1	ha	0	1	0	1
2	ed	1	0	1	0
5	ca	1	0	1	0

permuted order
1 ha
2 ed
3 ab
4 bc
5 ca
6 ah
7 de

Minhash function: h

- Based on permutation of rows in the characteristic matrix, h maps sets to first row where set appears.

$$h(S_1) = ed \text{ \#permuted row 2}$$

$$h(S_2) = ha \text{ \#permuted row 1}$$

$$h(S_3) = ed \text{ \#permuted row 2}$$

$$h(S_4) =$$

Minhashing

Characteristic Matrix:

		S_1	S_2	S_3	S_4
3	ab	1	0	1	0
4	bc	1	0	0	1
7	de	0	1	0	1
6	ah	0	1	0	1
1	ha	0	1	0	1
2	ed	1	0	1	0
5	ca	1	0	1	0

permuted order
1 ha
2 ed
3 ab
4 bc
5 ca
6 ah
7 de

Minhash function: h

- Based on permutation of rows in the characteristic matrix, h maps sets to first row where set appears.

$$h(S_1) = ed \text{ \#permuted row 2}$$

$$h(S_2) = ha \text{ \#permuted row 1}$$

$$h(S_3) = ed \text{ \#permuted row 2}$$

$$h(S_4) = ha \text{ \#permuted row 1}$$

Minhashing

Characteristic Matrix:

		S_1	S_2	S_3	S_4
3	ab	1	0	1	0
4	bc	1	0	0	1
7	de	0	1	0	1
6	ah	0	1	0	1
1	ha	0	1	0	1
2	ed	1	0	1	0
5	ca	1	0	1	0

(Leskovec et al., 2014; <http://www.mmms.org/>)

Minhash function: h

- Based on permutation of rows in the characteristic matrix, h maps sets to rows.

Signature matrix: M

- Record first row where each set had a 1 in the given permutation

	S_1	S_2	S_3	S_4
h_1	2	1	2	1

$h_1(S_1) = ed$ #permuted row 2

$h_1(S_2) = ha$ #permuted row 1

$h_1(S_3) = ed$ #permuted row 2

$h_1(S_4) = ha$ #permuted row 1

Minhashing

Characteristic Matrix:

		S_1	S_2	S_3	S_4
3	ab	1	0	1	0
4	bc	1	0	0	1
7	de	0	1	0	1
6	ah	0	1	0	1
1	ha	0	1	0	1
2	ed	1	0	1	0
5	ca	1	0	1	0

(Leskovec et al., 2014; <http://www.mmms.org/>)

Minhash function: h

- Based on permutation of rows in the characteristic matrix, h maps sets to rows.

Signature matrix: M

- Record first row where each set had a 1 in the given permutation

	S_1	S_2	S_3	S_4
h_1	2	1	2	1

$$h_1(S_1) = \text{ed} \text{ \#permutated row}$$

2

$$h_1(S_2) = \text{ha} \text{ \#permutated row}$$

1

$$h(S_3) = \text{ed} \text{ \#permutated row}$$

Minhashing

Characteristic Matrix:

		S_1	S_2	S_3	S_4
3	ab	1	0	1	0
4	bc	1	0	0	1
7	de	0	1	0	1
6	ah	0	1	0	1
1	ha	0	1	0	1
2	ed	1	0	1	0
5	ca	1	0	1	0

(Leskovec et al., 2014; <http://www.mmms.org/>)

Minhash function: h

- Based on permutation of rows in the characteristic matrix, h maps sets to rows.

Signature matrix: M

- Record first row where each set had a 1 in the given permutation

	S_1	S_2	S_3	S_4
h_1	2	1	2	1

$h_1(S_1) = ed$ #permutated row

2

$h_1(S_2) = ha$ #permutated row

1

$h(S_3) = ed$ #permutated row

Minhashing

Characteristic Matrix:

			S_1	S_2	S_3	S_4
4	3	ab	1	0	1	0
2	4	bc	1	0	0	1
1	7	de	0	1	0	1
3	6	ah	0	1	0	1
6	1	ha	0	1	0	1
7	2	ed	1	0	1	0
5	5	ca	1	0	1	0

Minhash function: h

- Based on permutation of rows in the characteristic matrix, h maps sets to rows.

Signature matrix: M

- Record first row where each set had a 1 in the given permutation

	S_1	S_2	S_3	S_4
h_1	2	1	2	1
h_2				

Minhashing

Characteristic Matrix:

			S_1	S_2	S_3	S_4
4	3	ab	1	0	1	0
2	4	bc	1	0	0	1
1	7	de	0	1	0	1
3	6	ah	0	1	0	1
6	1	ha	0	1	0	1
7	2	ed	1	0	1	0
5	5	ca	1	0	1	0

Minhash function: h

- Based on permutation of rows in the characteristic matrix, h maps sets to rows.

Signature matrix: M

- Record first row where each set had a 1 in the given permutation

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

Minhashing

Characteristic Matrix:

				S_1	S_2	S_3	S_4
1	4	3	ab	1	0	1	0
3	2	4	bc	1	0	0	1
7	1	7	de	0	1	0	1
6	3	6	ah	0	1	0	1
2	6	1	ha	0	1	0	1
5	7	2	ed	1	0	1	0
4	5	5	ca	1	0	1	0

Minhash function: h

- Based on permutation of rows in the characteristic matrix, h maps sets to rows.

Signature matrix: M

- Record first row where each set had a 1 in the given permutation

	S_1	S_2	S_3	S_4
h_1	2	1	2	1
h_2	2	1	4	1
h_3				

Minhashing

Characteristic Matrix:

				S_1	S_2	S_3	S_4
1	4	3	ab	1	0	1	0
3	2	4	bc	1	0	0	1
7	1	7	de	0	1	0	1
6	3	6	ah	0	1	0	1
2	6	1	ha	0	1	0	1
5	7	2	ed	1	0	1	0
4	5	5	ca	1	0	1	0

Minhash function: h

- Based on permutation of rows in the characteristic matrix, h maps sets to rows.

Signature matrix: M

- Record first row where each set had a 1 in the given permutation

	S_1	S_2	S_3	S_4
h_1	2	1	2	1
h_2	2	1	4	1
h_3	1	2	1	2

Minhashing

Characteristic Matrix:

				S_1	S_2	S_3	S_4
1	4	3	ab	1	0	1	0
3	2	4	bc	1	0	0	1
7	1	7	de	0	1	0	1
6	3	6	ah	0	1	0	1
2	6	1	ha	0	1	0	1
5	7	2	ed	1	0	1	0
4	5	5	ca	1	0	1	0

(Leskovec et al., 2014; <http://www.mmms.org/>)

Minhash function: h

- Based on permutation of rows in the characteristic matrix, h maps sets to rows.

Signature matrix: M

- Record first row where each set had a 1 in the given permutation

	S_1	S_2	S_3	S_4
h_1	2	1	2	1
h_2	2	1	4	1
h_3	1	2	1	2
...				
...				

Minhashing

Property of signature matrix:
The probability for any h_i (i.e. any row), that $h_i(S_1) = h_i(S_2)$ is the same as $\text{Sim}(S_1, S_2)$

Characteristic Matrix:

				S_1	S_2	S_3	S_4
1	4	3	ab	1	0	1	0
3	2	4	bc	1	0	0	1
7	1	7	de	0	1	0	1
6	3	6	ah	0	1	0	1
2	6	1	ha	0	1	0	1
5	7	2	ed	1	0	1	0
4	5	5	ca	1	0	1	0

	S_1	S_2	S_3	S_4
h_1	2	1	2	1
h_2	2	1	4	1
h_3	1	2	1	2
...				
...				

Minhashing

Characteristic Matrix:

				S_1	S_2	S_3	S_4
1	4	3	ab	1	0	1	0
3	2	4	bc	1	0	0	1
7	1	7	de	0	1	0	1
6	3	6	ah	0	1	0	1
2	6	1	ha	0	1	0	1
5	7	2	ed	1	0	1	0
4	5	5	ca	1	0	1	0

Property of signature matrix:

The probability for any h_i (i.e. any row), that $h_i(S_1) = h_i(S_2)$ is the same as $\text{Sim}(S_1, S_2)$

Thus, similarity of signatures S_1, S_2 is the fraction of minhash functions (i.e. rows) in which they agree.

	S_1	S_2	S_3	S_4
h_1	2	1	2	1
h_2	2	1	4	1
h_3	1	2	1	2
...				
...				

Minhashing

Characteristic Matrix:

				S_1	S_2	S_3	S_4
1	4	3	ab	1	0	0	0
3	2	4	an	0	1	0	0
7	1	5	ca	1	0	1	0
6	3	6	an	0	1	0	0
2	6	1	ha	0	1	0	1
5	7	2	ed	1	0	1	0
4	5	5	ca	1	0	1	0

Estimate with a random sample of permutations (i.e. ~100)

Property of signature matrix:
The probability for any h_i (i.e. any row), that $h_i(S_1) = h_i(S_2)$ is the same as $\text{Sim}(S_1, S_2)$

Thus, similarity of signatures S_1, S_2 is the fraction of minhash functions (i.e. rows) in which they agree.

	S_1	S_2	S_3	S_4
h_1	2	1	2	1
h_2	2	1	4	1
h_3	1	2	1	2
...				
...				

Minhashing

Characteristic Matrix:

				S_1	S_2	S_3	S_4
1	4	3	ab	1	0	0	0
3	2	4	ba	0	1	0	0
7	6	1	ca	1	0	1	0
6	3	6	ac	0	1	0	1
2	6	1	ha	0	1	0	1
5	7	2	ed	1	0	1	0
4	5	5	ca	1	0	1	0

Estimate with a random sample of permutations (i.e. ~100)

	S_1	S_2	S_3	S_4
h_1	2	1	2	1
h_2	2	1	4	1
h_3	1	2	1	2

Property of signature matrix:

The probability for any h_i (i.e. any row), that $h_i(S_1) = h_i(S_2)$ is the same as $\text{Sim}(S_1, S_2)$

Thus, similarity of signatures S_1, S_2 is the fraction of minhash functions (i.e. rows) in which they agree.

Estimated $\text{Sim}(S_1, S_3) =$
agree / all = $2/3$

Minhashing

Characteristic Matrix:

				S_1	S_2	S_3	S_4
1	4	3	ab	<u>1</u>	0	<u>1</u>	0
3	2	4	bc	<u>1</u>	0	0	1
7	1	7	de	0	1	0	1
6	3	6	ah	0	1	0	1
2	6	1	ha	0	1	0	1
5	7	2	ed	<u>1</u>	0	<u>1</u>	0
4	5	5	ca	<u>1</u>	0	<u>1</u>	0

Property of signature matrix:

The probability for any h_i (i.e. any row), that $h_i(S_1) = h_i(S_2)$ is the same as $\text{Sim}(S_1, S_2)$

Thus, similarity of signatures S_1, S_2 is the fraction of minhash functions (i.e. rows) in which they agree.

	S_1	S_2	S_3	S_4
h_1	2	1	2	1
h_2	2	1	4	1
h_3	1	2	1	2

Estimated $\text{Sim}(S_1, S_3) =$
agree / all = $2/3$

Real $\text{Sim}(S_1, S_3) =$
Type a / (a + b + c) = $3/4$

Minhashing

Characteristic Matrix:

				S_1	S_2	S_3	S_4
1	4	3	ab	<u>1</u>	0	<u>1</u>	0
3	2	4	bc	<u>1</u>	0	0	1
7	1	7	de	0	1	0	1
6	3	6	ah	0	1	0	1
2	6	1	ha	0	1	0	1
5	7	2	ed	<u>1</u>	0	<u>1</u>	0
4	5	5	ca	<u>1</u>	0	<u>1</u>	0

Property of signature matrix:

The probability for any h_i (i.e. any row), that $h_i(S_1) = h_i(S_2)$ is the same as $\text{Sim}(S_1, S_2)$

Thus, similarity of signatures S_1, S_2 is the fraction of minhash functions (i.e. rows) in which they agree.

	S_1	S_2	S_3	S_4
h_1	2	1	2	1
h_2	2	1	4	1
h_3	1	2	1	2

Estimated $\text{Sim}(S_1, S_3) =$
agree / all = $2/3$

Real $\text{Sim}(S_1, S_3) =$
Type a / (a + b + c) = $3/4$

Try $\text{Sim}(S_2, S_4)$ and
 $\text{Sim}(S_1, S_2)$

Minhashing

Error Bound?

Characteristic Matrix:

				S_1	S_2	S_3	S_4
1	4	3	ab	<u>1</u>	0	<u>1</u>	0
3	2	4	bc	<u>1</u>	0	0	1
7	1	7	de	0	1	0	1
6	3	6	ah	0	1	0	1
2	6	1	ha	0	1	0	1
5	7	2	ed	<u>1</u>	0	<u>1</u>	0
4	5	5	ca	<u>1</u>	0	<u>1</u>	0

	S_1	S_2	S_3	S_4
h_1	2	1	2	1
h_2	2	1	4	1
h_3	1	2	1	2

Estimated $\text{Sim}(S_1, S_3) =$
agree / all = $2/3$

Real $\text{Sim}(S_1, S_3) =$
Type a / (a + b + c) = $3/4$

Try $\text{Sim}(S_2, S_4)$ and
 $\text{Sim}(S_1, S_2)$

Minhashing

Characteristic Matrix:

				S_1	S_2	S_3	S_4
1	4	3	ab	<u>1</u>	0	<u>1</u>	0
3	2	4	bc	<u>1</u>	0	0	1
7	1	7	de	0	1	0	1
6	3	6	ah	0	1	0	1
2	6	1	ha	0	1	0	1
5	7	2	ed	<u>1</u>	0	<u>1</u>	0
4	5	5	ca	<u>1</u>	0	<u>1</u>	0

Error Bound?

Expect error: $O(1/\sqrt{k})$ (k hashes)

Why? Each row is a random observation of 1 or 0 (match or not) with $P(\text{match}=1) = \text{Sim}(S_1, S_2)$.

	S_1	S_2	S_3	S_4
h_1	2	1	2	1
h_2	2	1	4	1
h_3	1	2	1	2

Estimated $\text{Sim}(S_1, S_3) =$
agree / all = $2/3$

Real $\text{Sim}(S_1, S_3) =$
Type a / (a + b + c) = $3/4$

Try $\text{Sim}(S_2, S_4)$ and
 $\text{Sim}(S_1, S_2)$

Minhashing

Characteristic Matrix:

				S_1	S_2	S_3	S_4
1	4	3	ab	<u>1</u>	0	<u>1</u>	0
3	2	4	bc	<u>1</u>	0	0	1
7	1	7	de	0	1	0	1
6	3	6	ah	0	1	0	1
2	6	1	ha	0	1	0	1
5	7	2	ed	<u>1</u>	0	<u>1</u>	0
4	5	5	ca	<u>1</u>	0	<u>1</u>	0

(Leskovec et al., 2014; <http://www.mmms.org/>)

Error Bound?

Expect error: $O(1/\sqrt{k})$ (k hashes)

Why? Each row is a random observation of 1 or 0 (match or not) with $P(\text{match}=1) = \text{Sim}(S_1, S_2)$.

$N = k$ observations

Standard deviation(*std*)? < 1 (worst case is 0.5)

	S_1	S_2	S_3	S_4
h_1	2	1	2	1
h_2	2	1	4	1
h_3	1	2	1	2

Estimated $\text{Sim}(S_1, S_3) =$
agree / all = $2/3$

Real $\text{Sim}(S_1, S_3) =$
Type a / (a + b + c) = $3/4$

Try $\text{Sim}(S_2, S_4)$ and
 $\text{Sim}(S_1, S_2)$

Minhashing

Characteristic Matrix:

				S_1	S_2	S_3	S_4
1	4	3	ab	<u>1</u>	0	<u>1</u>	0
3	2	4	bc	<u>1</u>	0	0	1
7	1	7	de	0	1	0	1
6	3	6	ah	0	1	0	1
2	6	1	ha	0	1	0	1
5	7	2	ed	<u>1</u>	0	<u>1</u>	0
4	5	5	ca	<u>1</u>	0	<u>1</u>	0

(Leskovec et al., 2014; <http://www.mmms.org/>)

Error Bound?

Expect error: $O(1/\sqrt{k})$ (k hashes)

Why? Each row is a random observation of 1 or 0 (match or not) with $P(\text{match}=1) = \text{Sim}(S_1, S_2)$.

$N = k$ observations

Standard deviation(*std*)? < 1 (worst case is 0.5)

Standard Error of Mean = std/\sqrt{N}

	S_1	S_2	S_3	S_4
h_1	2	1	2	1
h_2	2	1	4	1
h_3	1	2	1	2

Estimated $\text{Sim}(S_1, S_3) =$
agree / all = $2/3$

Real $\text{Sim}(S_1, S_3) =$
Type a / (a + b + c) = $3/4$

Try $\text{Sim}(S_2, S_4)$ and
 $\text{Sim}(S_1, S_2)$

Minhashing

In Practice

Problem:

- Can't reasonably do permutations (huge space)
- Can't randomly grab rows according to an order (random disk seeks = slow!)

Minhashing

In Practice

Problem:

- Can't reasonably do permutations (huge space)
- Can't randomly grab rows according to an order (random disk seeks = slow!)

Solution: Use “random” hash functions.

- Setup:
 - Pick ~ 100 hash functions, hashes
 - Store $M[i][s] = \text{a potential minimum } h_i(r)$
#initialized to infinity (num hashes x num sets)

Minhashing

Solution: Use “random” hash functions.

Setup:

```
hashes = [getHfunc(i) for i in rand(1, num=100)]
```

#100 hash functions, seeded random

```
for i in hashes: for s in sets:
```

```
    Sig[i][s] = np.inf #represents a potential minimum  $h_i(r)$  ; initially infinity
```

Minhashing

Solution: Use “random” hash functions.

Setup:

```
hashes = [getHfunc(i) for i in rand(1, num=100)]
```

#100 hash functions, seeded random

```
for i in hashes: for s in sets:
```

```
    Sig[i][s] = np.inf #represents a potential minimum  $h_i(r)$ ; initially infinity
```

Algorithm (“efficient minhashing”):

```
for r in rows of cm: #cm is characteristic matrix
```

```
    compute  $h_i(r)$  for all i in hashes #precompute 100 values
```

```
    for each set s in sets: #columns of cm
```

```
        if cm[r][s] == 1:
```

```
            for i in hashes: #check which hash produces smallest value
```

```
                if  $h_i(r) < \text{Sig}[i][s]$ :  $\text{Sig}[i][s] = h_i(r)$ 
```

Minhashing

Solution: Use “random” hash functions.

Setup:

```
hashes = [getHfunc(i) for i in rand(1, num=100)]
```

#100 hash functions, seeded random

```
for i in hashes: for s in sets:
```

```
    Sig[i][s] = np.inf #represents a potential minimum  $h_i(r)$ ; initially infinity
```

Algorithm (“efficient minhashing”) without charact matrix:

```
for feat in shins: #shins is all unique shingles
```

```
    compute  $h_i(\textit{feat})$  for all i in hashes #precompute 100 values
```

```
    for each set s in sets: #sets is list of shingle sets
```

```
        if feat in s:
```

```
            for i in hashes: #check which hash produces smallest value
```

```
                if  $h_i(\textit{feat}) < \text{Sig}[i][s_{id}]$ :  $\text{Sig}[i][s_{id}] = h_i(\textit{feat})$ 
```

Minhashing

Problem: Even if hashing, sets of shingles are large (e.g. 4 bytes => 4x the size of the document).

Minhashing

Problem: Even if hashing, sets of shingles are large (e.g. 4 bytes => 4x the size of the document).

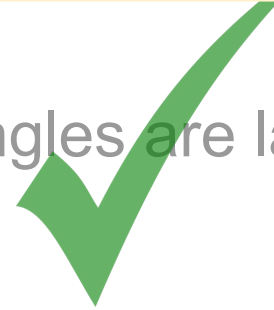


New Problem: Even if the size of signatures are small, it can be computationally expensive to find similar pairs.

E.g. 1m documents; $1,000,000 \text{ choose } 2 = 500,000,000,000$ pairs!

Minhashing

Problem: Even if hashing, sets of shingles are large (e.g. 4 bytes => 4x the size of the document).

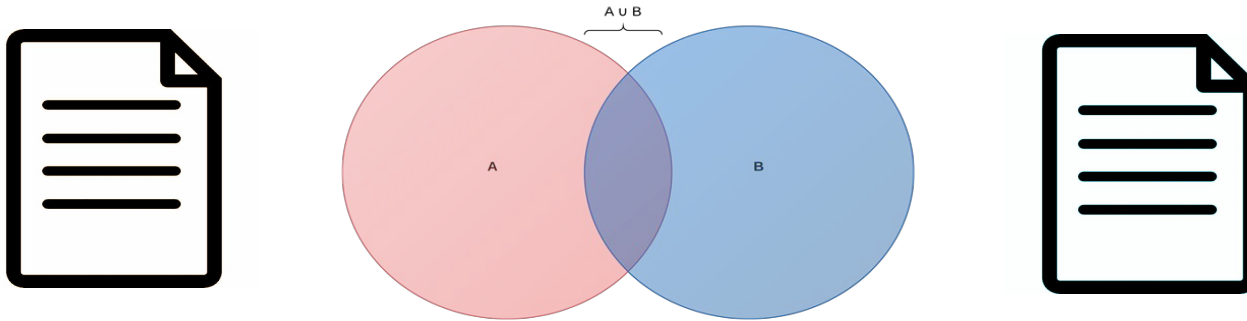


New Problem: Even if the size of signatures are small, it can be computationally expensive to find similar pairs.

E.g. 1m documents; 1,000,000 choose 2 = 500,000,000,000 pairs!

(1m documents isn't even "big data")

Document Similarity



Duplicate web pages (useful for ranking

Plagiarism

Cluster News Articles

Anything similar to documents: movie/music/art tastes, product characteristics

COVID-19 Report matching

Locality-Sensitive Hashing

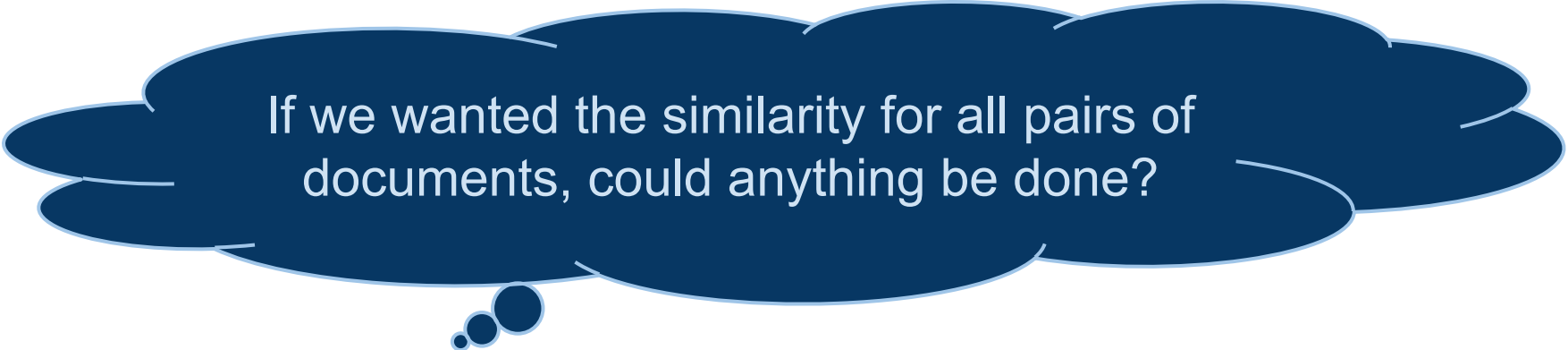
Goal: find pairs of minhashes *likely* to be similar (in order to then test more precisely for similarity).

Candidate pairs: pairs of elements to be evaluated for similarity.

Locality-Sensitive Hashing

Goal: find pairs of minhashes *likely* to be similar (in order to then test more precisely for similarity).

Candidate pairs: pairs of elements to be evaluated for similarity.



If we wanted the similarity for all pairs of documents, could anything be done?

Locality-Sensitive Hashing

Goal: find pairs of minhashes likely to be similar (in order to then test more precisely for similarity).

Candidate pairs: pairs of elements to be evaluated for similarity.

Approach: Hash multiple times over subsets of data: similar items are likely in the same bucket once.

Locality-Sensitive Hashing

Goal: find pairs of minhashes likely to be similar (in order to then test more precisely for similarity).

Candidate pairs: pairs of elements to be evaluated for similarity.

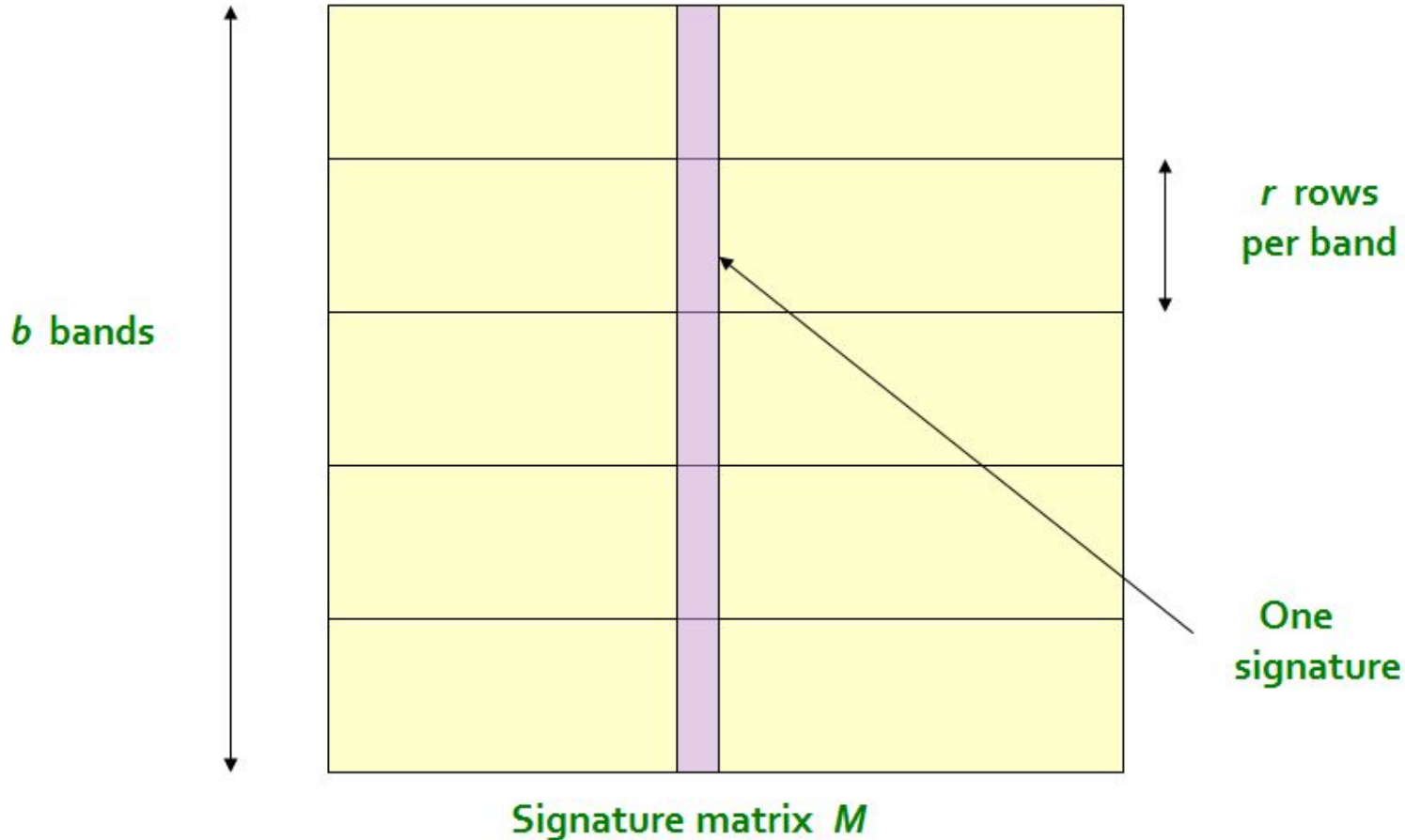
Approach: Hash multiple times over subsets of data: similar items are likely in the same bucket once.

Approach from MinHash: Hash columns of signature matrix

➡ Candidate pairs end up in the same bucket.

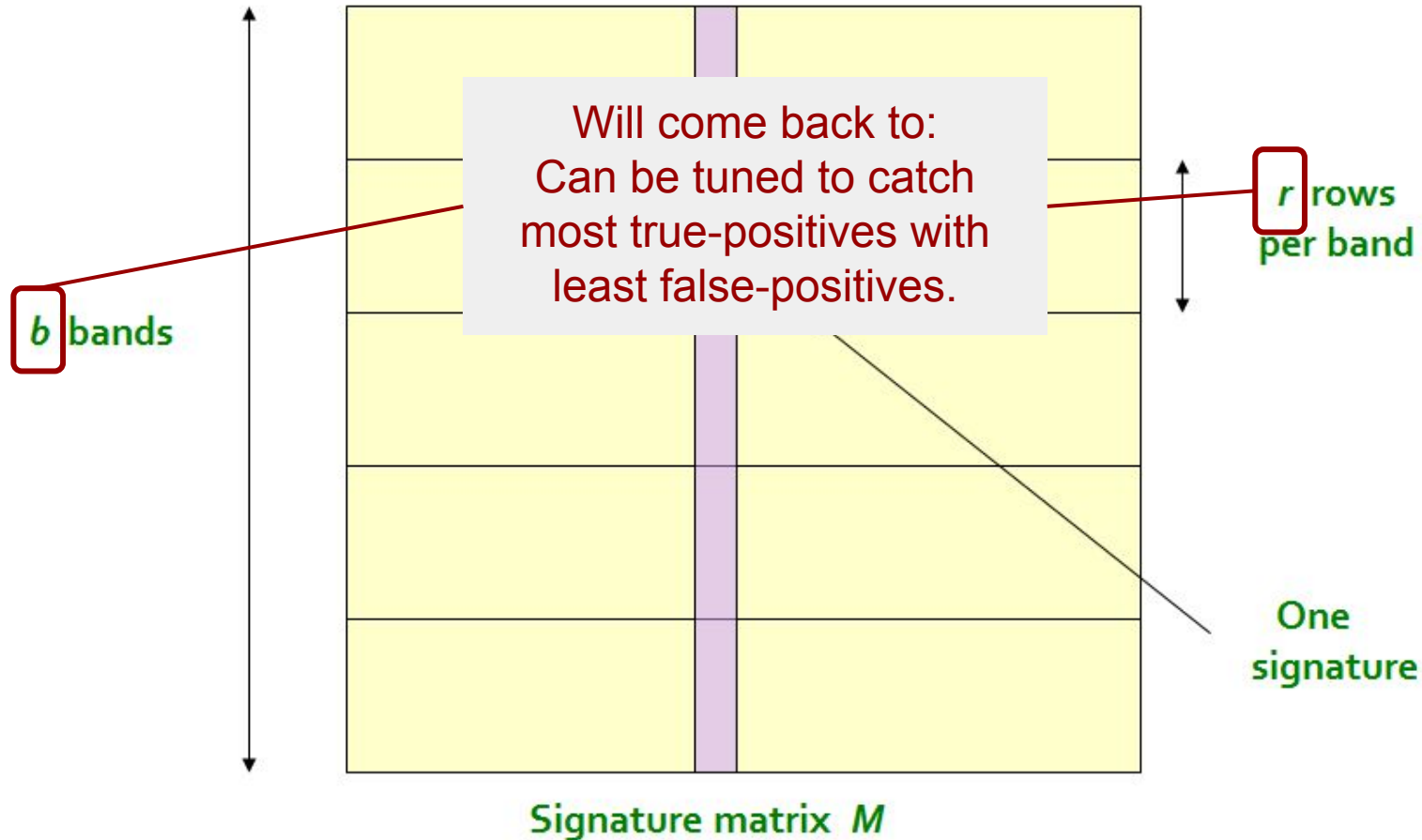
Locality-Sensitive Hashing

Step 1: Divide signature matrix into b bands



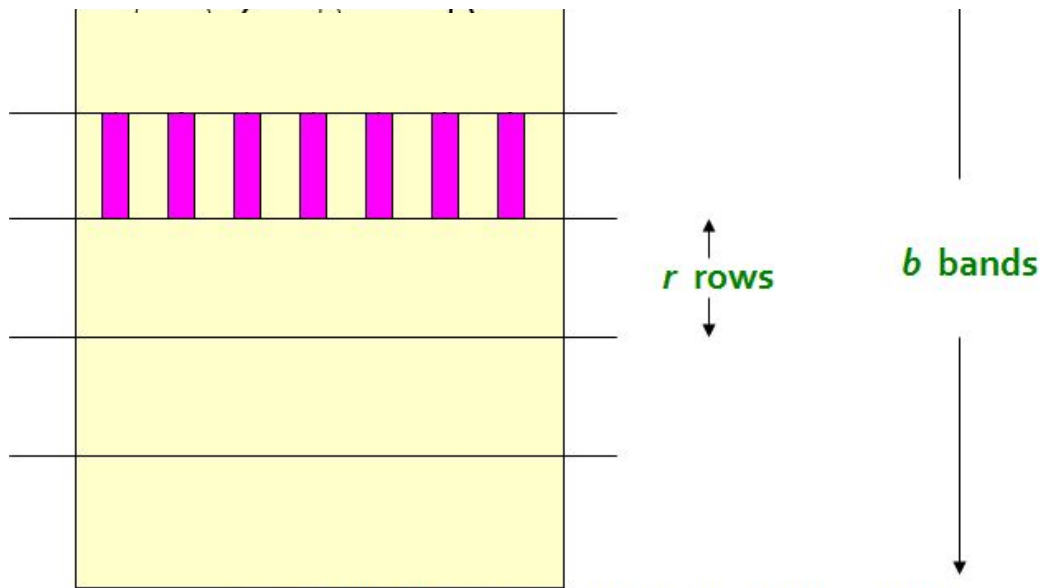
Locality-Sensitive Hashing

Step 1: Divide into b bands



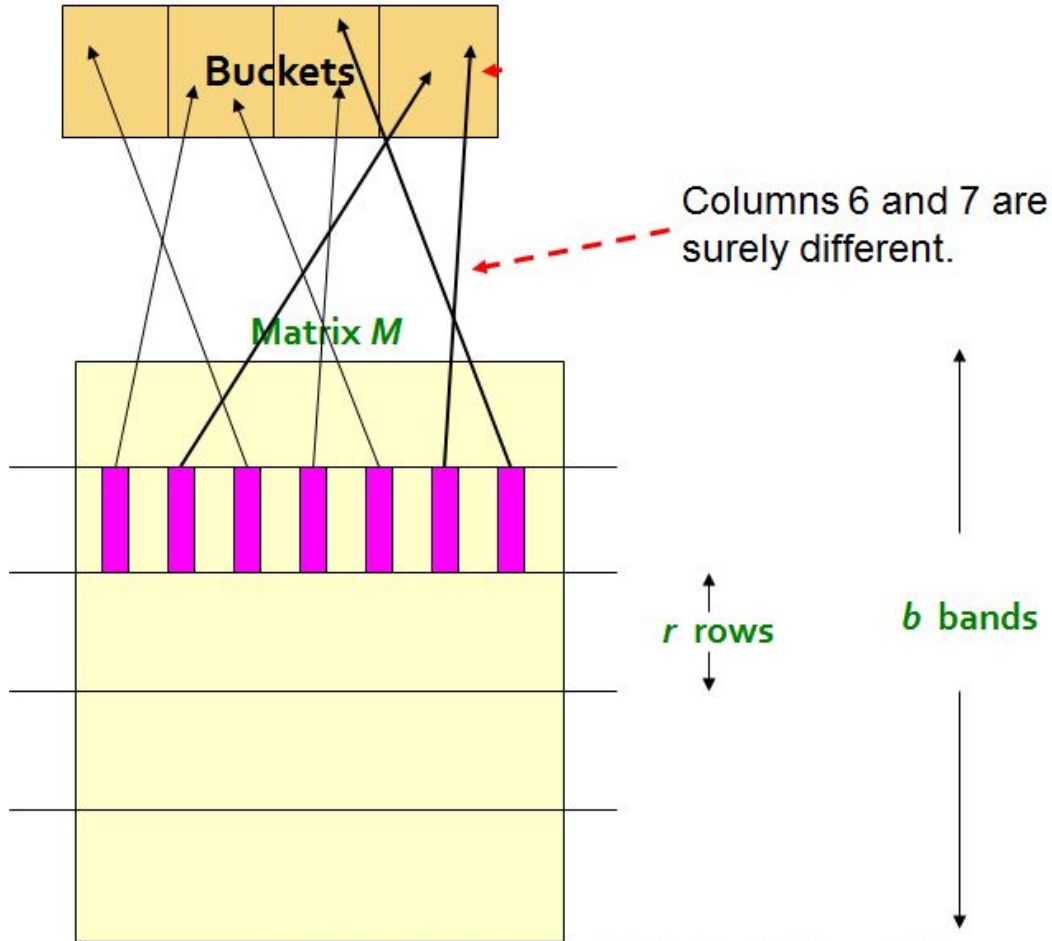
Locality-Sensitive Hashing

- Step 1: Divide into b bands
- Step 2: Hash columns within bands (one hash per band)



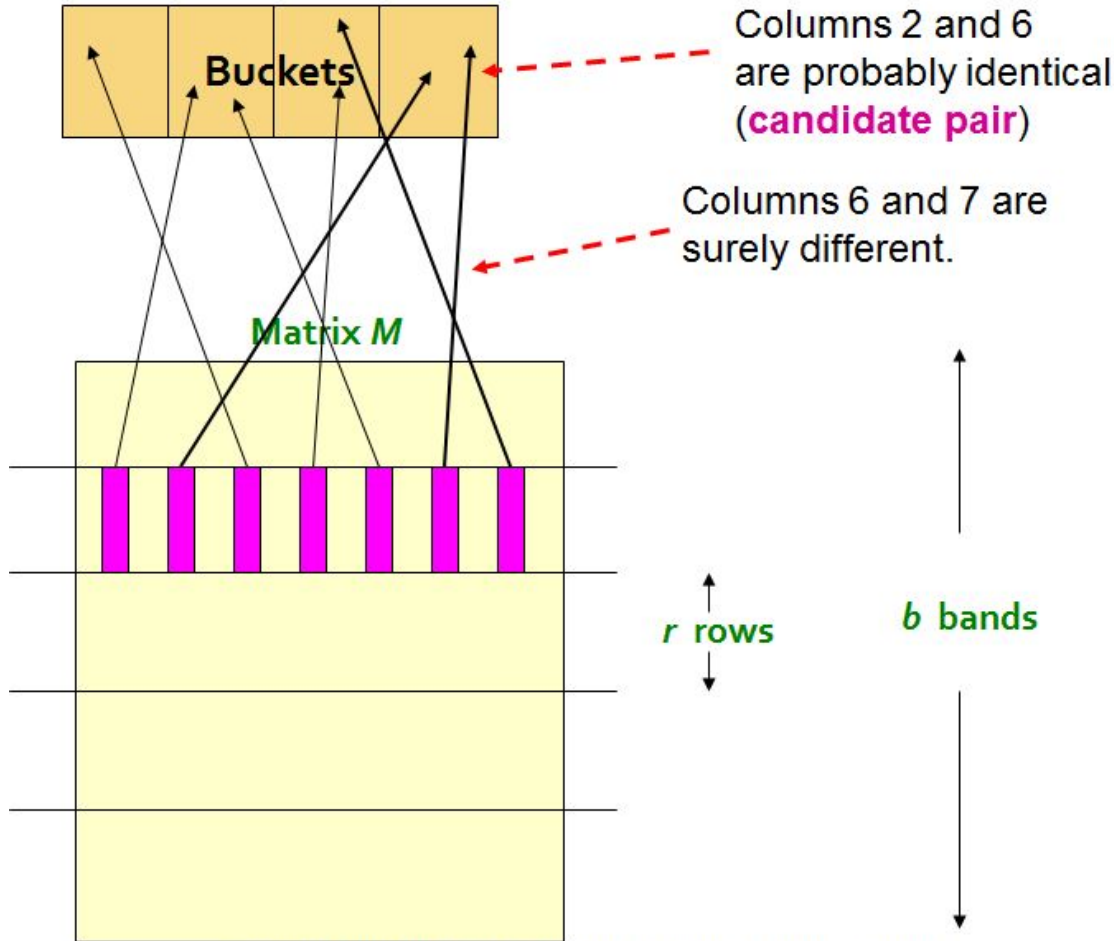
Locality-Sensitive Hashing

- Step 1: Divide into b bands
- Step 2: Hash columns within bands (one hash per band)



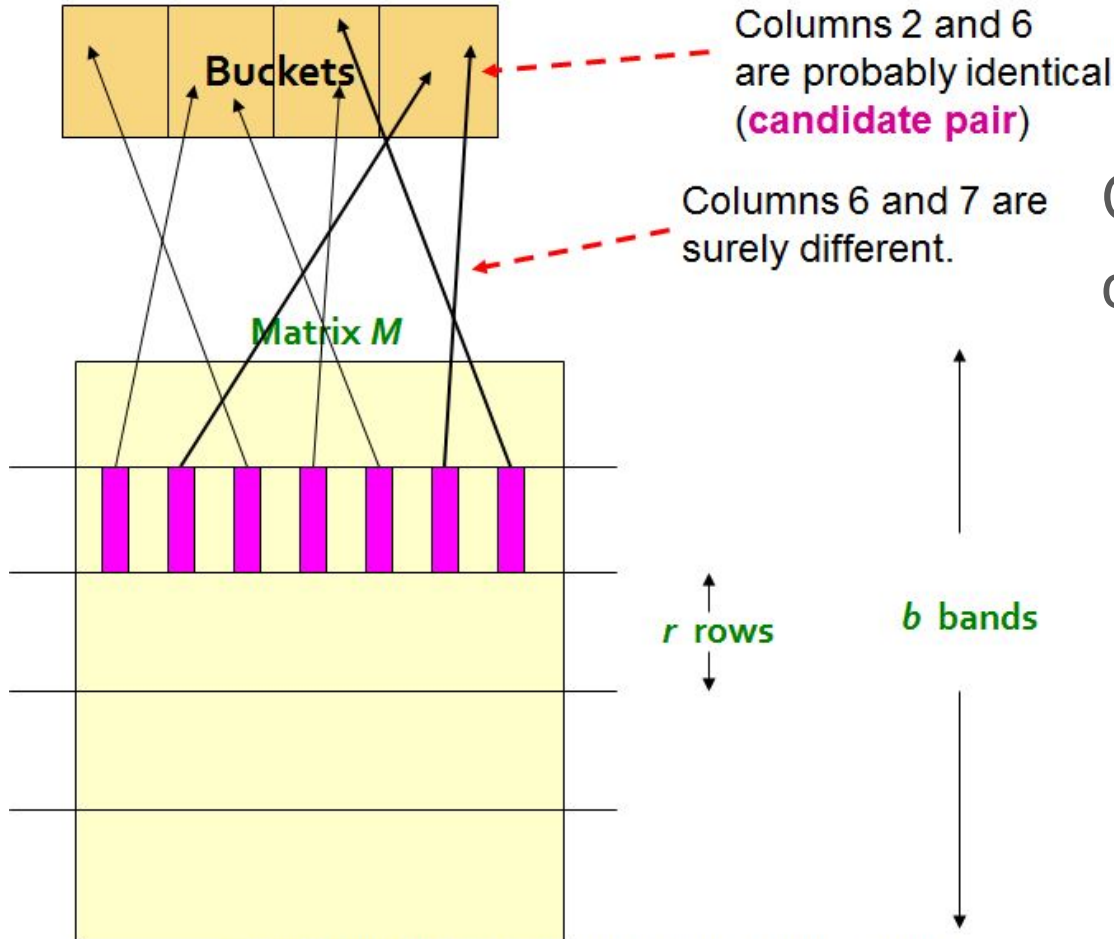
Locality-Sensitive Hashing

- Step 1: Divide into b bands
- Step 2: Hash columns within bands (one hash per band)



Locality-Sensitive Hashing

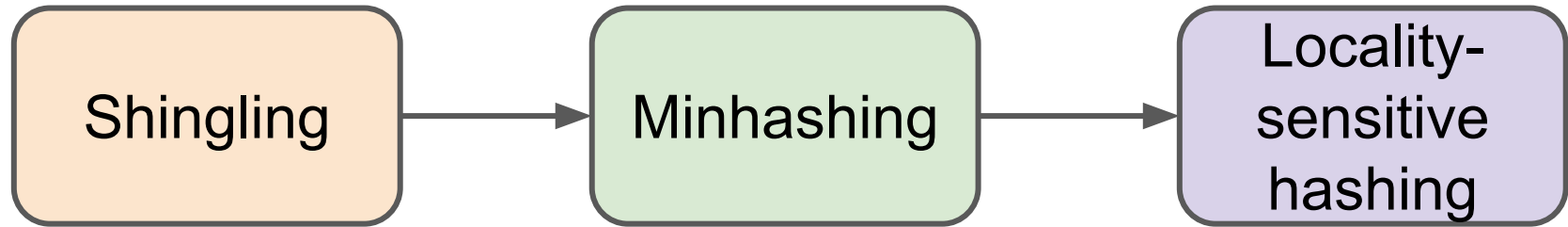
- Step 1: Divide into b bands
- Step 2: Hash columns within bands (one hash per band)



Criteria for being candidate pair:

- They end up in same bucket for at least 1 band.

Document-Similarity Pipeline



Probability of Agreement

- 100,000 documents
- 100 random permutations/hash functions/rows
 - => if 4byte integers then 40Mb to hold signature matrix
 - => still 100k choose 2 is a lot (~5billion)

Probability of Agreement

- 100,000 documents
- 100 random permutations/hash functions/rows
 - => if 4byte integers then 40Mb to hold signature matrix
 - => still 100k choose 2 is a lot (~5billion)
- 20 bands of 5 rows
- Want 80% Jaccard Similarity ; for any row $p(S_1 == S_2) = .8$

Probability of Agreement

- 100,000 documents
- 100 random permutations/hash functions/rows
 - => if 4byte integers then 40Mb to hold signature matrix
 - => still 100k choose 2 is a lot (~5billion)
- 20 bands of 5 rows
- Want 80% Jaccard Similarity ; for any row $p(S_1 == S_2) = .8$

$P(S_1 == S_2 \mid b^{(5)})$: probability S1 and S2 agree within a given band

Probability of Agreement

- 100,000 documents
- 100 random permutations/hash functions/rows
 - => if 4byte integers then 40Mb to hold signature matrix
 - => still 100k choose 2 is a lot (~5billion)
- 20 bands of 5 rows
- Want 80% Jaccard Similarity ; for any row $p(S_1 == S_2) = .8$

$P(S_1 == S_2 \mid b^{(5)})$: probability S1 and S2 agree within a given band
 $= 0.8^5 = .328$

Probability of Agreement

- 100,000 documents
- 100 random permutations/hash functions/rows
 - => if 4byte integers then 40Mb to hold signature matrix
 - => still 100k choose 2 is a lot (~5billion)
- 20 bands of 5 rows
- Want 80% Jaccard Similarity ; for any row $p(S_1 == S_2) = .8$

$$P(S_1 == S_2 \mid b^{(5)}): \text{probability } S_1 \text{ and } S_2 \text{ agree within a given band} \\ = 0.8^5 = .328 \quad \Rightarrow \quad P(S_1 \neq S_2 \mid b) = 1 - .328 = .672$$

Probability of Agreement

- 100,000 documents
- 100 random permutations/hash functions/rows
 - => if 4byte integers then 40Mb to hold signature matrix
 - => still 100k choose 2 is a lot (~5billion)
- 20 bands of 5 rows
- Want 80% Jaccard Similarity ; for any row $p(S_1 == S_2) = .8$

$P(S_1 == S_2 \mid b^{(5)})$: probability S1 and S2 agree within a given band
 $= 0.8^5 = .328 \Rightarrow P(S_1 != S_2 \mid b) = 1 - .328 = .672$

$P(S_1 != S_2)$: probability S1 and S2 do not agree in any band

Probability of Agreement

- 100,000 documents
- 100 random permutations/hash functions/rows
 - => if 4byte integers then 40Mb to hold signature matrix
 - => still 100k choose 2 is a lot (~5billion)
- 20 bands of 5 rows
- Want 80% Jaccard Similarity ; for any row $p(S_1 == S_2) = .8$

$P(S_1 == S_2 \mid b^{(5)})$: probability S1 and S2 agree within a given band
 $= 0.8^5 = .328 \Rightarrow P(S_1 \neq S_2 \mid b) = 1 - .328 = .672$

$P(S_1 \neq S_2)$: probability S1 and S2 do not agree in any band
 $= .672^{20} = .00035$

Probability of Agreement

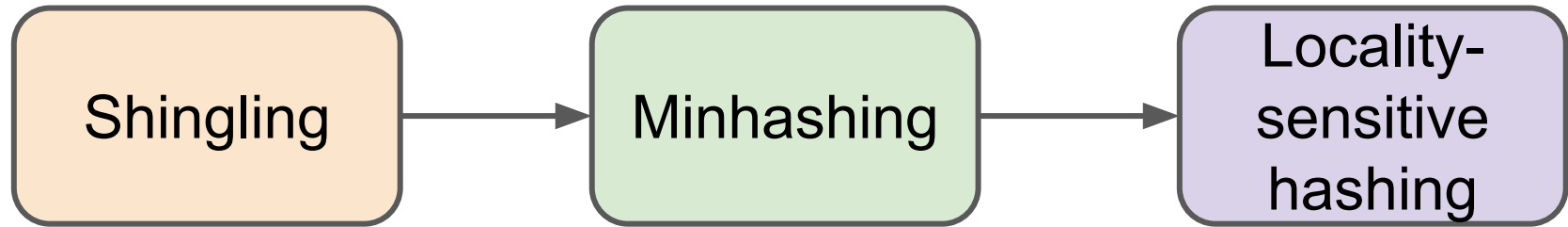
- 100,000 documents
- 100 random permutations/hash functions/rows
 - => if 4byte integers then 40Mb to hold signature matrix
 - => still 100k choose 2 is a lot (~5billion)
- 20 bands of 5 rows
- Want 80% Jaccard Similarity ; for any row $p(S_1 == S_2) = .8$

$P(S_1 == S_2 \mid b)$: probability S1 and S2 agree within a given band
 $= 0.8^5 = .328 \Rightarrow P(S_1 != S_2 \mid b) = 1 - .328 = .672$

$P(S_1 != S_2)$: probability S1 and S2 do not agree in any band
 $= .672^{20} = .00035$

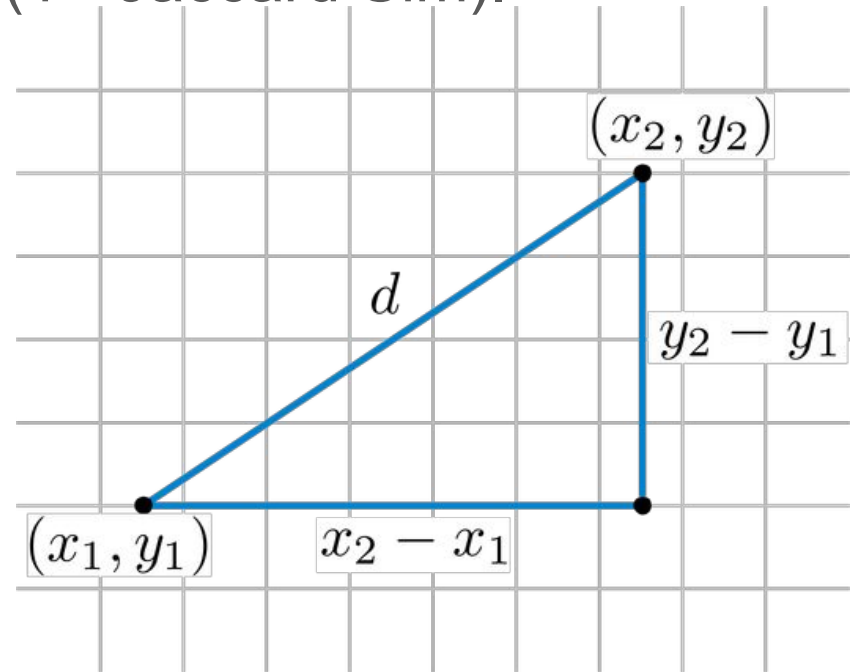
What if wanting 40% Jaccard Similarity?

Similarity Search



Distance Metrics

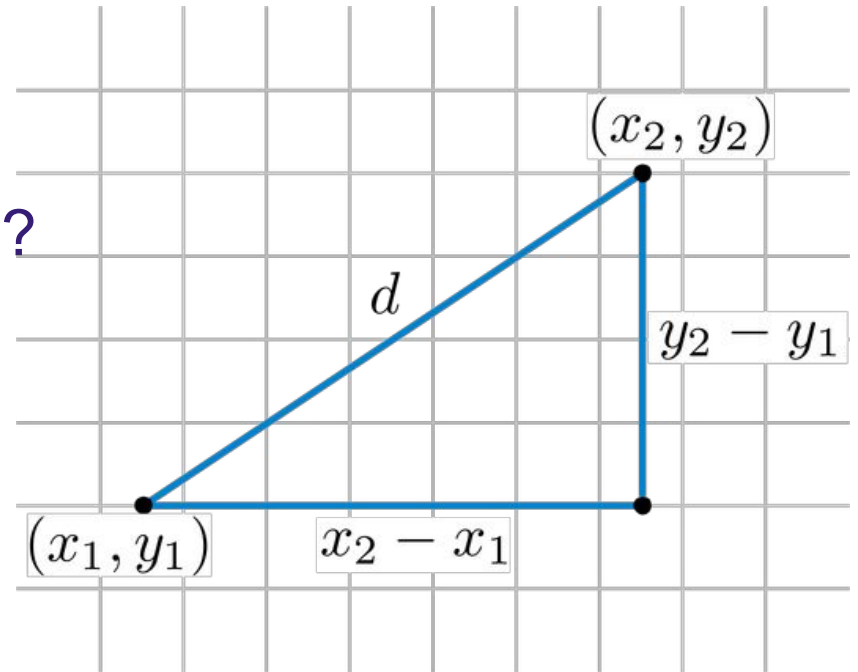
Pipeline gives us a way to find *near-neighbors* in *high-dimensional space* based on Jaccard Distance (1 - Jaccard Sim).



Distance Metrics

Pipeline gives us a way to find *near-neighbors* in *high-dimensional space* based on Jaccard Distance (1 - Jaccard Sim).

Typical properties of a distance metric, $d(\text{point1}, \text{point2})$?



Distance Metrics

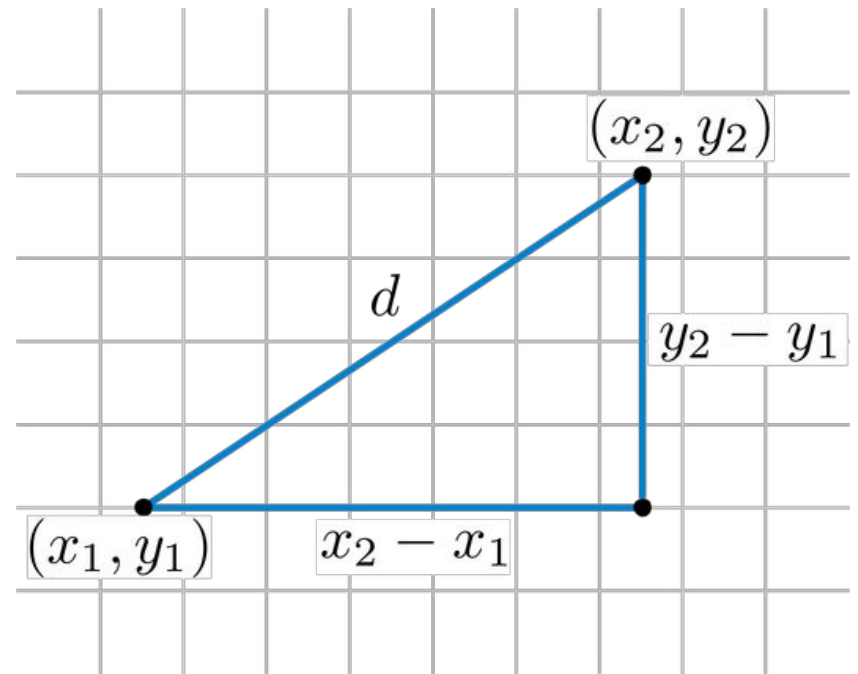
Pipeline gives us a way to find *near-neighbors* in *high-dimensional space* based on Jaccard Distance (1 - Jaccard Sim).

Typical properties of a distance metric, d :

$$d(a, a) = 0$$

$$d(a, b) = d(b, a)$$

$$d(a, b) \leq d(a, c) + d(c, b)$$



Distance Metrics

Pipeline gives us a way to find *near-neighbors* in *high-dimensional space* based on Jaccard Distance (1 - Jaccard Sim).

There are other metrics of similarity. e.g:

- Euclidean Distance
- Cosine Distance
- ...
- Edit Distance
- Hamming Distance

Distance Metrics

Pipeline gives us a way to find *near-neighbors* in *high-dimensional space* based on Jaccard Distance (1 - Jaccard Sim).

There are other metrics of similarity. e.g:

- Euclidean Distance

$$distance(X, Y) = \sqrt{\sum_i^n (x_i - y_i)^2} \quad (\text{"L2 Norm"})$$

- Cosine Distance

...

- Edit Distance

- Hamming Distance

Distance Metrics

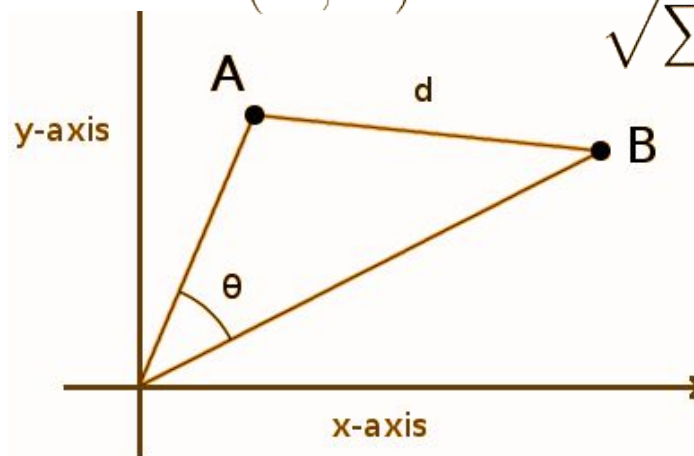
Pipeline gives us a way to find *near-neighbors* in *high-dimensional space* based on Jaccard Distance (1 - Jaccard Sim).

There are other metrics of similarity. e.g:

- Euclidean Distance
- Cosine Distance
- ...
- Edit Distance
- Hamming Distance

$$distance(X, Y) = \sqrt{\sum_i^n (x_i - y_i)^2} \quad (\text{"L2 Norm"})$$

$$distance(X, Y) = 1 - \frac{\sum x_i y_i}{\sqrt{\sum x_i^2} \sqrt{\sum y_i^2}}$$



Distance Metrics

Locality Sensitive Hashing - Theory

LSH Can be generalized to many distance metrics by converting output to a probability and providing a lower bound on probability of being similar.

E.g. for euclidean distance:

- Choose random lines (analogous to hash functions in minhashing)
- Project the two points onto each line; match if two points within an interval