Locality-Sensitive Hashing (LSH)

Universitat Pompeu Fabra Barcelona

Mining Massive Datasets

Materials provided by Prof. Carlos Castillo — https://chato.cl/teach

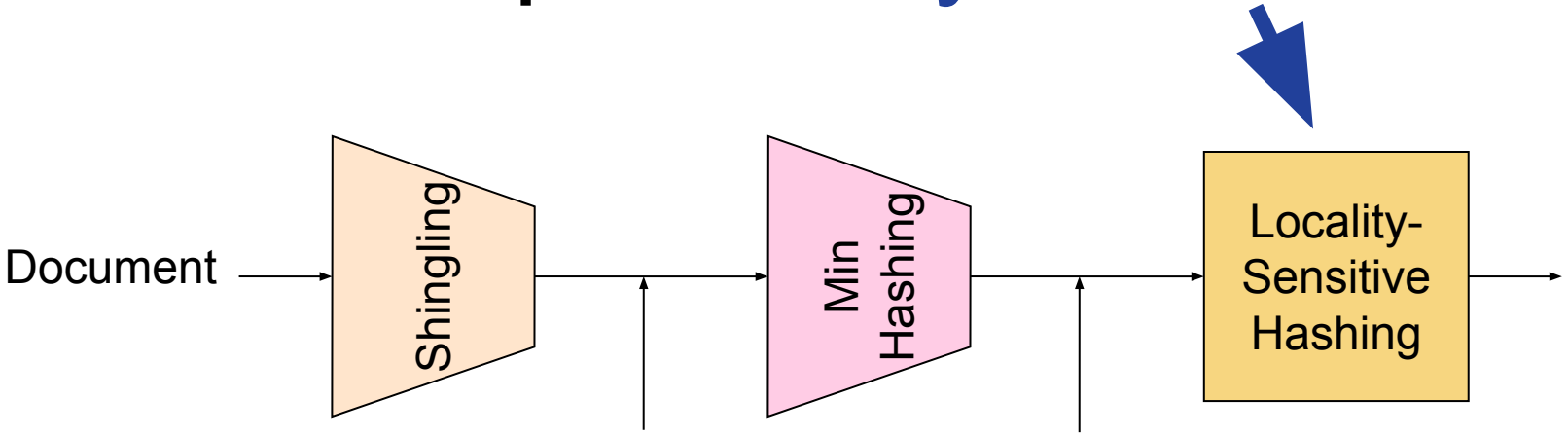
Instructor: Dr. Teodora Sandra Buda — https://tbuda.github.io/

Source for this deck

- Mining of Massive Datasets 2nd edition (2014) by Leskovec et al. (Chapter 3) [slides ch3]

Locality-sensitive hashing

Final step: locality-sensitive hashing



Sets of ***k*** *letters or words* that appear consecutively in the document

Signatures: short integer vectors that represent the sets, and reflect their similarity

Candidate pairs: those pairs of signatures that we need to test for similarity

LSH: first idea

- **Goal:** Find documents with Jaccard similarity at least ***s*** (for some similarity threshold, e.g., ***s***=0.8)

- **LSH – General idea:** Use a function ***f(x,y)*** that tells whether ***(x**,**y)*** is a "***candidate pair***", *with* similarity **likely to be ≥ *s***

- We will compute an auxiliary structure over ***M***
  1) Hash each column of the signature matrix ***M*** to a bucket
  2) A pair of columns that hashes to the same bucket is a **candidate pair**

Signature matrix *M*

d1	d2	d3	d4
2	1	4	1
1	2	1	2
2	1	2	1

Selecting candidates

- **Pick a similarity threshold *s* (0 < s < 1)**

- Columns ***x*** and ***y*** of ***M*** are a **candidate pair** if their signatures agree
  (***M*** **(*i*, *x*)** **=** ***M*** **(*i*, *y*)**) on at least fraction ***s*** of their rows

- Remember we showed that documents ***x*** and ***y*** will have a similar (Jaccard) similarity as their signatures

Signature matrix *M*

d1	d2	d3	d4
2	1	4	1
1	2	1	2
2	1	2	1

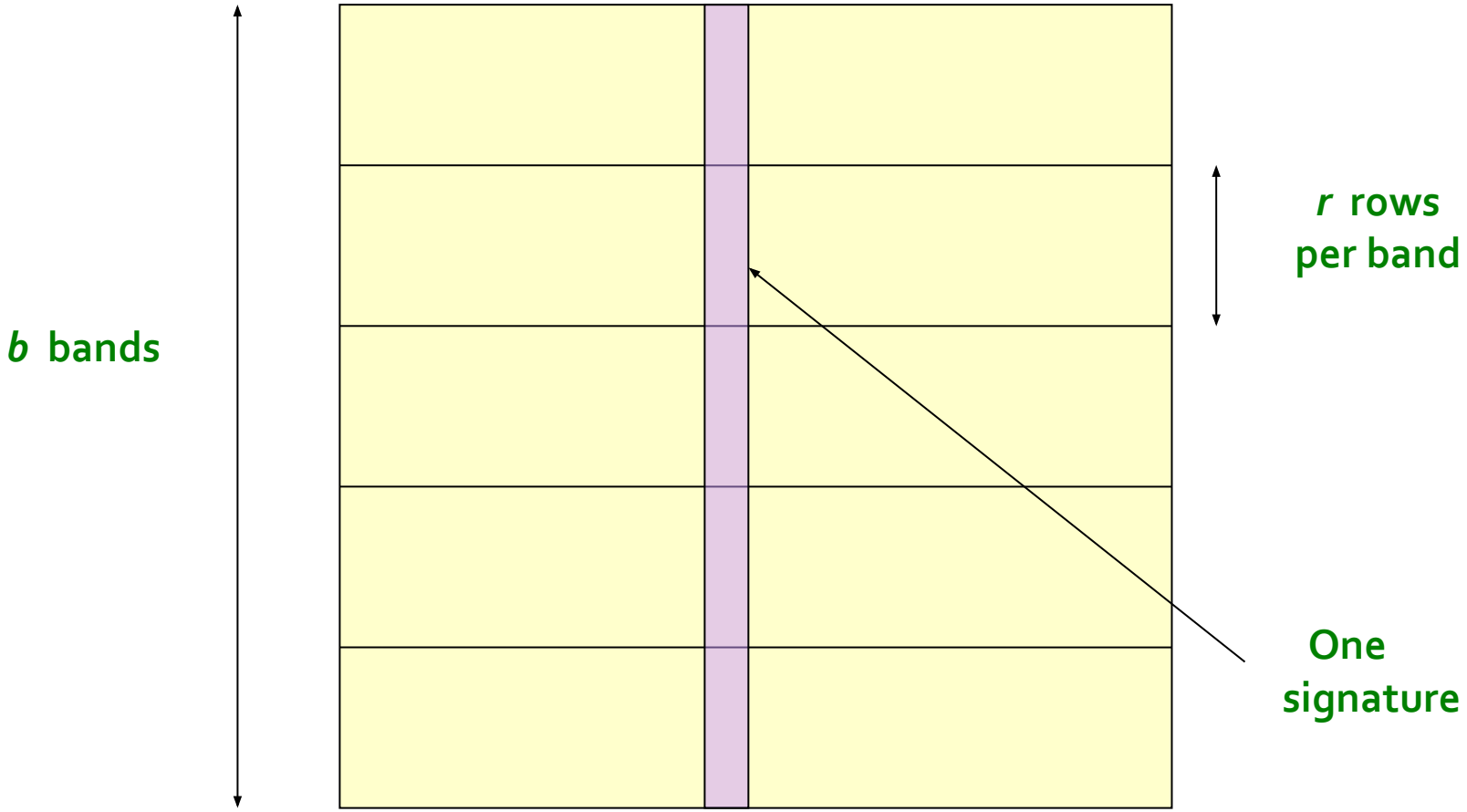
Creating buckets of similar documents

- **Hash columns of signature matrix *M***
- Make sure that (only) **similar columns** are likely to **hash to the same bucket**, with high probability
- **Only check the pairs that hash to the same bucket**

Signature matrix *M*

d1	d2	d3	d4
2	1	4	1
1	2	1	2
2	1	2	1

Partition M into b bands of size r



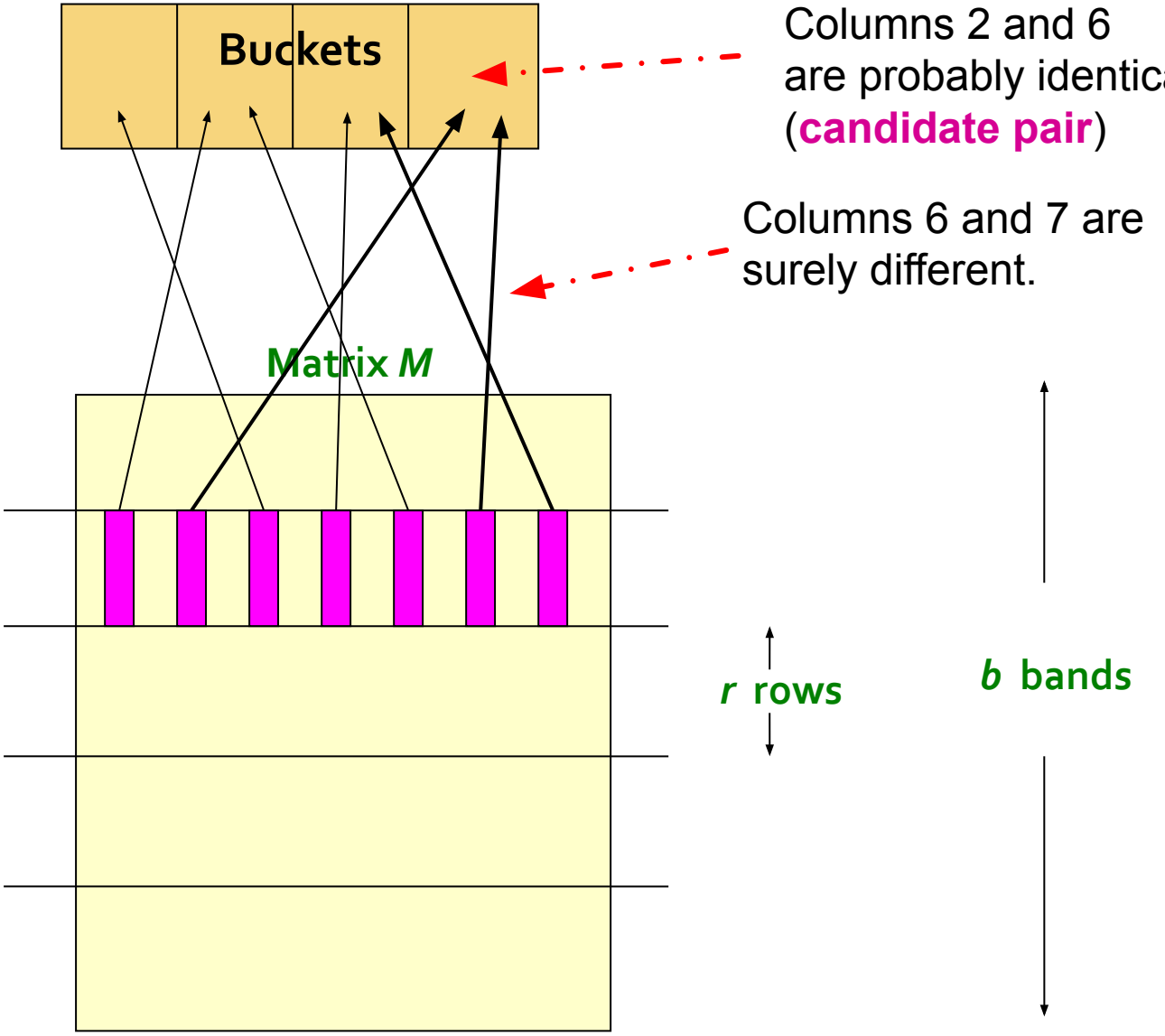
Partition M into b bands of size r (cont.)

- Remember that M has one column per document and as many rows as the signature length
- Partition matrix ***M*** into ***b*** bands of ***r*** rows
- For each band, hash its portion of each column to a hash table with ***k*** buckets
  - If ***k*** is large we use more memory but there are less spurious collisions
- ***Candidate*** column pairs are those that hash to the same bucket for **≥ 1** band
- Tune ***b*** and ***r*** to catch many similar pairs, but few non-similar pairs

Signature matrix *M*

d1	d2	d3	d4
2	1	4	1
1	2	1	2
2	1	2	1

Hashing bands



Simplifying assumption: no collisions (no false positives)

- We will assume there are **enough buckets** that columns are unlikely to hash to the same bucket unless they are **identical** in a particular band
- Hereafter, we assume that "**same bucket**" means "**identical in that band**"
- Assumption needed only to simplify analysis, not for correctness of algorithm

Computing LSH errors

- Assume the following case:
  - 100,000 documents = 100,000 columns in M
  - 100 integers/signature = 100 rows in M
  - 100,000 x 100 = 10M integers x 4 bytes/integer = 40 Mb of disk space
- Choose b = 20 bands of r = 5 integers/band
  - Note that b x r should be the number of integers in each signature
- **Suppose our goal** is to find pairs of documents that are at least 0.8 similar

Computing LSH errors (cont.)

- **Find pairs having at least 0.8 similarity with b=20, r=5**
- Whenever sim(C1, C2) > s, we want C1, C2 to be a candidate pair
  - We want them to hash to at least 1 common bucket (at least one band is identical)
- Probability C1, C2 identical in one particular band: $(0.8)^5 = 0.328$
- Probability C1, C2 are not similar in any of the 20 bands:
  - $(1\text{-}0.328)^{20} = 0.00035$
  - i.e., about 1/3000th of the 80%-similar column pairs are false negatives (we will miss them)
- We would find 99.965% pairs of truly similar documents

Computing LSH errors (cont.)

- **Find pairs having at least 0.8 similarity with b=20, r=5**
- Whenever sim(C1, C2) < s, we do not want C1, C2 to be a candidate pair
- Suppose sim(C1, C2) = 0.3; the probability that C1, C2 are identical in one particular band:
  - $(0.3)^5 = 0.00243$
- Probability C1, C2 identical in at least 1 of 20 bands:
  - $1 - (1 - 0.00243)^{20} = 0.0474$
- In other words, approximately 4.74% pairs of docs with similarity 0.3 end up becoming candidate pairs -- they are false positives since we will have to examine them but then it will turn out their similarity is below threshold s

Designing a good LSH scheme

- Tune the number of ***permutations (b x 3), the number of bands (b), and the number of rows per band (r)*** to
  - get almost all pairs with similar signatures
  - eliminate most pairs that do not have similar signatures

- After finding candidates, we always have to check in main memory that **candidate pairs** really do have **similar signatures**

Summary

Things to remember

- **Locality-Sensitive Hashing** allows us to focus on pairs of signatures likely to be from similar documents
- Remember the general idea and what are bands/rows
- *Additional materials on LSH available from the theory page of the course*

Exercises for TT08-TT09

- Mining of Massive Datasets 2nd edition (2014) by Leskovec et al.
  - Exercises 3.1.4 (Jaccard similarity)
  - Exercises 3.2.5 (Shingling)
  - Exercises 3.3.6 (Min hashing)
  - Exercises 3.4.4 (Locality-sensitive hashing)