

LSML #6

Трюки с хэшированием (продолжаем)

Хэширование полезно

- Уже знаем:
 - Хэширование признаков (vowpal wabbit)
- Узнаем сегодня для **больших** множеств:
 - Определение принадлежности множеству (Bloom Filter)
 - Оценка частот элементов множества в потоке (Count-min sketch)
 - Оценка похожести двух множеств (MinHash)
 - Отбор самых похожих множеств на заданное (LSH)

Будем решать приближенно!

MinHash (1997)

- **Оценка похожести двух множеств (больших)**
- Мера Жаккара для двух множеств: $J(S_1, S_2) = \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|}$
- Представим множества S_1 и S_2 как столбцы в пространстве универсальных элементов E_i :

	S_1	S_2	
E_1	1	1	
E_2	0	1	$J(S_1, S_2) = \frac{2}{3}$
E_3	1	1	
E_4	0	0	

MinHash

Всего 4 группы элементов:


	S_1	S_2
A	1	1
B	1	0
C	0	1
D	0	0

$|A| = A$

$$J(S_1, S_2) = \frac{A}{A + B + C}$$

MinHash

Всего 4 группы элементов:



	S_1	S_2
A	1	1
B	1	0
C	0	1
D	0	0

$|A| = A$

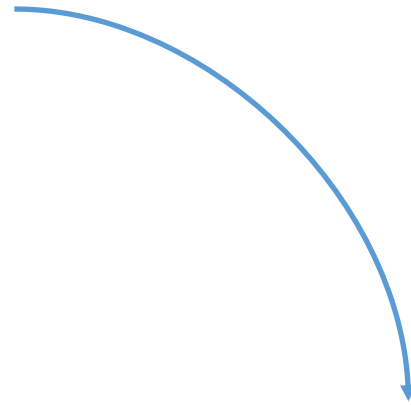
- Случайно перемешаем строки
- Хэш $h(S_i)$ = индекс первой строки с 1

$$P(h(S_1) = h(S_2)) = J(S_1, S_2)$$

- После случайной перестановки строк идем до **первой** строки не из группы D
- При этом хэши совпадут только если это строка из группы A

$$J(S_1, S_2) = \frac{A}{A + B + C}$$

MinHash сигнатура

- По одному хэшу ничего не понятно (совпал или нет)
 - Выберем k случайных перестановок
 - Сигнатура $sig(S) = (h_1(S), \dots, h_k(S))$
 - Пусть $sim(sig(S_i), sig(S_j))$ – процент совпадений в сигнатурах
 - Мат. ожидание этой похожести – $J(S_i, S_j)$
 - В силу ЗБЧ с ростом k ошибка убывает как $= O\left(\frac{1}{\sqrt{k}}\right)$
 - Для ошибки 0.05 хватит ≈ 400 хэш-функций
- 

MinHash сигнатура: пример

	C_1	C_2	C_3
R_1	1	0	1
R_2	0	1	1
R_3	1	0	0
R_4	1	0	1
R_5	0	1	0

Signatures

	S_1	S_2	S_3
Perm 1 = (12345)	1	2	1
Perm 2 = (54321)	4	5	4
Perm 3 = (34512)	3	5	4

Similarities

	1-2	1-3	2-3
Col-Col	0.00	0.50	0.25
Sig-Sig	0.00	0.67	0.00

Заменяли сравнение больших множеств на
сравнение маленьких сигнатур!

MinHash сигнатура: имплементация

- Генерировать перестановку элементов дорого!
- Давайте возьмем хорошую хэш-функцию $h(x)$ (murmur32 например)
- Сортировка по индексу корзинки $h(x)$ даст случайную перестановку!
- Тогда MinHash считается легко: хэшируем все элементы и берем минимальное значение хэша

MinHash сигнатура: имплементация

	C_1	C_2
R_1	1	0
R_2	0	1
R_3	1	1
R_4	1	0
R_5	0	1

Хэш-функции:

$$h(x) = x \bmod 5$$

$$g(x) = 2x+1 \bmod 5$$

	C_1 slots	C_2 slots	
$h(1) = 1$	1	-	Один проход
$g(1) = 3$	3	-	
$h(2) = 2$	1	2	
$g(2) = 0$	3	0	
$h(3) = 3$	1	2	Сигнатуры
$g(3) = 2$	2	0	
$h(4) = 4$	1	2	
$g(4) = 4$	2	0	
$h(5) = 0$	1	0	
$g(5) = 1$	2	0	

MinHash сигнатура: использование

- Придумали для поисковика AltaVista для поиска дубликатов веб-страниц и удаления из выдачи (страница как мешок слов)
- Применяют в биологии для сравнения геномов

Locality-Sensitive Hashing (LSH)

- **Отбор самых похожих множеств (много больших) на заданное**
- Если множеств много, то медленно сравнивать даже MinHash сигнатуры
- Ускорим сравнение MinHash сигнатур!
- **Идея: придумаем хэш, который в одну корзинку будет складывать похожие**

LSH для MinHash сигнатур

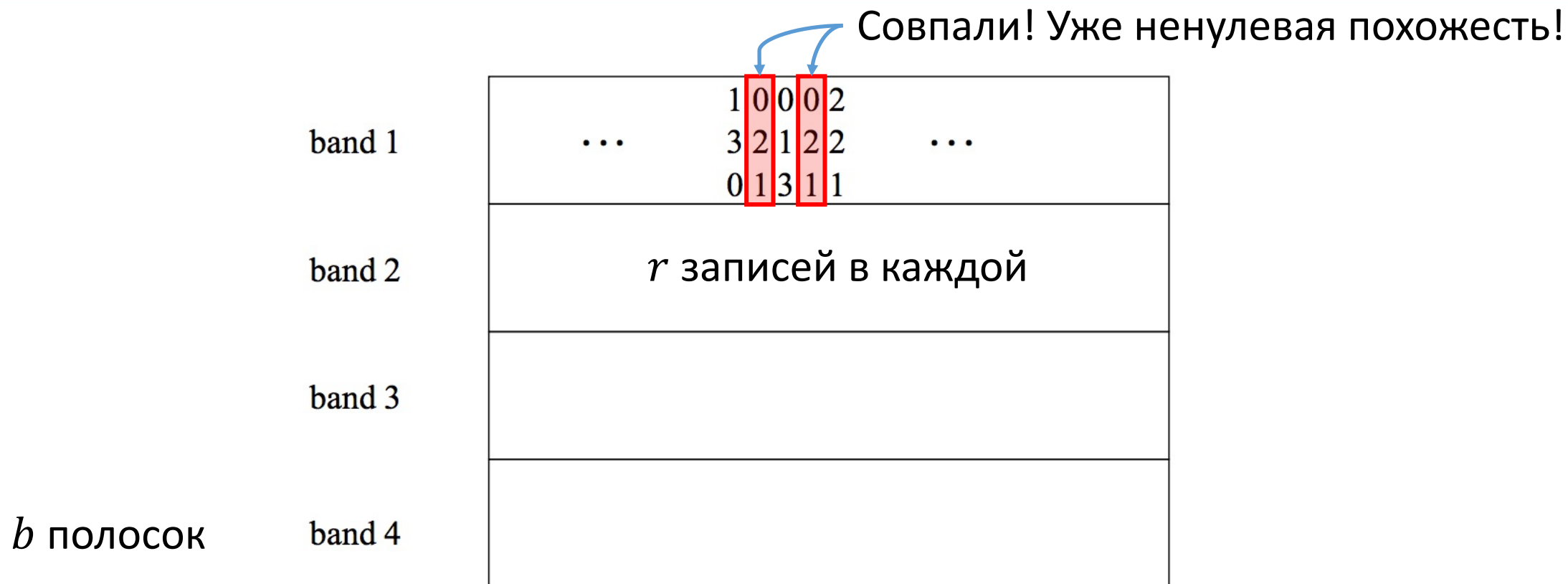


Figure 3.6: Dividing a signature matrix into four bands of three rows per band

Заменим сравнение в полосе на совпадение хэшей **кусочков сигнатур**
(пусть корзинок много, коллизии маловероятны)!

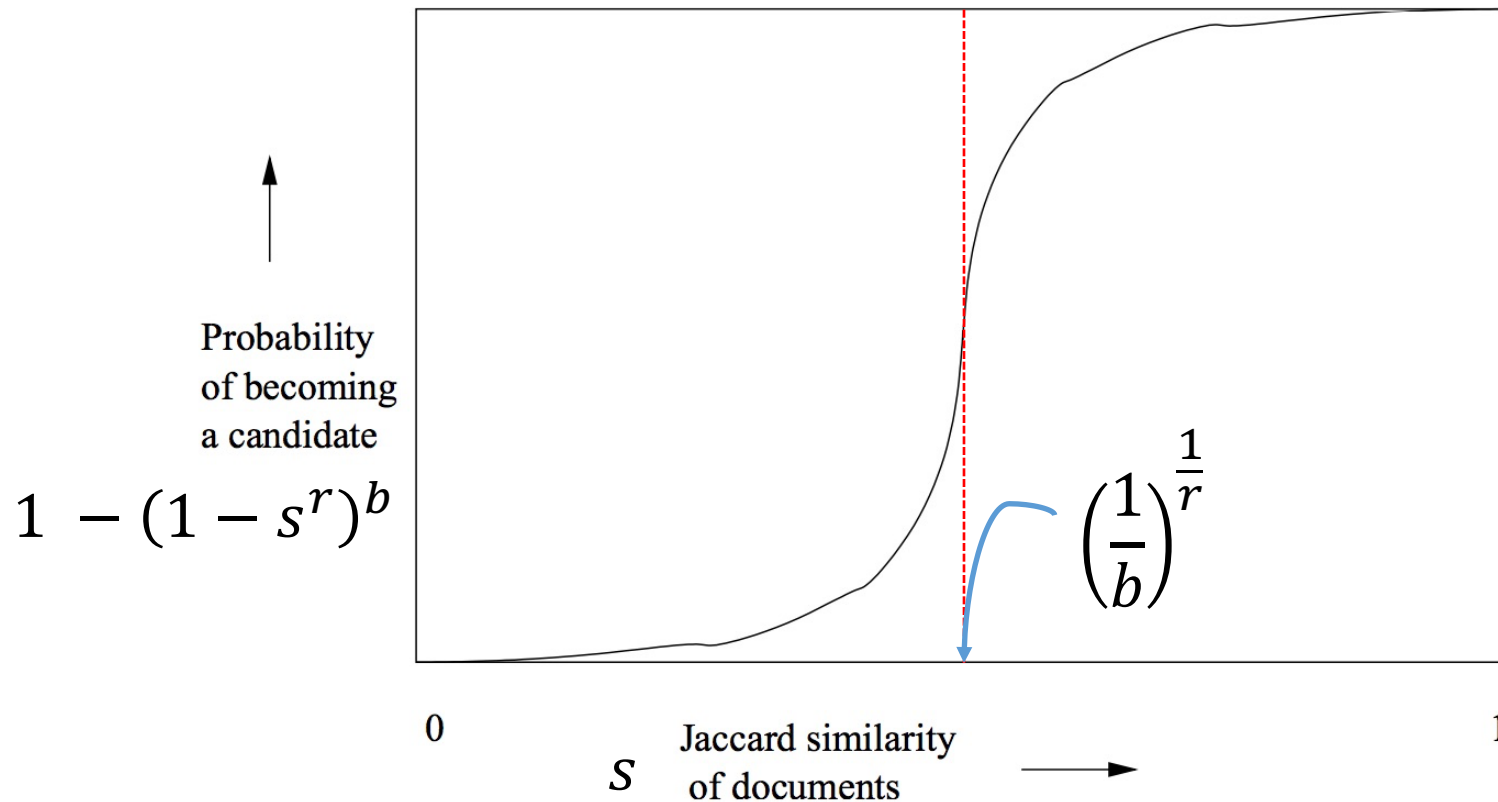
LSH для MinHash сигнатур

- Пусть два множества имеют $J(S_1, S_2) = s$
- Оценим вероятность того, что хэши **хотя бы в одной** полосе совпадут:
 1. The probability that the signatures agree in all rows of one particular band is s^r .
 2. The probability that the signatures disagree in at least one row of a particular band is $1 - s^r$.
 3. The probability that the signatures disagree in at least one row of each of the bands is $(1 - s^r)^b$.
 4. The probability that the signatures agree in all the rows of at least one band, and therefore become a candidate pair, is $1 - (1 - s^r)^b$.

Кандидатов будем проверять уже более тщательно!

LSH для MinHash сигнатур

- Вероятность стать кандидатом на *медленную* проверку похожести:



$$b = 20 \quad r = 5$$

s	$1 - (1 - s^r)^b$
.2	.006
.3	.047
.4	.186
.5	.470
.6	.802
.7	.975
.8	.9996

LSH

- Существуют реализации для других мер:
 - Евклидова
 - Косинусная
 - ...
- Позволяет быстро сузить круг похожих множеств для медленной проверки

LSH: примеры применения

- Google News (для поиска похожих сюжетов)
- Поиск похожих картинок (по нейросетевым сигнатурам)
- Audio similarity identification

Хэширование полезно

- Уже знаем:
 - Хэширование признаков (Vowpal Wabbit)
 - Определение принадлежности множеству (Bloom Filter)
 - Оценка частот элементов множества в потоке (Count-min sketch)
 - Оценка похожести двух множеств (MinHash)
 - Отбор самых похожих множеств на заданное (LSH)