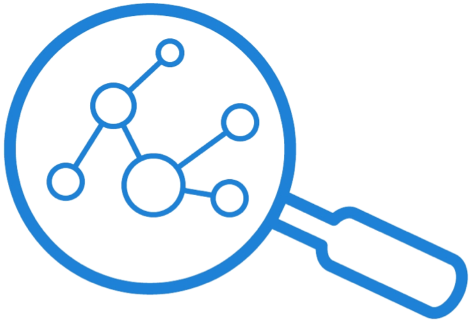
数据科学与大数据技术的数学基础

第六讲



计算机学院
余皓然

2023/5/12

课程内容

Part1 随机化方法
一致性哈希 布隆过滤器 CM Sketch方法 最小哈希 欧氏距离下的相似搜索 Jaccard相似度下的相似搜索
Part2 谱分析方法
主成分分析 奇异值分解 谱图论
Part3 最优化方法
压缩感知

Jaccard相似度下的相似搜索

Jaccard相似度

上讲回顾

欧几里得距离（Euclidean Distance）/l_2距离：

若$x, y \in \mathbb{R}^d$，它们之间的欧式距离为

$$D_{euclidean}(x,y) = ||x-y||_2 = \sqrt{\sum_{i=1}^{d}(x(i)-y(i))^2}.$$

相似搜索：k维树（若是高维数据，先通过JL转换降维成低维数据）

上讲回顾

欧几里得距离（Euclidean Distance）$/l_2$距离：

若$x, y \in \mathbb{R}^d$，它们之间的欧式距离为

$$D_{euclidean}(x,y) = ||x-y||_2 = \sqrt{\sum_{i=1}^{d}(x(i)-y(i))^2}.$$

相似搜索：k维树（若是高维数据，先通过JL转换降维成低维数据）

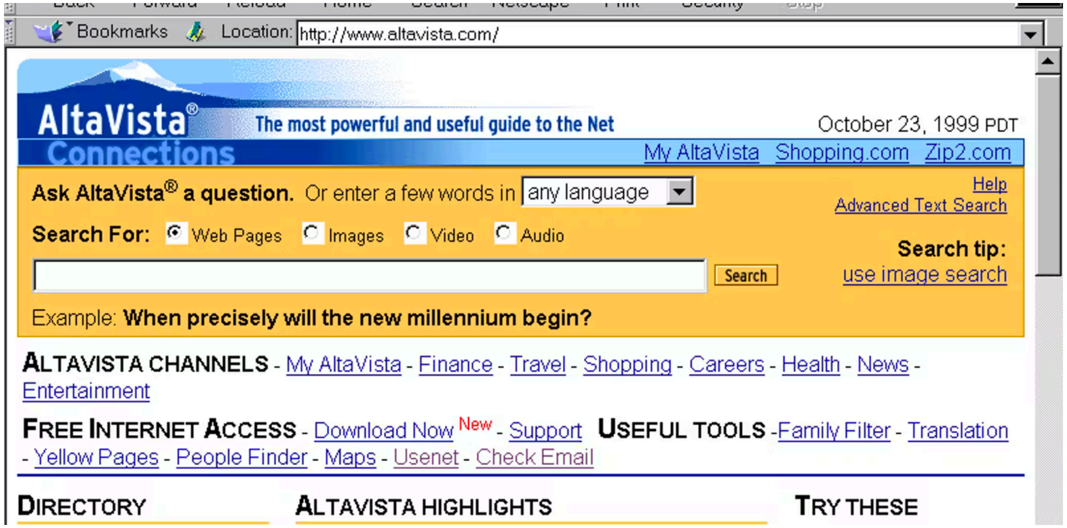
当数据的类别是集合怎么做相似搜索？

文档（比如一条微博）可以看作是词语的集合

给定一堆文档（比如若干条微博）和一个新的文档，如何做相似搜索？

Jaccard相似度

当数据的类别是集合怎么做相似搜索?



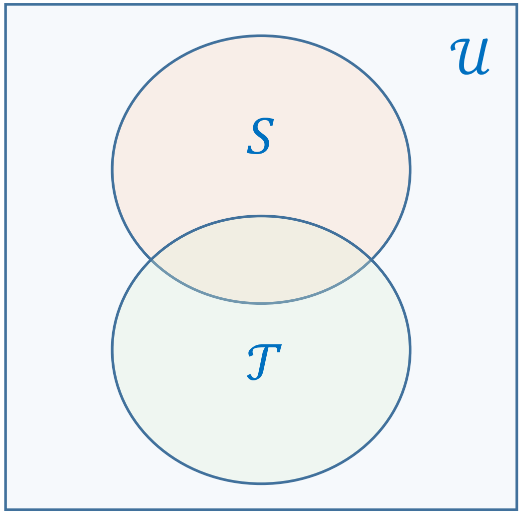
上世纪九十年代，网页搜索引擎Alta Vista需要滤除相似的网页

Alta Vista将每个网页视作一个集合，采取Jaccard相似度定义不同网页的相似度

Jaccard相似度

杰卡德相似度（Jaccard Similarity）：刻画两个集合之间的距离

$$J(S,T) = \frac{|S \cap T|}{|S \cup T|}$$



Jaccard相似度

$$J(S,T) = \frac{|S \cap T|}{|S \cup T|}$$

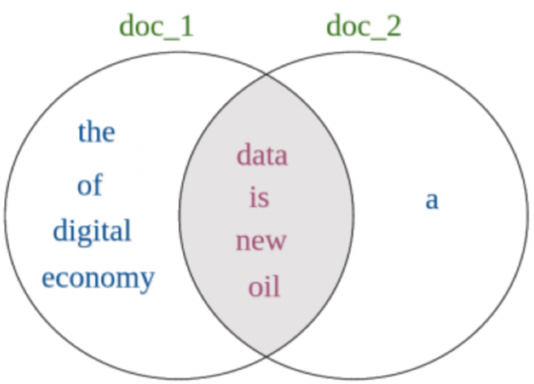
```
doc_1 = "Data is the new oil of the digital economy"
doc_2 = "Data is a new oil"
```

```
words_doc1 = {'data', 'is', 'the', 'new', 'oil', 'of', 'digital', 'economy'}
words_doc2 = {'data', 'is', 'a', 'new', 'oil'}
```

$$J(doc_1, doc_2) = \frac{\{\text{'data', 'is', 'the', 'new', 'oil', 'of', 'digital', 'economy'}\} \bigcap \{\text{'data', 'is', 'a', 'new', 'oil'}\}}{\{\text{'data', 'is', 'the', 'new', 'oil', 'of', 'digital', 'economy'}\} \bigcup \{\text{'data', 'is', 'a', 'new', 'oil'}\}}$$

$$= \frac{\{\text{'data', 'is', 'new', 'oil'}\}}{\{\text{'data', 'a', 'of', 'is', 'economy', 'the', 'new', 'digital', 'oil'}\}}$$

$$= \frac{4}{9} = 0.444$$



例子取自Bhavika Kanani <Jaccard Similarity – Text Similarity Metric in NLP> from studymachinelearning.com

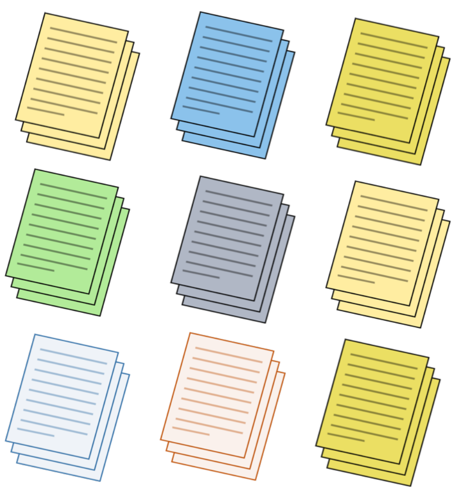
Jaccard相似度

杰卡德相似度（Jaccard Similarity）：刻画两个集合之间的距离

$$J(S,T)=\frac{|S\cap T|}{|S\cup T|}$$

如何在Jaccard相似度下做相似搜索?

为简化对方法的介绍，假设每个集合（文件）不包含重复元素

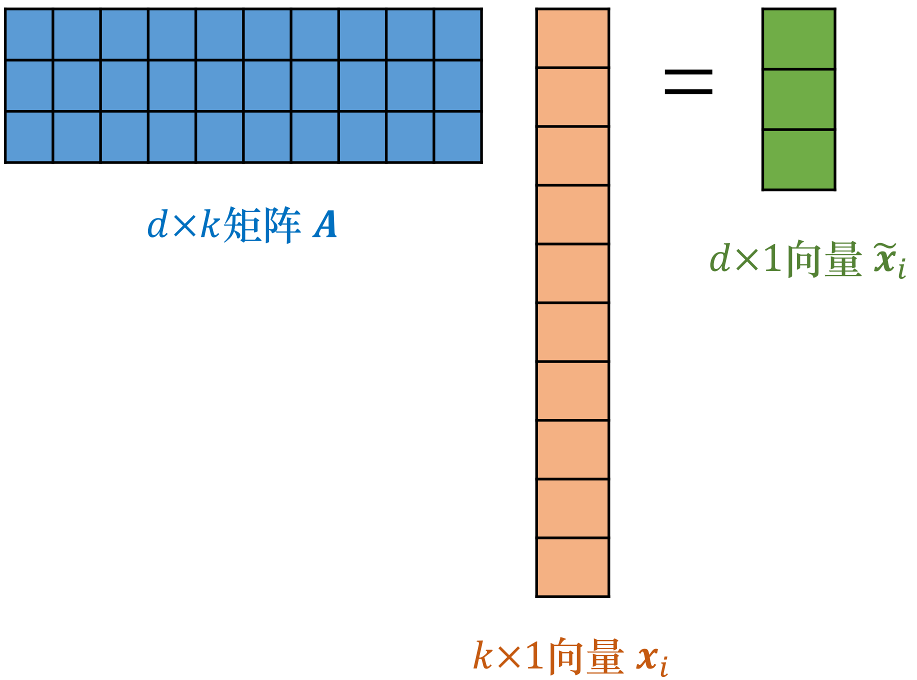


Jaccard相似度下的相似搜索

局部敏感哈希

最小哈希

- 回顾欧式距离下高维数据的相似搜索方法，先通过JL转换把高维数据（随机）映射到低维数据，从而方便做相似搜索



- 此处，可以用类似思路，将集合类数据（随机）映射为一个（或若干个）实数，从而方便做相似搜索

最小哈希

➢ 回顾欧式距离下高维数据的相似搜索方法，先通过JL转换把高维数据（随机）映射到低维数据，从而方便做相似搜索

➢ 此处，可以用类似思路，将集合类数据（随机）映射为一个（或若干个）实数，从而方便做相似搜索

最理想的情况：利用这个实数做相似搜索，如先寻找实数值和集合A一样的集合，再具体计算这些集合与集合A之间的Jaccard相似度

最小哈希

首先，需要找到一个映射$f(\cdot)$将输入集合映射为一个实数，并确保如下性质：

$$\mathbf{Pr}[f(A)=f(B)]=J(A,B)=\frac{|A\cap B|}{|A\cup B|}$$

集合A

集合B

即$f(\cdot)$需要满足：两个集合的Jaccard相似度越高，$f(A)$与$f(B)$相等的概率越大

有没有这样的映射$f(\cdot)$?

之前介绍其它技术时提到过

内容回顾

不同元素统计问题（Distinct Element Counting Problem）

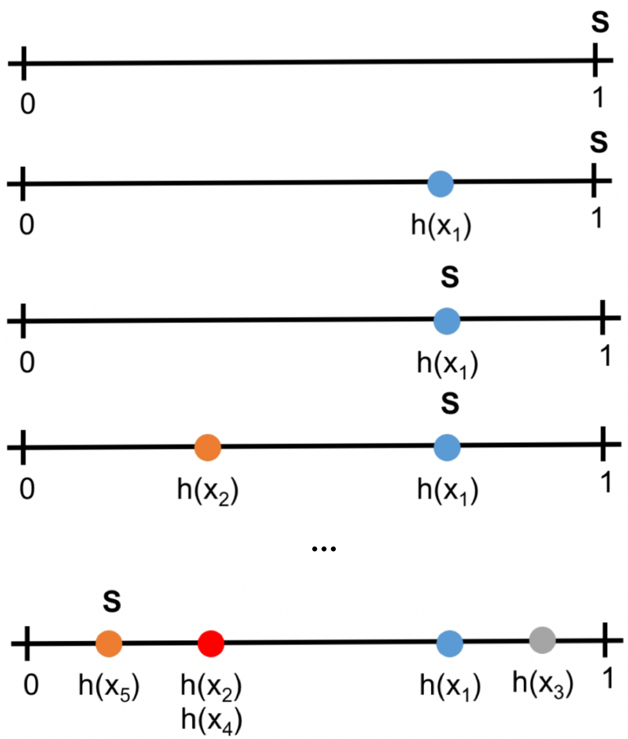
给定长度为n的**数据流**$x_1, \cdots, x_n$，如何计算其中不同元素的个数?

假如有一个随机哈希函数 $h\colon \mathcal{U} \to [0,1]$，即输出是连续而非离散值

初始化 $s \leftarrow 1$

对$i = 1, \dots, n$:　$s \leftarrow \min\{s, h(x_i)\}$

取$\frac{1}{s} - 1$作为对不同元素个数的估计



图例取自Cameron Musco <COMPSCI 514 - Algorithms for Data Science>

最小哈希

首先，需要找到一个映射$f(\cdot)$将输入集合映射为一个实数，并确保如下性质：

$$\mathbf{Pr}[f(A)=f(B)]=J(A,B)=\frac{|A\cap B|}{|A\cup B|}$$

集合A

集合B

即$f(\cdot)$需要满足：两个集合的Jaccard相似度越高，$f(A)$与$f(B)$相等的概率越大

有没有这样的映射$f(\cdot)$?

为集合的所有元素分别计算哈希值，然后取最小哈希值

最小哈希

需要找到一个映射$f(\cdot)$将输入集合映射为一个实数

为集合的所有元素分别计算哈希值，然后取最小哈希值

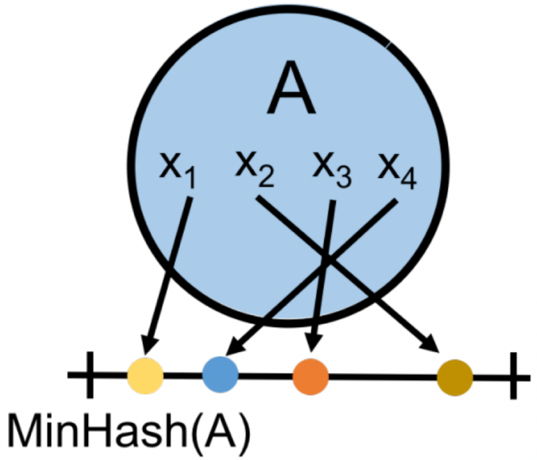
假如有一个随机哈希函数 $h\colon \mathcal{U} \to [0,1]$，即输出是连续而非离散值

初始化 $s \leftarrow 1$

对$x_i \in A, i = 1, \ldots, |A|$:　$s \leftarrow \min\{s, h(x_i)\}$

取s作为$f(A)$

将$f(A)$记为MinHash(A)



图例取自Cameron Musco <COMPSCI 514 - Algorithms for Data Science>

最小哈希

验证MinHash(·)是否满足如下性质：

$$\mathbf{Pr}[\mathrm{MinHash}(A) = \mathrm{MinHash}(B)] = J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

即两个集合的Jaccard相似度越高， MinHash(A)与 MinHash(B)相等的概率越大?

集合A

集合B

初始化 $s \leftarrow 1$

对$x_i \in A, i = 1, \dots, |A|$:　$s \leftarrow \min\{s, h(x_i)\}$

取s作为MinHash(A)

初始化 $s \leftarrow 1$

对$x_i \in B, i = 1, \dots, |B|$:　$s \leftarrow \min\{s, h(x_i)\}$

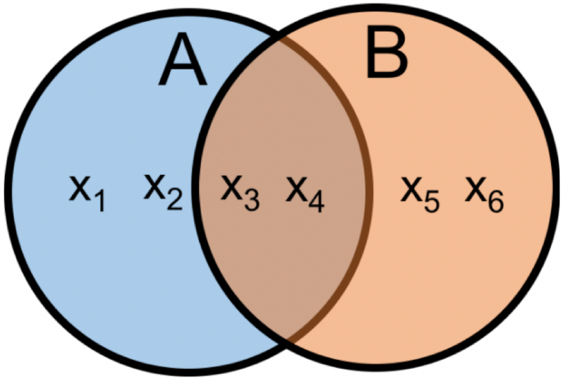
取s作为MinHash(B)

最小哈希

验证MinHash(·)是否满足如下性质：

$$\mathbf{Pr}[\text{MinHash}(A) = \text{MinHash}(B)] = J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

即两个集合的Jaccard相似度越高， MinHash(A)与 MinHash(B)相等的概率越大?



注意，因为随机哈希函数h的取值范围是连续区间$[0,1]$而且$x_1 \neq \cdots \neq x_6$，有$h(x_1) \neq \cdots \neq h(x_6)$

什么情况下MinHash(A) = MinHash(B) ?

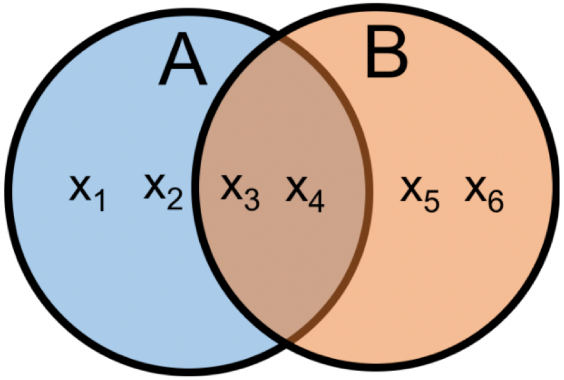
图例取自Cameron Musco <COMPSCI 514 - Algorithms for Data Science>

最小哈希

验证MinHash(·)是否满足如下性质:

$$\mathbf{Pr}[\mathrm{MinHash}(A) = \mathrm{MinHash}(B)] = J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

即两个集合的Jaccard相似度越高， MinHash(A)与 MinHash(B)相等的概率越大?



注意，因为随机哈希函数h的取值范围是连续区间$[0,1]$而且$x_1 \neq \cdots \neq x_6$，有$h(x_1) \neq \cdots \neq h(x_6)$

什么情况下MinHash(A) = MinHash(B) ?

当$\min\{h(x_1), \ldots, h(x_6)\}$为$h(x_3)$或$h(x_4)$时

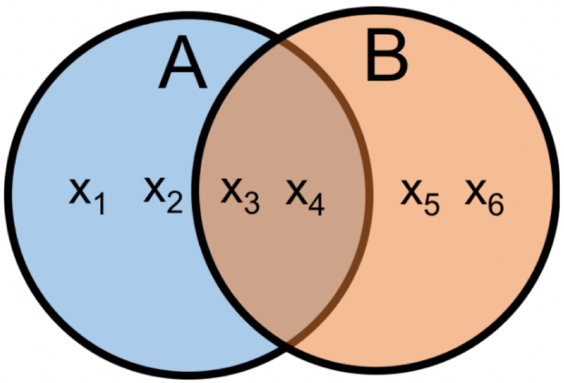
图例取自Cameron Musco <COMPSCI 514 - Algorithms for Data Science>

最小哈希

验证MinHash(·)是否满足如下性质:

$$\mathbf{Pr}[\mathrm{MinHash}(A) = \mathrm{MinHash}(B)] = J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

即两个集合的Jaccard相似度越高， MinHash(A)与 MinHash(B)相等的概率越大?



$\mathbf{Pr}[\mathrm{MinHash}(A) = \mathrm{MinHash}(B)] = \mathbf{Pr}[A \cup B$中哈希值最小的元素属于$A \cap B]$

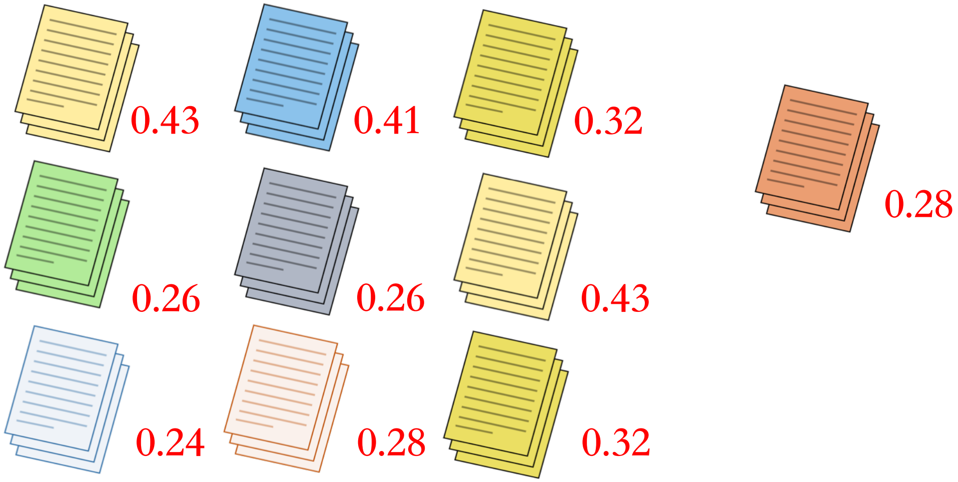
图例取自Cameron Musco <COMPSCI 514 - Algorithms for Data Science>

基于最小哈希的相似搜索

$\text{MinHash}(\cdot)$满足性质：$\mathbf{Pr}[\text{MinHash}(A) = \text{MinHash}(B)] = J(A, B)$

如何用最小哈希做相似搜索？

为每个集合（文件）计算MinHash值，然后搜索所有与新集合具有相同MinHash值的集合？最后，逐个计算搜得集合与新集合的准确Jaccard相似度

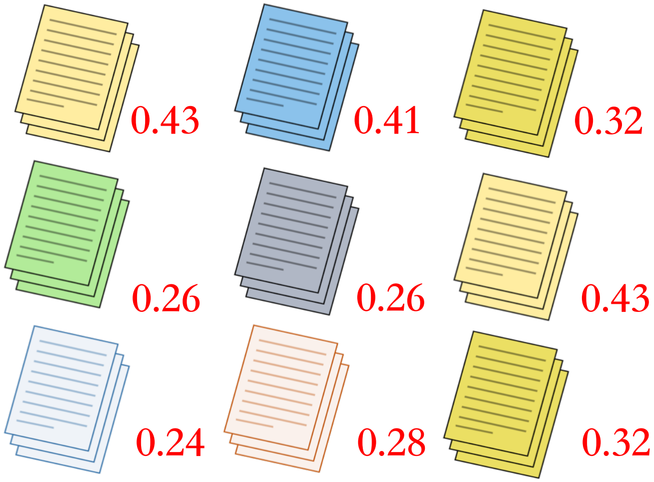


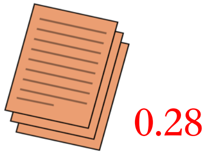
基于最小哈希的相似搜索

MinHash(⋅)满足性质：$\mathbf{Pr}[\text{MinHash}(A)=\text{MinHash}(B)]=J(A,B)$

如何用最小哈希做相似搜索?

为每个集合（文件）计算MinHash值，然后搜索所有与新集合具有相同MinHash值的集合？最后，逐个计算搜得集合与新集合的准确Jaccard相似度



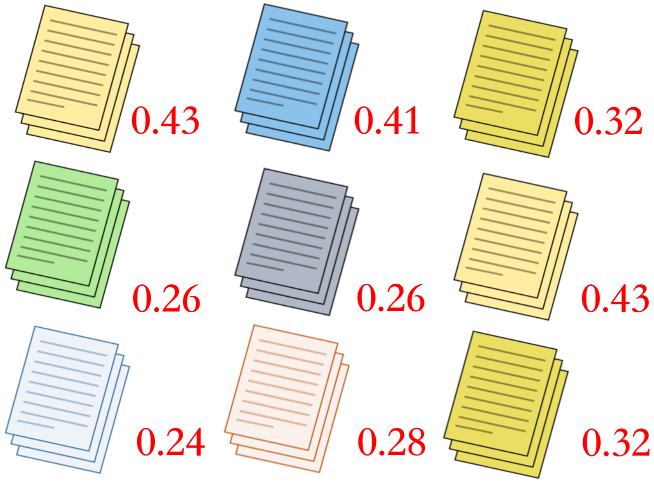


可以先将已有文件根据MinHash值分入不同的“桶”，然后只需在相应的“桶”中进行比对

基于最小哈希的相似搜索

用随机哈希函数$g(\cdot): [0,1] \rightarrow \{0, \dots, m-1\}$将MinHash值映射为$0, \dots, m-1$（“桶”编号）

注意，该随机哈希函数$g(\cdot)$不同于在计算MinHash值时用的哈希函数$h(\cdot)$

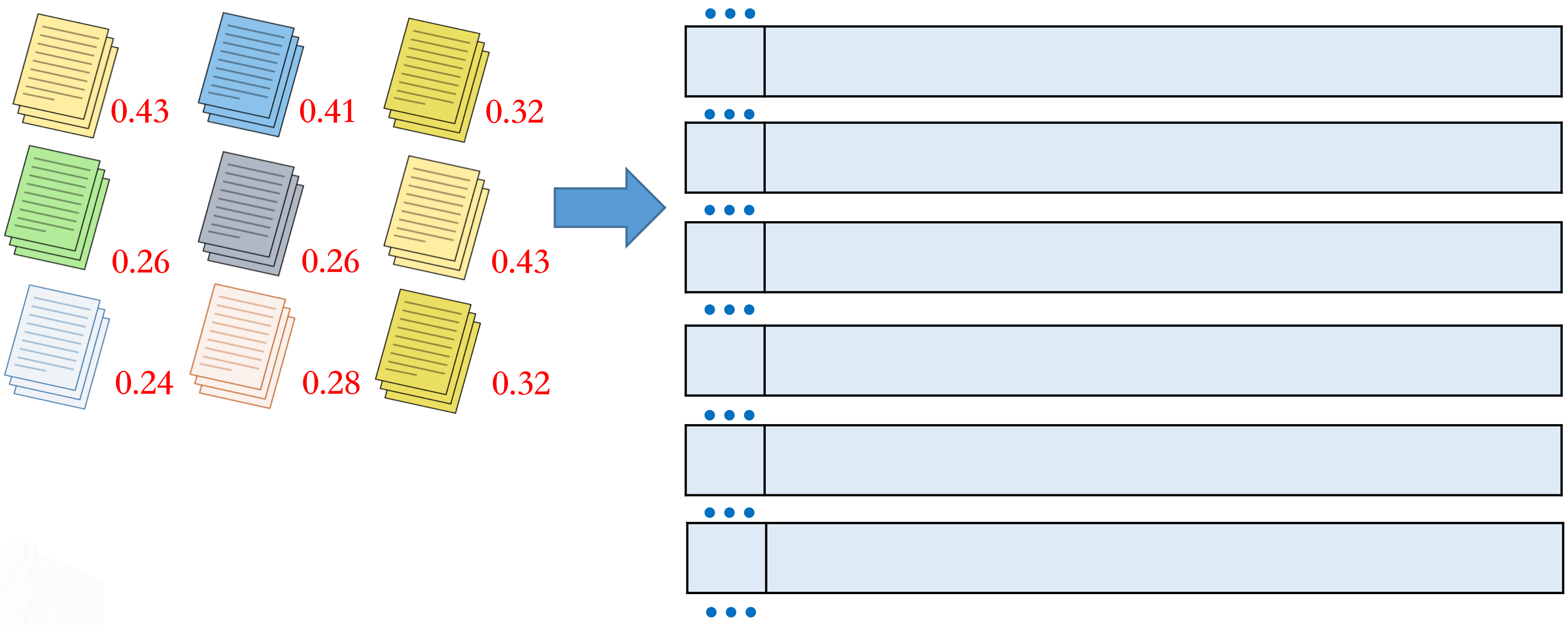


基于最小哈希的相似搜索

用随机哈希函数$g(\cdot): [0,1] \rightarrow \{0, \dots, m-1\}$将MinHash值映射为$0, \dots, m-1$（“桶”编号）

注意，该随机哈希函数$g(\cdot)$不同于在计算MinHash值时用的哈希函数$h(\cdot)$

实际选取的m值很大，因此不同的MinHash值被映射到同一个桶的概率很小



基于最小哈希的相似搜索

用随机哈希函数$g(\cdot)$: $[0,1] \rightarrow \{0, \ldots, m-1\}$将MinHash值映射为$0, \ldots, m-1$（“桶”编号）

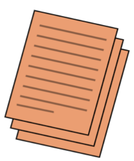
注意，该随机哈希函数$g(\cdot)$不同于在计算MinHash值时用的哈希函数$h(\cdot)$

实际选取的m值很大，因此不同的MinHash值被映射到同一个桶的概率很小

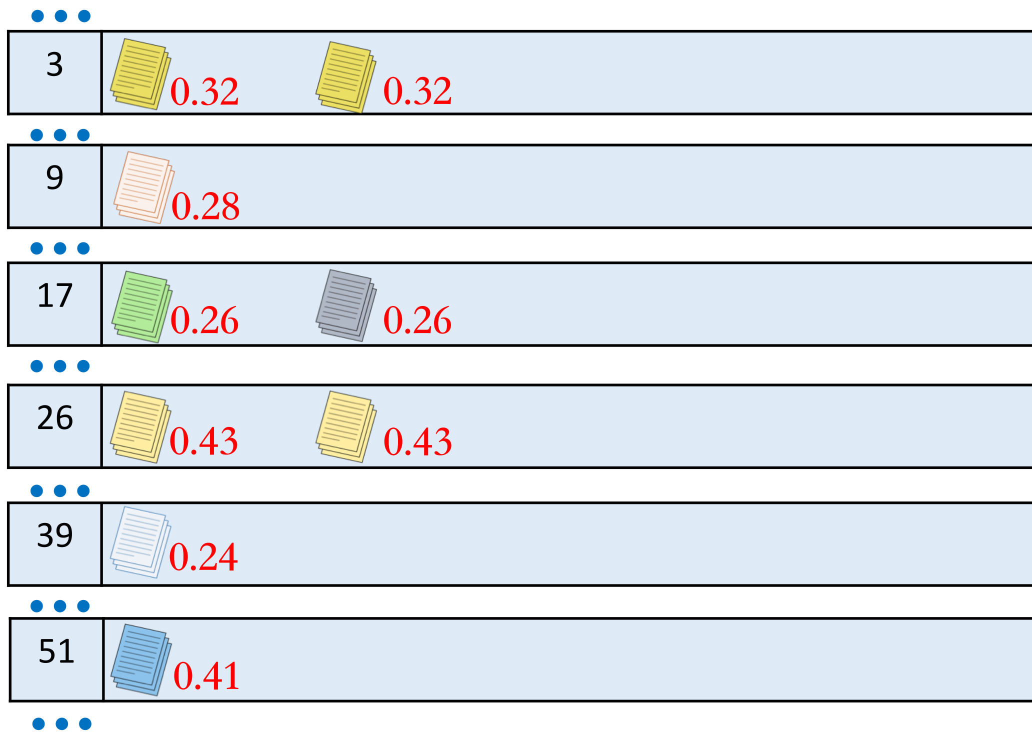
基于最小哈希的相似搜索

用随机哈希函数$g(\cdot): [0,1] \rightarrow \{0, \dots, m-1\}$将MinHash值映射为$0, \dots, m-1$（“桶”编号）

注意，该随机哈希函数$g(\cdot)$不同于在计算MinHash值时用的哈希函数$h(\cdot)$

需要寻找与  相似度最高的文件

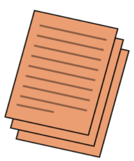
$0.28,\ g(0.28) = 9$



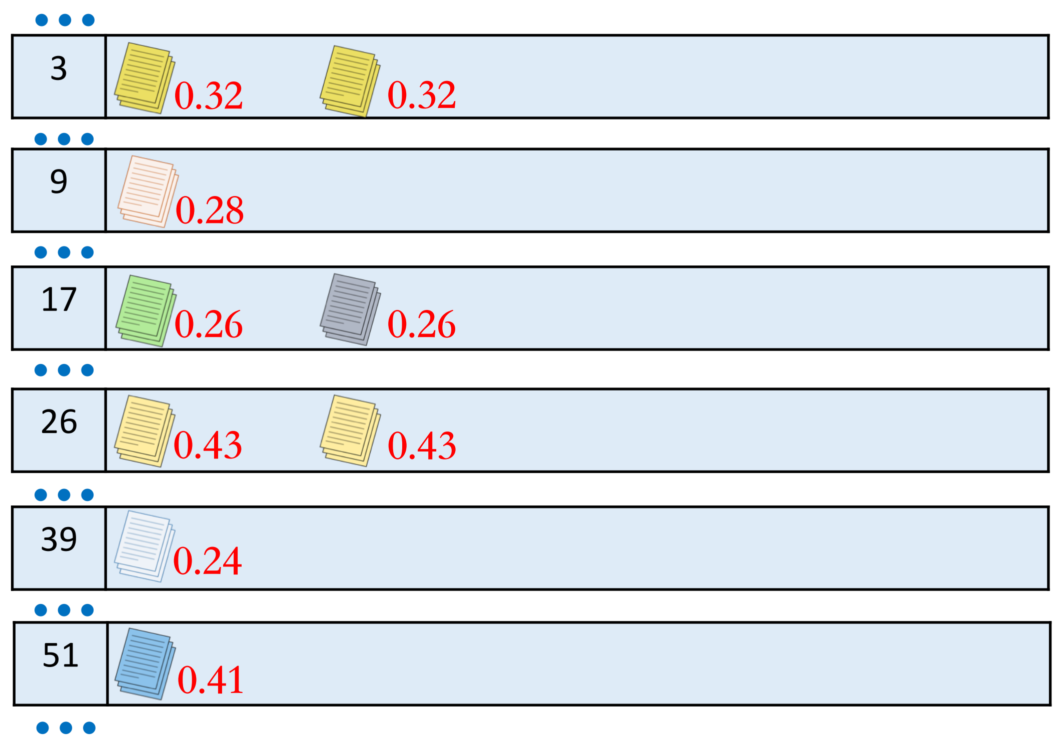
基于最小哈希的相似搜索

用随机哈希函数$g(\cdot):[0,1]\rightarrow\{0,\ldots,m-1\}$将MinHash值映射为$0,\ldots,m-1$（“桶”编号）

注意，该随机哈希函数$g(\cdot)$不同于在计算MinHash值时用的哈希函数$h(\cdot)$

需要寻找与  相似度最高的文件

$0.28,\ g(0.28)=9$



优点：

如果存在与  相似度100%的文件，那么一定能通过此方法搜索出来。

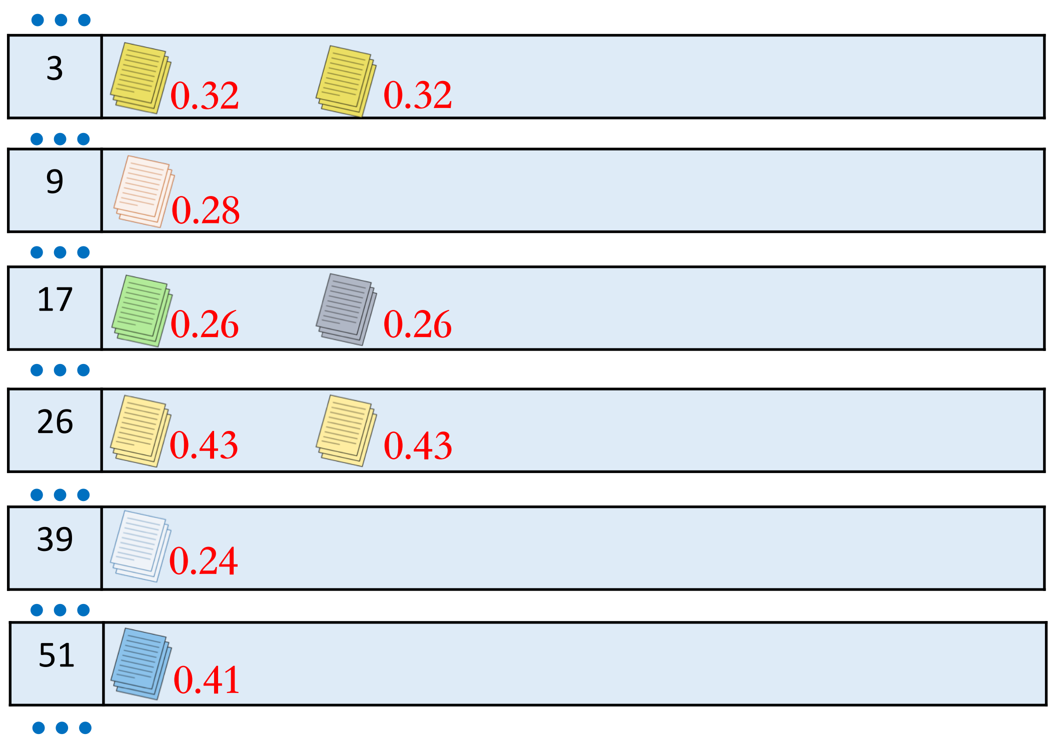
基于最小哈希的相似搜索

用随机哈希函数$g(\cdot): [0,1] \rightarrow \{0, \dots, m-1\}$将MinHash值映射为$0, \dots, m-1$（“桶”编号）

注意，该随机哈希函数$g(\cdot)$不同于在计算MinHash值时用的哈希函数$h(\cdot)$

需要寻找与  相似度最高的文件

$0.28,\ g(0.28) = 9$



缺点：

如果存在与  相似度非常高的文件C，有一定概率$\mathrm{MinHash}(C) \neq 0.28$且$g(\mathrm{MinHash}(C)) \neq 9$。此时，将出现“false negative”（漏报），无法搜索出该文件C

基于最小哈希的相似搜索

用随机哈希函数$g(\cdot): [0,1] \rightarrow \{0, \ldots, m-1\}$将MinHash值映射为$0, \ldots, m-1$（“桶”编号）

注意，该随机哈希函数$g(\cdot)$不同于在计算MinHash值时用的哈希函数$h(\cdot)$

例，当文件A与C的Jaccard相似度$J(A, C) = 0.8$，那么$\mathbf{Pr}\left[g\big(\mathrm{MinHash}(A)\big) = g\big(\mathrm{MinHash}(C)\big)\right] \approx ?$

（假设m值足够大，使得不同的MinHash值被映射到同一个桶的概率很小）

基于最小哈希的相似搜索

用随机哈希函数$g(\cdot): [0,1] \rightarrow \{0, \ldots, m-1\}$将MinHash值映射为$0, \ldots, m-1$（“桶”编号）

注意，该随机哈希函数$g(\cdot)$不同于在计算MinHash值时用的哈希函数$h(\cdot)$

例，当文件A与C的Jaccard相似度$J(A,C) = 0.8$，那么$\mathbf{Pr}\left[g\left(\text{MinHash}(A)\right) = g\left(\text{MinHash}(C)\right)\right] \approx 0.8$

（假设m值足够大，使得不同的MinHash值被映射到同一个桶的概率很小）

$$\mathbf{Pr}\left[g\left(\text{MinHash}(A)\right) = g\left(\text{MinHash}(C)\right)\right] \approx \mathbf{Pr}[\text{MinHash}(A) = \text{MinHash}(C)] = J(A,C) = 0.8$$

说明如果文件A是新文件，文件C在一堆文件中，有约20%的概率无法在前述相似搜索中找出C

如何降低该概率?

基于最小哈希的相似搜索

用随机哈希函数$g(\cdot)$: $[0,1] \rightarrow \{0, \ldots, m-1\}$将MinHash值映射为$0, \ldots, m-1$（“桶”编号）

注意，该随机哈希函数$g(\cdot)$不同于在计算MinHash值时用的哈希函数$h(\cdot)$

例，当文件A与C的Jaccard相似度$J(A, C) = 0.8$，那么$\mathbf{Pr}\left[g\left(\mathrm{MinHash}(A)\right) = g\left(\mathrm{MinHash}(C)\right)\right] \approx 0.8$

（假设m值足够大，使得不同的MinHash值被映射到同一个桶的概率很小）

$$\mathbf{Pr}\left[g\left(\mathrm{MinHash}(A)\right) = g\left(\mathrm{MinHash}(C)\right)\right] \approx \mathbf{Pr}[\mathrm{MinHash}(A) = \mathrm{MinHash}(C)] = J(A, C) = 0.8$$

说明如果文件A是新文件，文件C在一堆文件中，有约20%的概率无法在前述相似搜索中找出C

如何降低该概率?

利用多个不同的哈希函数，多个不同的$h(\cdot)$还是多个不同的$g(\cdot)$？

基于最小哈希的相似搜索

用随机哈希函数$g(\cdot): [0,1] \rightarrow \{0, \dots, m-1\}$将MinHash值映射为$0, \dots, m-1$（“桶”编号）

注意，该随机哈希函数$g(\cdot)$不同于在计算MinHash值时用的哈希函数$h(\cdot)$

例，当文件A与C的Jaccard相似度$J(A, C) = 0.8$，那么$\mathbf{Pr}\left[g\left(\text{MinHash}(A)\right) = g\left(\text{MinHash}(C)\right)\right] \approx 0.8$

（假设m值足够大，使得不同的MinHash值被映射到同一个桶的概率很小）

$$\mathbf{Pr}\left[g\left(\text{MinHash}(A)\right) = g\left(\text{MinHash}(C)\right)\right] \approx \mathbf{Pr}[\text{MinHash}(A) = \text{MinHash}(C)] = J(A, C) = 0.8$$

说明如果文件A是新文件，文件C在一堆文件中，有约20%的概率无法在前述相似搜索中找出C

如何降低该概率?

利用多个不同的哈希函数，多个不同的$h(\cdot)$还是多个不同的$g(\cdot)$？

利用多个不同的$h(\cdot)$，为每个文件计算多个MinHash值

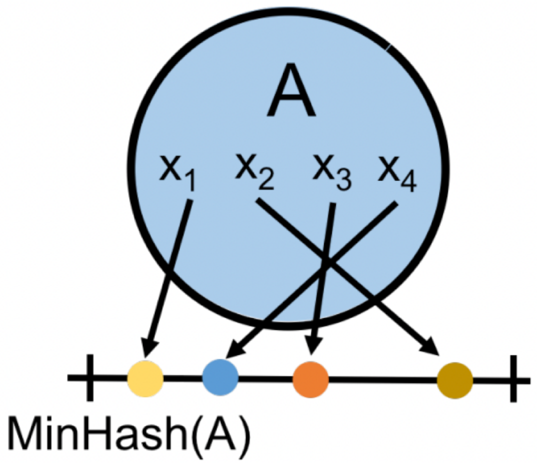
基于最小哈希的相似搜索

假如有一个随机哈希函数 $h: \mathcal{U} \rightarrow [0,1]$

初始化 $s \leftarrow 1$

对$x_i \in A, i = 1, \dots, |A|$: $s \leftarrow \min\{s, h(x_i)\}$

取s作为MinHash(A)

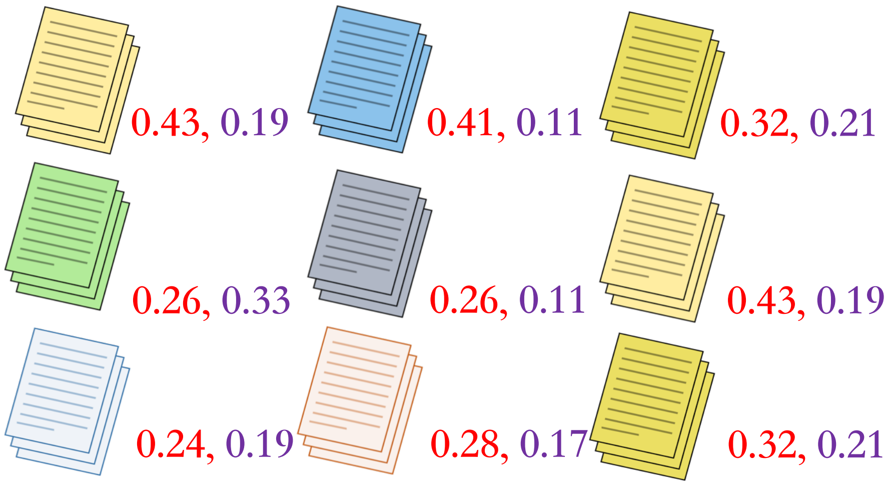


通过采用t个不同的哈希函数，可以得到t个不同的MinHash值

可以记为$\text{MinHash}_1(A), \text{MinHash}_2(A), \dots, \text{MinHash}_t(A)$

基于最小哈希的相似搜索

假设$t = 2$

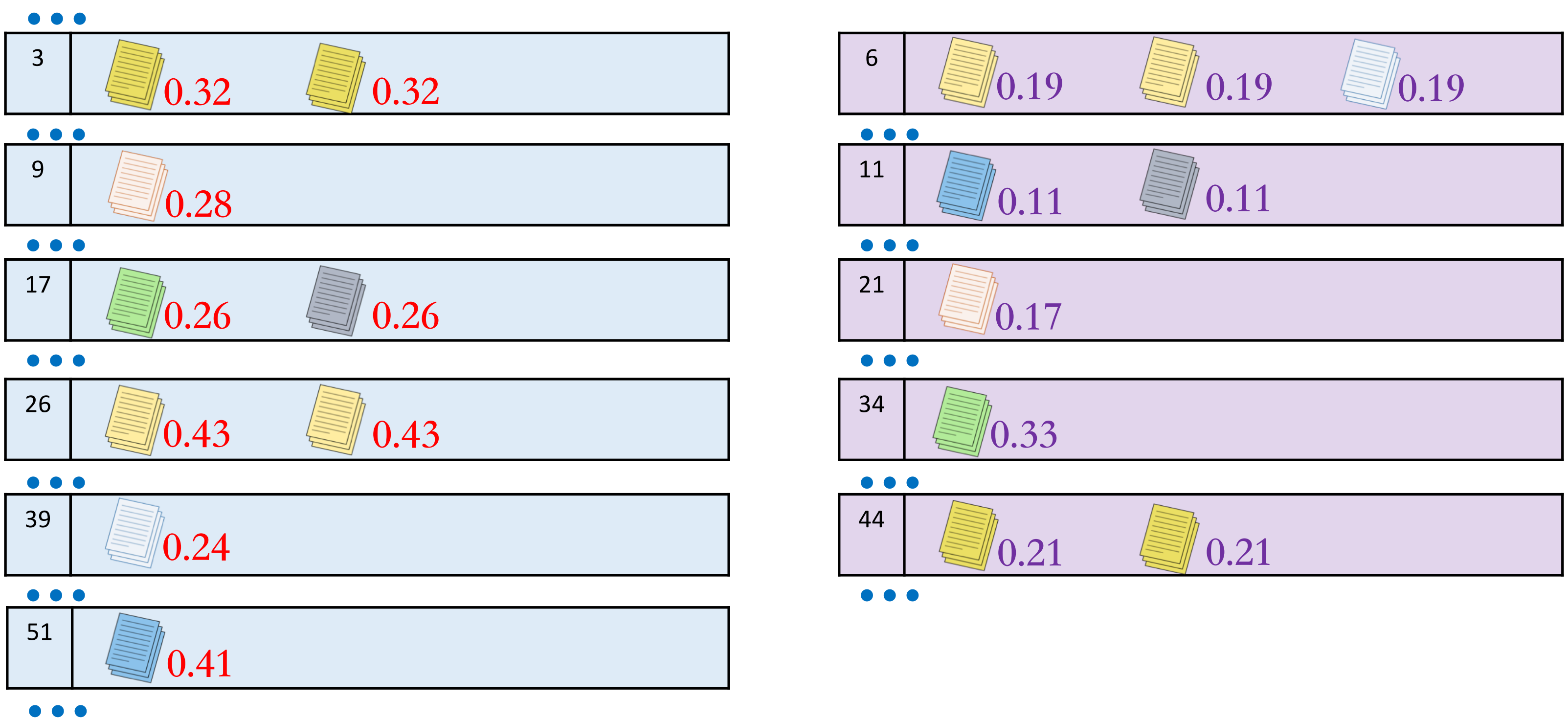


基于最小哈希的相似搜索

假设$t=2$，用随机哈希函数$g(\cdot):[0,1]\to\{0,\dots,m-1\}$将每个MinHash值映射为$0,\dots,m-1$

根据$\mathrm{MinHash}_1$值将文件映射到“桶”

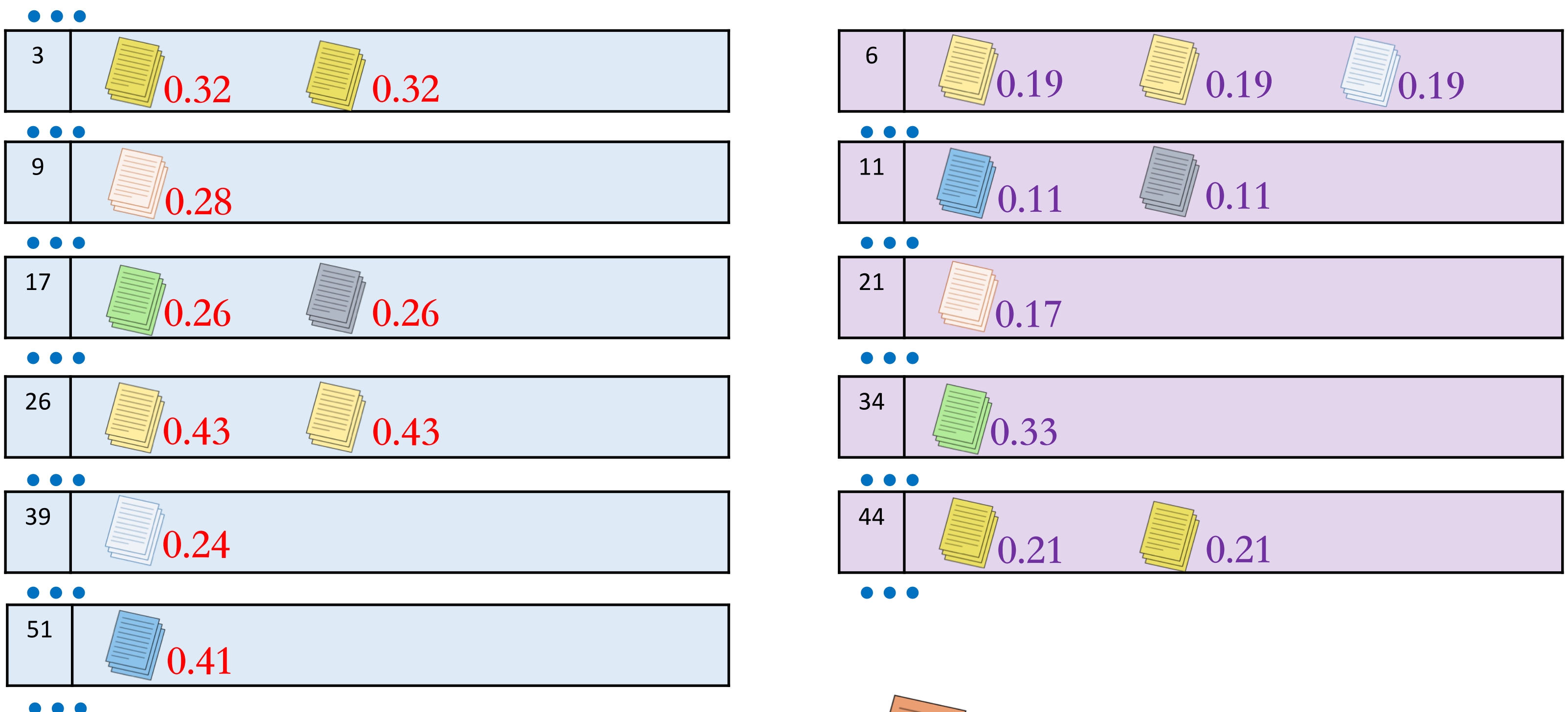
根据$\mathrm{MinHash}_2$值将文件映射到“桶”



基于最小哈希的相似搜索

假设$t=2$，用随机哈希函数$g(\cdot):[0,1]\rightarrow\{0,\ldots,m-1\}$将每个MinHash值映射为$0,\ldots,m-1$

根据$\mathrm{MinHash}_1$值将文件映射到“桶”　　根据$\mathrm{MinHash}_2$值将文件映射到“桶”

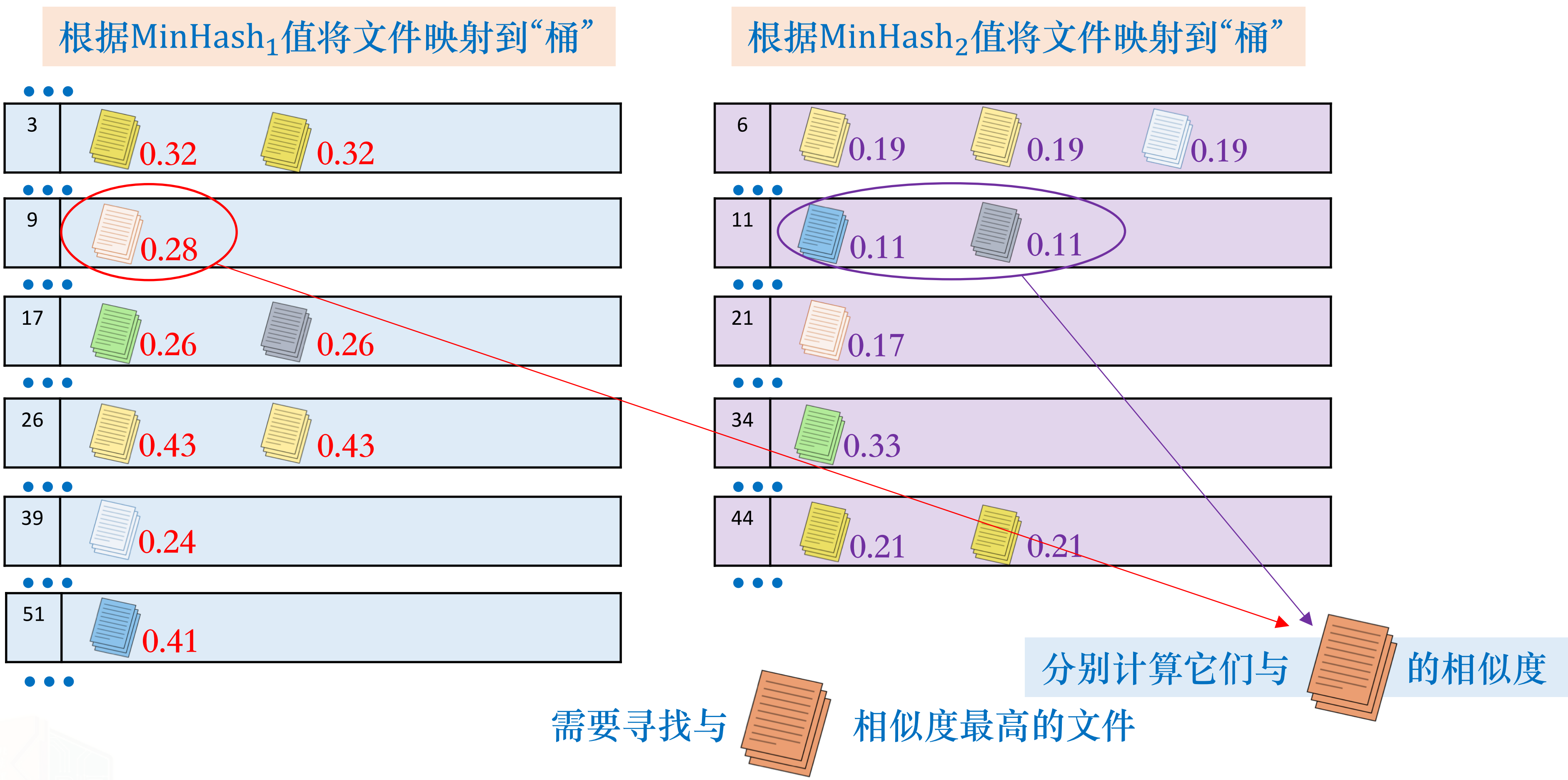


需要寻找与  相似度最高的文件

$(0.28, 0.11), g(0.28)=9, g(0.11)=11$

基于最小哈希的相似搜索

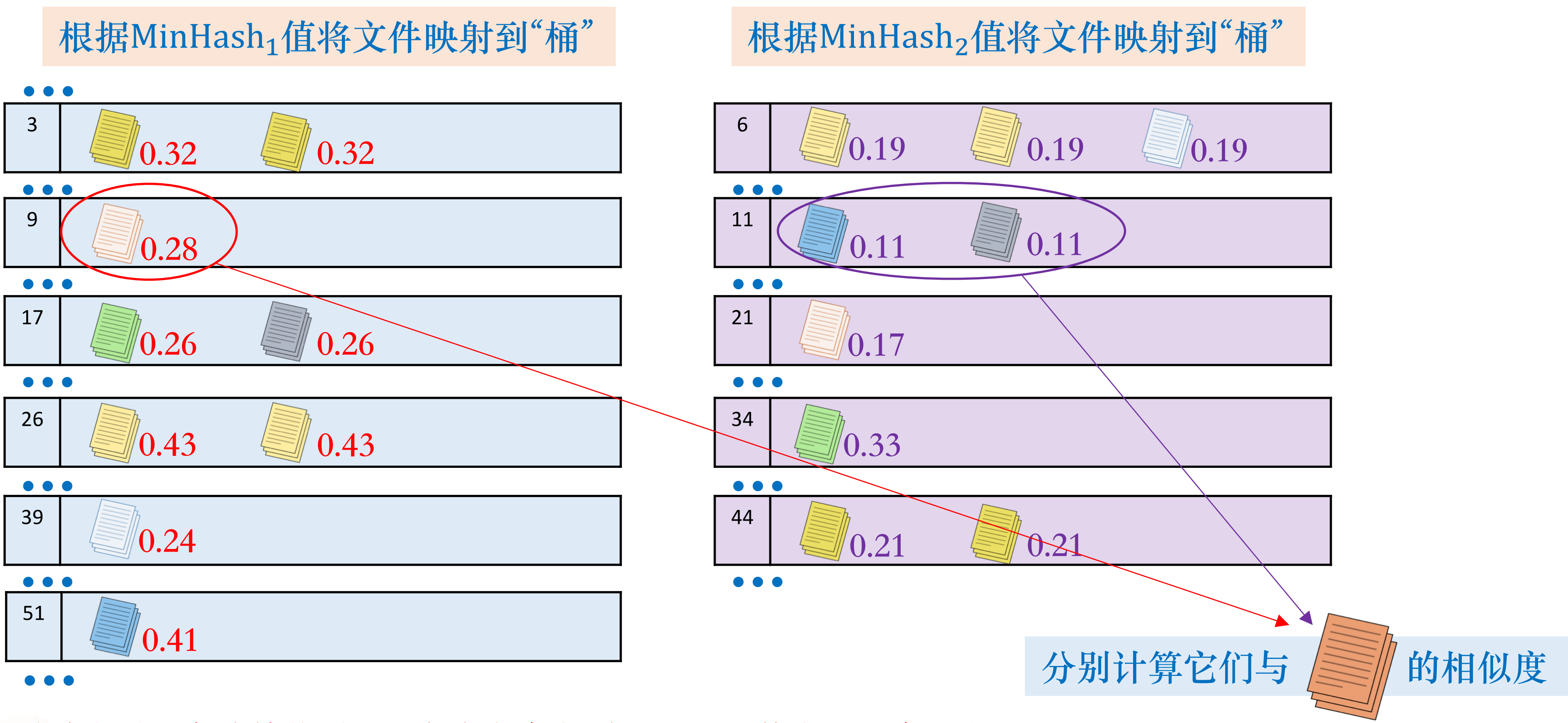
假设$t = 2$，用随机哈希函数$g(\cdot): [0,1] \rightarrow \{0, \ldots, m-1\}$将每个MinHash值映射为$0, \ldots, m-1$



$(0.28, 0.11), g(0.28) = 9, g(0.11) = 11$

基于最小哈希的相似搜索

假设$t = 2$，用随机哈希函数$g(\cdot)$: $[0,1] \rightarrow \{0, \dots, m-1\}$将每个MinHash值映射为$0, \dots, m-1$



通过扩大对比文件的范围（即考虑所有在t个MinHash值中至少有一个与 相同的文件），减少“false negative”（漏报）

基于最小哈希的相似搜索

用随机哈希函数$g(\cdot)\colon [0,1] \rightarrow \{0, \dots, m-1\}$将每个MinHash值映射为$0, \dots, m-1$

例，当文件A与C的Jaccard相似度$J(A, C) = 0.8$，计算如下概率：

$$\mathbf{Pr}\left[g\left(\text{MinHash}_1(A)\right) = g\left(\text{MinHash}_1(C)\right) \textbf{ OR } \dots \textbf{ OR } g\left(\text{MinHash}_t(A)\right) = g\left(\text{MinHash}_t(C)\right)\right]$$

（假设m值足够大，使得不同的MinHash值被映射到同一个桶的概率很小）

基于最小哈希的相似搜索

用随机哈希函数$g(\cdot): [0,1] \rightarrow \{0, \dots, m-1\}$将每个MinHash值映射为$0, \dots, m-1$

例，当文件A与C的Jaccard相似度$J(A, C) = 0.8$，计算如下概率：

$$\mathbf{Pr}\left[g\left(\text{MinHash}_1(A)\right) = g\left(\text{MinHash}_1(C)\right) \textbf{ OR } \dots \textbf{ OR } g\left(\text{MinHash}_t(A)\right) = g\left(\text{MinHash}_t(C)\right)\right]$$

（假设m值足够大，使得不同的MinHash值被映射到同一个桶的概率很小）

约为$1 - (1 - 0.8)^t$

当$t = 2$时，概率约为0.96
（说明如果文件A是新文件，文件C在一堆文件中，有约4%的概率无法在相似搜索中找出C）
当$t = 3$时，概率约为0.992
（说明如果文件A是新文件，文件C在一堆文件中，有约0.8%的概率无法在相似搜索中找出C）

通过扩大对比文件的范围可以减少“false negative”（漏报）的可能性，但是附带的问题是?

基于最小哈希的相似搜索

用随机哈希函数$g(\cdot):[0,1]\rightarrow\{0,\dots,m-1\}$将每个MinHash值映射为$0,\dots,m-1$

例，当文件A与C的Jaccard相似度$J(A,C)=0.8$，计算如下概率：

$$\mathbf{Pr}\left[g\big(\mathrm{MinHash}_1(A)\big)=g\big(\mathrm{MinHash}_1(C)\big)\ \mathbf{OR}\ \dots\ \mathbf{OR}\ g\big(\mathrm{MinHash}_t(A)\big)=g\big(\mathrm{MinHash}_t(C)\big)\right]$$

（假设m值足够大，使得不同的MinHash值被映射到同一个桶的概率很小）

约为$1-(1-0.8)^t$

当$t=2$时，概率约为0.96
（说明如果文件A是新文件，文件C在一堆文件中，有约4%的概率无法在相似搜索中找出C）
当$t=3$时，概率约为0.992
（说明如果文件A是新文件，文件C在一堆文件中，有约0.8%的概率无法在相似搜索中找出C）

通过扩大对比文件的范围可以减少“false negative”（漏报）的可能性，但是附带的问题是?

需对比的文件的数目增多（包括实际相似度低的文件），需计算与更多文件之间的Jaccard相似度

基于最小哈希的相似搜索

用随机哈希函数$g(\cdot): [0,1] \rightarrow \{0, \dots, m-1\}$将每个MinHash值映射为$0, \dots, m-1$

例，当文件A与C的Jaccard相似度$J(A, C) = 0.3$，计算如下概率：

$$\mathbf{Pr}\left[g\left(\text{MinHash}_1(A)\right) = g\left(\text{MinHash}_1(C)\right) \textbf{ OR } \dots \textbf{ OR } g\left(\text{MinHash}_t(A)\right) = g\left(\text{MinHash}_t(C)\right)\right]$$

（假设m值足够大，使得不同的MinHash值被映射到同一个桶的概率很小）

约为$1-(1-0.3)^t$

当$t = 3$时，概率约为0.657

（说明如果文件A是新文件，文件C在一堆文件中，即便实际相似度不高，仍有65.7%的概率需要具体计算文件A与文件C之间的Jaccard相似度值）

基于最小哈希的相似搜索

用随机哈希函数$g(\cdot):[0,1]\rightarrow\{0,\ldots,m-1\}$将每个MinHash值映射为$0,\ldots,m-1$

例，当文件A与C的Jaccard相似度$J(A,C)=0.3$，计算如下概率：

$$\mathbf{Pr}\left[g\left(\mathrm{MinHash}_1(A)\right)=g\left(\mathrm{MinHash}_1(C)\right)\ \mathbf{OR}\ \ldots\ \mathbf{OR}\ g\left(\mathrm{MinHash}_t(A)\right)=g\left(\mathrm{MinHash}_t(C)\right)\right]$$

（假设m值足够大，使得不同的MinHash值被映射到同一个桶的概率很小）

约为$1-(1-0.3)^t$

当$t=3$时，概率约为0.657
（说明如果文件A是新文件，文件C在一堆文件中，即便实际相似度不高，仍有65.7%的概率需要具体计算文件A与文件C之间的Jaccard相似度值）

如何尽量令低相似度的两个文件的$\mathrm{MinHash}_1,\ldots,\mathrm{MinHash}_t$值都不相等?

此时，概率值$\mathbf{Pr}\left[g\left(\mathrm{MinHash}_1(A)\right)=g\left(\mathrm{MinHash}_1(C)\right)\ \mathbf{OR}\ \ldots\ \mathbf{OR}\ g\left(\mathrm{MinHash}_t(A)\right)=g\left(\mathrm{MinHash}_t(C)\right)\right]$非常小

基于最小哈希的相似搜索

$$\mathbf{Pr}[\mathrm{MinHash}(A) = \mathrm{MinHash}(B)] = J(A, B)$$

集合A

初始化 $s \leftarrow 1$

对$x_i \in A, i = 1, \dots, |A|$: $s \leftarrow \min\{s, h(x_i)\}$

取s作为$\mathrm{MinHash}(A)$

集合B

初始化 $s \leftarrow 1$

对$x_i \in B, i = 1, \dots, |B|$: $s \leftarrow \min\{s, h(x_i)\}$

取s作为$\mathrm{MinHash}(B)$

如何修改可以令当$J(A, B)$较小时，A和B对应的数以较大概率不同?

基于最小哈希的相似搜索

$$\mathbf{Pr}[\mathrm{MinHash}(A) = \mathrm{MinHash}(B)] = J(A, B)$$

集合A

初始化 $s \leftarrow 1$

对$x_i \in A, i = 1, \ldots, |A|$： $s \leftarrow \min\{s, h(x_i)\}$

取s作为$\mathrm{MinHash}(A)$

集合B

初始化 $s \leftarrow 1$

对$x_i \in B, i = 1, \ldots, |B|$： $s \leftarrow \min\{s, h(x_i)\}$

取s作为$\mathrm{MinHash}(B)$

如何修改可以令当$J(A, B)$较小时，A和B对应的数以较大概率不同?

利用多个不同的哈希函数，为每个文件计算多个MinHash值

基于最小哈希的相似搜索

采用r个不同的哈希函数

集合A

$$\left(\mathrm{MinHash}_1(A), \ldots, \mathrm{MinHash}_r(A)\right)$$

集合B

$$\left(\mathrm{MinHash}_1(B), \ldots, \mathrm{MinHash}_r(B)\right)$$

$$\mathbf{Pr}\left[\left(\mathrm{MinHash}_1(A), \ldots, \mathrm{MinHash}_r(A)\right) = \left(\mathrm{MinHash}_1(B), \ldots, \mathrm{MinHash}_r(B)\right)\right] = J(A, B)^r$$

局部敏感哈希

- 为每个集合（文件）用不同的哈希函数计算共$t \times r$个最小哈希值

集合X

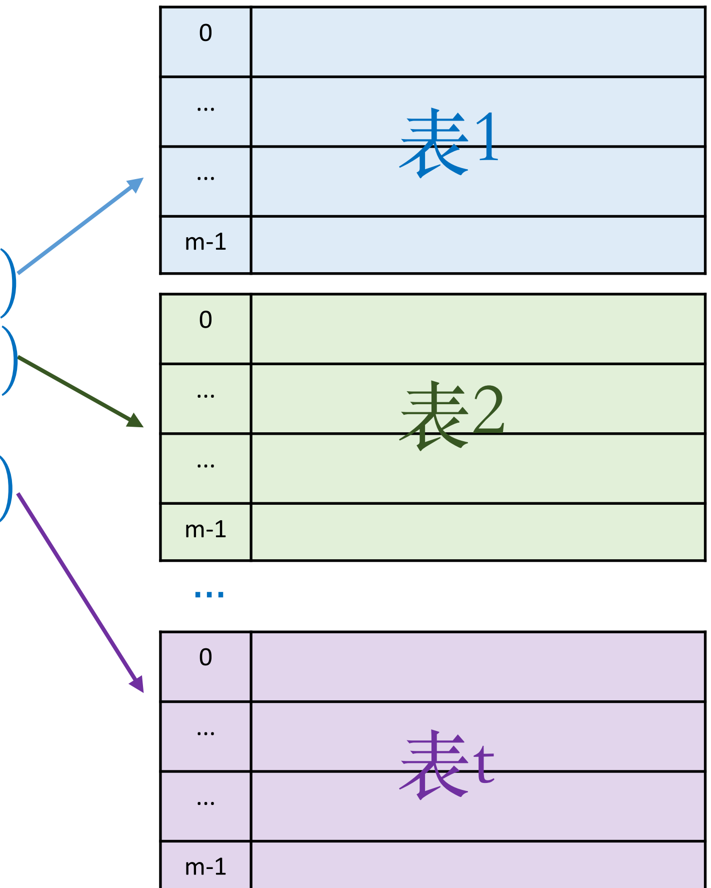
$$\left(\mathrm{MinHash}_{1,1}(X), \mathrm{MinHash}_{1,2}(X), \ldots, \mathrm{MinHash}_{1,r}(X)\right)$$
$$\left(\mathrm{MinHash}_{2,1}(X), \mathrm{MinHash}_{2,2}(X), \ldots, \mathrm{MinHash}_{2,r}(X)\right)$$
$$\ldots$$
$$\left(\mathrm{MinHash}_{t,1}(X), \mathrm{MinHash}_{t,2}(X), \ldots, \mathrm{MinHash}_{t,r}(X)\right)$$

局部敏感哈希

- 为每个集合（文件）用不同的哈希函数计算共$t \times r$个最小哈希值
- 用哈希函数$g(\cdot)\colon [0,1]^r \rightarrow \{0, \ldots, m-1\}$将每行MinHash向量映射为$0, \ldots, m-1$

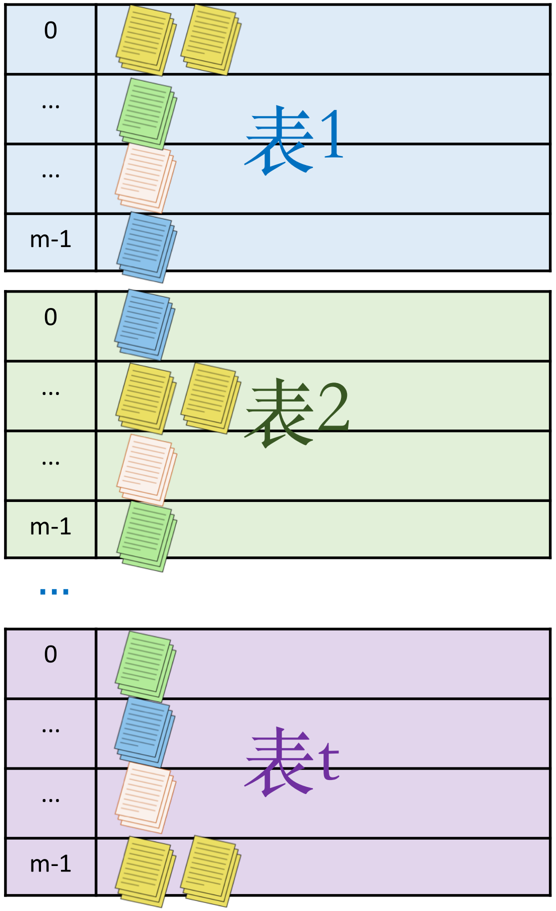
集合X

$$\left(\text{MinHash}_{1,1}(X), \text{MinHash}_{1,2}(X), \ldots, \text{MinHash}_{1,r}(X)\right)$$
$$\left(\text{MinHash}_{2,1}(X), \text{MinHash}_{2,2}(X), \ldots, \text{MinHash}_{2,r}(X)\right)$$
$$\ldots$$
$$\left(\text{MinHash}_{t,1}(X), \text{MinHash}_{t,2}(X), \ldots, \text{MinHash}_{t,r}(X)\right)$$



局部敏感哈希

- 为每个集合（文件）用不同的哈希函数计算共$t \times r$个最小哈希值
- 用哈希函数$g(\cdot)$: $[0,1]^r \rightarrow \{0, \dots, m-1\}$将每行MinHash向量映射为$0, \dots, m-1$

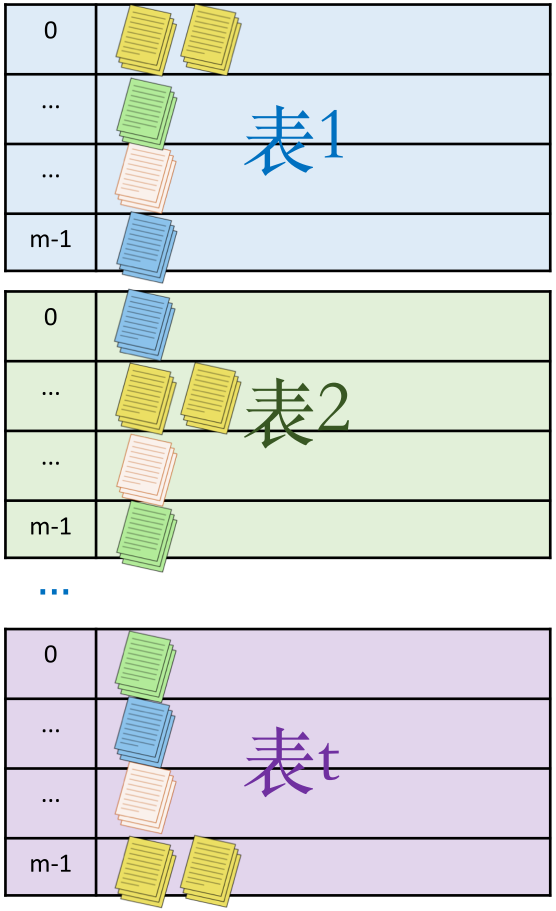


局部敏感哈希

- 为每个集合（文件）用不同的哈希函数计算共$t \times r$个最小哈希值
- 用哈希函数$g(\cdot)$: $[0,1]^r \rightarrow \{0, \ldots, m-1\}$将每行MinHash向量映射为$0, \ldots, m-1$
- 对需要做相似搜索的新集合（文件）**A**计算

$$g\left(\mathrm{MinHash}_{1,1}(A), \mathrm{MinHash}_{1,2}(A), \ldots, \mathrm{MinHash}_{1,r}(A)\right)$$
$$g\left(\mathrm{MinHash}_{2,1}(A), \mathrm{MinHash}_{2,2}(A), \ldots, \mathrm{MinHash}_{2,r}(A)\right)$$
$$\ldots$$
$$g\left(\mathrm{MinHash}_{t,1}(A), \mathrm{MinHash}_{t,2}(A), \ldots, \mathrm{MinHash}_{t,r}(A)\right)$$

把相应的t个“桶”（每个表出一个“桶”）中所有集合提取出来，分别计算它们与集合**A**的具体**Jaccard**相似度



局部敏感哈希

- 为每个集合（文件）用不同的哈希函数计算共$t \times r$个最小哈希值
- 用哈希函数$g(\cdot)$: $[0,1]^r \rightarrow \{0, \ldots, m-1\}$将**每行MinHash向量**映射为$0, \ldots, m-1$

假设集合A是新文件，集合C是原有文件之一而且$J(A, C) = s$，那么把C找出来与A比对的概率?

$$
\begin{aligned}
&\mathbf{Pr}[\text{将C找出来与A比对}] \\
&= 1 - \mathbf{Pr}[\text{未将C找出来与A比对}] \\
&= 1 - (\mathbf{Pr}[\text{在第}t\text{个表中A和C不在一个“桶”}])^t \\
&= 1 - (\mathbf{Pr}[\text{A和C的第}t\text{行MinHash向量不相同}])^t \\
&= 1 - (1 - \mathbf{Pr}[\text{A和C的第}t\text{行MinHash向量相同}])^t \\
&= 1 - (1 - s^r)^t
\end{aligned}
$$

局部敏感哈希

- 为每个集合（文件）用不同的哈希函数计算共$t \times r$个最小哈希值
- 用哈希函数$g(\cdot)$: $[0,1]^r \rightarrow \{0, \ldots, m-1\}$将每行MinHash向量映射为$0, \ldots, m-1$

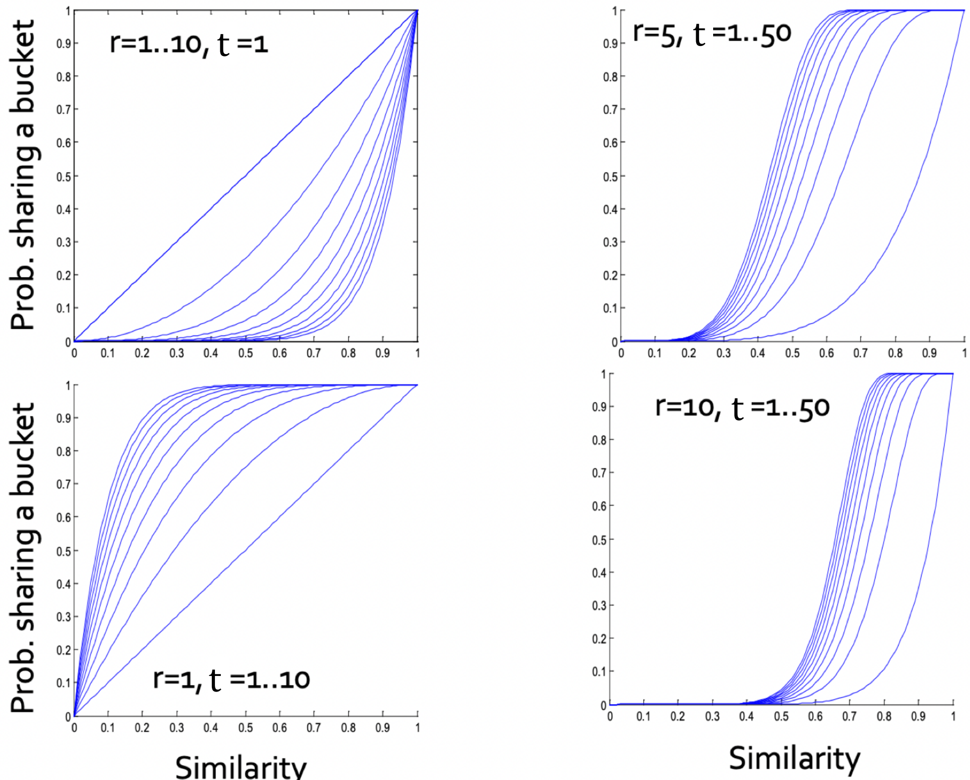
假设**A**是新文件，**C**是原有文件之一且$J(A, C) = s$，那么**Pr**[将C找出来与A比对] $= 1 - (1 - s^r)^t$

应该如何选择t和r?

局部敏感哈希

- 为每个集合（文件）用不同的哈希函数计算共$t \times r$个最小哈希值
- 用哈希函数$g(\cdot)$: $[0,1]^r \to \{0, \dots, m-1\}$将每行MinHash向量映射为$0, \dots, m-1$

Pr[将C找出来与A比对] $= 1 - (1 - s^r)^t$

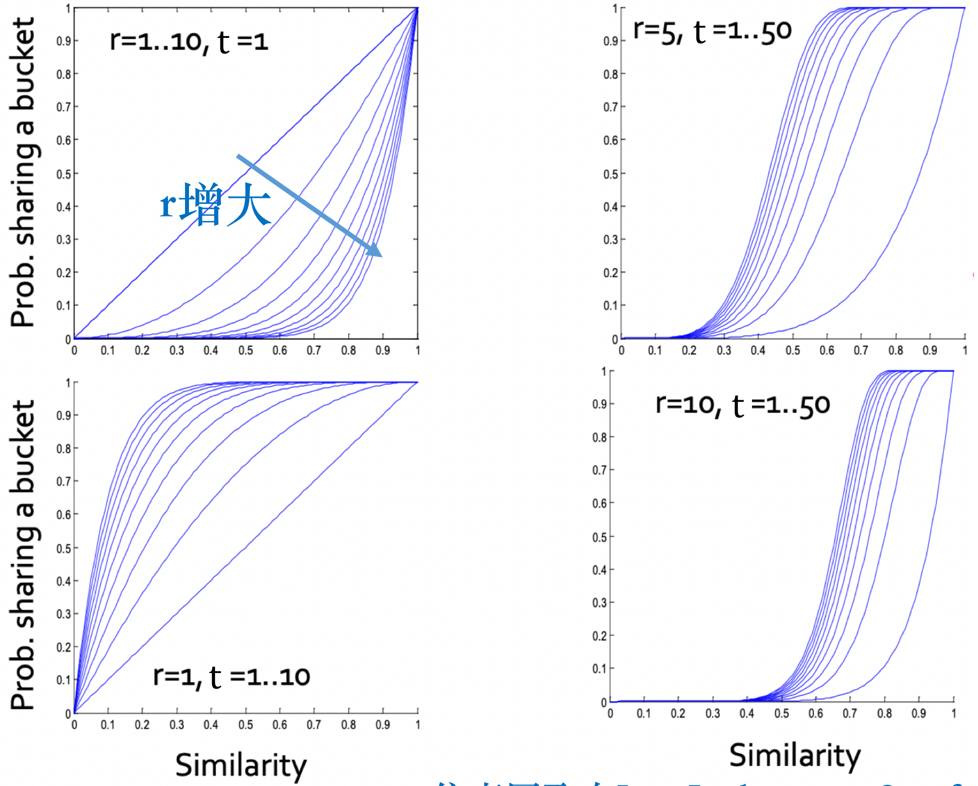


仿真图取自Jure Leskovec < Stanford CS246: Mining Massive Datasets>

局部敏感哈希

- 为每个集合（文件）用不同的哈希函数计算共$t \times r$个最小哈希值
- 用哈希函数$g(\cdot)$: $[0,1]^r \rightarrow \{0, \dots, m-1\}$将每行MinHash向量映射为$0, \dots, m-1$

$$\mathbf{Pr}[\text{将C找出来与A比对}] = 1 - (1 - s^r)^t$$

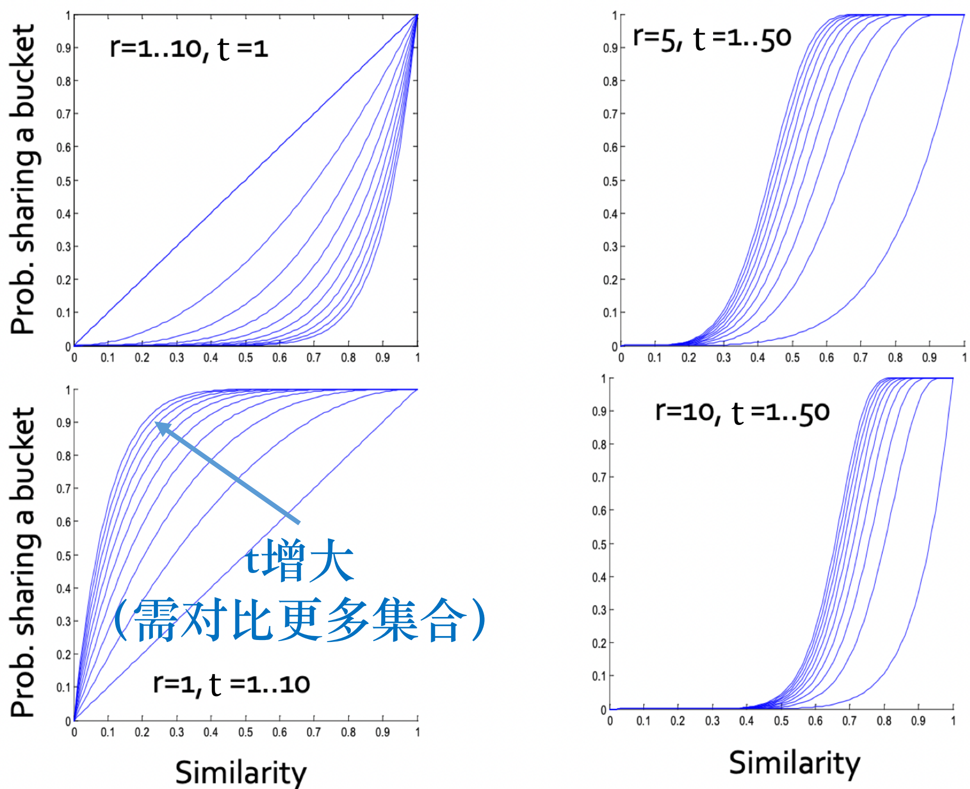


仿真图取自Jure Leskovec < Stanford CS246: Mining Massive Datasets>

局部敏感哈希

- 为每个集合（文件）用不同的哈希函数计算共$t\times r$个最小哈希值
- 用哈希函数$g(\cdot)$: $[0,1]^r \rightarrow \{0,\dots,m-1\}$将每行MinHash向量映射为$0,\dots,m-1$

$$\mathbf{Pr}[\text{将C找出来与A比对}] = 1-(1-s^r)^t$$

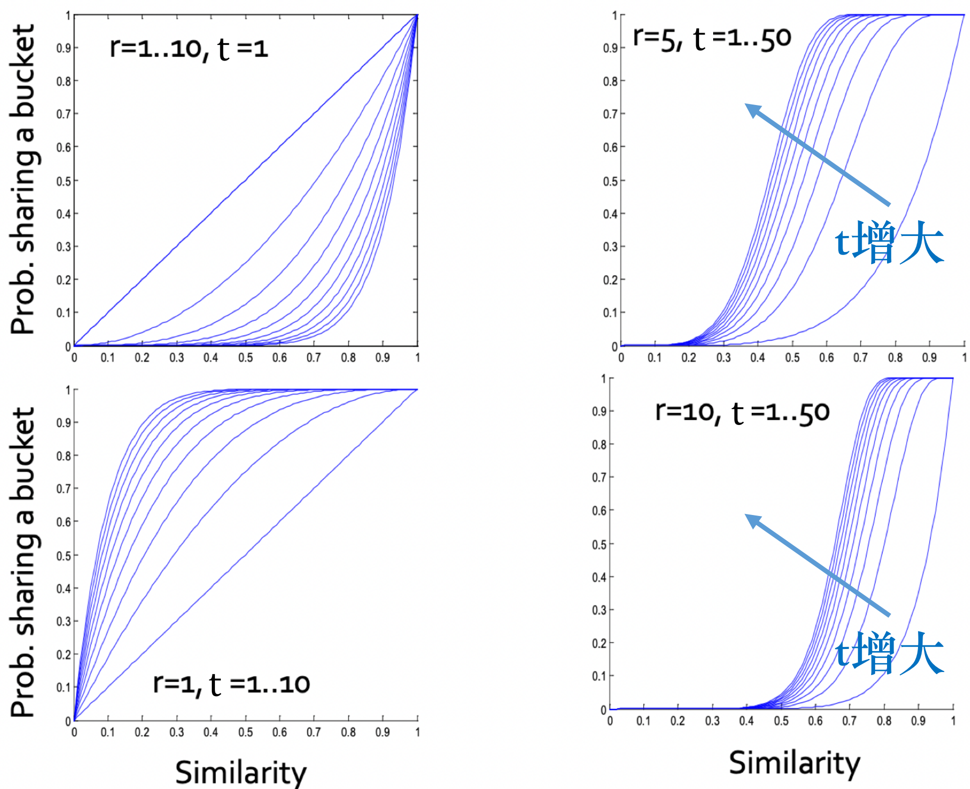


仿真图取自Jure Leskovec < Stanford CS246: Mining Massive Datasets>

局部敏感哈希

- 为每个集合（文件）用不同的哈希函数计算共$t \times r$个最小哈希值
- 用哈希函数$g(\cdot)$: $[0,1]^r \to \{0,\ldots,m-1\}$将每行MinHash向量映射为$0,\ldots,m-1$

$$\mathbf{Pr}[\text{将C找出来与A比对}] = 1-(1-s^r)^t$$

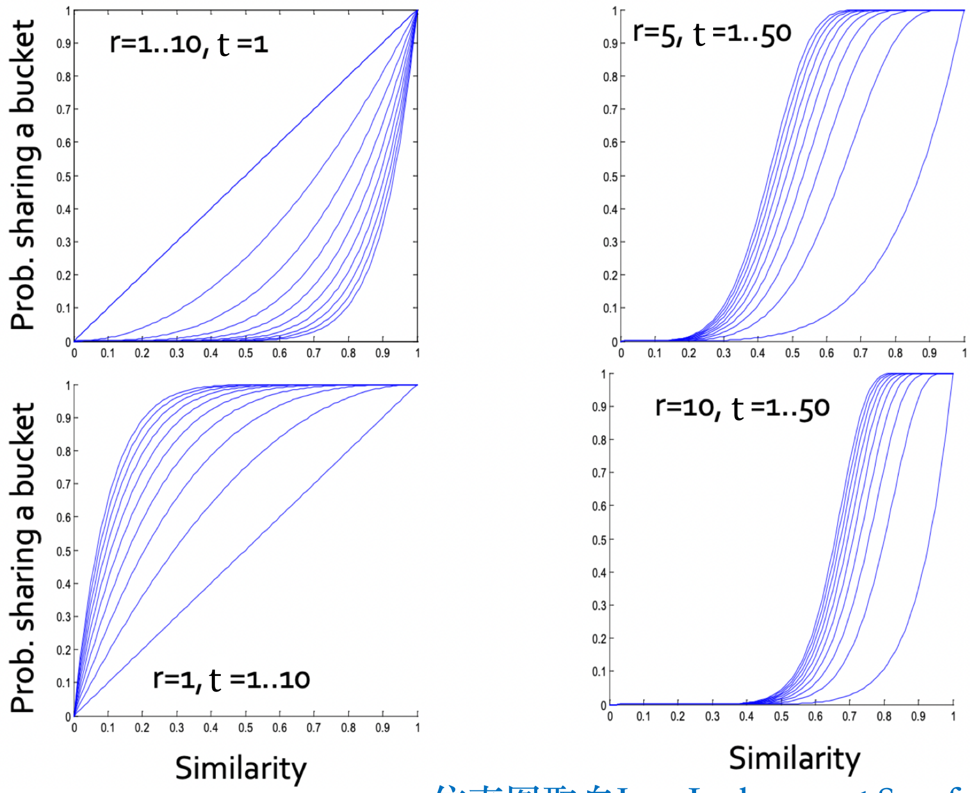


仿真图取自Jure Leskovec < Stanford CS246: Mining Massive Datasets>

局部敏感哈希

- 为每个集合（文件）用不同的哈希函数计算共$t \times r$个最小哈希值
- 用哈希函数$g(\cdot)$: $[0,1]^r \rightarrow \{0, \ldots, m-1\}$将每行MinHash向量映射为$0, \ldots, m-1$

Pr[将C找出来与A比对] $= 1 - (1 - s^r)^t$



r增大时，进入相同“桶”的条件更苛刻
将概率值向0压

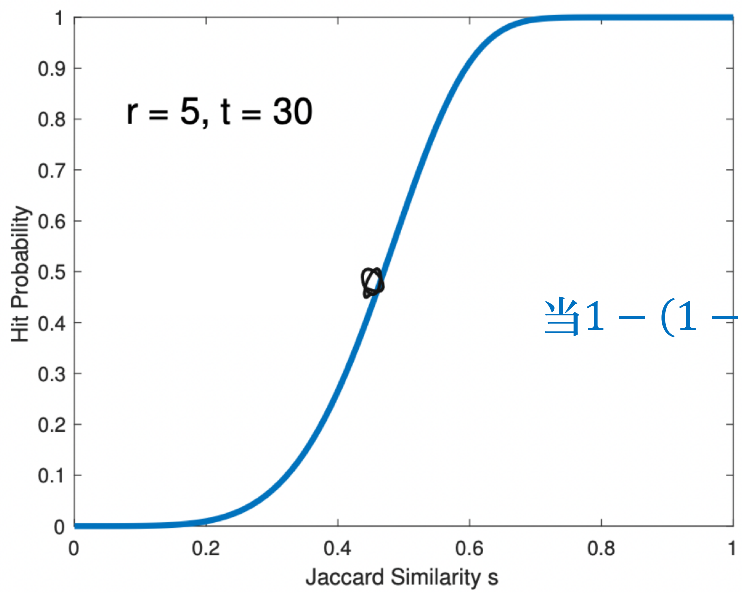
仿真图取自Jure Leskovec < Stanford CS246: Mining Massive Datasets>

局部敏感哈希

- 为每个集合（文件）用不同的哈希函数计算共$t \times r$个最小哈希值
- 用哈希函数$g(\cdot)$: $[0,1]^r \rightarrow \{0, \dots, m-1\}$将每行MinHash向量映射为$0, \dots, m-1$

$$\mathbf{Pr}[\text{将C找出来与A比对}] = 1 - (1 - s^r)^t$$

可以考虑的标准：令曲线经过(0.5, 0.5)



当$1 - (1 - s^r)^t = 0.5$时，有$s \approx 0.47$

图例取自Cameron Musco <COMPSCI 514 - Algorithms for Data Science>

局部敏感哈希

利用局部敏感哈希加速相似搜索的例子：

For example: Consider a database with 10,000,000 audio clips. You are given a clip x and want to find any y in the database with $J(x, y) \geq .9$.

- There are 10 true matches in the database with $J(x, y) \geq .9$.
- There are 10,000 near matches with $J(x, y) \in [.7, .9]$.

With signature length $r = 25$ and repetitions $t = 50$, hit probability for $J(x, y) = s$ is $1 - (1 - s^{25})^{50}$.

- Hit probability for $J(x, y) \geq .9$ is $\geq 1 - (1 - .9^{25})^{50} \approx .98$
- Hit probability for $J(x, y) \in [.7, .9]$ is $\leq 1 - (1 - .9^{25})^{50} \approx .98$
- Hit probability for $J(x, y) \leq .7$ is $\leq 1 - (1 - .7^{25})^{50} \approx .007$

Expected Number of Items Scanned: (proportional to query time)

$$\leq 10 + .98 * 10,000 + .007 * 9,989,990 \approx 80,000 \ll 10,000,000.$$

例子取自Cameron Musco <COMPSCI 514 - Algorithms for Data Science>

本讲小结

- Jaccard相似度
- 局部敏感哈希

主要参考资料

Tim Roughgarden and Gregory Valiant <CS 168 - The Modern Algorithmic Toolbox> Lecture Notes

Cameron Musco <COMPSCI 514 - Algorithms for Data Science> Slides

Bhavika Kanani <Jaccard Similarity – Text Similarity Metric in NLP> Article

Jure Leskovec < Stanford CS246: Mining Massive Datasets>

谢谢！