COMPSCI 514: ALGORITHMS FOR DATA SCIENCE

Andrew McGregor

Lecture 8

Jaccard Index: A similarity measure between two sets.

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{\#\text{ shared elements}}{\#\text{ total elements}}.$$

Want Fast Implementations For:

Jaccard Index: A similarity measure between two sets.

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{\#\text{ shared elements}}{\#\text{ total elements}}.$$

Want Fast Implementations For:

- **Near Neighbor Search:** Have a database of n sets and given a set A, want to find if it has high Jaccard similarity to anything in the database. $\Omega(n)$ time with a linear scan.

SEARCH WITH JACCARD SIMILARITY

Jaccard Index: A similarity measure between two sets.

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{\#\text{ shared elements}}{\#\text{ total elements}}.$$

Want Fast Implementations For:

- **Near Neighbor Search:** Have a database of n sets and given a set A, want to find if it has high Jaccard similarity to anything in the database. $\Omega(n)$ time with a linear scan.
- **All-pairs Similarity Search:** Have n different sets and want to find all pairs with high Jaccard similarity. $\Omega(n^2)$ time if we check all pairs explicitly.

Will speed up via randomized locality sensitive hashing.

Goal: Speed up Jaccard similarity search.

MINHASHING

Goal: Speed up Jaccard similarity search.

Strategy: Use random hashing to map each set to a very compressed representation. Jaccard similarity can be estimated from these.

Goal: Speed up Jaccard similarity search.

Strategy: Use random hashing to map each set to a very compressed representation. Jaccard similarity can be estimated from these.

MinHash(A): [Andrei Broder, 1997 at Altavista]

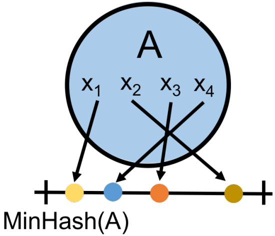
- Let $\mathbf{h} : U \rightarrow [0, 1]$ be a random hash function
- $\mathbf{s} := 1$
- For $x_1, \ldots, x_{|A|} \in A$
  - $\mathbf{s} := \min(\mathbf{s}, \mathbf{h}(x_k))$
- Return $\mathbf{s}$

Goal: Speed up Jaccard similarity search.

Strategy: Use random hashing to map each set to a very compressed representation. Jaccard similarity can be estimated from these.

MinHash(A): [Andrei Broder, 1997 at Altavista]

- Let $\mathbf{h} : U \to [0, 1]$ be a random hash function
- $\mathbf{s} := 1$
- For $x_1, \ldots, x_{|A|} \in A$
  - $\mathbf{s} := \min(\mathbf{s}, \mathbf{h}(x_k))$
- Return $\mathbf{s}$

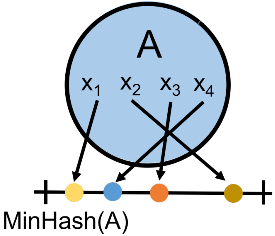


Goal: Speed up Jaccard similarity search.

Strategy: Use random hashing to map each set to a very compressed representation. Jaccard similarity can be estimated from these.

MinHash(A): [Andrei Broder, 1997 at Altavista]

- Let $\mathbf{h} : U \rightarrow [0, 1]$ be a random hash function
- $\mathbf{s} := 1$
- For $x_1, \ldots, x_{|A|} \in A$
    - $\mathbf{s} := \min(\mathbf{s}, \mathbf{h}(x_k))$
- Return $\mathbf{s}$

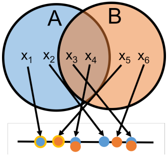


Identical to our distinct elements sketch!

For two sets A and B, what is $\Pr(MinHash(A) = MinHash(B))$?

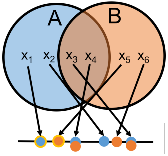
For two sets A and B, what is $\Pr(\mathit{MinHash}(A) = \mathit{MinHash}(B))$?

- Since we are hashing into the continuous range $[0, 1]$, we will never have $\mathbf{h}(x) = \mathbf{h}(y)$ for $x \neq y$ (i.e., no spurious collisions)



For two sets A and B, what is $\Pr(MinHash(A) = MinHash(B))$?

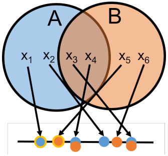
- Since we are hashing into the continuous range $[0, 1]$, we will never have $\mathbf{h}(x) = \mathbf{h}(y)$ for $x \neq y$ (i.e., no spurious collisions)



- $MH(A) = MH(B)$ iff an item in $A \cap B$ has the minimum hash value in both sets.

For two sets A and B, what is $\Pr(MinHash(A) = MinHash(B))$?

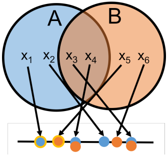
- Since we are hashing into the continuous range $[0, 1]$, we will never have $\mathbf{h}(x) = \mathbf{h}(y)$ for $x \neq y$ (i.e., no spurious collisions)



- $MH(A) = MH(B)$ iff an item in $A \cap B$ has the minimum hash value in both sets. Therefore,

For two sets A and B, what is $\Pr(MinHash(A) = MinHash(B))$?

- Since we are hashing into the continuous range $[0,1]$, we will never have $\mathbf{h}(x) = \mathbf{h}(y)$ for $x \neq y$ (i.e., no spurious collisions)



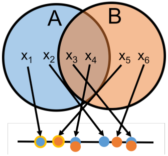
- $MH(A) = MH(B)$ iff an item in $A \cap B$ has the minimum hash value in both sets. Therefore,

$$\Pr(MH(A) = MH(B)) = \sum_{x \in A \cap B} \Pr(MH(A) = \mathbf{h}(x) \cap MH(B) = \mathbf{h}(x))$$

$$= \sum_{x \in A \cap B} \Pr(x = \arg\min_{y \in A \cup B} \mathbf{h}(y))$$

For two sets A and B, what is $\Pr(MinHash(A) = MinHash(B))$?

- Since we are hashing into the continuous range $[0, 1]$, we will never have $\mathbf{h}(x) = \mathbf{h}(y)$ for $x \neq y$ (i.e., no spurious collisions)

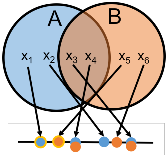


- $MH(A) = MH(B)$ iff an item in $A \cap B$ has the minimum hash value in both sets. Therefore,

$$\begin{aligned}
\Pr(MH(A) = MH(B)) &= \sum_{x \in A \cap B} \Pr(MH(A) = \mathbf{h}(x) \cap MH(B) = \mathbf{h}(x)) \\
&= \sum_{x \in A \cap B} \Pr(x = \arg\min_{y \in A \cup B} \mathbf{h}(y)) \\
&= \sum_{x \in A \cap B} \frac{1}{|A \cup B|}
\end{aligned}$$

For two sets A and B, what is $\Pr(MinHash(A) = MinHash(B))$?

- Since we are hashing into the continuous range $[0,1]$, we will never have $\mathbf{h}(x) = \mathbf{h}(y)$ for $x \neq y$ (i.e., no spurious collisions)



- $MH(A) = MH(B)$ iff an item in $A \cap B$ has the minimum hash value in both sets. Therefore,

$$\begin{aligned}
\Pr(MH(A) = MH(B)) &= \sum_{x \in A \cap B} \Pr(MH(A) = \mathbf{h}(x) \cap MH(B) = \mathbf{h}(x)) \\
&= \sum_{x \in A \cap B} \Pr(x = \arg\min_{y \in A \cup B} \mathbf{h}(y)) \\
&= \sum_{x \in A \cap B} \frac{1}{|A \cup B|} = \frac{|A \cap B|}{|A \cup B|} = J(A, B)
\end{aligned}$$

Upshot: MinHash reduces estimating the Jaccard similarity to checking equality of a *single number*.

$$\Pr(MinHash(A) = MinHash(B)) = J(A, B).$$

LOCALITY SENSITIVE HASHING

Upshot: MinHash reduces estimating the Jaccard similarity to checking equality of a *single number*.

$$\Pr(MinHash(A) = MinHash(B)) = J(A, B).$$

- An instance of locality sensitive hashing (LSH).

Upshot: MinHash reduces estimating the Jaccard similarity to checking equality of a *single number*.

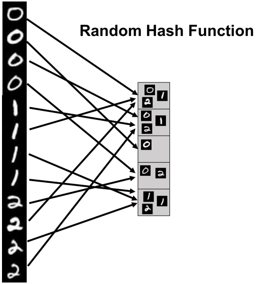
$$\Pr(MinHash(A) = MinHash(B)) = J(A, B).$$

- An instance of locality sensitive hashing (LSH).
- A hash function where the collision probability is higher when two inputs are more similar (can design different functions for different similarity metrics.)

Upshot: MinHash reduces estimating the Jaccard similarity to checking equality of a *single number*.

$$\Pr(MinHash(A) = MinHash(B)) = J(A, B).$$

- An instance of locality sensitive hashing (LSH).
- A hash function where the collision probability is higher when two inputs are more similar (can design different functions for different similarity metrics.)

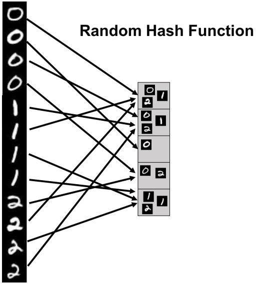


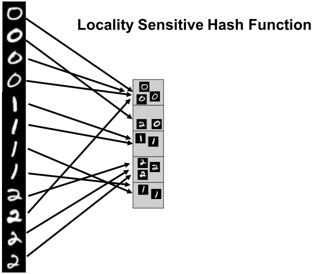
LOCALITY SENSITIVE HASHING

Upshot: MinHash reduces estimating the Jaccard similarity to checking equality of a *single number*.

$$\Pr(MinHash(A) = MinHash(B)) = J(A, B).$$

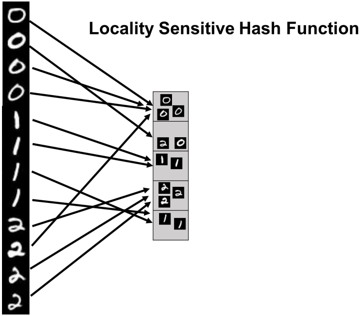
- An instance of locality sensitive hashing (LSH).
- A hash function where the collision probability is higher when two inputs are more similar (can design different functions for different similarity metrics.)





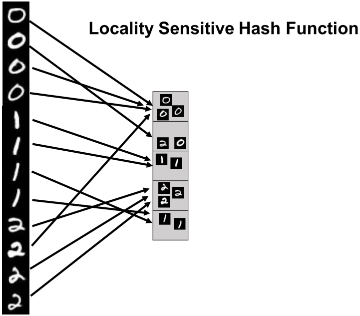
LSH FOR SIMILARITY SEARCH

How does locality sensitive hashing help for similarity search?



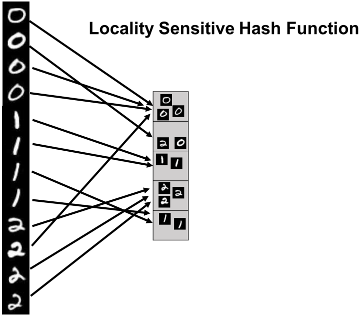
LSH FOR SIMILARITY SEARCH

How does locality sensitive hashing help for similarity search?



- **Near Neighbor Search:** Given item x, compute $\mathbf{h}(x)$. Only search for similar items in the $\mathbf{h}(x)$ bucket of the hash table.

How does locality sensitive hashing help for similarity search?



- **Near Neighbor Search:** Given item x, compute $\mathbf{h}(x)$. Only search for similar items in the $\mathbf{h}(x)$ bucket of the hash table.
- **All-pairs Similarity Search:** Scan through all buckets of the hash table and look for similar pairs within each bucket.

Goal: Given a document y, identify all documents x in a database with Jaccard similarity (of their shingle sets) $J(x, y) \geq 1/2$.

Goal: Given a document y, identify all documents x in a database with Jaccard similarity (of their shingle sets) $J(x, y) \geq 1/2$.

Our Approach:

Goal: Given a document y, identify all documents x in a database with Jaccard similarity (of their shingle sets) $J(x, y) \geq 1/2$.

Our Approach:

- Create a hash table of size m, choose a random hash function $\mathbf{g} : [0, 1] \to [m]$, and insert each item x into bucket $\mathbf{g}(MH(x))$. Search for items similar to y in bucket $\mathbf{g}(MH(y))$.

Goal: Given a document y, identify all documents x in a database with Jaccard similarity (of their shingle sets) $J(x, y) \geq 1/2$.

Our Approach:

- Create a hash table of size m, choose a random hash function $\mathbf{g} : [0, 1] \rightarrow [m]$, and insert each item x into bucket $\mathbf{g}(MH(x))$. Search for items similar to y in bucket $\mathbf{g}(MH(y))$.
- What is $\Pr[\mathbf{g}(MH(z)) = \mathbf{g}(MH(y))]$ assuming $J(z, y) \leq 1/3$ and $\mathbf{g}$ is collision free?

Goal: Given a document y, identify all documents x in a database with Jaccard similarity (of their shingle sets) $J(x, y) \geq 1/2$.

Our Approach:

- Create a hash table of size m, choose a random hash function $\mathbf{g} : [0, 1] \to [m]$, and insert each item x into bucket $\mathbf{g}(MH(x))$. Search for items similar to y in bucket $\mathbf{g}(MH(y))$.
- What is $\Pr[\mathbf{g}(MH(z)) = \mathbf{g}(MH(y))]$ assuming $J(z, y) \leq 1/3$ and $\mathbf{g}$ is collision free? At most $1/3$

Goal: Given a document y, identify all documents x in a database with Jaccard similarity (of their shingle sets) $J(x, y) \geq 1/2$.

Our Approach:

- Create a hash table of size m, choose a random hash function $\mathbf{g} : [0, 1] \rightarrow [m]$, and insert each item x into bucket $\mathbf{g}(MH(x))$. Search for items similar to y in bucket $\mathbf{g}(MH(y))$.
- What is $\Pr[\mathbf{g}(MH(z)) = \mathbf{g}(MH(y))]$ assuming $J(z, y) \leq 1/3$ and $\mathbf{g}$ is collision free? At most $1/3$
- For each document x in your database with $J(x, y) \geq 1/2$ what is the probability you will find x in bucket $\mathbf{g}(MH(y))$?

Goal: Given a document y, identify all documents x in a database with Jaccard similarity (of their shingle sets) $J(x, y) \geq 1/2$.

Our Approach:

- Create a hash table of size m, choose a random hash function $\mathbf{g} : [0, 1] \rightarrow [m]$, and insert each item x into bucket $\mathbf{g}(MH(x))$. Search for items similar to y in bucket $\mathbf{g}(MH(y))$.
- What is $\Pr[\mathbf{g}(MH(z)) = \mathbf{g}(MH(y))]$ assuming $J(z, y) \leq 1/3$ and $\mathbf{g}$ is collision free? At most $1/3$
- For each document x in your database with $J(x, y) \geq 1/2$ what is the probability you will find x in bucket $\mathbf{g}(MH(y))$? At least $1/2$

REDUCING FALSE NEGATIVES

With a simple use of MinHash, we miss a match x with $J(x, y) = 1/2$ with probability $1/2$. How can we reduce this false negative rate?

REDUCING FALSE NEGATIVES

With a simple use of MinHash, we miss a match x with $J(x, y) = 1/2$ with probability $1/2$. How can we reduce this false negative rate?

Repetition: Run MinHash t times independently, to produce hash values $MH_1(x), \ldots, MH_t(x)$. Apply random hash function $\mathbf{g}$ to map all these values to locations in t hash tables.

REDUCING FALSE NEGATIVES

With a simple use of MinHash, we miss a match x with $J(x, y) = 1/2$ with probability $1/2$. How can we reduce this false negative rate?

Repetition: Run MinHash t times independently, to produce hash values $MH_1(x), \ldots, MH_t(x)$. Apply random hash function $\mathbf{g}$ to map all these values to locations in t hash tables.

- To search for items similar to y, look at all items in bucket $\mathbf{g}(MH_1(y))$ of the 1^{st} table, bucket $\mathbf{g}(MH_2(y))$ of the 2^{nd} table, etc.

REDUCING FALSE NEGATIVES

With a simple use of MinHash, we miss a match x with $J(x, y) = 1/2$ with probability $1/2$. How can we reduce this false negative rate?

Repetition: Run MinHash t times independently, to produce hash values $MH_1(x), \ldots, MH_t(x)$. Apply random hash function $\mathbf{g}$ to map all these values to locations in t hash tables.

- To search for items similar to y, look at all items in bucket $\mathbf{g}(MH_1(y))$ of the 1^{st} table, bucket $\mathbf{g}(MH_2(y))$ of the 2^{nd} table, etc.
- What is the probability that x with $J(x, y) = 1/2$ is in at least one of these buckets, assuming for simplicity $\mathbf{g}$ has no collisions?

REDUCING FALSE NEGATIVES

With a simple use of MinHash, we miss a match x with $J(x, y) = 1/2$ with probability $1/2$. How can we reduce this false negative rate?

Repetition: Run MinHash t times independently, to produce hash values $MH_1(x), \ldots, MH_t(x)$. Apply random hash function $\mathbf{g}$ to map all these values to locations in t hash tables.

- To search for items similar to y, look at all items in bucket $\mathbf{g}(MH_1(y))$ of the 1^{st} table, bucket $\mathbf{g}(MH_2(y))$ of the 2^{nd} table, etc.
- What is the probability that x with $J(x, y) = 1/2$ is in at least one of these buckets, assuming for simplicity $\mathbf{g}$ has no collisions?
  $1-$ (probability in *no* buckets)

REDUCING FALSE NEGATIVES

With a simple use of MinHash, we miss a match x with $J(x, y) = 1/2$ with probability $1/2$. How can we reduce this false negative rate?

Repetition: Run MinHash t times independently, to produce hash values $MH_1(x), \ldots, MH_t(x)$. Apply random hash function $\mathbf{g}$ to map all these values to locations in t hash tables.

- To search for items similar to y, look at all items in bucket $\mathbf{g}(MH_1(y))$ of the 1^{st} table, bucket $\mathbf{g}(MH_2(y))$ of the 2^{nd} table, etc.
- What is the probability that x with $J(x, y) = 1/2$ is in at least one of these buckets, assuming for simplicity $\mathbf{g}$ has no collisions?
  $1-$ (probability in *no* buckets) $= 1 - \left(\frac{1}{2}\right)^t$

REDUCING FALSE NEGATIVES

With a simple use of MinHash, we miss a match x with $J(x, y) = 1/2$ with probability $1/2$. How can we reduce this false negative rate?

Repetition: Run MinHash t times independently, to produce hash values $MH_1(x), \ldots, MH_t(x)$. Apply random hash function $\mathbf{g}$ to map all these values to locations in t hash tables.

- To search for items similar to y, look at all items in bucket $\mathbf{g}(MH_1(y))$ of the 1^{st} table, bucket $\mathbf{g}(MH_2(y))$ of the 2^{nd} table, etc.
- What is the probability that x with $J(x, y) = 1/2$ is in at least one of these buckets, assuming for simplicity $\mathbf{g}$ has no collisions?
  $1-$ (probability in *no* buckets) $= 1 - \left(\frac{1}{2}\right)^t \approx .99$ for $t = 7$.

REDUCING FALSE NEGATIVES

With a simple use of MinHash, we miss a match x with $J(x, y) = 1/2$ with probability $1/2$. How can we reduce this false negative rate?

Repetition: Run MinHash t times independently, to produce hash values $MH_1(x), \ldots, MH_t(x)$. Apply random hash function $\mathbf{g}$ to map all these values to locations in t hash tables.

- To search for items similar to y, look at all items in bucket $\mathbf{g}(MH_1(y))$ of the 1^{st} table, bucket $\mathbf{g}(MH_2(y))$ of the 2^{nd} table, etc.
- What is the probability that x with $J(x, y) = 1/2$ is in at least one of these buckets, assuming for simplicity $\mathbf{g}$ has no collisions?
  $1-$ (probability in *no* buckets) $= 1 - \left(\frac{1}{2}\right)^t \approx .99$ for $t = 7$.
- What is the probability that x with $J(x, y) = 1/4$ is in at least one of these buckets, assuming for simplicity $\mathbf{g}$ has no collisions?

REDUCING FALSE NEGATIVES

With a simple use of MinHash, we miss a match x with $J(x, y) = 1/2$ with probability $1/2$. How can we reduce this false negative rate?

Repetition: Run MinHash t times independently, to produce hash values $MH_1(x), \ldots, MH_t(x)$. Apply random hash function $\mathbf{g}$ to map all these values to locations in t hash tables.

- To search for items similar to y, look at all items in bucket $\mathbf{g}(MH_1(y))$ of the 1^{st} table, bucket $\mathbf{g}(MH_2(y))$ of the 2^{nd} table, etc.
- What is the probability that x with $J(x, y) = 1/2$ is in at least one of these buckets, assuming for simplicity $\mathbf{g}$ has no collisions?
  $1-$ (probability in *no* buckets) $= 1 - \left(\frac{1}{2}\right)^t \approx .99$ for $t = 7$.
- What is the probability that x with $J(x, y) = 1/4$ is in at least one of these buckets, assuming for simplicity $\mathbf{g}$ has no collisions?
  $1-$ (probability in *no* buckets) $= 1 - \left(\frac{3}{4}\right)^t$

REDUCING FALSE NEGATIVES

With a simple use of MinHash, we miss a match x with $J(x, y) = 1/2$ with probability $1/2$. How can we reduce this false negative rate?

Repetition: Run MinHash t times independently, to produce hash values $MH_1(x), \ldots, MH_t(x)$. Apply random hash function $\mathbf{g}$ to map all these values to locations in t hash tables.

- To search for items similar to y, look at all items in bucket $\mathbf{g}(MH_1(y))$ of the 1^{st} table, bucket $\mathbf{g}(MH_2(y))$ of the 2^{nd} table, etc.
- What is the probability that x with $J(x, y) = 1/2$ is in at least one of these buckets, assuming for simplicity $\mathbf{g}$ has no collisions?
  $1-$ (probability in *no* buckets) $= 1 - \left(\frac{1}{2}\right)^t \approx .99$ for $t = 7$.
- What is the probability that x with $J(x, y) = 1/4$ is in at least one of these buckets, assuming for simplicity $\mathbf{g}$ has no collisions?
  $1-$ (probability in *no* buckets) $= 1 - \left(\frac{3}{4}\right)^t \approx .87$ for $t = 7$.

REDUCING FALSE NEGATIVES

With a simple use of MinHash, we miss a match x with $J(x, y) = 1/2$ with probability $1/2$. How can we reduce this false negative rate?

Repetition: Run MinHash t times independently, to produce hash values $MH_1(x), \ldots, MH_t(x)$. Apply random hash function $\mathbf{g}$ to map all these values to locations in t hash tables.

- To search for items similar to y, look at all items in bucket $\mathbf{g}(MH_1(y))$ of the 1^{st} table, bucket $\mathbf{g}(MH_2(y))$ of the 2^{nd} table, etc.
- What is the probability that x with $J(x, y) = 1/2$ is in at least one of these buckets, assuming for simplicity $\mathbf{g}$ has no collisions?
  $1-$ (probability in *no* buckets) $= 1 - \left(\frac{1}{2}\right)^t \approx .99$ for $t = 7$.
- What is the probability that x with $J(x, y) = 1/4$ is in at least one of these buckets, assuming for simplicity $\mathbf{g}$ has no collisions?
  $1-$ (probability in *no* buckets) $= 1 - \left(\frac{3}{4}\right)^t \approx .87$ for $t = 7$.

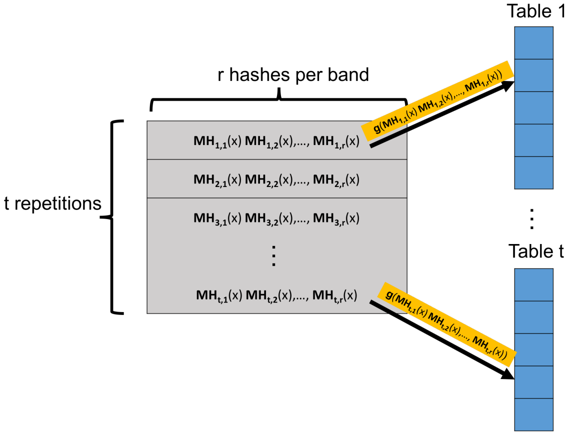
Potential for a lot of false positives! Slows down search time.

BALANCING HIT RATE AND QUERY TIME

We want to balance a small probability of false negatives (a high hit rate) with a small probability of false positives (a small query time.)

BALANCING HIT RATE AND QUERY TIME

We want to balance a small probability of false negatives (a high hit rate) with a small probability of false positives (a small query time.)



Create t hash tables. Each is indexed into not with a single MinHash value, but with r values, appended together. A length r signature.

BALANCING HIT RATE AND QUERY TIME

Consider searching for matches in t hash tables, using MinHash signatures of length r. For x and y with Jaccard similarity $J(x, y) = s$:

BALANCING HIT RATE AND QUERY TIME

Consider searching for matches in t hash tables, using MinHash signatures of length r. For x and y with Jaccard similarity $J(x, y) = s$:

- Probability that a single hash matches.
  $\Pr[MH_{i,j}(x) = MH_{i,j}(y)] = J(x, y) = s$.

BALANCING HIT RATE AND QUERY TIME

Consider searching for matches in t hash tables, using MinHash signatures of length r. For x and y with Jaccard similarity $J(x, y) = s$:

- Probability that a single hash matches.
  $\Pr[MH_{i,j}(x) = MH_{i,j}(y)] = J(x, y) = s$.
- Probability that x and y having matching signatures in repetition i.
  $\Pr[MH_{i,1}(x), \ldots, MH_{i,r}(x) = MH_{i,1}(y), \ldots, MH_{i,r}(y)]$.

BALANCING HIT RATE AND QUERY TIME

Consider searching for matches in t hash tables, using MinHash signatures of length r. For x and y with Jaccard similarity $J(x, y) = s$:

- Probability that a single hash matches.
  $\Pr[MH_{i,j}(x) = MH_{i,j}(y)] = J(x, y) = s$.
- Probability that x and y having matching signatures in repetition i.
  $\Pr[MH_{i,1}(x), \ldots, MH_{i,r}(x) = MH_{i,1}(y), \ldots, MH_{i,r}(y)] = s^r$.

BALANCING HIT RATE AND QUERY TIME

Consider searching for matches in t hash tables, using MinHash signatures of length r. For x and y with Jaccard similarity $J(x, y) = s$:

- Probability that a single hash matches.
  $\Pr[MH_{i,j}(x) = MH_{i,j}(y)] = J(x, y) = s$.
- Probability that x and y having matching signatures in repetition i.
  $\Pr[MH_{i,1}(x), \ldots, MH_{i,r}(x) = MH_{i,1}(y), \ldots, MH_{i,r}(y)] = s^r$.
- Probability that x and y don't match in repetition i:

BALANCING HIT RATE AND QUERY TIME

Consider searching for matches in t hash tables, using MinHash signatures of length r. For x and y with Jaccard similarity $J(x,y) = s$:

- Probability that a single hash matches.
  $\Pr\left[MH_{i,j}(x) = MH_{i,j}(y)\right] = J(x,y) = s$.
- Probability that x and y having matching signatures in repetition i.
  $\Pr\left[MH_{i,1}(x), \ldots, MH_{i,r}(x) = MH_{i,1}(y), \ldots, MH_{i,r}(y)\right] = s^r$.
- Probability that x and y don't match in repetition i: $1 - s^r$.

BALANCING HIT RATE AND QUERY TIME

Consider searching for matches in t hash tables, using MinHash signatures of length r. For x and y with Jaccard similarity $J(x, y) = s$:

- Probability that a single hash matches.
  $\Pr[MH_{i,j}(x) = MH_{i,j}(y)] = J(x, y) = s$.
- Probability that x and y having matching signatures in repetition i.
  $\Pr[MH_{i,1}(x), \ldots, MH_{i,r}(x) = MH_{i,1}(y), \ldots, MH_{i,r}(y)] = s^r$.
- Probability that x and y don't match in repetition i: $1 - s^r$.
- Probability that x and y don't match in *all repetitions*:

BALANCING HIT RATE AND QUERY TIME

Consider searching for matches in t hash tables, using MinHash signatures of length r. For x and y with Jaccard similarity $J(x, y) = s$:

- Probability that a single hash matches.
  $\Pr[MH_{i,j}(x) = MH_{i,j}(y)] = J(x, y) = s$.
- Probability that x and y having matching signatures in repetition i.
  $\Pr[MH_{i,1}(x), \ldots, MH_{i,r}(x) = MH_{i,1}(y), \ldots, MH_{i,r}(y)] = s^r$.
- Probability that x and y don't match in repetition i: $1 - s^r$.
- Probability that x and y don't match in *all repetitions*: $(1 - s^r)^t$.

BALANCING HIT RATE AND QUERY TIME

Consider searching for matches in t hash tables, using MinHash signatures of length r. For x and y with Jaccard similarity $J(x, y) = s$:

- Probability that a single hash matches.
  $\Pr[MH_{i,j}(x) = MH_{i,j}(y)] = J(x, y) = s$.
- Probability that x and y having matching signatures in repetition i.
  $\Pr[MH_{i,1}(x), \ldots, MH_{i,r}(x) = MH_{i,1}(y), \ldots, MH_{i,r}(y)] = s^r$.
- Probability that x and y don't match in repetition i: $1 - s^r$.
- Probability that x and y don't match in *all repetitions*: $(1 - s^r)^t$.
- Probability that x and y match in at least one repetition:

BALANCING HIT RATE AND QUERY TIME

Consider searching for matches in t hash tables, using MinHash signatures of length r. For x and y with Jaccard similarity $J(x, y) = s$:

- Probability that a single hash matches.
  $\Pr[MH_{i,j}(x) = MH_{i,j}(y)] = J(x, y) = s$.
- Probability that x and y having matching signatures in repetition i.
  $\Pr[MH_{i,1}(x), \ldots, MH_{i,r}(x) = MH_{i,1}(y), \ldots, MH_{i,r}(y)] = s^r$.
- Probability that x and y don't match in repetition i: $1 - s^r$.
- Probability that x and y don't match in *all repetitions*: $(1 - s^r)^t$.
- Probability that x and y match in at least one repetition:

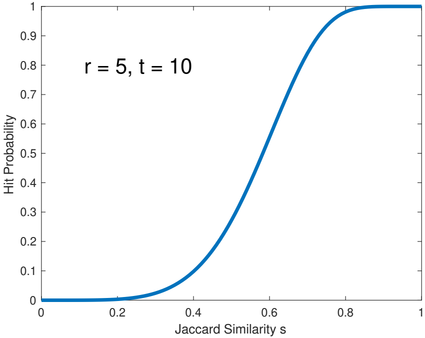
$$\text{Hit Probability: } 1 - (1 - s^r)^t.$$

THE s-CURVE

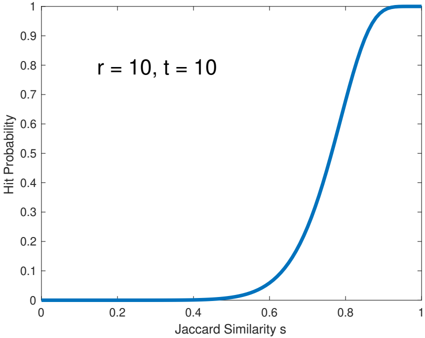
Using t repetitions each with a signature of r MinHash values, the probability that x and y with Jaccard similarity $J(x, y) = s$ match in at least one repetition is: $1 - (1 - s^r)^t$.

THE s-CURVE

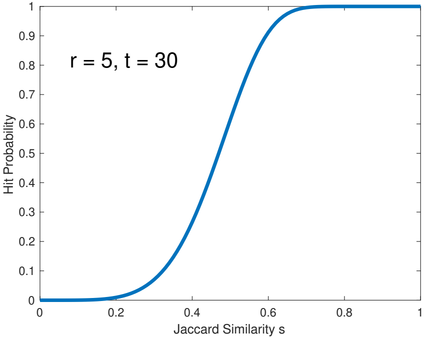
Using t repetitions each with a signature of r MinHash values, the probability that x and y with Jaccard similarity $J(x, y) = s$ match in at least one repetition is: $1 - (1 - s^r)^t$.



Using t repetitions each with a signature of r MinHash values, the probability that x and y with Jaccard similarity $J(x, y) = s$ match in at least one repetition is: $1 - (1 - s^r)^t$.

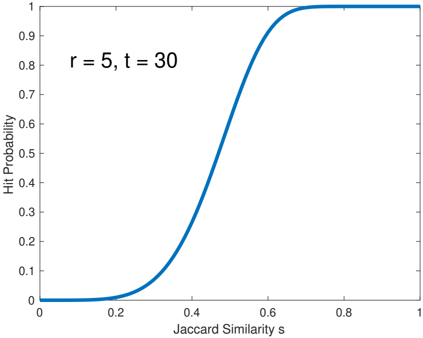


Using t repetitions each with a signature of r MinHash values, the probability that x and y with Jaccard similarity $J(x, y) = s$ match in at least one repetition is: $1 - (1 - s^r)^t$.



THE s-CURVE

Using t repetitions each with a signature of r MinHash values, the probability that x and y with Jaccard similarity $J(x, y) = s$ match in at least one repetition is: $1 - (1 - s^r)^t$.



r and t are tuned depending on application. 'Threshold' when hit probability is $1/2$ is $\approx (1/t)^{1/r}$. E.g., $\approx (1/30)^{1/5} = .51$ in this case.

s-CURVE EXAMPLE

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 $J(x,y) \geq .9$.

s-CURVE EXAMPLE

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 $J(x, y) \in [.7, .9]$.

s-CURVE EXAMPLE

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 $1 - (1 - s^{25})^{50}$.

s-CURVE EXAMPLE

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 $\leq 1-(1-.7^{25})^{50} \approx .007$

s-CURVE EXAMPLE

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)

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

s-CURVE EXAMPLE

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)

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

Repetition and s-curve tuning can be used for fast similarity search with other similarity metrics:

GENERALIZING LOCALITY SENSITIVE HASHING

Repetition and s-curve tuning can be used for fast similarity search with other similarity metrics:

- LSH schemes exist for many similarity/distance measures: hamming distance, cosine similarity, etc.

GENERALIZING LOCALITY SENSITIVE HASHING

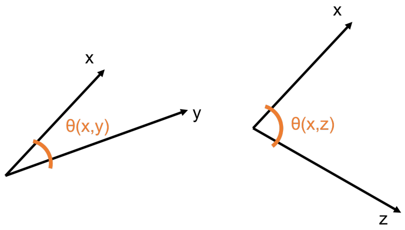
Repetition and s-curve tuning can be used for fast similarity search with other similarity metrics:

- LSH schemes exist for many similarity/distance measures: hamming distance, cosine similarity, etc.

GENERALIZING LOCALITY SENSITIVE HASHING

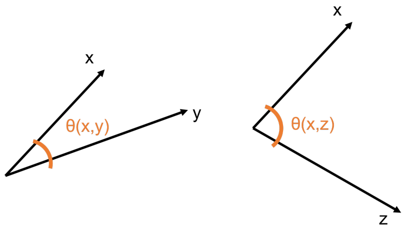
Repetition and s-curve tuning can be used for fast similarity search with other similarity metrics:

- LSH schemes exist for many similarity/distance measures: hamming distance, cosine similarity, etc.



Repetition and *s*-curve tuning can be used for fast similarity search with other similarity metrics:

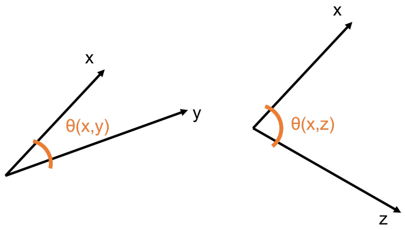
- LSH schemes exist for many similarity/distance measures: hamming distance, cosine similarity, etc.



Cosine Similarity: $\cos(\theta(x, y))$

Repetition and *s*-curve tuning can be used for fast similarity search with other similarity metrics:

- LSH schemes exist for many similarity/distance measures: hamming distance, cosine similarity, etc.



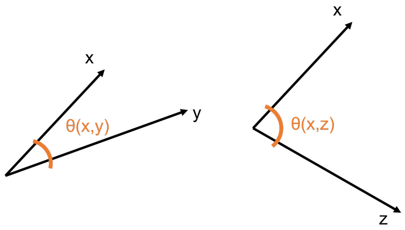
Cosine Similarity: $\cos(\theta(x, y))$

- $\cos(\theta(x, y)) = 1$ when $\theta(x, y) = 0^\circ$ and $\cos(\theta(x, y)) = 0$ when $\theta(x, y) = 90^\circ$, and $\cos(\theta(x, y)) = -1$ when $\theta(x, y) = 180^\circ$

GENERALIZING LOCALITY SENSITIVE HASHING

Repetition and s-curve tuning can be used for fast similarity search with other similarity metrics:

- LSH schemes exist for many similarity/distance measures: hamming distance, cosine similarity, etc.



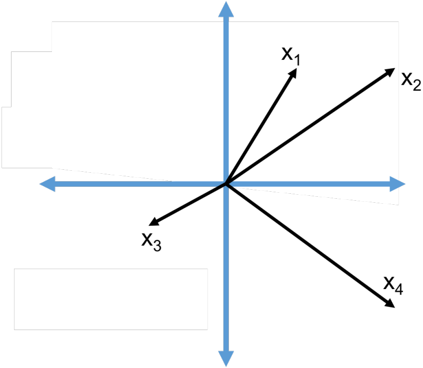
Cosine Similarity: $\cos(\theta(x,y)) = \frac{\langle x,y \rangle}{\|x\|_2 \cdot \|y\|_2}$.

- $\cos(\theta(x,y)) = 1$ when $\theta(x,y) = 0^\circ$ and $\cos(\theta(x,y)) = 0$ when $\theta(x,y) = 90^\circ$, and $\cos(\theta(x,y)) = -1$ when $\theta(x,y) = 180^\circ$

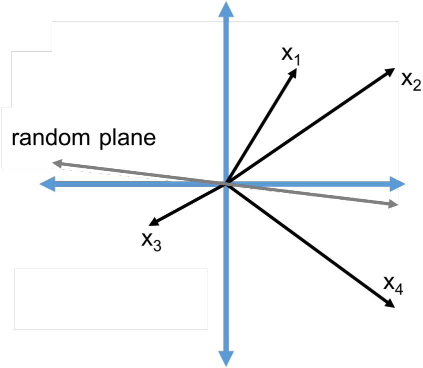
SIMHASH FOR COSINE SIMILARITY

SimHash Algorithm: LSH for cosine similarity.

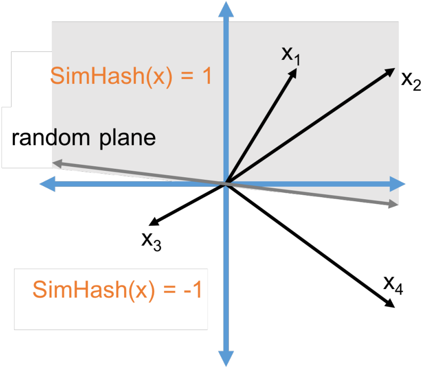
SimHash Algorithm: LSH for cosine similarity.



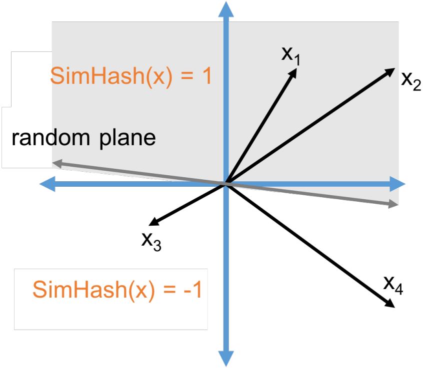
SimHash Algorithm: LSH for cosine similarity.



SimHash Algorithm: LSH for cosine similarity.



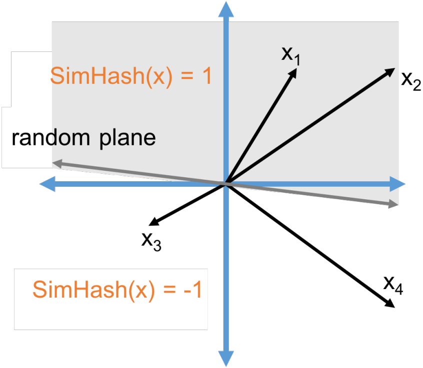
SimHash Algorithm: LSH for cosine similarity.



$$SimHash(x) = \text{sign}(\langle x, t \rangle) \text{ for a random vector } t.$$

SIMHASH FOR COSINE SIMILARITY

SimHash Algorithm: LSH for cosine similarity.



$$\mathit{SimHash}(x) = \mathrm{sign}(\langle x, t \rangle) \text{ for a random vector } t.$$

What is $\Pr[\mathit{SimHash}(x) = \mathit{SimHash}(y)]$?

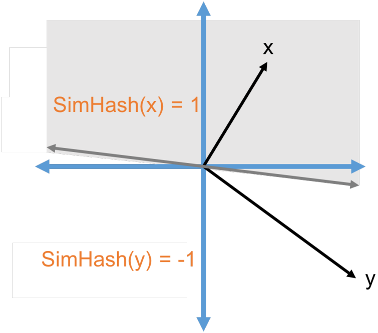
SIMHASH FOR COSINE SIMILARITY

What is $\Pr[SimHash(x) = SimHash(y)]$?

SIMHASH FOR COSINE SIMILARITY

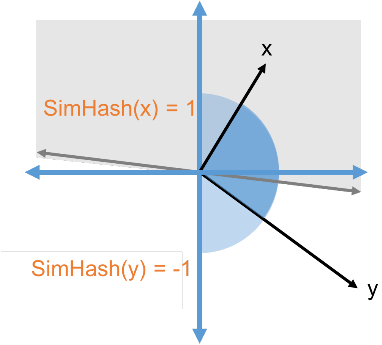
What is $\Pr[SimHash(x) = SimHash(y)]$?

$SimHash(x) \neq SimHash(y)$ when the plane separates x from y.



What is $\Pr[SimHash(x) = SimHash(y)]$?

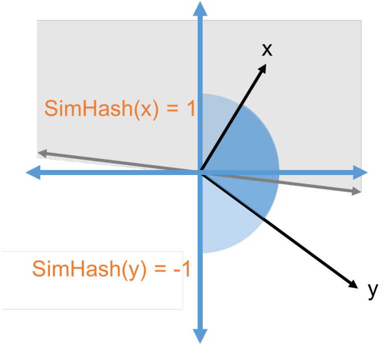
$SimHash(x) \neq SimHash(y)$ when the plane separates x from y.



SIMHASH FOR COSINE SIMILARITY

What is $\Pr[SimHash(x) = SimHash(y)]$?

$SimHash(x) \neq SimHash(y)$ when the plane separates x from y.

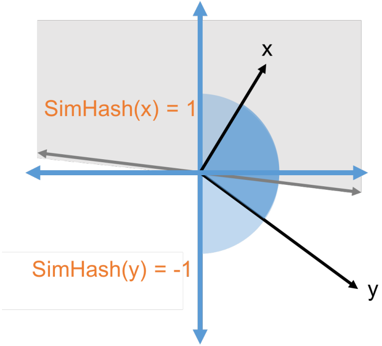


- $\Pr[SimHash(x) \neq SimHash(y)] = \frac{\theta(x,y)}{180}$

SIMHASH FOR COSINE SIMILARITY

What is $\Pr[SimHash(x) = SimHash(y)]$?

$SimHash(x) \neq SimHash(y)$ when the plane separates x from y.



- $\Pr[SimHash(x) \neq SimHash(y)] = \frac{\theta(x,y)}{180}$
- $\Pr[SimHash(x) = SimHash(y)] = 1 - \frac{\theta(x,y)}{180} \approx \cos\theta$ for small θ.

Questions on MinHash and Locality Sensitive Hashing?