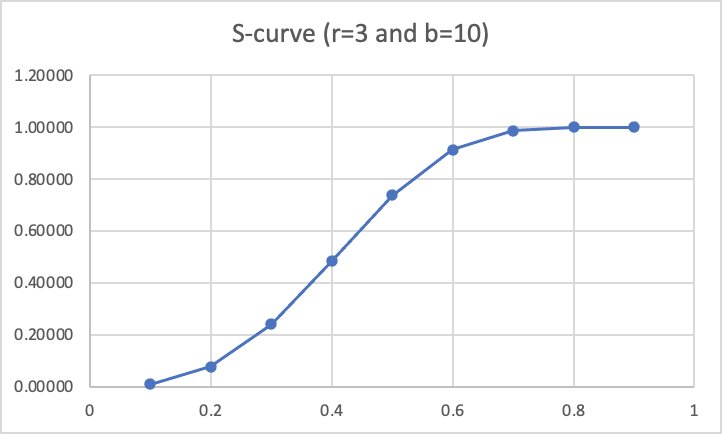
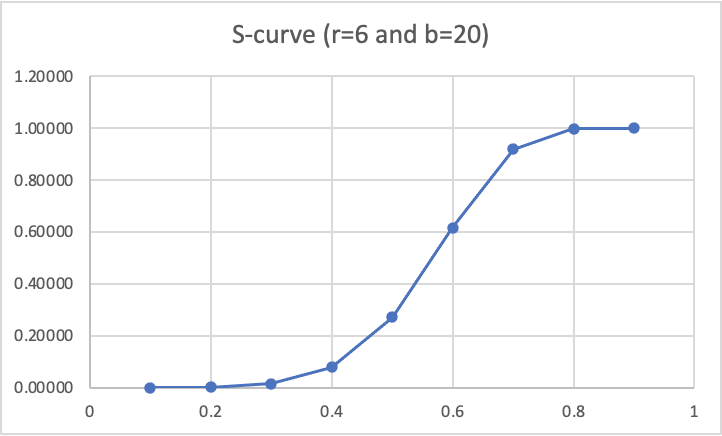
### MBD Assignment 2 - Team 6

Yingyao Lu a1784870 Jin Yin aXXXX

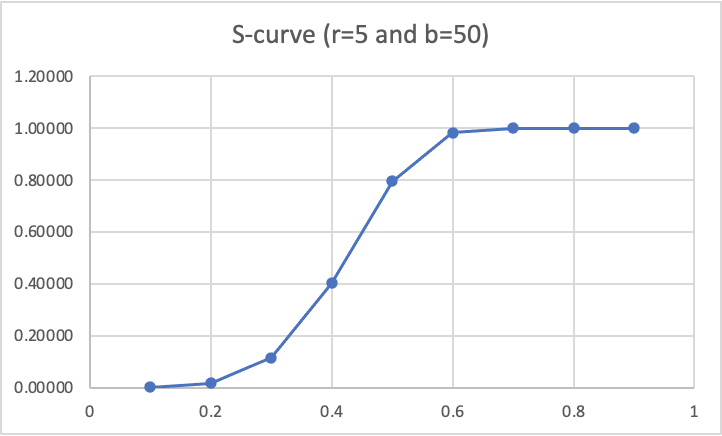
## Exercise 1 S-curve (5+5+5 points)

1.S-curve for r=3 and b=10

|  |  |
| --- | --- |
| s | S-curve (r=3 and b=10) |
| 0.1 | 0.00996 |
| 0.2 | 0.07718 |
| 0.3 | 0.23945 |
| 0.4 | 0.48387 |
| 0.5 | 0.73692 |
| 0.6 | 0.91227 |
| 0.7 | 0.98502 |
| 0.8 | 0.99923 |
| 0.9 | 1.00000 |

2. S-curve for r=6 and b=20

|  |  |
| --- | --- |
| s | S-curve (r=6 and b=20) |
| 0.1 | 0.00002 |
| 0.2 | 0.00128 |
| 0.3 | 0.01448 |
| 0.4 | 0.07881 |
| 0.5 | 0.27019 |
| 0.6 | 0.61541 |
| 0.7 | 0.91819 |
| 0.8 | 0.99771 |
| 0.9 | 1.00000 |

3. S-curve for r=5 and b=50

|  |  |
| --- | --- |
| s | S-curve (r=5 and b=50) |
| 0.1 | 0.00050 |
| 0.2 | 0.01588 |
| 0.3 | 0.11454 |
| 0.4 | 0.40228 |
| 0.5 | 0.79555 |
| 0.6 | 0.98253 |
| 0.7 | 0.99990 |
| 0.8 | 1.00000 |
| 0.9 | 1.00000 |

**Exercise 2 Filtering Streams (8 + 8 points)**

1. In general, the probability of a false positive is the probability of a 1-bit raised to the power of k, I, e :
   1. m=2bil, n=10bil, k=3= (1-0.54881)^3 = 0.4512^3 = 0.09185
   2. m=2bil, n=10bil, k=4= (1-0.44933)^4 = 0.55067^4 = 0.09195

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| m=2 n=10 | K=3 | K=4 | K=5 | K=6 |
|  | 0.0918487 | 0.09195348 | 0.10092499 | 0.11644961 |

2 . basing on the formular:

We want a minimal false-positive rate is the result is 0.

For that result, we can get a equation : =1

So we get the k’s value.

k =

## Exercise 3 PageRank (22+13 points)

Text, letter

Description automatically generated

## The top 10 highest rank of the pages:

|  |  |  |
| --- | --- | --- |
| No | Node ID | Rank |
| 1 | 163075 | 6.57930344155782E-4 |
| 2 | 597621 | 6.538465907537884E-4 |
| 3 | 537039 | 6.41168046820425E-4 |
| 4 | 41909 | 6.250066824853455E-4 |
| 5 | 384666 | 5.398350690554848E-4 |
| 6 | 504140 | 5.375654569547856E-4 |
| 7 | 605856 | 5.267175479252007E-4 |
| 8 | 551829 | 5.168942917752945E-4 |
| 9 | 486980 | 5.135998255216842E-4 |
| 10 | 558791 | 5.046655488181088E-4 |

The codes and the relative document are in the zip.

## Exercise 4 Data streams (7 + 7 points)

## 4.1 Input stream = 3, 1, 4, 6, 5, 9

Q1: h(x) = (2x+1) mod 32

h(3) = 7 = 00111 = 0

h(1) = 3 = 00011= 0

h(4) = 9 = 01001= 0

h(6) = 13 = 01101= 0

h(5) = 11 = 01011= 0

h(9) = 19 = 10011= 0

Max Tail Length R = 0

Estimate number of distinct elements 2^R= 2^0 = 1

Q2: h(x) = (3x+7) mod 32

h(3) = 16 = 10000=4

h(1) = 10 = 11010=1

h(4) = 19 = 10011=0

h(6) = 25 = 11001=0

h(5) = 22 = 10110=1

h(9) = 2 = 00010=1

Max Tail Length R = 4

Estimate number of distinct elements 2^R = 2^4 =16

Q3: h(x) = 4x mod 32

h(3) = 12 = 10000=4

h(1) = 4 = 00100=2

h(4) = 16 = 10000=4

h(6) = 24 = 11000=3

h(5) = 20 = 10100=2

h(9) = 4 = 00100=2

Max Tail Length R = 4

Estimate number of distinct elements 2^R = 2^4 = 16

Q4: h(x) = (6x+2) mod 32

h(4) = 26 = 11010 = 1

h(5) = 0 = 00000= 0

h(6) = 6 = 00110= 1

h(7) = 12 = 01100= 2

h(10) = 30 = 11110= 1

h(15) = 28 = 11100= 2

Max Tail Length R = 2

Estimate number of distinct elements 2^R= 2^2 = 4

Q5: h(x) = (2x+5) mod 32

h(4) = 13 = 01101 = 0

h(5) = 15 = 01111= 0

h(6) = 17 = 10001= 0

h(7) = 19 = 10011= 0

h(10) = 25 = 11001= 0

h(15) = 3 = 00011= 0

Max Tail Length R = 0

Estimate number of distinct elements 2^0 = 2^0 =1

Q6: h(x) = 2x mod 32

h(4) = 8 = 01000 = 3

h(5) = 10 = 01010= 1

h(6) = 12 = 01100= 2

h(7) = 14 = 01110= 1

h(10) = 20 = 10100= 2

h(15) = 30 = 11110= 1

Max Tail Length R = 3

Estimate number of distinct elements 2^R = 2^3 = 8

## Exercise 5 Summary of 3.6 and 3.7 (10 +10 points)

Summarize the content of 3.6 in your own words (600 words).

3.6 The Theory of Locality-Sensitive Functions

Locality-sensitive hashing (LSH) is an algorithm that hashes similar inputs into the same buckets with high probability. The objective of LSH is to map high dimensional points into a lower dimensional space in such a way that points that are close to each other.

Besides the minhash functions, this section 3.6 discusses about some other families of functions that can produce candidate pairs efficiently.

There Locality-Sensitive Functions have three conditions in common:

1. They must be more possible to choose close pairs as candidate pairs.
2. They must be statistically independent.
3. They must be efficient to

(a) identify candidate pairs in time much less than the time it takes to look at all pairs.

(b) be combinable to avoid false positives and false negatives.

These functions can apply to the space of sets and the Jaccard distance, or to another space and/or another distance measure.

## 3.6.1 Locality-Sensitive Functions definition

In situation that we don’t want to check all n items, but we want to somehow figure out explore whether two items are similar/ might be close to each other that we could looked like a candidate pair.

The function F needs to be a good hashing function to hash items, and the equality of the results relate to decision of pairs

f(x) = f(y) demonstrates that x and y is a candidate pair.

d(x,y) is the distance between x and y

d1, d2 are two distances metric, where d1 < d2.

P1, p2 are two possibilities that between 0 and 1, and p1< p2

Compare d(x,y) with d1 and d2,

LSF states that a family of functions F is said to be a (d1, d2, p1, p2) locally sensitive family:

1. If d(x,y) ≤ d1 probability of f(x) = f(y) >= p1.

2. If d(x,y) ≥ d2, probability of f(x) = f(y) is <= p2.

The (d1, d2, p1, p2)-sensitive function show as follows:

Line chart

Description automatically generated with medium confidence

3.6.2 Locality-Sensitive Families for Jaccard Distance

We will use minhash functions and d is the Jaccard distance to find a family of Locality-Sensitive functions.

We interpret a minhash function h, if and only if h(x) = h(y), x and y is a candidate pair.

P1 -> 1-d1, p2 -> 1-d2, where0<=d1 <d2 <=1.

SIM(x, y) = 1 − d(x, y) ≥ 1 − d1.

Therefore, we could say the family of minhash functions is a (d1, d2, 1-d1, 1-d2)-sensitive family.

3.6.3 Amplification of Locality-Sensitive Families

A family of Locality-Sensitive functions F as defined above (satisfying p1 > p2) can be combined using AND and OR operations to produce new functions that approach this ideal LSH function. Given a (d1,d2,p1,p2){\displaystyle (d\_{1},d\_{2},p\_{1},p\_{2})}()-sensitive family F{\displaystyle {\mathcal {F}}}FF, we can construct new families {\displaystyle {\mathcal {G}}}by either the AND-construction or OR-construction of F.{\displaystyle {\mathcal {F}}}..

## The AND operation

Given a (d1,d2,p1,p2) sensitive family F, we say F′ is a (d1,d2,(p1)^r,(p2)^r)-sensitive family.

Where r functions h1, h2, . . . hr without replacement from F. Thus, while reducing the probability of a collision, AND amplifies the difference in probabilities of collisions between nearby and far points.

## The OR operation

The OR construction turns a (d1, d2, p1, p2)-sensitive family F into a (d1, d2, 1 − (1 − p1)^b, 1 − (1 − p2)^b)- sensitive family F′.

The OR operation boosts the chances of a collision in F’. Thus while boosting the probability of a collision, OR also boosts the probability of collision more nearby points than for points farther away, and thus is also an amplifying operation for hash functions.

## Concatenation of AND and OR

Concatenation of AND and OR can be used to combine hash functions to produce amplified hash functions that are near-ideal. The following equation shows a family of hash functions H with collision probabilities p transforms after AND and OR in sequence.

## Summarize the content of 3.7 in your own words (600 words).

This part is about locality-sensitive families construction for Hamming distance, the cosine distance and for the normal Euclidean distance.

Terms in this section:

**Hyperplane**: a hyperplane is a subspace whose dimension is one less than that of its ambient space.

It sperate the space into two space.

**Normal** vector: a normal is an object such as a line, ray, or vector that is perpendicular to a given object. In this section is a vector that is perpendicular to a plane (or a hyperplane).

In a given space (it could be multiple-dimension space), two intersecting vectors will define a plane, and there must be an angle, is used in the textbook, between two vectors. And this angle is the cosine distance.

For the hyperplane choosing, there are uncountable Hyperplanes through intersection point of two vectors, origin is used in the textbook. Two hyperplanes are chosen randomly.

Two normal lines, dash line (a) and dot line (b), is provide in following picture.

Chart

Description automatically generated with low confidence

Vector x and y are in the different sides of a, which means the point on the x line has a different sign to the points on the y line. But if we choose line b as the normal vector, vector x and vector y will have a same sign.

The formular of the calculating the probability that a normal vector is like line a.

Only the line in the angle can be similar to line a. In this case, the probability is . Of course the probability that a normal vector as line b is 1- .

To make expression simple, we use +1 and -1 to express the normal vector, let’s say v, to multiple the vector x and y.

It can be written as: . If the result is positive, we record it as 1, otherwise, it si -1. We collection the result as a new vector which is the sketch. The sketches only contain +1, and -1. At last we can compare two sketches to calculate the ratio of the same index, use this percentage multiple 180.

However, this method is not accurate enough.

In a 2-dimensional Euclidean space.

Chart, diagram

Description automatically generated

All the hash function in the family will be associated with a randomly chosen line in this space.

The buckets on the line have a length of a. the points on d line would have a projection point in one of these buckets, which means it will be hashed into different bucket by a hashing function.

In this picture, it is clear that the length of d and the width of a, will decide the probability that a point hash into a bucket. If the two points is close enough, they will hash into the same bucket. These two points can be seen as equal.

For the example given by the textbook, d = a/2, so there are 50% chance the two points hash to the same bucket. I think 50% is made by 25%+ 25%.

For the picture above, the chance is that the probability that two points hashing into the left bucket add the probability for middle bucket and add the probability for the right bucket.

We may find that if the d is longer than a, the probability that two different point hashing into the same bucket is lower.

Of course, another element is the angle , if this angle is big enough, the probability two points hashing into one bucket is greater, if the angle is 90 degree, the percentage would be 100%.

For multiple-dimension space, there will be a locality-sensitive-function (d1, d2, p1, p2). It is still possible to use a random line and buckets with width of a to partition a line.