Exercise 1) Minhashing:

a. Compute the minhash signature for each column if we use the following three hash functions: $h_1(x) = 2x + 1 \bmod 6$; $h_2(x) = 3x + 2 \bmod 6$; $h_3(x)=5x+2 \bmod 6$.

Element	**S1**	**S2**	**S3**	**S4**	$h_1(x) = 2x + 1 \bmod 6$	$h_2(x) = 3x + 2 \bmod 6$	$h_3(x)=5x+2 \bmod 6$
0	**0**	**1**	**0**	**1**	**1**	**2**	**2**
1	**0**	**1**	**0**	**0**	**3**	**5**	**1**
2	**1**	**0**	**0**	**1**	**5**	**2**	**0**
3	**0**	**0**	**1**	**0**	**1**	**5**	**5**
4	**0**	**0**	**1**	**1**	**3**	**2**	**4**
5	**1**	**0**	**0**	**0**	**5**	**5**	**3**

Initialize all hash values with max infinitiy value

Hash function	**S1**	**S2**	**S3**	**S4**
$h_1(x)$	∞	∞	∞	∞
$h_2(x)$	∞	∞	∞	∞
$h_3(x)$	∞	∞	∞	∞

Row 0 only S3 and S4 have a 1 so only they will be updated

Hash function	**S1**	**S2**	**S3**	**S4**
$h_1(x)$	∞	**1**	∞	**1**
$h_2(x)$	∞	**2**	∞	**2**
$h_3(x)$	∞	**2**	∞	**2**

Row 1, only s2 and $h_3(x)$ **was 2 now it is 1**

Hash function	**S1**	**S2**	**S3**	**S4**
$h_1(x)$	∞	**1**	∞	**1**
$h_2(x)$	∞	**2**	∞	**2**
$h_3(x)$	∞	**1**	∞	**2**

Row 2 changes for S1 and S4

Hash function	**S1**	**S2**	**S3**	**S4**
$h_1(x)$	**5**	**1**	∞	**1**

$h_2(x)$	**2**	**2**	∞	**2**
$h_3(x)$	**0**	**1**	∞	**0**

Row 3 changes for S3

Hash function	**S1**	**S2**	**S3**	**S4**
$h_1(x)$	**5**	**1**	**1**	**1**
$h_2(x)$	**2**	**2**	**5**	**2**
$h_3(x)$	**0**	**1**	**5**	**0**

Row 4 Changes for S3

Hash function	**S1**	**S2**	**S3**	**S4**
$h_1(x)$	**5**	**1**	**1**	**1**
$h_2(x)$	**2**	**2**	**2**	**2**
$h_3(x)$	**0**	**1**	**4**	**0**

Row 5 no change for S1

Hash function	**S1**	**S2**	**S3**	**S4**
$h_1(x)$	**5**	**1**	**1**	**1**
$h_2(x)$	**2**	**2**	**2**	**2**
$h_3(x)$	**0**	**1**	**4**	**0**

b. Which of these hash functions are true permutations?

Only function h3 is a true permutation.

H2 and h1 are not true permutations as there are duplicate row numbers for each of the hash functions.

c. How close are the estimated Jaccard similarities for the six pairs of columns to the true Jaccard similarities? Compute the true Jaccard similarities and compare them.

Hash function	**J(S1,S2)**	**J(S1,S3)**	**J(S1,S4)**	**J(S2,S3)**	**J(S2,S4)**	**J(S3,S4)**
Columns	**0**	**0**	**0.25**	**0**	**0.25**	**0.25**

Minhash	**0.33**	**0.33**	**0.67**	**0.67**	**0.67**	**0.67**

Exercise 2)

In the lecture, you learned how to compute minhash signatures for a given binary matrix representation of documents and shingles. Suppose we want to use a MapReduce framework to compute minhash signatures. If the matrix is stored in chunks that correspond to some columns, then it is quite easy to exploit parallelism. Each Map task gets some of the columns and all the hash functions, and computes the minhash signatures of its given columns. However, suppose the matrix were chunked by rows, so that a Map task is given the hash functions and a set of rows to work on. Design Map and Reduce functions to exploit MapReduce with data in this form. You can write the map and reduce functions as a pseudocode or you can also just describe what mappers and reducers do in your own words.

In this form of row partitioning each mapper is given a subset of k-grams for every document. Since the mappers do not have the entire set of 3

k-grams, a min-hash value for each document cannot be computed and instead a hash value for the set of k-grams must be computed. The hash values are then grouped by the document ID and the reducers then calculate the min-hash value for every document ID it is responsible for.

The MapReduce system is given a pair of key,values where the keys are the document IDs and the values are the k-grams for every document. The map function then computes the hash value:

map (k: docID, v: kgram) { emit(ik = k, iv = hash(v))

}

The MapReduce system groups the hashes by document ID and then the reduce function finds the minimum hash value:

reduce (ik: docID, ivs: hashval[]) {

var minhash := INFINITY for each iv in ivs {

if iv <minhash {
minhash := iv

}}
emit(fk = ik, fv = minhash)

}

Exercise 3) Prove that if the Jaccard similarity of two columns is 0, then minhashing always gives a correct estimate of the Jaccard similarity

If the Jaccard similarity of two columns is 0, the probability of two documents getting the same minhash signature is 0. From the lecture slides (see the proof) you can verify that, the type D rows never influence the signature. And with two documents/columns having no chance of having similar signature in any permutation their Jaccard similarity using min hashes are also 0.

Exercise 4)

For any random permutation π on the binary shingle matrix Prove that $Pr[h_\pi(C_1) = h_\pi(C_2)] = sim(C_1, C_2)$

Choose a random permutation π

Claim: Pr[hπ(C1) = hπ(C2)] = sim(C1, C2)

Proof: Intuition:

Size of the universe of all possible vals of min(π ($C_1 \cup C_2$)) is $|C_1 \cup C_2|$ and in $|C_1 \cap C_2|$ of cases it can be that min(π (C_1))=min(π(C_2)) which exactly the jaccard between C1 and C2

For two columns A and B, we have h_π(A) = h_π(B) exactly when the minimum hash value of the union A ∪ B lies in the intersection A ∩ B. Thus Pr[h_π(A) = h_π(B)] = |A ∩ B| / |A ∪ B|.

Let X be a doc (set of shingles), y∈ X is a shingle Then: Pr[π(y) = min(π(X))] = 1/|X|

It is equally likely that any y∈ X is mapped to the min element Let y be s.t. π(y) = min(π(C1∪C2))
Then either: π(y) = min(π(C1)) if y ∈ C1 , or

π(y) = min(π(C2)) if y ∈ C2 So the prob. that both are true is the prob. y ∈ C1 ∩ C2

Pr[min(π(C1))=min(π(C2))]=|C1∩C2|/|C1∪C2|= sim(C1, C2)