Q1: HDFS

1. Yes.

Consider X = [block1, block2 … block6] T and G =

Then we can have P = G \* X = \* =

Therefore, we can calculate any last three parties from six cells by which means six cells and three parities from a stripe.

1. Maximum toleration of (6,3) is 3. The maximum toleration of (x,y) is depends on the value of y.

In that case, x and y should satisfied the situation that y = 3

Q2. Spark and MapReduce

1. def createCombiner(x,y):

lambda x: [x], lambda x,y:x+[y],lambda x,y:x+y

def mergeValue(x,y):

lambda: x, y: sorted(x+y)[:2]

def mergeCombiners(x,y) :

lambda x, y: (x [0], (y[0], y[1]))

1. False. Each time after update the candidates, we need to compare the total number of candidates (cand\_num) with the value of beta\_n. In the code provided in the question the student forgets this <if statement>. The following is the correct code while the yellow part needs to be added.

def c2lsh(data\_hashes, query\_hashes, alpha\_m, beta\_n):

offset = 0

cand\_num = 0

while cand\_num < beta\_n :

candidates = data\_hashes.flatMap(lambda x :

[x[0]] if collision\_count(x[1], query\_hashes, offset)>=alpha\_m else [])

cand\_num = candidates.count()

if cand\_num < beta\_n:

offset += 1

return candidates

Q3: LSH

1. For OR-AND composition, p = (1 – ( )  S

P q,o = 0.8, here R = 4, S = 5, therefore, P = (1 – ( )  S

= (1- (1- 0.8)4)5 = 0.992

1. OALSH ： P = (1 – ( )  S  with P q,o ≥ 0.5

P >= (1- (1- 0.5)2)5 = 0.237

If the recall of LSH scheme required to be higher than OALSH scheme, it needs P and P q,o are the same, therefore,

P >= 0. 237=

L >= 9

Thus, minimum l is 9.

1. Yes.

If the Jaccard similarity is fixed to be required no less than t, it means that the total number of positives is fixed. To make the recall of OALSH higher, the number of returns positive should be larger. In that case, P (OALSH) >= P (LSH).

(1 – ( )  S   >=

* (1 – ( )  S  + for P q,o >= t

Q4: Spark SQL

maxRec = record.groupBy('Id').agg({"Score":"max"})

minRec = record.groupBy('Id').agg({"Score":"min"})

maxmin = maxRec.join(minRec, on = ['Id']).orderBy('Id')

maxmin=maxmin.select(maxmin['Id'],(maxmin['max(Score)']).alias('max'),(maxmin['min(Score)']).alias('min'))

Q5:

1. Assume there are n number of labels

Type 1: K-fold cross validation

5 group \* 3 classifiers \* n labels = 15n

Type 2: train as a whole

1 \* 3 classifiers \* n labels = 3n

In total there are 18n classifiers

1. Since there are 5 groups in total and has 3 base classifiers, the number of times that an instance in the training set by all the classifiers is 3 \* 5 = 15 times.
2. There are 5 groups and meta classifiers is used for prediction. In that case, there are 1 \* 5 = 5 times in total.

Q6. Mining Data Streams

1. h1(hello) = (7 + 4 + 11 + 11 + 14) mod 8 = 7

h1(map) = (12 + 0 + 15) mod 8 = 3

h1 (reduce)= (17 + 4 + 3 + 20 + 2 + 4) mod 8 = 2

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 |

h2(hello) = 5 mod 8 = 5

h2(map) = 3 mod 8 = 3

h2(reduce) = 6 mod 8 = 6

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 |

1. h1(spark) = (18 + 15+ 0 +17 + 10) mod 8 = 4

h2(spark) = 5 mod 8 = 5

h1(spark) is not in the first hash checking while h2(spark) is in the second hash checking.

Therefore, it isn’t contained in S by bloom filter checking.

1. k = 2

m = 3

n = 8

false positive probability = (1 - e ≈ 27.8%

Q7：

1. Avg score for all movie u= (3 + 5 + 2 + 4 + 1 + 4 + 5 + 2) / 8 = 3.25

Line 1: b(xi) = u + bx + bi = 3.25 +(5-3.25) + ((3+5+2)/3 – 3.25) =5.08

Line 2: b(xi) = u + bx + bi = 3.25 + (2-3.25) + ((4+1)/2 – 3.25) = 1.25

Line 3: b(xi) = u + bx + bi = 3.25 + ((4+5)/2 – 3.25) + ((4+5+2)/3 – 3.25) = 4.92

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Users | | | | |
| Movie | 3 | 5 | 5.08 |  | 2 |
|  | 4 |  | 1 | 1.25 |
| 4 | 4.92 | 5 | 2 |  |

1. R = Q \* P^T

Line 1 = 2.3 \* 0.8 + 1.2 \* 0.6 + 1.5 \* 0.7 + 0.4 \* 0.8 = 3.93

Line 2 = 1.5 \* 0.4 + 3.2 \* 0.6 + 0.6\* 0.5 + 1.7 \* 0.7 = 4.01

Line 3 = 2.1 \* 0.7 + 1.3 \* 0.9 + 2.8 \* 0.8 + 0.4 \* 0.3 = 5

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Users | | | | |
| Movie | 3 | 5 | 3.93 |  | 2 |
|  | 4 |  | 1 | 4.01 |
| 4 | 5 | 5 | 2 |  |

1. RMSE

Baseline = = 3.57

Matrix factor = = 1.37

Thus, estimation with matrix factorization based on RMSE is better.