Z5175081
Xinyu Xu

Q1: HDFS

1. Yes.

   Consider X = [block1, block2 ⋯ block6] ^{T} and G = $\begin{bmatrix} I6 \\ g1 \\ g2 \\ g3 \end{bmatrix}$

   Then we can have P = G ⋆ X = $\begin{pmatrix} 1\ 0\ 0\ 0\ 0\ 0 \\ 0\ 1\ 0\ 0\ 0\ 0 \\ 0\ 0\ 1\ 0\ 0\ 0 \\ 0\ 0\ 0\ 0\ 1\ 0 \\ 0\ 0\ 0\ 0\ 0\ 1 \\ block\ 7 \\ block\ 8 \\ block\ 9 \end{pmatrix} * \begin{pmatrix} block\ 1 \\ block\ 2 \\ block\ 3 \\ block\ 4 \\ block\ 5 \\ block\ 6 \end{pmatrix} = \begin{pmatrix} block\ 1 \\ block\ 2 \\ block\ 3 \\ block\ 4 \\ block\ 5 \\ block\ 6 \\ block\ 7 \\ block\ 8 \\ block\ 9 \end{pmatrix}$

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

2. 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

Z5175081
Xinyu Xu

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]))

2. 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
```

Z5175081
Xinyu Xu

Q3: LSH

1. For OR-AND composition, p = $(1 - (1 - \mathrm{P\,q,o})^{R})^{S}$
   P q,o = 0.8, here R = 4, S = 5, therefore, P = $(1 - (1 - \mathrm{P\,q,o})^{R})^{S}$
   = $(1- (1- 0.8)^{4})^{5}$ = 0.992

2. OALSH： P = $(1 - (1 - \mathrm{P\,q,o})^{R})^{S}$ 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= $1 - (1 - \mathrm{P\,q,o}^{k})^{l}$

   L >= 9
   Thus, minimum l is 9.

3. 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).

   $(1 - (1 - \mathrm{P\,q,o})^{R})^{S}$ >= $1 - (1 - \mathrm{P\,q,o}^{k})^{l}$

   ⇨ $(1 - (1 - \mathrm{P\,q,o})^{R})^{S} + (1 - \mathrm{P\,q,o}^{k})^{l} \geq 1$ for P q,o >= t

Z5175081
Xinyu Xu

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

2. 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.
3. There are 5 groups and meta classifiers is used for prediction. In that case, there are 1 * 5 = 5 times in total.

Z5175081
Xinyu Xu

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

2. 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.

3. k = 2
   m = 3
   n = 8
   false positive probability = (1 - e$-\frac{km}{n}$)^{k} ≈ 27.8%

Z5175081
Xinyu Xu

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	

2. 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	

3. RMSE
   Baseline = $\sqrt{(5.08-3)^2+(1.25-4)^2+(4.92-4)^2}$ = 3.57
   Matrix factor = $\sqrt{(3.93-3)^2+(4.01-4)^2+(5-4)^2}$ = 1.37
   Thus, estimation with matrix factorization based on RMSE is better.