COMP9313 FINAL

Name: Yu ZHANG

Student ID: z5238743

Q1. HDFS

1.

$X = G'^{-1} * P$

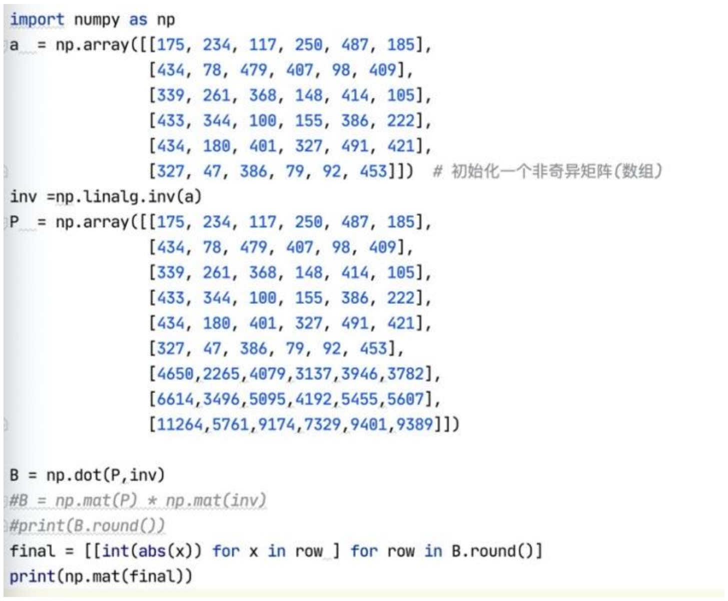
$G = P * X^{-1}$

$$X = \begin{pmatrix} 175 & 234 & 117 & 250 & 487 & 185 \\ 434 & 78 & 479 & 407 & 98 & 409 \\ 339 & 261 & 368 & 148 & 414 & 105 \\ 422 & 344 & 100 & 155 & 386 & 222 \\ 438 & 180 & 401 & 327 & 491 & 222 \\ 327 & 47 & 386 & 79 & 92 & 453 \end{pmatrix}$$

$$P = \begin{pmatrix} 175 & 234 & 117 & 250 & 487 & 185 \\ 434 & 78 & 479 & 407 & 98 & 409 \\ 339 & 261 & 368 & 148 & 414 & 105 \\ 422 & 344 & 100 & 155 & 386 & 222 \\ 438 & 180 & 401 & 327 & 491 & 421 \\ 327 & 47 & 386 & 79 & 92 & 453 \\ 4650 & 2265 & 4079 & 3137 & 3946 & 3782 \\ 6614 & 3496 & 5095 & 4192 & 5455 & 5607 \\ 11264 & 5761 & 9174 & 7322 & 9401 & 9389 \end{pmatrix}$$

$$\therefore G = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 3 & 2 & 2 & 3 & 1 \\ 3 & 4 & 1 & 5 & 2 & 3 \\ 4 & 7 & 3 & 7 & 5 & 4 \end{pmatrix}$$

The code is Figure 1:



```
import numpy as np
a  = np.array([[175, 234, 117, 250, 487, 185],
               [434, 78, 479, 407, 98, 409],
               [339, 261, 368, 148, 414, 105],
               [433, 344, 100, 155, 386, 222],
               [434, 180, 401, 327, 491, 421],
               [327, 47, 386, 79, 92, 453]])  # 初始化一个非奇异矩阵(数组)
inv =np.linalg.inv(a)
P  = np.array([[175, 234, 117, 250, 487, 185],
               [434, 78, 479, 407, 98, 409],
               [339, 261, 368, 148, 414, 105],
               [433, 344, 100, 155, 386, 222],
               [434, 180, 401, 327, 491, 421],
               [327, 47, 386, 79, 92, 453],
               [4650,2265,4079,3137,3946,3782],
               [6614,3496,5095,4192,5455,5607],
               [11264,5761,9174,7329,9401,9389]])

B = np.dot(P,inv)
#B = np.mat(P) * np.mat(inv)
#print(B.round())
final = [[int(abs(x)) for x in row ] for row in B.round()]
print(np.mat(final))
```

Figure 1

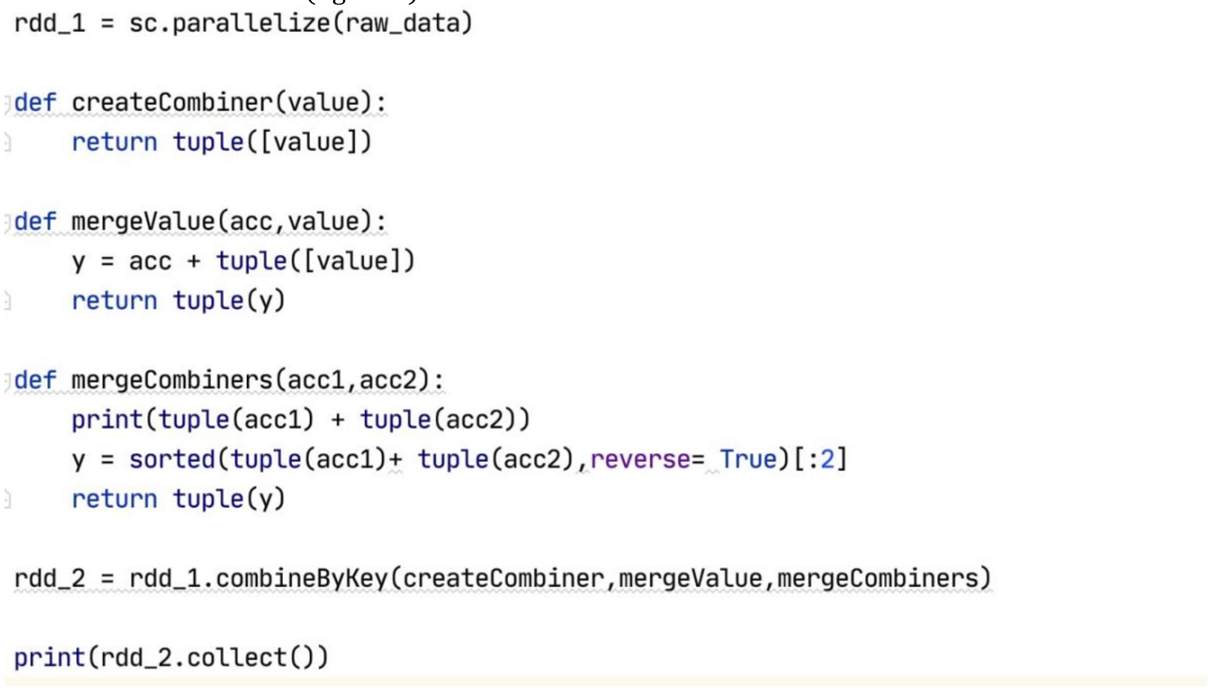
2.

The same maximum toleration as (6,3)-Reed-Solomon coding is 3, because when we lose 3 parities, we cannot recover the data from any 6 internal block.

Q2. Spark and MapReduce

1.

The code is shown below(figure 2)



```python
rdd_1 = sc.parallelize(raw_data)

def createCombiner(value):
    return tuple([value])

def mergeValue(acc,value):
    y = acc + tuple([value])
    return tuple(y)

def mergeCombiners(acc1,acc2):
    print(tuple(acc1) + tuple(acc2))
    y = sorted(tuple(acc1)+ tuple(acc2),reverse= True)[:2]
    return tuple(y)

rdd_2 = rdd_1.combineByKey(createCombiner,mergeValue,mergeCombiners)

print(rdd_2.collect())
```

Figure2

2.

This implementation is not correct, in the while loop, offset can not put in the last, it must put in the beginning.

Such like that

```
While cand_num < beta_n:
    offset += 1
    Candidates = data_hashes.flatMap(lambda x:
    [x[0] ] if collision_count(x[1], query_hashes, offset)>=alpha_m
else [])
    cand_num = candidates.count()
return candidates
```

Because of lazy evaluation, the data inside RDD is not available or transformed until an action is executed that triggers the execution.
In this codes, when in the last loop, because the action not trigger, the flatMap do not transformed. Thus, this implementation is not correct.

Q3: LSH

1.

Because Jaccard similarity between o and q is 0.8, R = 4, S =5

Thus $p_{q,o} = 0.8$

Because it is OALSH

First, using OR operation: $Pr[H_{OR}(O_1)= H_{OR}(O_2)] = (1-p_{q,o})^R$
Second, using AND operation: $Pr[H_{AND}(O_1)= H_{AND}(O_2)] = (1-(1-p_{q,o})^R)^S$

Thus, the probability of o is a nearest neighbor candidate of q is $(1-(1-p_{q,o})^R)^S$

Thus, $(1-(1-p_{q,o})^R)^S = (1-(1-0.8)^4)^5) = 0.9921$

2.

Because threshold t = 0.5
So Jaccard similarity between o and q >= 0.5, R = 2, S =5
Thus $p_{q,o} >=0.5$

From Q1, in OALSH, the probability of o is a nearest neighbor candidate of q is $(1-(1-p_{q,o})^R)^S$

Thus, $(1-(1-p_{q,o})^R)^S =$ （$1-(1-0.5)^2)^5) <= 0.2373$

Because LSH schema with k = 5, in AOLSH, the probability of o is a nearest neighbor candidate of q is $1-(1-p^k_{q,o})^l$

Thus, $1-(1-p^5_{q,o})^l = 1-(1-0.5^5)^l > 0.2373$

The minimal of l is 9

3.

Assume threshold t, Jaccard similarity between o and q equal to t, k,l,R,S for LSH and OALSH scheme,

Recall = number of returned positive/ total number of positive

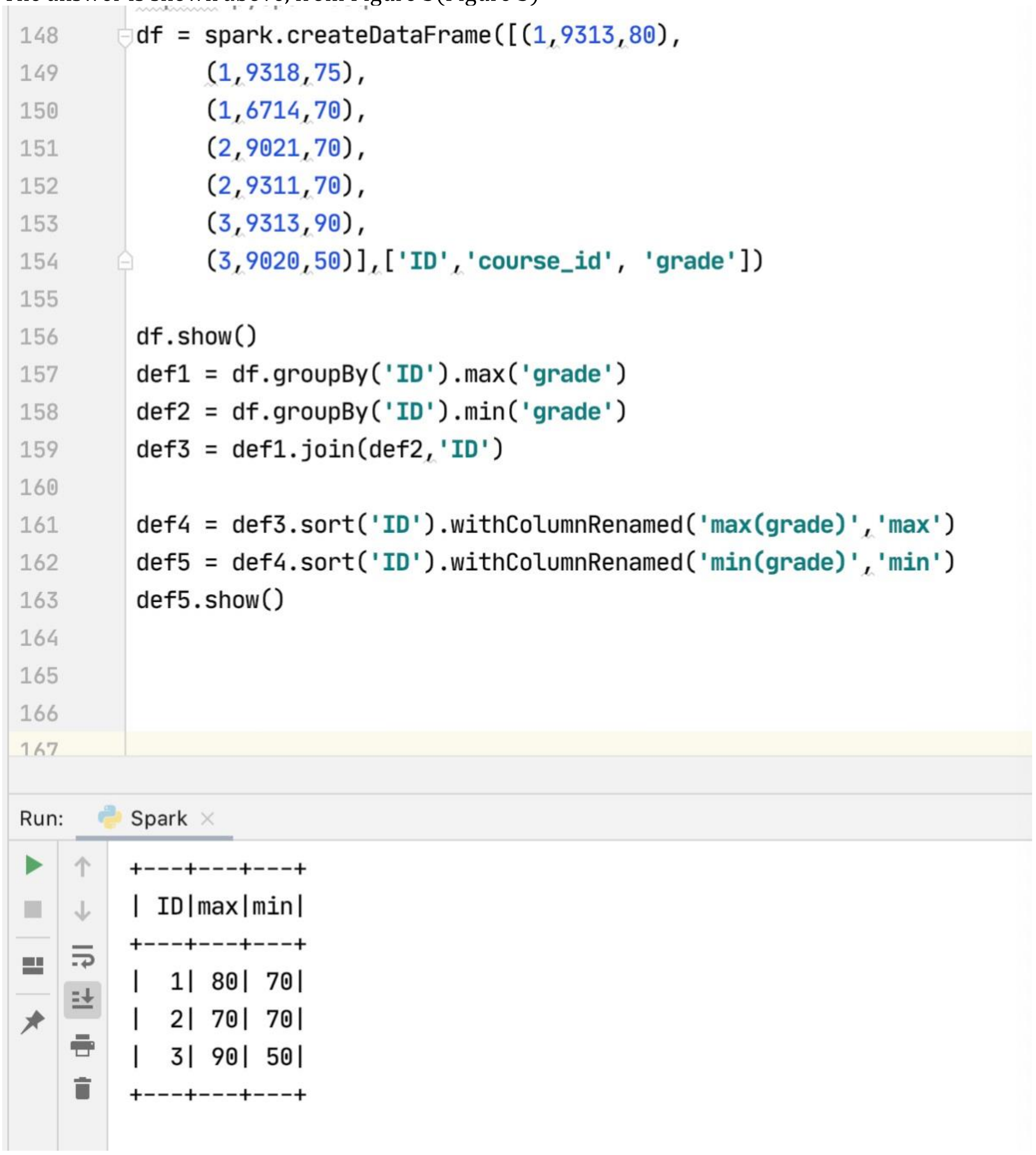
$Recall_{(OALSH)} > Recall_{(LSH)}$

According to OALSH, the probability of o is a nearest neighbor candidate of q is $(1-(1-p_{q,o})^R)^S$ and in AOLSH, the probability of o is a nearest neighbor candidate of q is $1-(1-p^k_{q,o})^l$

Thus , $(1-(1-t)^R)^S > 1-(1-t^k)^l$

Q4: Spark SQL

The answer is shown above, from Figure 3(Figure 3)



```
df = spark.createDataFrame([(1,9313,80),
     (1,9318,75),
     (1,6714,70),
     (2,9021,70),
     (2,9311,70),
     (3,9313,90),
     (3,9020,50)],['ID','course_id', 'grade'])

df.show()
def1 = df.groupBy('ID').max('grade')
def2 = df.groupBy('ID').min('grade')
def3 = def1.join(def2,'ID')

def4 = def3.sort('ID').withColumnRenamed('max(grade)','max')
def5 = def4.sort('ID').withColumnRenamed('min(grade)','min')
def5.show()
```

```
+---+---+---+
| ID|max|min|
+---+---+---+
|  1| 80| 70|
|  2| 70| 70|
|  3| 90| 50|
+---+---+---+
```

Figure 3

Q5: Stacking

1.

Type A： 3 labels
5 group
1 metaclassifier
Type B: In the end, 3 base classifiers train the whole data respectively

Thus 3*5+3+1= 19

2.

Because we have 3 base classifiers and the instance are trained in the 4 loops,So we have 3*4 12 times, then In the end, 3 base classifiers train the whole data, so we have 3 times, finally， the metaclassifer train the data one times.
Thus, 3*4 +3 +1 =16 times

3.

3 base classifiers train the training set 3 times, metaclassifier do not predict the training the data 1 time
Thus, 3 times

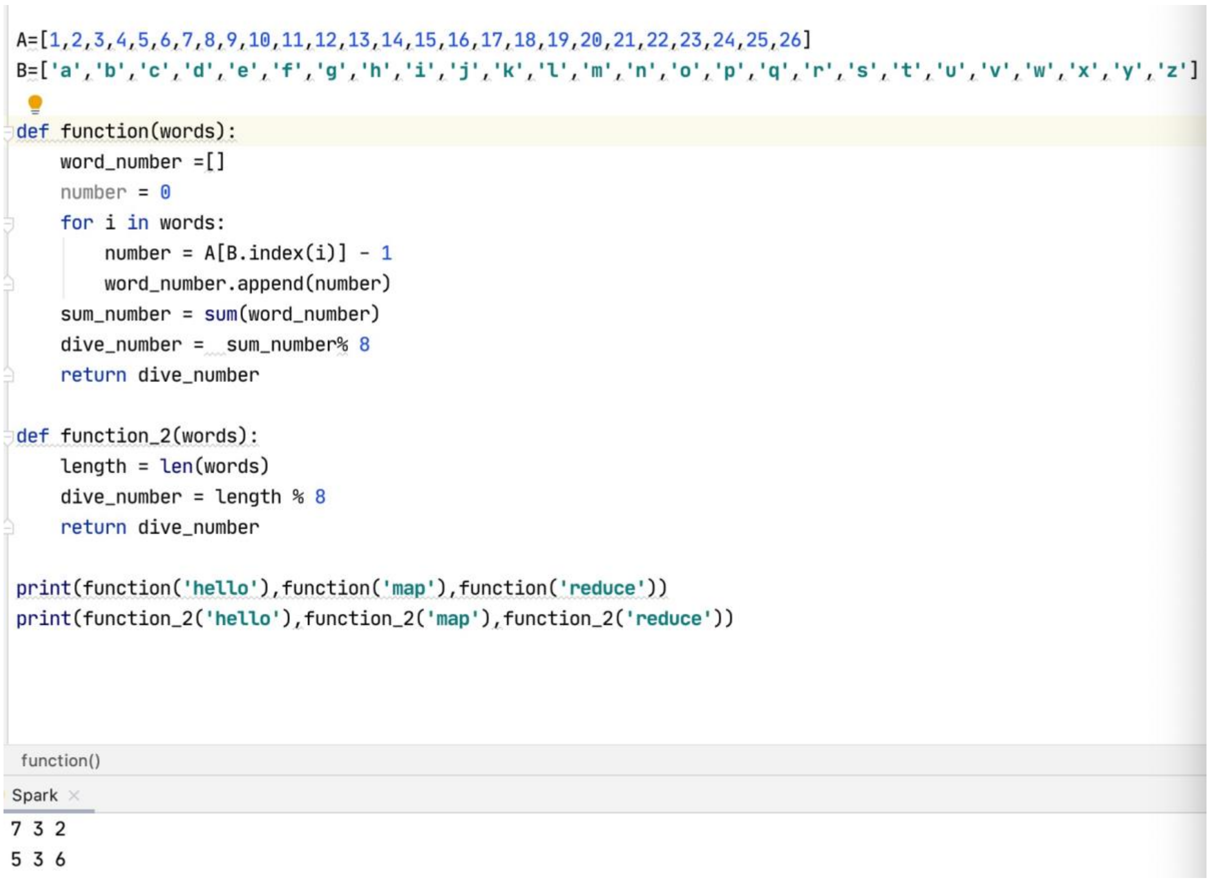
Q6: Mining Data Streams

1.

Because $h_1(str) = P_{c \in str}(c - 'a')$ mod 8

word	Hello	map	reduce
$h_1(str)$	7	3	2
$h_2(str)$	5	3	6

The processes is shown in code(Figure 4):



```
A=[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26]
B=['a','b','c','d','e','f','g','h','i','j','k','l','m','n','o','p','q','r','s','t','u','v','w','x','y','z']

def function(words):
    word_number =[]
    number = 0
    for i in words:
        number = A[B.index(i)] - 1
        word_number.append(number)
    sum_number = sum(word_number)
    dive_number =  sum_number% 8
    return dive_number

def function_2(words):
    length = len(words)
    dive_number = length % 8
    return dive_number

print(function('hello'),function('map'),function('reduce'))
print(function_2('hello'),function_2('map'),function_2('reduce'))
```

```
7 3 2
5 3 6
```

Figure 4

$Sh_1(str)$ and $Sh_2(str)$

	0	1	2	3	4	5	6	7
$h_1(str)$	0	0	1	1	0	0	0	1
$h_2(str)$	0	0	0	1	0	1	1	0

S:

	0	1	2	3	4	5	6	7
$h_1(str)$	0	0	1	1	0	1	1	1

2.

Spark is not contained in S.
Because， according to the code in Q4.1 $h_1(spark) = 4$, $h_5(spark) = 5$.
Although $h_2(spark)$ is in table $h_2(str)$, $h_1(spark)$ is not in $h_1(str)$
They must both in S $h_1(str)$ and S $h_2(str)$.
Thus Spark is not contained in S.

3.
Because we have 2 independent hash function and the size of Bloom filter of size is 8,Besides,the number of elements in S is 5
Thus $k = 2$, $m=8$, $n= 5$

According to False positive probability = $(1-e^{-km/n})^k$
$P = (1-e^{-2*8/5})^2 = 0.92013$

Q7: Recommender System

1.

	users				
movies	3	5	?	0	2
	0	4	0	1	?
	4	?	5	2	0

According to baseline estimator = u + bx + bi

for movie 1: u = 3.25
bx = 5 - 3.25 = 1.75
bi = 3/10 – 3.25 =0.08333
Thus, baseline estimator = 3.25 +1.75 +0.08333=5.0833

for movie 2: u = 3.25
bx = 2 – 3.25 = -1.25
bi = 2.5– 3.25 = -0.75
Thus, baseline estimator = 3.25 -1.25 -0.75 = 1.25

for movie 3: u = 3.25
bx = 4.5 – 3.25 = 1.25
bi = 3.66667– 3.25 = 0.4166667
Thus, baseline estimator = 3.25 +1.25 + 0.75 = 4.9166667

2.

$R_{xi} = q_i * P^T$

$$P^T = \begin{pmatrix} 0.7 & 0.7 & 0.8 & 0.1 & 0.4 \\ 0.7 & 0.9 & 0.6 & 0.1 & 0.6 \\ 0.7 & 0.8 & 0.7 & 0.6 & 0.5 \\ 0.5 & 0.3 & 0.8 & 0.4 & 0.7 \end{pmatrix}$$

for movie 1: $R_{xi} = \begin{pmatrix} 2.3 & 1.2 & 1.5 & 0.4 \end{pmatrix} \begin{pmatrix} 0.8 \\ 0.6 \\ 0.7 \\ 0.8 \end{pmatrix} = 3.93$

for movie 2: $R_{xi} = \begin{pmatrix} 1.5 & 3.2 & 0.6 & 1.7 \end{pmatrix} \begin{pmatrix} 0.4 \\ 0.6 \\ 0.5 \\ 0.7 \end{pmatrix} = 4.01$

for movie 3: $R_{xi} = \begin{pmatrix} 2.1 & 1.3 & 2.8 & 0.4 \end{pmatrix} \begin{pmatrix} 0.7 \\ 0.9 \\ 0.8 \\ 0.3 \end{pmatrix} = 5$

3.

For question 1'result:

$RMSE = (1/|R|)*(\sum_{(i,x)\in R}(R_{xi}\hat{}- R_{xi}))^{(1/2)}$
$= (1/3)*((5.0833-3)^2 + (1.25-4)^2 +(4.9166667-4))^{(1/2)}$
$= 1.1896$

For question 2'result:

$RMSE = (1/|R|)*(\sum_{(i,x)\in R}(R_{xi}\hat{}- R_{xi}))^{(1/2)}$
$= (1/3)*((3.93-3)^2 + (4.01-4)^2 +(5-4))^{(1/2)}$
$= 0.4552$

Because question 1’result > question 2’result:
Thus, using baseline estimator is better than using matrix factorization