Assignment2 Report

R0685971 Enyan Dai

1. Problem Analysis

Assignment2 contains two major tasks. First, parameter analysis based on a subset of Twitter dataset. Second, find a good parameter configuration for obtaining the pairs of tweets in the full dataset that have a Jaccard Similarity > 0.9 when using 3-shingles at the character level.

2. Parameter analysis

In the LSH algorithm, there are three parameters we can adjust:

1) The number of rows in the signature matrix (rows*bands)
2) The number of bands (bands)
3) The number of rows in each band (rows)

What's more, in the universal hashing:

$$h_{a,b}(x) = ((a \bullet x + b) \bmod p) \bmod N$$

a,b are random integers, p is a prime number (p > N),N is number of hash values(nShingles in this code)

In this section, I made a experiment to check how these four parameters affect the run time, precision and recall of candidates proposed by LSH.

$$Precision = \frac{TP}{TP + FP}$$

Precision is the percent of true positives in the candidates that LSH algorithm proposed.

$$Recall = \frac{TP}{TP + FN}$$

Recall is in the pairs similarity larger than a threshold , what fraction the LSH found.

The ground truth is proposed by the BruteSearh. Experiment is based on first 5000 lines of tweets. And the domain of each parameters is list below:

nShingles	{1000，10000，100000}
Bands	{2，3}
Rows	{4~15}
Rows*bands	{8~42}

Table 1 domain of each parameters

nShingles	Band	Row	recall	precision	Time(ms)
10000	2	4	0.9573529	0.05915947	145
10000	2	5	0.9397059	0.09824529	156

Table 2 two samples of dataset

2.1. nShingles

The nShingles may affect the number of false positives. Because when we use hash function, the collision give us false positives. And in universal hash, if nShingles become larger, false positive will decrease and the precision will increase. As for the recall and run time, nShingles won't affect at all.

2.1.1. precision

hypothesis test 1:

$$H_0\text{:}\ \ \text{precision}_{\text{n}S\text{hingles}=1000} = \text{precision}_{\text{n}S\text{hingles}=10000}$$
$$H_1\text{:}\ \ \text{precision}_{\text{n}S\text{hingles}=1000} < \text{precision}_{\text{n}S\text{hingles}=10000}$$

Result of **paired t test**: p-value = 4.25e-07, mean of the differences = -0.025. Because p-value is less that 0.05,we can reject H0. So we can make a conclusion that when nShingles is 10000 the precision is larger than when nShingles is 1000.
hypothesis test 2:

$$H_0\text{:}\ \ \text{precision}_{\text{n}S\text{hingles}=10000} = \text{precision}_{\text{n}S\text{hingles}=100000}$$
$$H_1\text{:}\ \ \text{precision}_{\text{n}S\text{hingles}=10000} < \text{precision}_{\text{n}S\text{hingles}=100000}$$

Result of **paired t test**: p-value = 0.9928, mean of the differences = -0.007. Because p-value is larger that 0.05,we can not reject H0. So we can make a conclusion that when nShingles is 10000 the precision is the same when nShingles is 1000.
This phenomenon shows that when we increase nShingles to a certain value, the false positives are very rare, it is useless to keep increasing nShingles. So we can choose nShingles as 10000 in the following analysis.

2.1.2. recall and run time

The nShingles doesn't affect recall too much. Mean of the differences are 0.006 and 0.007 for recall. As for the time mean of differences are -4.3 and 0.5. Which are all really small compared to the total value.

2.2. Rows, bands and rows*bands

In 2.1, we have already decided that nShingles equals to 10000 is the best parameter. The dataset we used in this section are the subset which nShingles are 10000.

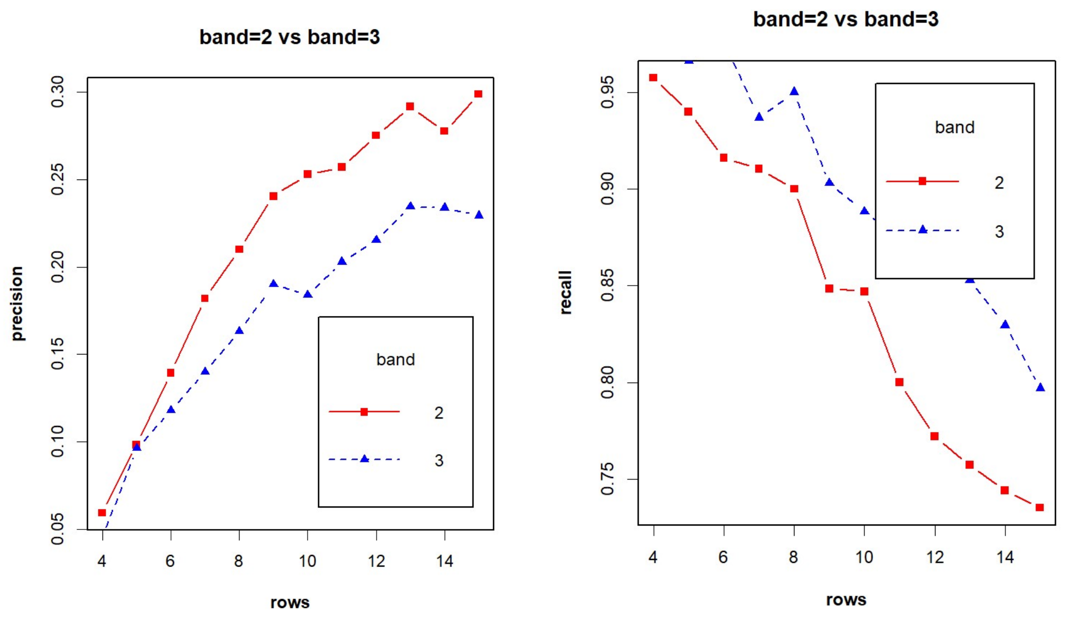


figure 1 how the bands and rows affect the precision and recall .

Now we can build a linear regression model to analyse the effect of rows, bands and rows*bands to precision and recall.

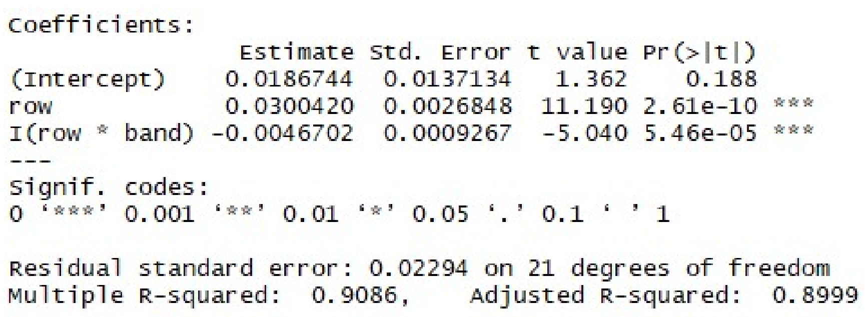
2.2.1. Precision

1) Use this formula: $precision = k_1 * rows + k_2 * bands + k_3 * (rows * bands) + b$

After we get the result, we can find the p-value for k2 = 0 is 0.96. So we can eliminate bands and make a new linear model.

2) New formula: $precision = k_1 * rows + k_3 * (rows * bands) + b$

Result of R studio:



```
Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)    0.0186744  0.0137134   1.362    0.188
row            0.0300420  0.0026848  11.190 2.61e-10 ***
I(row * band) -0.0046702  0.0009267  -5.040 5.46e-05 ***
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.02294 on 21 degrees of freedom
Multiple R-squared:  0.9086,    Adjusted R-squared:  0.8999
```

The Pr are all very significant. We don't need to eliminate the variable this time. So the linear model is: $precision = 0.0300420 * rows + -0.0046702 * (rows * bands) + 0.0186744$. Now we can make a conclusion that rows and rows*bands play very important role. If rows are greater precision will be greater. If rows*bands are less the precision will be greater.

2.2.2. Recall

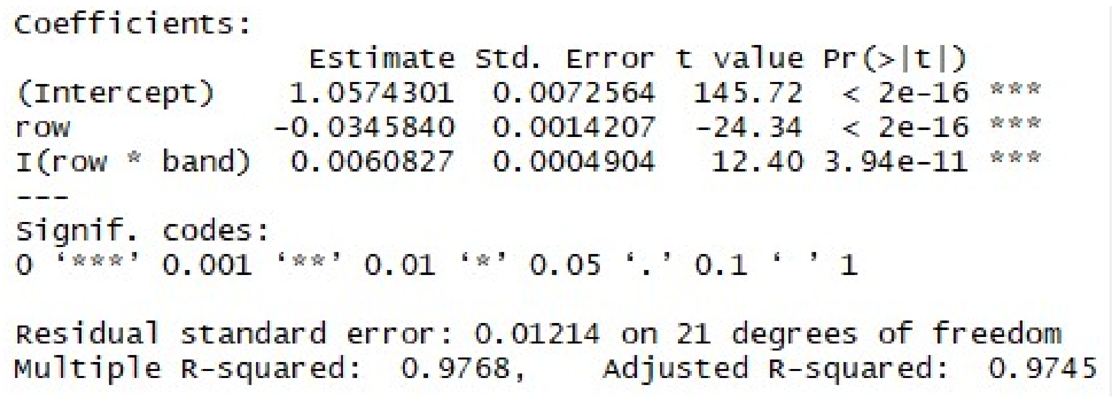
The same linear regression analysis step as precision. We begin with the formula:

$$recall = k_1 * rows + k_2 * bands + k_3 * (rows * bands) + b$$

After we eliminate the redundant variable, we end up with this formula:

$$recall = k_1 * rows + k_3 * (rows * bands) + b$$

Result of R studio:



```
Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)      1.0574301  0.0072564  145.72  < 2e-16 ***
row             -0.0345840  0.0014207  -24.34  < 2e-16 ***
I(row * band)    0.0060827  0.0004904   12.40 3.94e-11 ***
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.01214 on 21 degrees of freedom
Multiple R-squared:  0.9768,    Adjusted R-squared:  0.9745
```

Final linear model: $recall = -0.0345840 * rows + 0.0060827 * (rows * bands) + 1.0574301$

So if increase rows, the recall will decrease. If rows*bands are greater, the recall will be greater.

2.2.3. Run time

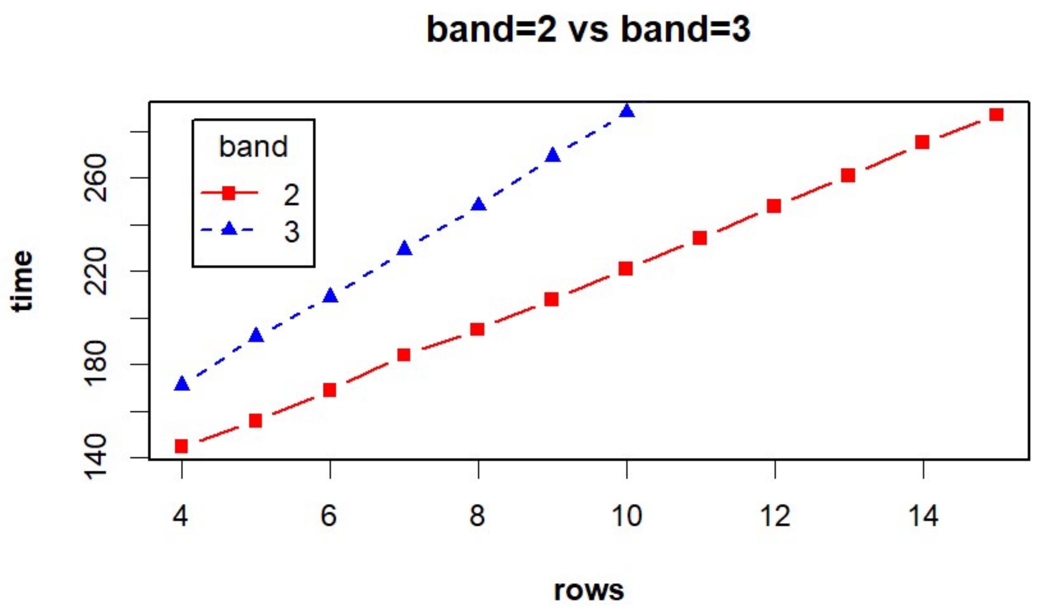


figure 2

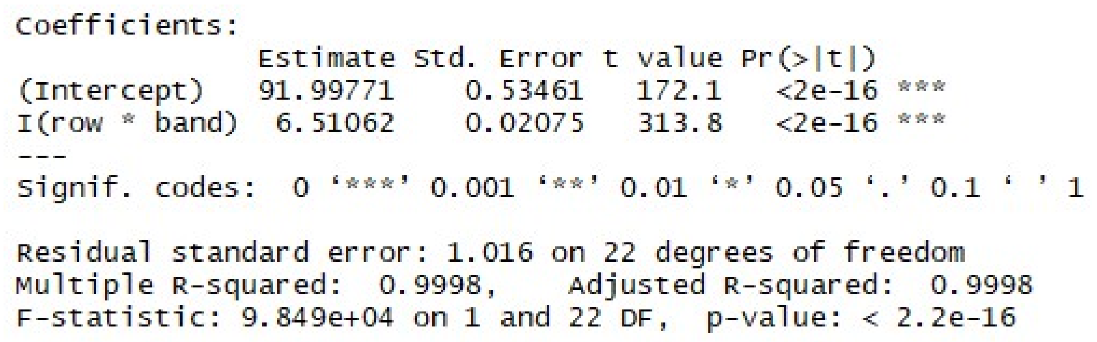
The same linear regression analysis step as precision. We begin with the formula:

$$\text{time} = k_1 * rows + k_2 * bands + k_3 * (rows * bands) + b$$

After we eliminate the redundant variable, we end up with this formula:

$$time = k_3 * (rows * bands) + b$$

Result of R studio:



```
Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)    91.99771    0.53461   172.1   <2e-16 ***
I(row * band)   6.51062    0.02075   313.8   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.016 on 22 degrees of freedom
Multiple R-squared:  0.9998,	Adjusted R-squared:  0.9998
F-statistic: 9.849e+04 on 1 and 22 DF,  p-value: < 2.2e-16
```

Final linear model: $recall = 6.51062 * (rows * bands) + 91.99771$

The rows*bands play a major role to the time.

3. Running on full dataset

In this part, just like the assignment mentioned 2GB memory is an very import limitation when we developing the solution. This limitation give us a domain of parameters we can adjust.

We need to store every signature array to get a similarity when we find a new candidate pair. And the amount of tweets in total is 7416113 which is about 2^13. And an int variable is 8 byte. The memory we can use is 2GB~2^21 byte. So the number of number of rows in the signature matrix should be less or equal 2^5.

And for this formula: $0.9 = \left(\frac{1}{b}\right)^{\frac{1}{r}}$,when b is 2, rows =6.5, and when b is 3, rows=10.4. The b can't be any larger because of memory limitation(b*r<=32). For this task, we want a recall as high as 0.9. And Also we want the precision as high as possible. So we choose bands=2, rows=8. nShingles = 10000.

3.1. Use hash function to record candidate

The possible pairs is k*2^13*2^13 which will cost very large memory if we restore the candidate pair. In this way, we can use a hash function and a array to record whether the pair has been recorded. In the code the size of array is 2^18, so the possibility of collision will be very small.

4. Perspective improvement

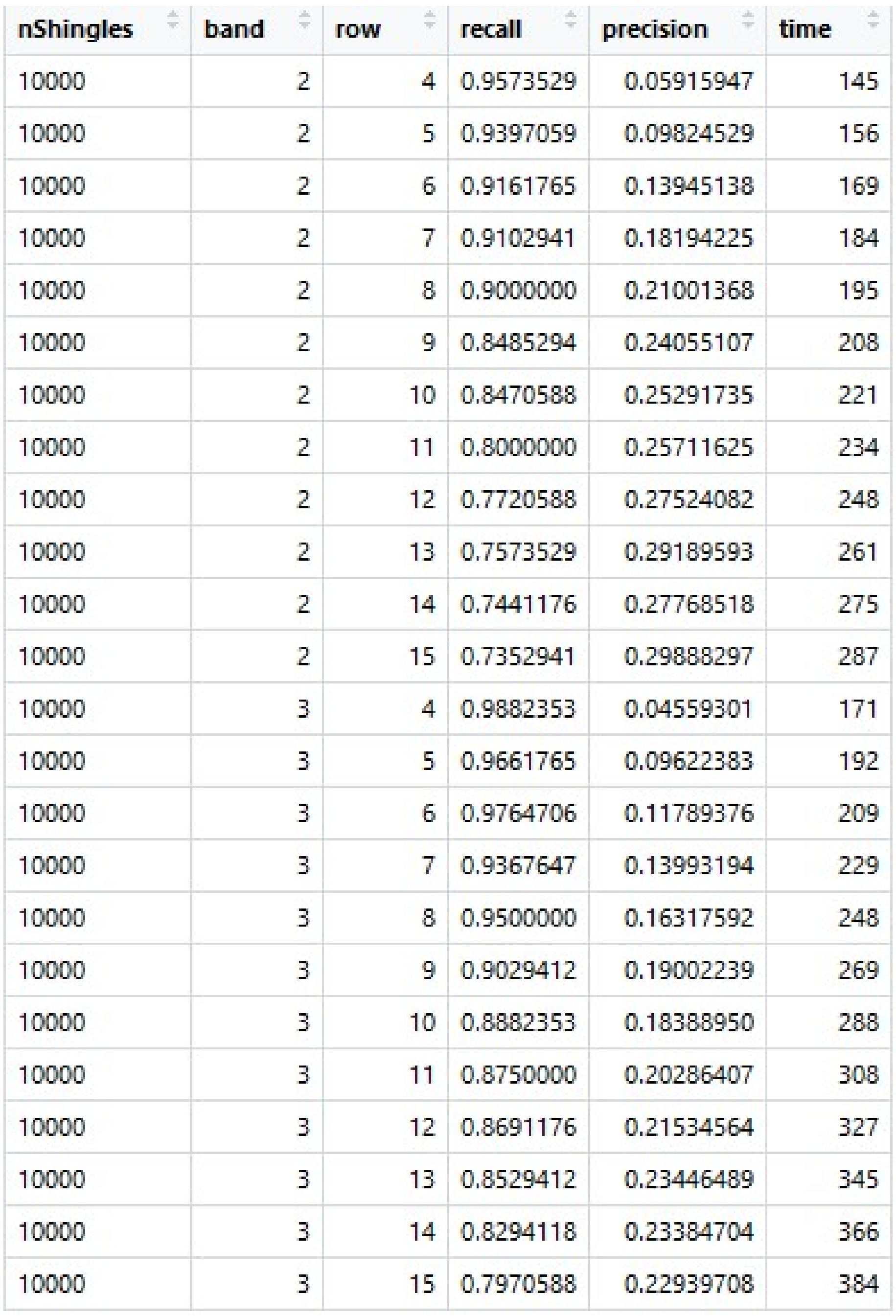
Because of the limitation of memory, the precision is low in order to get a high enough recall.

1. One possible improvement is increase the row and bands, if we have more memory.
2. Another one is that when we get the candidate pair and the signature similarity of the pair, we can use the threshold to filter the false positive. For instance, when we run the full dataset to get a pair that similarity is larger than 0.9, we can throw all the pair that the signature similarity is less than 0.9. However this may bring another problem, the signature similarity is not the real similarity of the pair similarity. If we throw all the pairs less that 0.9, we will also lose some pairs which similarity is larger that 0.9, but signature similarity is less than 0.9.

In other words, although setting a threshold to filter the signature matrix similarity can increase precision,increasing from about 0.2 to 0.4 , the recall will decrease from 0.90 to 0.80. It is a trade off here. And the the trade off is worth here. What's we may use a threshold less than 0.9 to make sure the recall is still very high and rule out the pair that is very unlikely. For instance we can set the threshold as 0.5 instead of 0.9. This method is implemented in my code.

Appendix

1. Importtant subset of DataSet



nShingles	band	row	recall	precision	time
10000	2	4	0.9573529	0.05915947	145
10000	2	5	0.9397059	0.09824529	156
10000	2	6	0.9161765	0.13945138	169
10000	2	7	0.9102941	0.18194225	184
10000	2	8	0.9000000	0.21001368	195
10000	2	9	0.8485294	0.24055107	208
10000	2	10	0.8470588	0.25291735	221
10000	2	11	0.8000000	0.25711625	234
10000	2	12	0.7720588	0.27524082	248
10000	2	13	0.7573529	0.29189593	261
10000	2	14	0.7441176	0.27768518	275
10000	2	15	0.7352941	0.29888297	287
10000	3	4	0.9882353	0.04559301	171
10000	3	5	0.9661765	0.09622383	192
10000	3	6	0.9764706	0.11789376	209
10000	3	7	0.9367647	0.13993194	229
10000	3	8	0.9500000	0.16317592	248
10000	3	9	0.9029412	0.19002239	269
10000	3	10	0.8882353	0.18388950	288
10000	3	11	0.8750000	0.20286407	308
10000	3	12	0.8691176	0.21534564	327
10000	3	13	0.8529412	0.23446489	345
10000	3	14	0.8294118	0.23384704	366
10000	3	15	0.7970588	0.22939708	384