Big Data Analytics
Assignment-3

Submitted By:
Nitindeep Singh(MT19069)
Mohd Zartab Ali(MT19041)

Pre-Processing Steps:

- Only the first 1 lakh rows have been taken into consideration for this assignment**.**

Assumptions:

- Questions are converted into tf-idf vectors using Spark API.
- No other NLP based preprocessing is done.
- Random Vector approach is used for LSH. (Text similarity is considered w.r.t Dot Product).

Methodology:

- The queries and their id's are extracted from the Data Frame and mapped.
- Then vectors are created for the queries (Initially the size of 30000) using the spark tf-idf vectorizer in this process.(Size of vector has been varied)
- These vectors now represent the queries.
- Now random vectors are created of size (equal to the size of vectors of the queries) for the purpose of generating min-hash signatures.
- Generation of min-hash signatures:
    1) Each of the question/query vectors is taken.
    2) Dot product of each question vector with the random vectors is done.
    3) If the dot product is greater than 0, value 1 is assigned otherwise value 0 in the final min-hash being created.
    4) Finally for all the question vectors, min-hash signatures are generated.

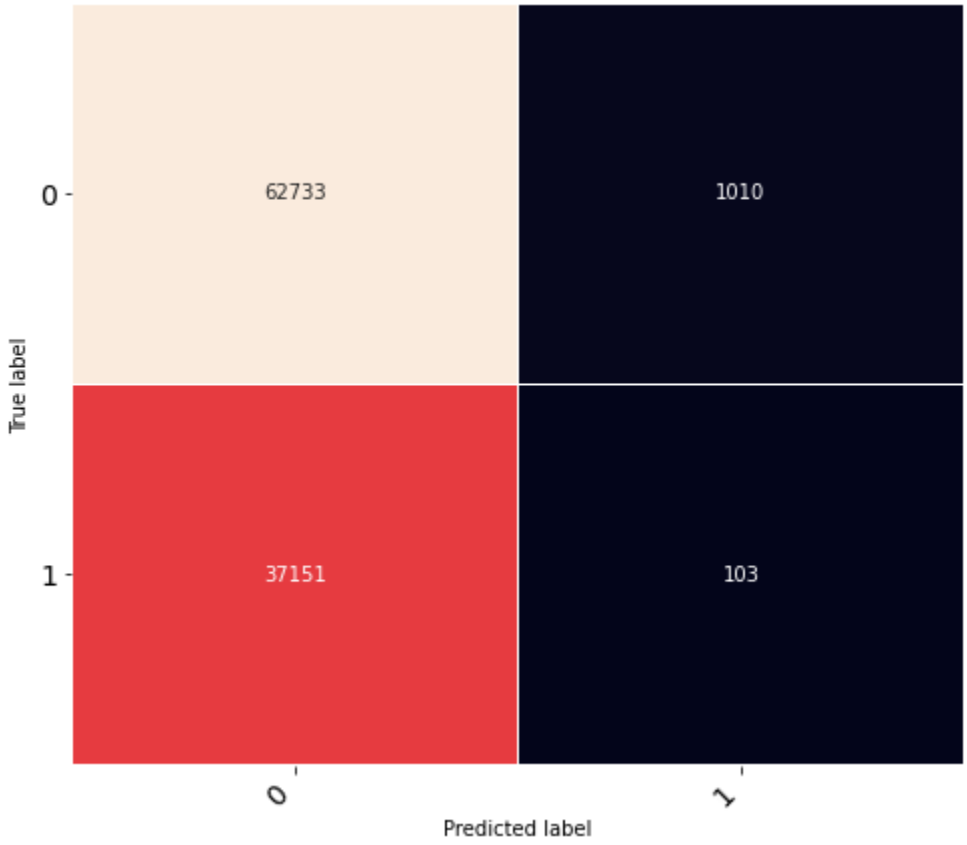
- In another variation, the minhash signatures are converted into a decimal number and then used a key to be hashed to the hash table.
- We also tried md5 hash on the minhash values generated.
- The choice of number of random vectors (i.e 30, 50, 55, 75) affects the precision and recall. It also affects the time complexity.
- The queries (transformed) are mapped to the hash table formed.
- Now candidate pairs that are similar will collide into the same hash table bucket.
- We finally count the candidate pairs and evaluate(precision and recall) on the basis of that.

Learning:

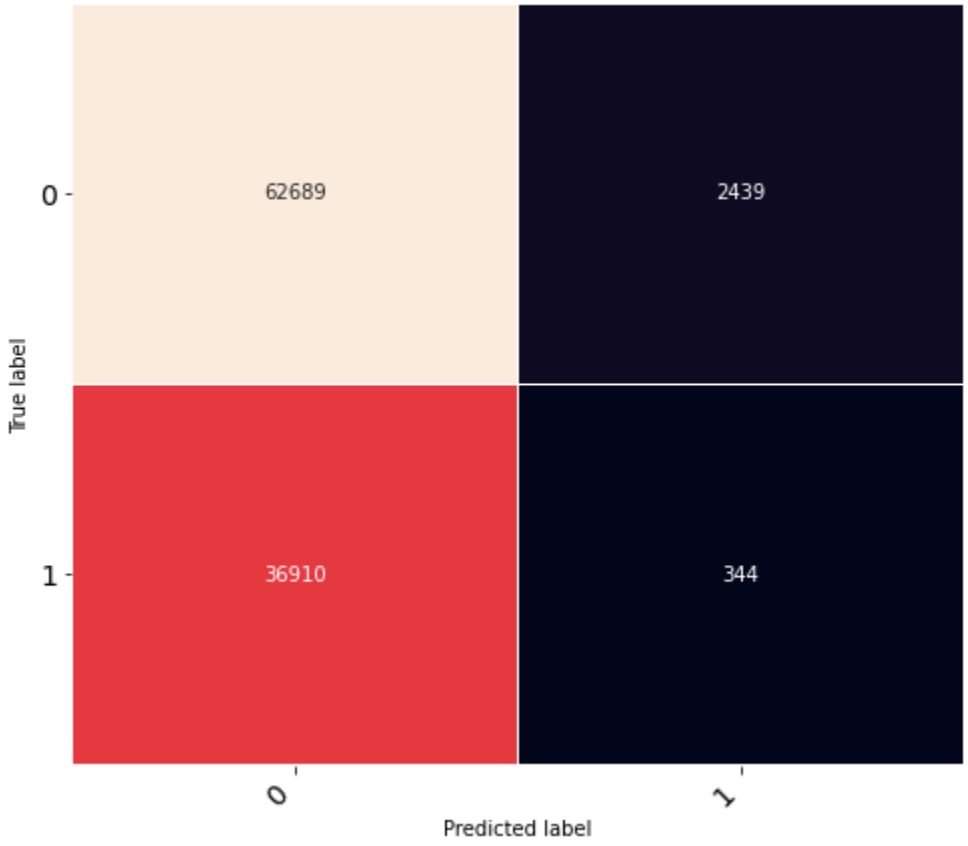
- The idea of minhash and concept of LSH on a practical level.
- The idea about the complexity with respect to the size of minhashes.
- A significant insight gained about how to derive better precision and recall using multiple Hash Tables.
- Learned about how to handle big data.
- Use of Spark API to generate vectors.

Result:

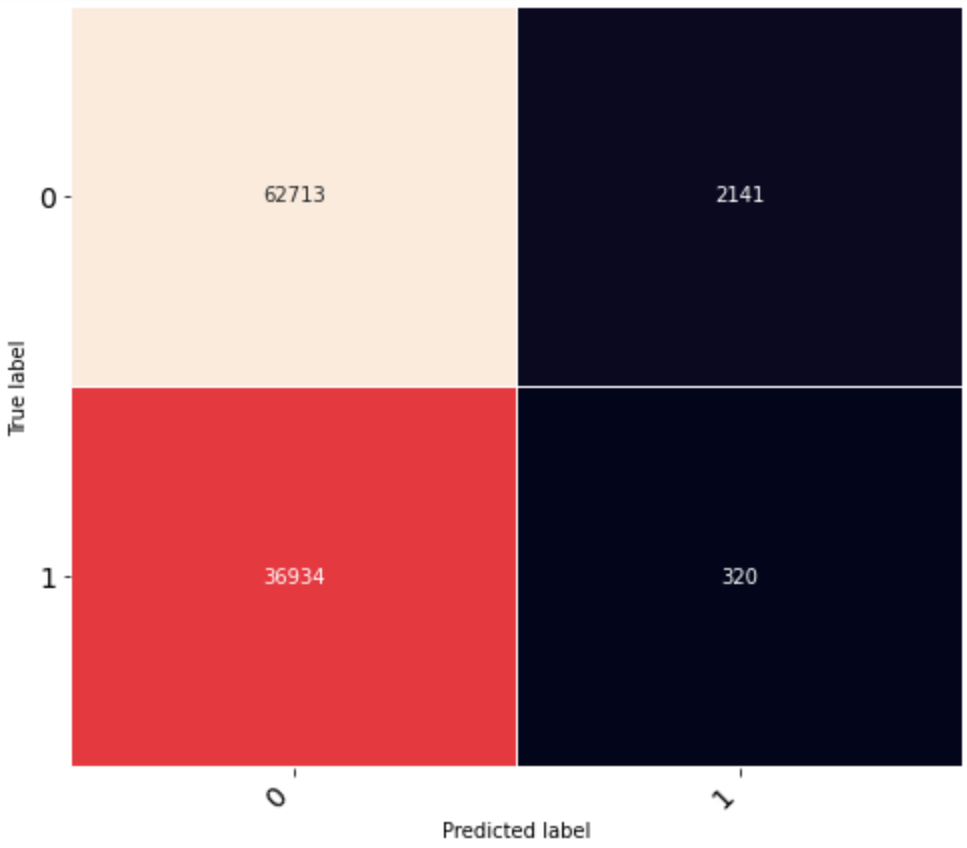
At number of random vectors= 75, Dimension of Question vector= 1*30000
Number of HashTable=1
Precision: 0.12424607961399277
Recall : 0.002764803779459924



At number of random vectors= 35, Dimension of Question vector= 1*20000
Number of HashTable=1
Precision: 0.16586306653809066
Recall : 0.009233907768293338



At number of random vectors= 55, Dimension of Question vector= 1*20000
Number of HashTable=3 (OR of HashTables)
Precision: 0.17438692098092642
Recall : 0.008589681644924034


62713
2141
36934
320
0
1
True label
0
1
Predicted label