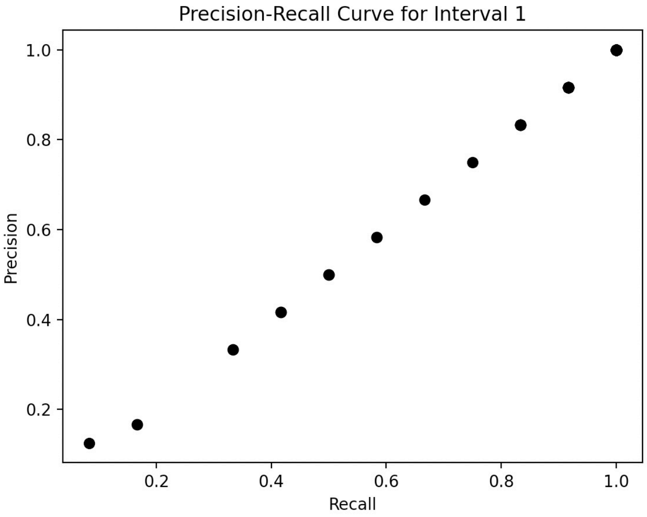
Monday, February 21, 2022

Report for Assignment One

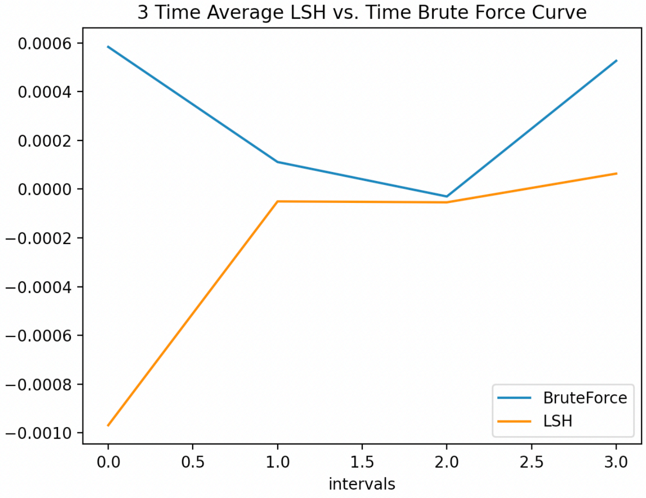
CSC 200H

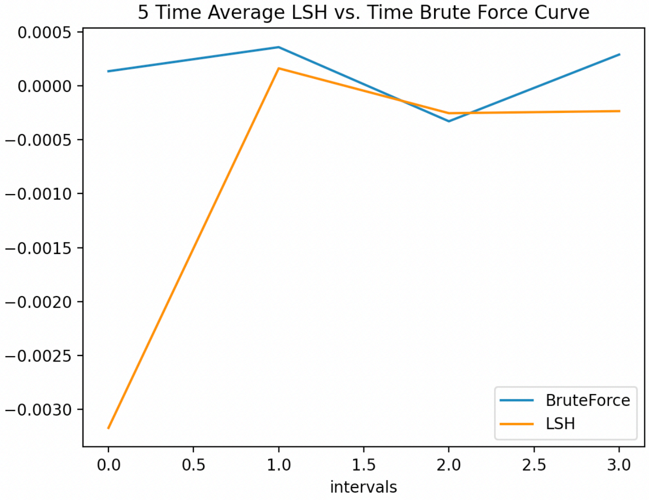
- Done by:
  - Bohan Cui , bcui2@u.rochester.edu
  - Chem Chikwez, cchikwez@u.rochester.edu
- To run :
  - Simply go to experiments.py and click run
- Top-k and hash function explained:
  - n=128 hash functions, for parameters b and r: we set b=8 and r=16, k=12.
  - We converted the documents as a sparse matrix in kshingle.py using the shingles in our data set. We then generated and stored on file the signature matrix using minhash.py Within lskbucket.py, we used the signature matrix file (sig.pkl), bands and hash function to hash documents to buckets. This helped build our index.
  - For each query, we got it's signature value from the signature matrix, hashed it to the band, collected all the documents it hashed to, computed their Jaccard values and retrieved the top-k.
- Recall vs. Precision:
  - We run our experiment multiple times, and the recall and precision always returns the same answer for same set, thus, our graph looks like a linear line. We run it for all 4 intervals and we have 25 return values, but only 11 of them are distinct, thus our graphs shows 11 plots.
  - Precision: [1.0, 0.9166666666666666, 0.4166666666666667, 0.3333333333333333, 0.9166666666666666, 1.0, 0.8333333333333334, 0.8333333333333334, 0.75, 1.0, 0.9166666666666666, 1.0, 0.5, 1.0, 0.8333333333333334, 1.0, 0.9166666666666666, 0.125, 1.0, 1.0, 0.6666666666666666, 0.9166666666666666, 0.3333333333333333, 1.0, 1.0]
  - Recall: [1.0, 0.9166666666666666, 0.4166666666666667, 0.3333333333333333, 0.9166666666666666, 1.0, 0.8333333333333334, 0.8333333333333334, 0.75, 1.0,

0.9166666666666666, 1.0, 0.5, 1.0, 0.8333333333333334, 1.0, 0.9166666666666666, 0.08333333333333333, 1.0, 1.0, 0.6666666666666666, 0.9166666666666666, 0.3333333333333333, 1.0, 1.0]

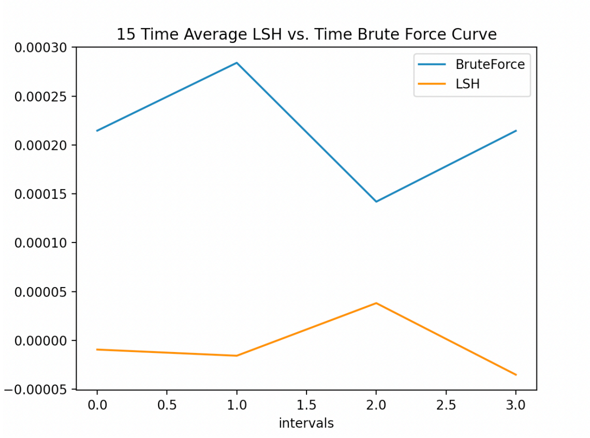


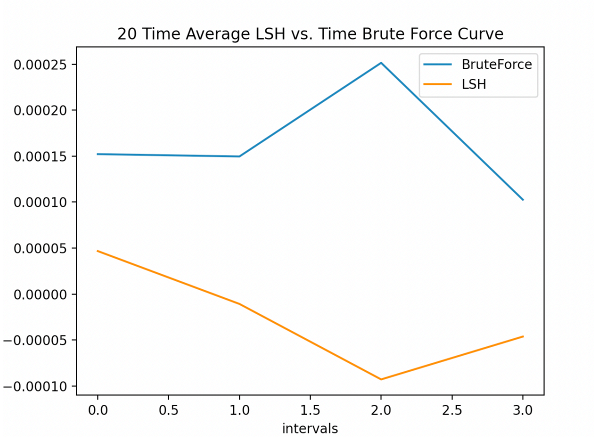
- Average running time for Brute Force method and MinHash LSH:
  - Average for running three times:
    - Brute force: [0.0005834026666661218, 0.00011113900000014887, -3.0277666667781016e-05, 0.0005262499999997653]
    - LSH: [-0.0009688466666666856, -5.045833333265174e-05, -5.422266666584482e-05, 6.355499999965986e-05]
  - Average for running five times:
    - Brute Force: [0.00013485000000019732, 0.0003582079999974895, -0.0003293000000002123, 0.0002893163999996062]
    - LSH: [-0.00317217519999705, 0.0001618245999987132, -0.0002542000000010702, -0.00023500840000068024]



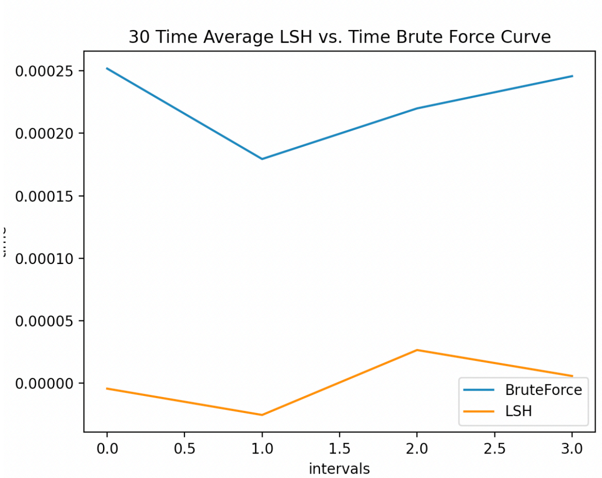


- Average for running 15 times:
  - Brute Force: [0.00021459166666769984, 0.00028406653333327497, 0.00014189419999998739, 0.00021439719999993018]
  - LSH: [-9.469333333737378e-06, -1.578313333370218e-05, 3.806373333035869e-05, -3.516920000106912e-05]
- Average for running twenty times:
  - Brute Force: [0.00015220609999971323, 0.00014969584999846575, 0.0002515626000004767, 0.00010265600000192165]
  - LSH: [4.673765000000607e-05, -1.0787549998325651e-05, -9.29021499999294e-05, -4.629389999832867e-05]





- Average for running 30 times:
  - Brute Force: [0.00025188466666698512, 0.00017942803333323443, 0.00021998473333250483, 0.00024579590000003054]
  - LSH: [-4.444400000795194e-06, -2.5523566666378154e-05, 2.647206666582432e-05, 5.687499997453216e-06]


30 Time Average LSH vs. Time Brute Force Curve
intervals
BruteForce
LSH