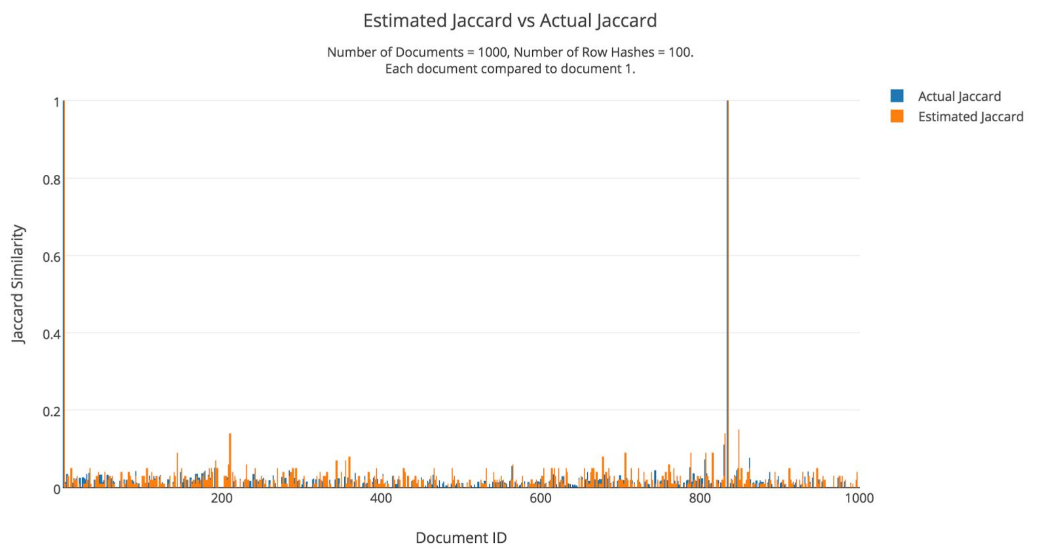
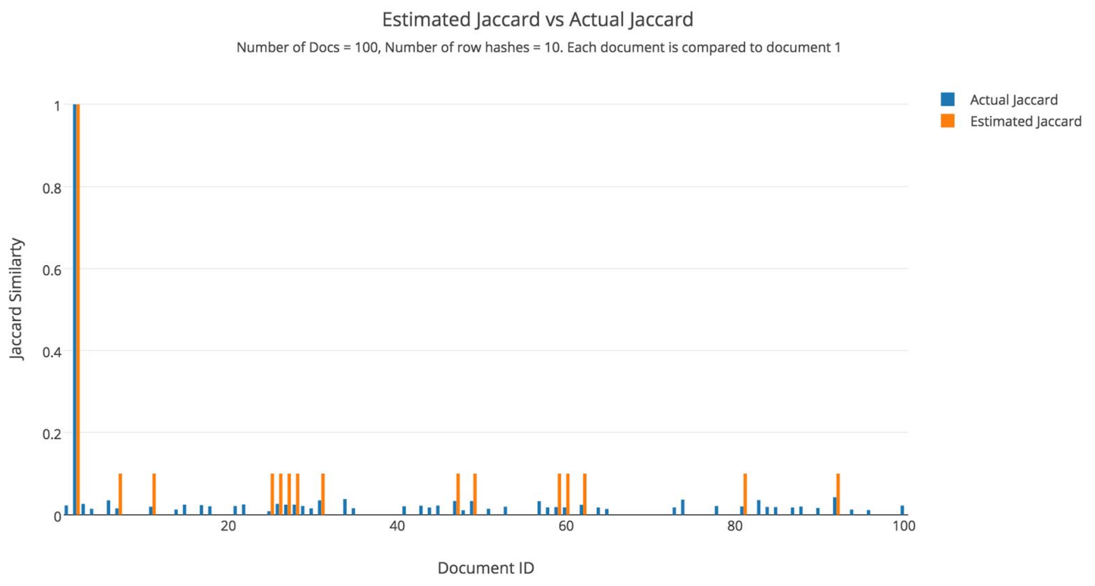
Sabastian Mugazambi & Simon Orlovsky
HMW 2 : Data Mining Spring 2017

Estimated Jaccard vs Jaccard Performance

Our estimated Jaccard and real Jaccard values had roughly similar results. Chart below which is a combined plot of actual jaccard similarities and estimated jaccard similarities shows very similar levels of jaccard similarities between the two.

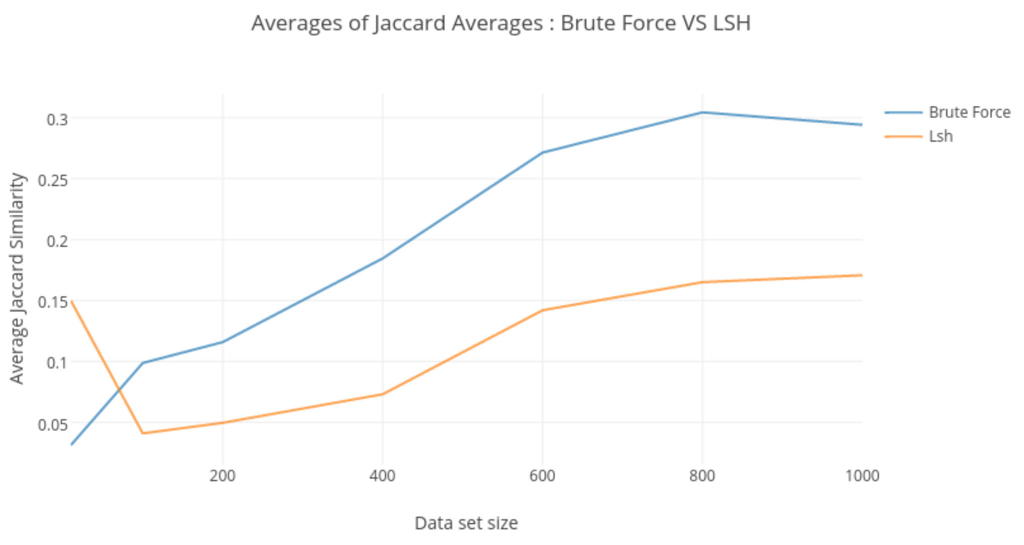




As the number of row hashes and the number of documents go down the precision of the estimated Jaccard goes down. We can possibly attribute the similarity in results from both methods to the algorithm used to estimate the jaccard similarities. The relationship; $P(h(S_x) = h(S_y)) = Jaccard(S_x,S_y)$ allows for this approximation to be similar. The inaccuracy caused by the low data set is probably due to insufficient data points for estimation.

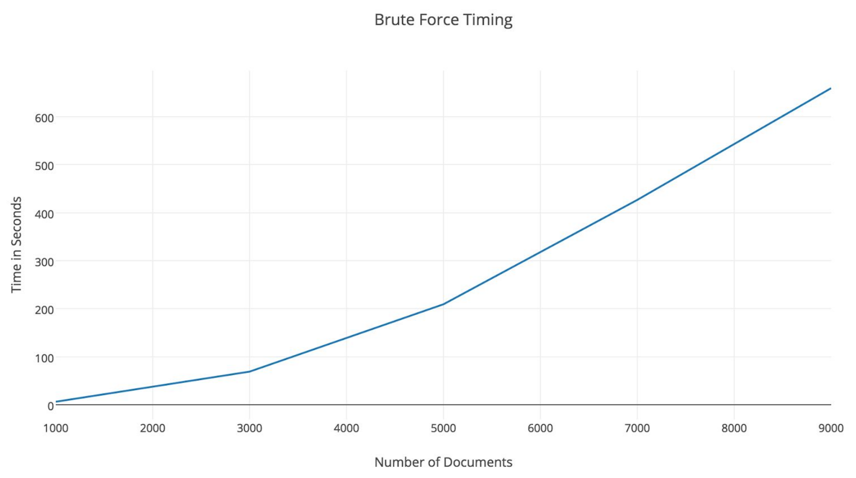
LSH vs Brute Force

Jaccard Averages



The plot above shows that when the data set size is very low LSH averages are very high, however as soon as the data set passes a certain size threshold the averages for LSH fall drastically before steadily rising again. In general the LSH averages are lower than the brute force averages due to the fact that we are using estimation to calculate jaccard similarities in LSh thus it results in lower averages. Notice that both increase, when the size of the data set increases.

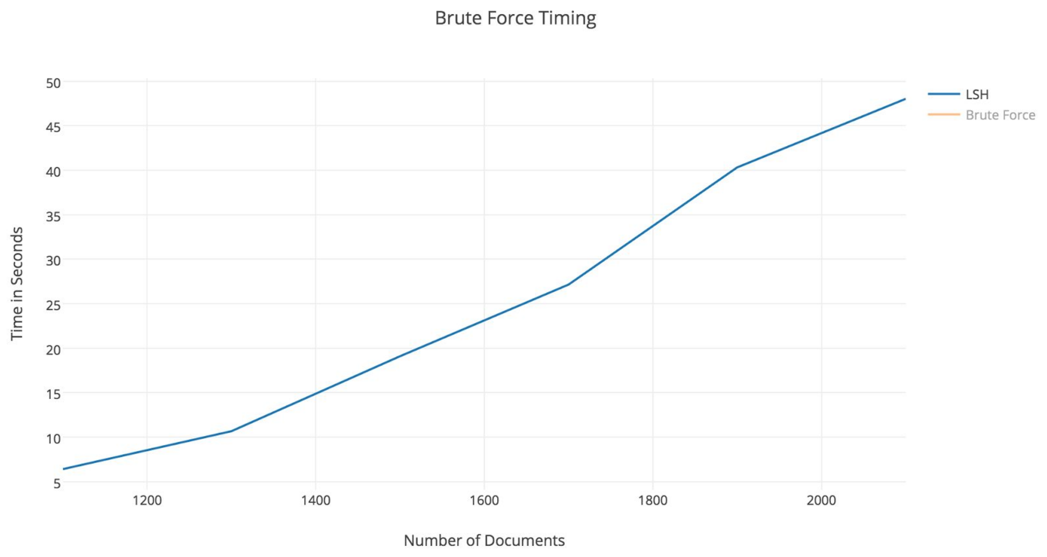
Running Time of Brute Force



.
As the number of documents increases the time required seems to grow faster than a linear pace. In this example we used k = 3.

Running Time of LSH

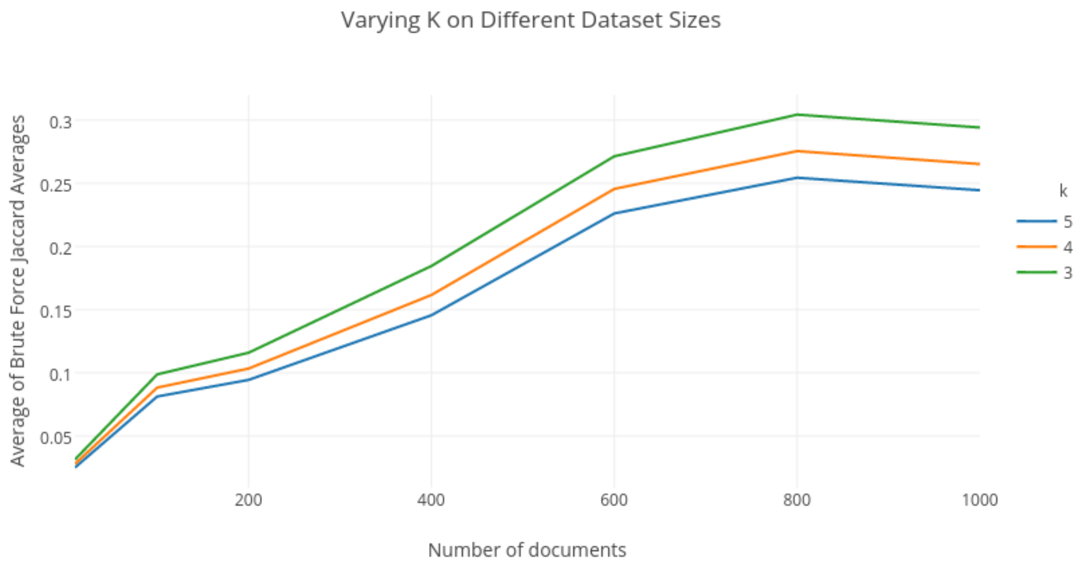
In this example we used k = 3, rows in signature matrix = 5, band rows = 3.



Our LSH algorithm with 100 rows performed about as well as the brute force algorithm, up until about 1000 documents. After this point the performance of LSH dropped significantly which is not what we were expecting. When we had a small number of rows in the character matrix the LSH algorithm ran more quickly (on a document size less than 1000) than the brute force algorithm. In general the LSH seems to perform better in runtime than the brute force since we are only calculating jaccard for fewer candidate pairs rather than throughout the entire dataset for each document.

Parameter Choice

Dataset size and k analysis

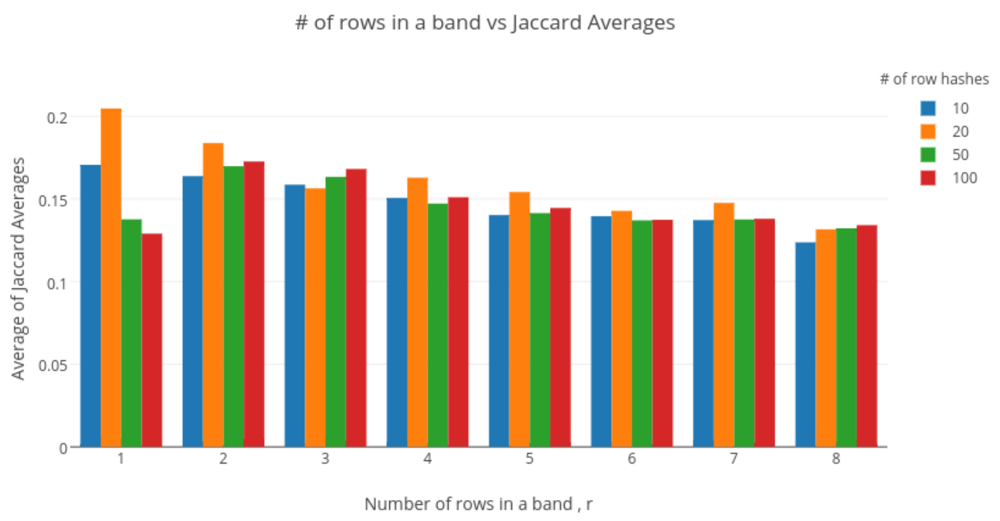


As seen in section one, number of documents and number of rows in the signature matrix have a big impact on the accuracy of the estimated Jaccard.

Size of k affects our algorithm because if k is too large and we can't find k candidate pairs we will randomly generate candidate pairs from the documents. This causes our algorithm to not be as accurate.

At a document size of 1000 the number of hashes doesn't seem to have an effect on accuracy. This behavior can be attributed to increase in averages within the k nearest neighbors as the data size increases.

<u>Number of rows in the signature matrix and r analysis</u>



As the number of rows in the signature matrix increases the running time for LSH will increase. However, if the number of rows in the signature matrix is small we trading off speed for accuracy of the Jaccard estimate.

If we have less bands the number of false positives will go down but the number of false negatives will go up. Choosing r will depend heavily on how we weigh those two scenarios in our use case. In general we can also see that the average jaccards decrease slowly as you increase the number of rows in the bands. This can be attributed to the fact that as the number of rows in a band increase it becomes difficult to identify candidate pairs.