lab3

Omer Ronen

10/19/2020

1 Parallelizing k-means

1.

(a). Code

(b). The important thing to note is that both the numerator and the denominator of the cosine (correlation) similarity are additive in the indices so we take advantage on this. In the $O(n^2)$ we simply iterate over all indices, compute each term separately and lastly divide. On the $O(k_1k_2n)$ implementation what iterate over each cluster combination and count all the indices for our two clusters vectors l_1, l_2 for which $l_1 = i \wedge l_2 = j$ for two cluster i, j. After counting we take the square to account for all pairs.

(c). Obviously the cpp code is much faster, we can also see that the implementation in $O(k_1k_2n)$ is indeed faster with the margin becoming larger (in log scale)

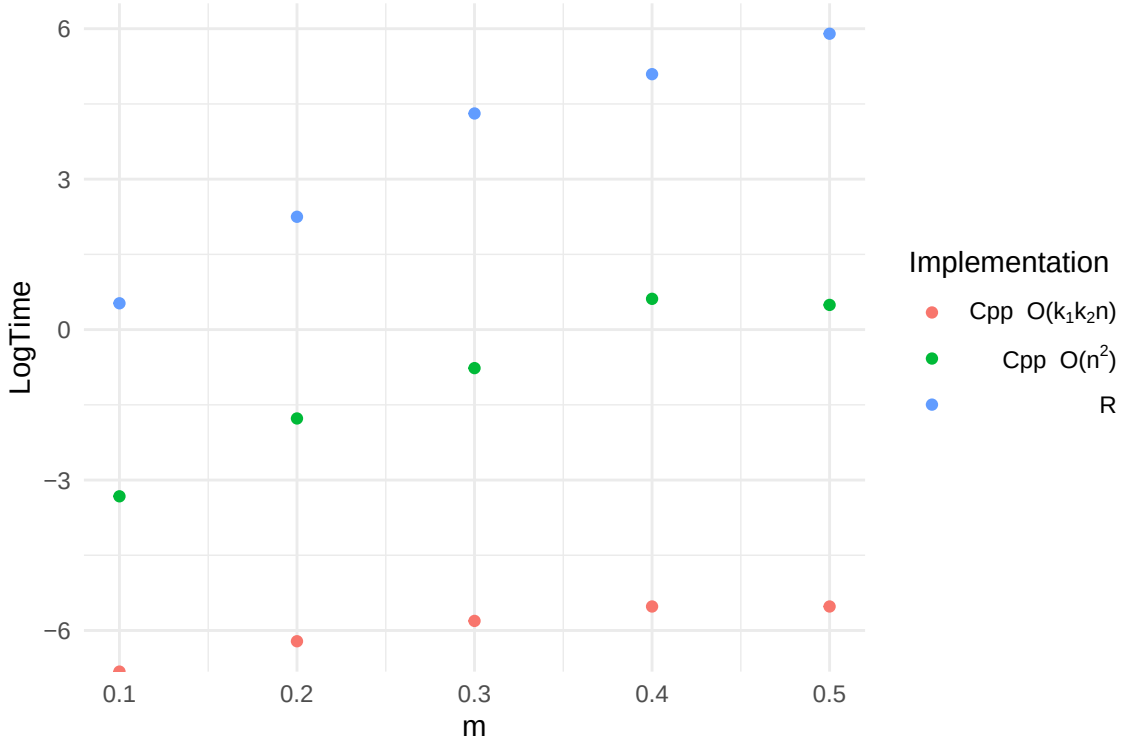

Figure 1: Run time (log scale) for R and Cpp implementations of similarity (correlation) score

2. code

3.

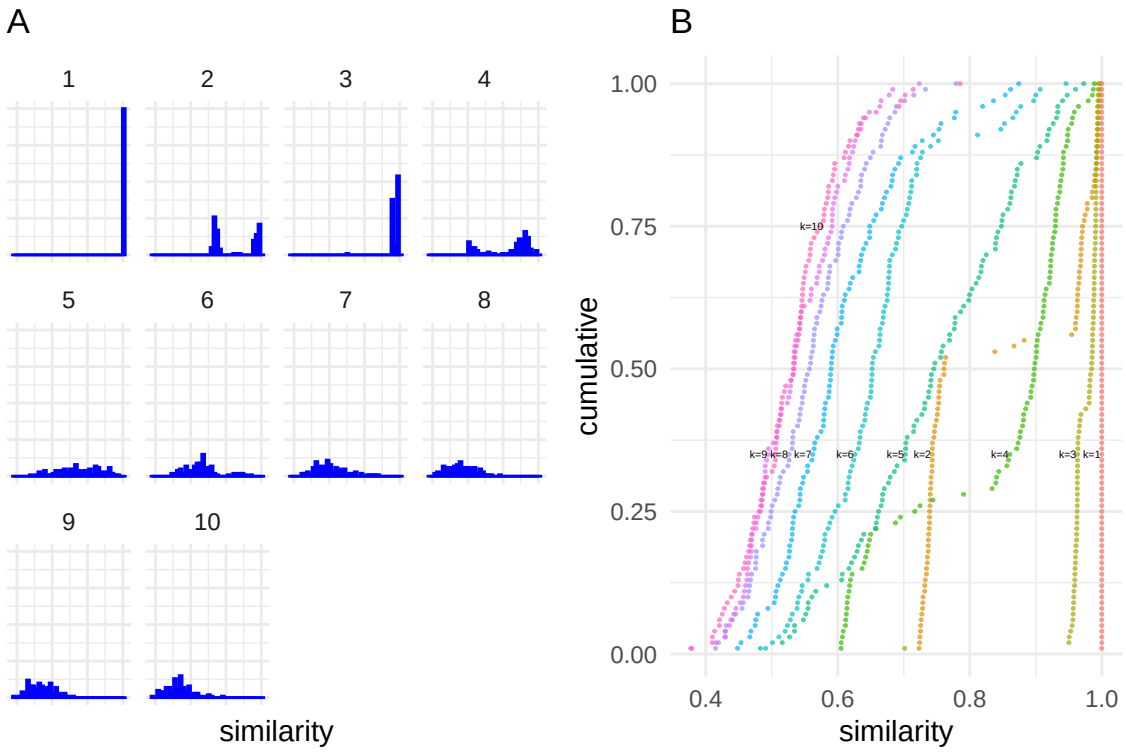

Figure 2: Stability analysis for choice of K (number of clusters). In A we can see the similiraity histograms for various choices of k. In B we show the empirical CDF obtained from calculating the similarity over 100 sub-samples of the dataset

On this plot the cluster than stands out is k=3, with it's distribution very dense around higher values of similarity.For k=2 we have an interesting distribution where in 50% of cases we get really good similarity and on the other 50% we're as good as random. k=1 serves as a sanity check.