Lab 3 - Parallelizing k-means Stat 215A, Fall 2017

24978168

October 20, 2017

Note: The code and other necessary files for generating the plots in this report are included in /data. Please refer to README in /data for a brief description of each file.

1 Introduction

Assessing the stability of statistical methods is an important part of data analysis as the level of stability can tell us whether the results of the analysis can be trusted or generalized. In this report, I will investigate the stability of k-means clustering on the linguistics dataset by perturbing the input data for different values of k, and use the stability results in return to choose an appropriate value of k for the data. Furthermore, I will discuss the computational issues that must be addressed when dealing with large datasets such as running time and memory efficiency.

2 Data and Method

To investigate the stability of k-means, I used the linguistics dataset from lab2 [1]. The data consisted of 45,152 observations (rows) and 468 columns that correspond to the binary responses to the survey questions. The data was used in its original form given in lingBinary.RData.

In order to assess the stability of k-means, I followed the procedure outlined in Figure 2 in Ben-Hur [2], with sampling proportion of 0.5 and number of trials of 100, for each $k = 2, ..., 10$. Furthermore, to measure the similarity between two different cluster assignments, I used the correlation similarity measure, *cor*, described in Equations 1-3 in [2].

3 Results and Discussion

3.1 Running Time Comparison of R and C++

Here I compare the running time of R and C++ for computing the correlation similarity. Both methods are implemented in a similar way, where the most computationally intensive component is a double for loop that iterates through every pair of observations, (i, j). Consequently the running time is $O(n^2)$ for both methods, where n is the number of observations. In addition, both methods do not explicitly store the n-by-n C matrix described in [2]. Instead, each element, C_{ij}, is computed only when it is needed, by iterating over the original cluster assignment vector of size n. Therefore the complexity for memory is $O(n)$.

Benchmarking both methods on simulated data with $n = 5000$ showed that the method implemented in C++ was on average 34 times faster than the method in R (2502 ms for R vs. 72 ms for C++). This highlights that the running time may vary greatly between different programming languages and thus we must think critically about computational efficiency, especially when working with large datasets. For the stability analysis of k-means, therefore, I used the method implemented in C++ in order to speed up the computation.

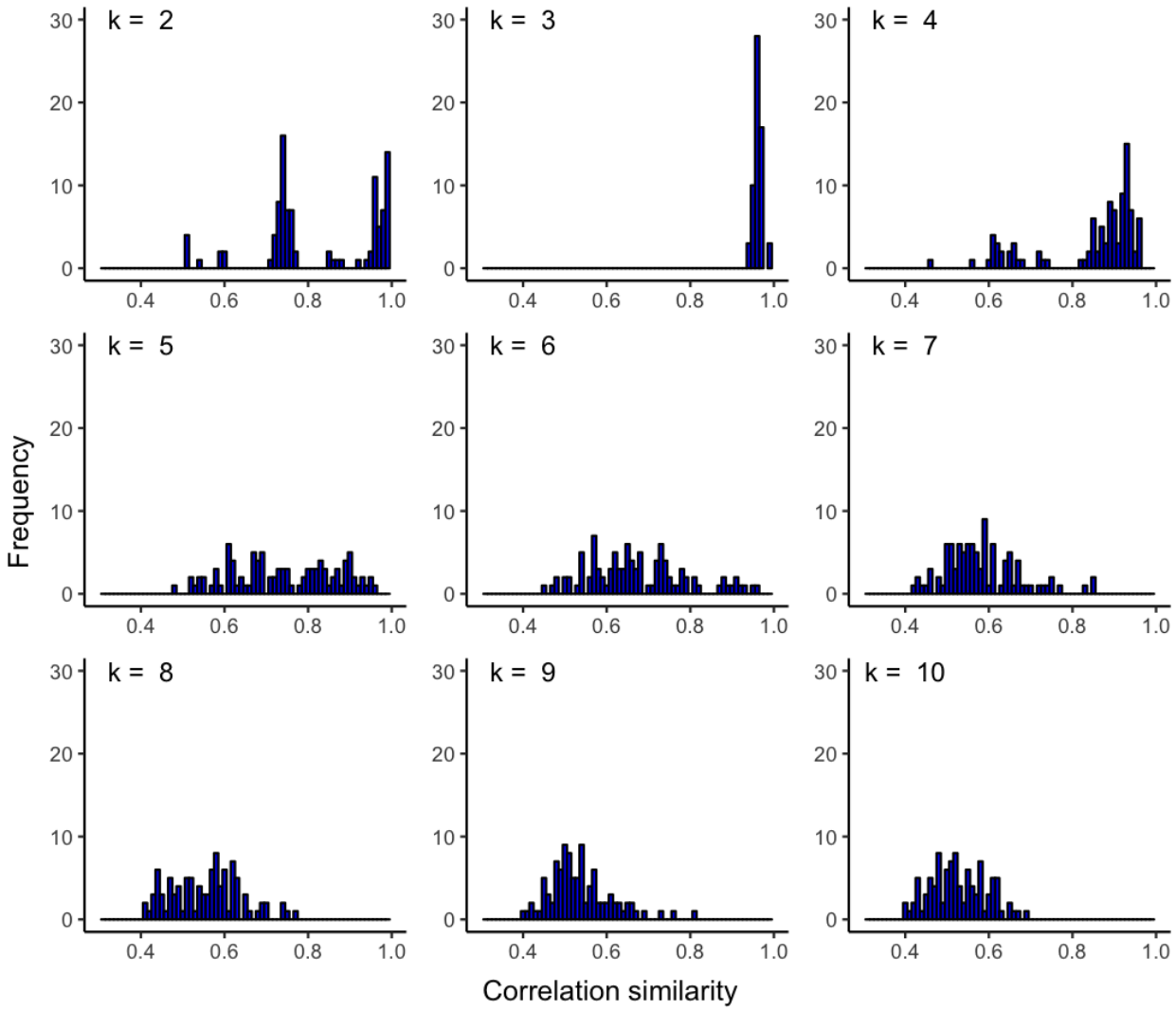

Figure 1: Distribution of the correlation similarity for different values of k.

3.2 Stability of k-means

Figure 1 shows the distribution of the correlation measure for different values of k. We can see that $k = 3$ produced the most stable results with high mean and low variance in the distribution. Interestingly, $k = 2$ produced a bimodal distribution, where about half of the runs produced correlation values near 1 whereas the other half gave lower values around .75. As the number of clusters increases beyond 3, the spread of the distribution generally becomes greater and the mean shifts to the left, indicating that k-means becomes increasingly unstable beyond $k = 3$. Figure 2 also tells the same story, where the cumulative density shows that more correlation values are near 1 for $k = 3$ than any other choice of k. Therefore, I would choose $k = 3$ to cluster the linguistics dataset. Choosing $k = 3$ would also agree with my previous finding that three clusters produced the lowest average silhouette value in lab 2, which suggests that using the stability measure in this way may be a reliable method for choosing an appropriate value of k.

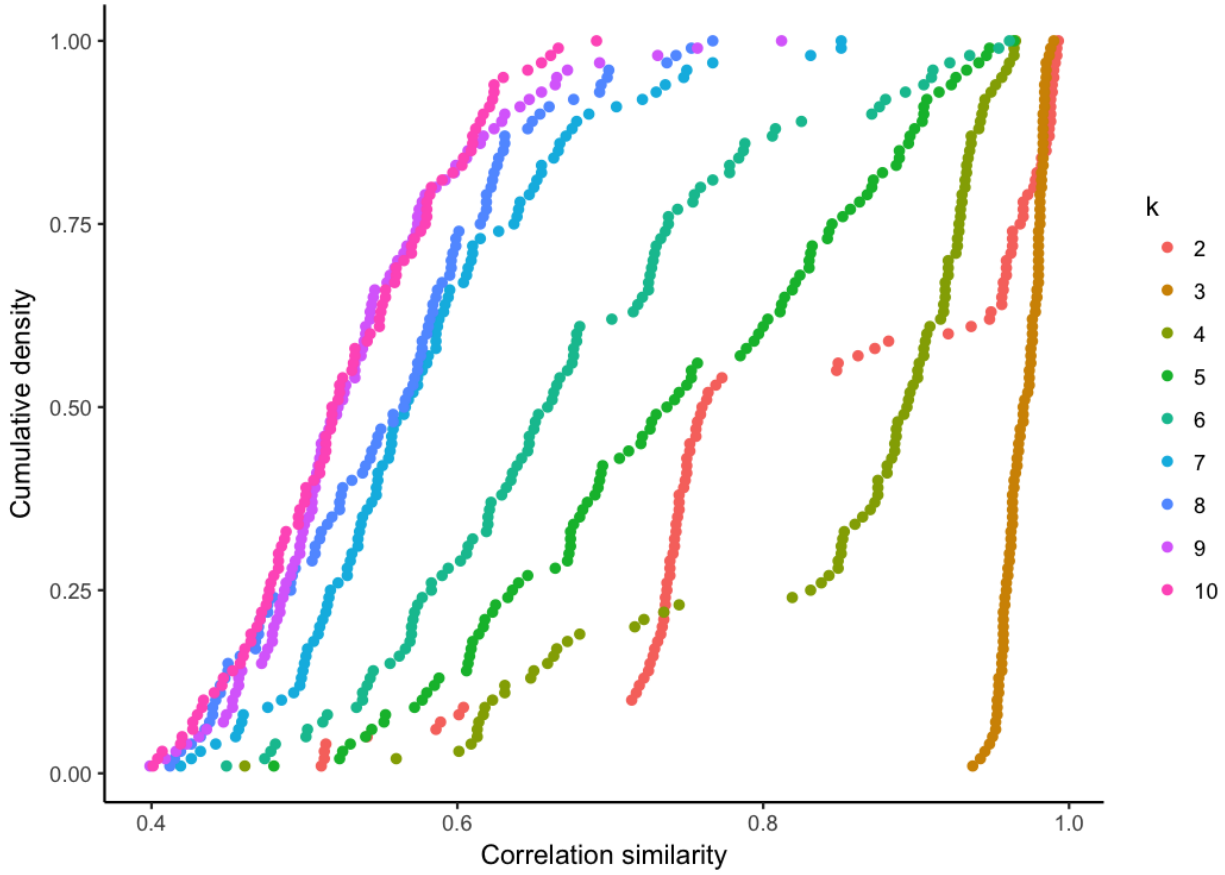

Figure 2: Cumulative distribution of the correlation similarity for different values of k.

4 Conclusion

In this report, we explored the stability of the k-means algorithm on the linguistics dataset by perturbing the input data, and using the results in return to choose an appropriate value of k. We saw that $k = 3$ produced the most stable results in terms of correlation similarity, which agrees with our previous finding that $k = 3$ also gave the lowest average silhouette value. Furthermore, we saw that C++ was about 34 times faster in computing the correlation measure than R, which emphasizes how the choice of programming language and how we implement the code may have a great impact on the computational efficiency of analyzing large datasets.

References

[1] Vaux, Bert. "Harvard Dialect Survey." dialect.redlog.net. Harvard University. 2003. Web.

[2] Asa Ben-Hur, Andr Elisseeff, and Isabelle Guyon. A stability based method for discovering structure in clustered data. In Pacific symposium on biocomputing, volume 7, pages 617, 2001.