Homework 4 (programming assignment due 3/20)

DS-563 / CS-543 @ Boston University

Spring 2024

Before you start...

Collaboration policy: You may verbally collaborate on programming assignments, however, you must write your code and final report independently, i.e., without seeing other students' solutions. If you choose to collaborate on a problem, you are allowed to discuss it with **at most three** other students currently enrolled in the class.

The header of each assignment you submit must include the field "Collaborators:" with the names of the students with whom you have had discussions concerning your solutions. A failure to list collaborators may result in a credit deduction.

You may use external resources such as textbooks, lecture notes, and videos to supplement your general understanding of the course topics. You may use references such as books and online resources for well known facts. However, you must always cite the source.

You may **not** look up solutions to a programming assignment in the published literature or on the web. You may **not** share written work with anyone else. If you wish to make your code public (for instance, by making it publicly available on GitHub or GitLab), please wait till the end of the semester. (If you want to publish it earlier for whatever reason, please check with us first.)

Submitting: Your solution is to be submitted via Gradescope (entry code: 6G4V6G). In particular, you should submit a pdf with your project report and a zip file with your code (or an equivalent of these as long as it's allowed by Gradescope). Don't forget to provide information how to obtain your data sets. More information on this is provided below.

Late policy: No extensions. Submitting a solution one day late may result in a deduction of 10% of points.

Your task

1. Implement **four** different versions of the Johnson–Lindenstrauss transform. Each of them should be of the form
$$f(x) = \alpha M x,$$
where $x \in \mathbb{R}^{k\times 1}$ $\mathbb{R}^{1\times k}$ is a vertical vector of height k, $M \in \mathbb{R}^{d\times k}$ is a randomized matrix with d rows and k columns, and $\alpha \in \mathbb{R}$ is a normalizing factor. The main difference between different versions should be how M is selected. You can come up with your own versions—even if you don't know how to prove that they work or they do not ultimately work because of the distribution used—but the following transformations are known to work for some selection of parameters:

(a) Select each entry independently from the Gaussian distribution, which is what we covered in class.

(b) Select each entry independently from the uniform distribution on $\{-1, 1\}$ (a.k.a. the Radamacher distribution).

(c) Select each entry independently from the following distribution: -1 with probability $1/6$, 0 with probability $2/3$, and 1 with probability $1/6$.

(d) For each column, select one third of its entries with no replacement. Set the entries that were not selected to 0. Each selected entry should be assigned a value drawn independently from the Radamacher distribution.

In each case, make α such that, in expectation, the square of the length of a vector (i.e., its ℓ_2 norm squared) gets preserved in the transformation. More formally, for any vector $v \in \mathbb{R}^{k\times 1}$,

$$\mathbb{E}\left[\|f(v)\|^2\right] = \|v\|^2,$$

where the expectation is taken over the selection of matrix M in the definition of f. In your implementation, when you generate each of the transforms, one of the parameters should be d, the target dimension.

2. Our goal is to compare different transforms to see how they perform for different target dimensions. To this end, we need some measure of their performance. In the spirit of the Johnson–Lindenstrauss lemma, we could for instance look at the worst (i.e., highest) value of

$$\left|1 - \frac{\|f(u) - f(v)\|^2}{\|u - v\|^2}\right|,$$

taken over all pairs u and v of points in the data set. This would correspond to the ϵ in the JL Lemma statement that this transform achieved. You could also skip the square operation to more directly relate this to the multiplicative approximation you get in the worst case on the lengths of vectors (as opposed to their squares).

Your task is to propose **three** different measures of performance and explain why you are using them. Some other examples would be the average divergence (this could be interpreted as the mean of values above taken over all the pairs of points) or the variance of the divergence.

3. Now your task is to compare your different versions of the JL transform on **two** different real–world data sets of your choice. (Select data sets that have the potential to make this comparison interesting.) Apply your measures of performance. Draw a graph how the performance of the different versions changes as a function of the target dimension d.

Note: It could be the case that there is a lot of variance in your results. That is, each time you generate a transform you get very different values of your measure of performance on your data set. If this is the case, you may want to run your test multiple times and take the average or median of the results you achieved. If so, please comment on this in your report and explain what you did.

4. What are your conclusions? Which of the versions seems to perform best? How would you rank them?

5. How much time (approximately) did you spend on this homework? Was it too easy/too hard?

Deliverables (General Guidelines)

Your final submission should include:

- Code:
  - Provide your implementation of different versions of the Johnson–Lindenstrauss transform.
  - Provide any auxiliary tools you develop for preprocessing data or generating artificial data sets.
  - Make your code readable (use reasonable variable and function names, etc.).
- Report:
  - It should include all important implementation details and discuss implementation decisions. Describe briefly how to run your code with an example.
  - It should address all questions from the previous section. Use tables and/or graphs to display numerical results of your experiments.
- Data: Describe your data sets and make sure we can access them. Think of what someone would need to reproduce your results.
  - For any public data set that you use, provide a link to where this data set can be obtained from and include code used for its preprocesssing (if needed).
  - For any artificial data set, provide code that generates it (try to make it deterministic, by for instance, fixing the random seed it uses) or share a link (Dropbox, Google Drive, etc.) that can be used to access the data set. If you provide a link, make sure it does not require logging in.