Programming Assignment 1

Nearest Neighboor

Problem Description

Input: A set of points in the plan, $\{p_1 = (x_1, y_1), p_2 = (x_2, y_x), \ldots, p_n = (x_n, y_n)\}$
Output: The euclidean distance between the the closest pair of points: that is, the pair $p_i \neq p_j$ for which the distance between p_i and p_j ($d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$) is minimized.

Brute Force

First, implement a brute force algorithm which compares each pair of points. You will then run this on the smallest example, and manually verify that it is correct. You will use this to verify the correctness of your divide-and-conquer solution as well as compare run time.

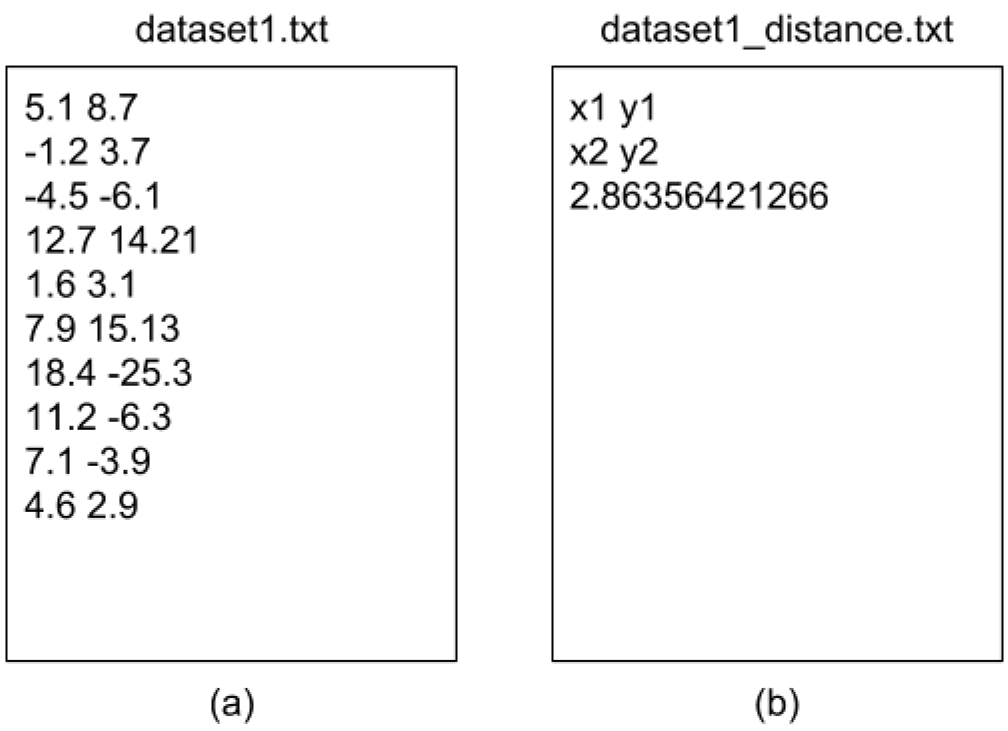
Divide-and-Conquer

Here is a high-level overview of the divide-and-conquer algorithm (from class):

- Find a value x for which exactly half of the points have $x_i < x$ and half have $x_i > x$. On this basis, split the points into two groups, L and R.
- Recursively find the closest pair in L and in R. Say these pairs are $p_L, q_L \in L$ and $p_R, q_r \in R$, with distance d_L and d_R respectively.
- Let d be the smaller of these two distances.
- It remains to be seen wheither there is a point in L and a point in R that are less than distance d apart from each other. To this end, discard all points with $x_i < x - d$ or $x_i > x + d$ and sort the remaining points by their y-coordinate.
- Now, go through this sorted list, and for each point, compute its distance to the subsequent points in the list. Let p_M, q_M be the closest pair found in this way.
- The answer is one of the three pairs $\{p_L, q_L\}, \{p_R, q_R\}, \{p_M, q_M\}$, whichever minimizes d.

Programming Requirements

- You will be writing the program in Python 3, using the provided template.
- The nearestNeighbor.py script should run through the command line. A argument parser has been added to the template. (https://docs.python.org/3/library/argparse.html)
- The output should be a file named <dataset>_distance.txt formatted according to Fig. 1
- The input will be a .txt file with the x y coordinate of the points separated by a space, as shown in Fig. 1



dataset1.txt

```
5.1 8.7
-1.2 3.7
-4.5 -6.1
12.7 14.21
1.6 3.1
7.9 15.13
18.4 -25.3
11.2 -6.3
7.1 -3.9
4.6 2.9
```

(a)

dataset1_distance.txt

```
x1 y1
x2 y2
2.86356421266
```

(b)

Figure 1. (a) Input and (b) output file formats. Command:
`$python3 --algorithm a dataset1.txt`

Profiling

You will be using the profile package in python to profile your program.

Report

The report needs to contain the following (use the provided LaTeX template):

- A run-time analysis of the brute force algorithm vs. the divide-and-conquer algorithm described
- A table showing the run-time of each algorithm on all data sets provided
- A discussion of the theoretical run-time (justify the theoretical run-time for both algorithms)
- A discussion of the theoretical run-time, and how it is shown in the real example
- A section discussing a real-life problem and how you can convert that problem into the nearest-neighbor problem to be solved by these algorithms.

Submission

You will be submitting a .zip file with the following components

- nearestNeighbor.py - *The code that you wrote*
- Report.pdf - *Report of the results of your experiments (written in LaTeX)*
- README.txt - *Describing how to run the program*