Assignment 2

Data Preparation Techniques (COMP 3400)

Fall 2024

Important notes:

1. You are required to submit your assignment in *one* file in *IPython Notebook* format (due date: Oct 13). Please note that:
   - You may use *Markdown* in your IPython Notebook.
   - How you develop your IPython Notebook is your choice. You may use *jupyter.org* or *Google Colab*, or a locally installed Jupyter platform on your machine.
2. If not instructed otherwise, use solely `NumPy` (imported as `np`) or `Pandas` (imported as `pd`) to solve the problems.
3. In this assignment the term "array" means `Python`'s `ndarray`.
4. You are not allowed to use loops, in any of the problems unless otherwise stated (or you'll get a mark of 0 for that problem).
5. For some of the problems you may have to refer to the `NumPy's` API.
6. Slides covered in this assignment: *2-07* to *3-01*.

Problem 1 (15 pts). The *Law of Large Numbers* states that the sample mean converges to the true mean (population mean) as the size of the sample grows. In this exercise you test this law using `NumPy`. First, create a population consisting of 10^6 random points between 0 and 1 from the *uniform* distribution. Then use *Fancy Indexing* and uniform random sampling to choose samples of size 5, 5×10^2, and 5×10^5 from the population. Finally, relying upon the collected samples, show that the Law of Large Numbers holds.

Problem 2 (20 pts). Find the closest number to 0.1 in each row of a 10×3 array of random numbers ranging from 0 to 1.

Problem 3 (50 pts). In the problem of *k-Nearest Neighbors* discussed in class, the underlying *distance metrics* is *Euclidean* where the distance d between two points $p_1 = (x_1, y_1)$ and $p_2 = (x_2, y_2)$ is measured by $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$. Rewrite the code with two other metrics: first with the *Manhattan* distance where $d = |x_1 - x_2| + |y_1 - y_2|$, and then with the *Chebyshev* distance where $d = \max(|x_1 - x_2|, |y_1 - y_2|)$. Run the three versions on the *same* set of array of 100 random points $p = (x, y)$ where $x, y \in [0, 1]$. Finally, use `NumPy` to determine in what percentage of all the 100 points, the closest point returned by k-Nearest Neighbors with Manhattan distance is *not the same* as the closest point returned by the k-Nearest Neighbors with the Euclidean distance. Do the same comparison between k-Nearest Neighbors with Euclidean distance and Chebyshev distance, and between k-Nearest Neighbors with Chebyshev distance and Manhattan distance.

Problem 4 (15 pts). Create a `pandas` Series called `squares`, containing the squares of integers from 1 to 10, indexed by those integers. Create a new Series called `cubes`, containing the cubes of integers from 2 to 11, indexed by those integers (you may use loops for creating `squares` and `cubes`). Finally, inspect the `ndaray` produced by `(squares+cubes).values` and justify the unexpected values in it.