**Week 03 - Python Optimization Assignment**

For this week’s application assignment, please complete the following tasks using the data provided in clinic.xls (see Week04 folder)

**1. Tabulate the execution times of each of the individual approaches for computing distance in Python (i.e., run the shared code on your computer, note the times, and tabulate them).**

**Execution Times in Python**

The execution times for different approaches in Python are as follows​:

| **Method** | **Execution Time** |
| --- | --- |
| For-loop (iterrows) | 794 µs ± 15 µs |
| Apply with Lambda | 468 µs ± 16.3 µs |
| Vectorized (NumPy) | 48.4 µs ± 1.47 µs |

**2. Next, replicate the for-loop based approach (the first one) and two different ways to make that version more efficient, in R. Profile these three approaches, and tabulate the results.**

The execution times for different approaches in R are​:

| **Method** | **Execution Time** |
| --- | --- |
| For-loop | 0.025 sec |
| Apply Function | 0.0098 sec |
| Vectorized Approach | 0.0052 sec |

**3. Based on the computational efficiency of implementations in Python and R, which one would you prefer? Based on a consideration of implementation (i.e., designing and implementing the code), which approach would you prefer? Taking both of these (run time and coding time), which approach would you prefer?**

Python's NumPy-based vectorized approach (48.4 µs) outperforms R's vectorized approach (0.0052 sec = 5.2 ms). The same pattern applies to for-loops and apply functions.

Ease of Implementation: R's apply and vectorized functions are identical to Python's apply(), although Python has more efficient NumPy and Pandas support.

Overall Preference: Python is superior in terms of performance, but R is also a good alternative for statistical tasks due to its easier built-in vectorized functions.

**4. Identify and describe one or two other considerations, in addition to these two, in determining which of the two environments – Python or R – is preferable to you.**

Scalability: Python's ability to integrate with optimized libraries (NumPy, Cython) makes it more suitable for huge datasets.

Domain-Specific Use Cases: R is more appropriate for statistical and data analysis workflows, but Python is more suited for general-purpose programming and machine learning.