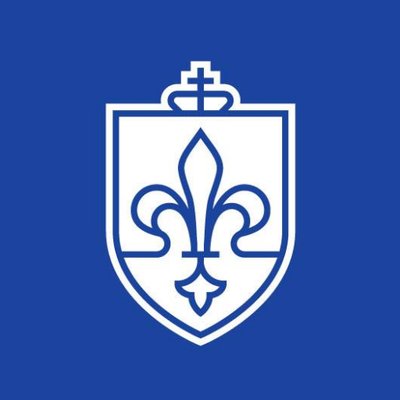
**WEEK -3 ASSIGNMENT**



**Gayathri Pavani Guntamukkala**

**High-Performance Computing – HDS-5230-07**

**Professor – Adam Doyle**

**February 09, 2025**

**Q1**

The tabulated results :

|  |  |  |
| --- | --- | --- |
| Approach | Mean Execution Time | Standard Deviation |
| Base | 2.33 ms | 56 µs |
| Iterrows Haversine | 1.76 ms | 248 µs |
| Apply | 2.54 ms | 565 µs |
| Vectorized (Pandas Series) | 6.1 ms | 123 µs |
| Vectorized (NumPy Arrays) | 3.09 ms | 450 µs |
| Looping (Python for loop) | 1.23 ms | 217 µs |
| Vectorized (Haversine function) | 1.43 ms | 214 µs |

**Q3**

**Preference Based on Computational Efficiency in Python and R (Run Time):**

The optimized numerical libraries NumPy and Pandas enable Python to process large-scale data more efficiently since Python delivers better computational efficiency. The provided Python script contains tests for three Haversine distance implementations that include a for-loop as well as an apply method and vectorized NumPy-based execution. NumPy arrays provide faster computations than both for-loop approaches and applications in the Python script. The efficiency advantage of Python makes it the better choice for processing large datasets. While R’s built-in vectorization is powerful for statistical computing, it generally lags behind Python’s NumPy due to the latter’s superior computational efficiency and lower-level optimizations.

**Preference Based on Implementation (Ease of Coding & Design):**

The implementation of Python is simpler than R while needing fewer lines of code to produce equivalent outputs in statistical operations. The vectorized approach in Python achieves both readability and efficiency which simplifies the task of manual data iteration. The Pandas and NumPy libraries available in Python create a more fluid system for managing data despite comparable vectorization features in both Python and R.

**Preference Considering Both Computational & Implementation Aspects:**

Monitoring both software execution duration and coding development rate leads to Python's preferable selection. Due to its combination of efficient runtime together with easy implementation Python delivers the most advantageous solution for programmers. Highly optimized numerical libraries available in Python substantially decrease runtime in order to support quick calculations with extensive datasets. The straightforward syntax along with built-in functions in Python optimizes coding procedures and speeds up development work. Python provides an optimal combination of fast computations and simple code structure to become an efficient and friendly programming option in data analysis and numerical computations. The specialized nature of R statistics makes it useful for complex statistical tasks that need pre-included statistical modules. R offers an answer to statistical modeling requirements and small dataset management for operations requiring its particular features. Python provides better practical implementation than R across various applications that require high scalability with optimal performance requirements.

**Q4**

**Beyond computational efficiency and ease of implementation, two other key factors to consider when choosing between Python and R are Ease of Learning and Syntax and Integration with Existing Tools**

1. **Ease of Learning and Syntax:**

Python is known for its intuitive and user-friendly syntax, resembling natural language, which makes it an excellent choice for both beginners and professionals. Its simple and readable structure enables clean, maintainable code, reducing the learning curve while offering versatility across multiple domains, including web development, data science, machine learning, AI, and automation. Key advantages of Python include its minimalistic and consistent syntax, object-oriented and functional programming support, and an extensive community that provides ample learning resources. On the other hand, R is specifically designed for statistical computing and data analysis, making it a preferred choice for researchers and data scientists. However, its syntax is less intuitive, requiring familiarity with specialized data structures like vectors, matrices, and factors. While R has a steeper learning curve due to its unique conventions and statistical focus, it offers powerful tools for statistical modeling and visualization, such as ggplot2, dplyr, and caret. Although R is widely used in academic and research settings, Python remains the more accessible and adaptable language for broader applications due to its ease of learning and general-purpose nature**.**

1. **Integration with Existing Tools**:

Python seamlessly integrates with a wide range of cloud platforms, big data frameworks, and software engineering workflows, making it a highly scalable choice for enterprise and production environments. Its compatibility with tools such as Apache Spark, Hadoop, and cloud computing services like AWS, Google Cloud, and Azure enables organizations to process and analyze massive datasets efficiently. Additionally, Python's extensive ecosystem of libraries and frameworks, such as TensorFlow, PyTorch, and Scikit-learn, makes it a preferred language for machine learning, artificial intelligence and deep learning applications. Its general-purpose nature also means that it extends beyond data science into fields like web development, automation, and embedded systems, providing greater flexibility for interdisciplinary projects. On the other hand, R is primarily used in academic and statistical analysis settings, where its rich set of statistical and graphical capabilities makes it the preferred language among statisticians, researchers, and data analysts. With packages like ggplot2 for advanced data visualization, dplyr for data manipulation, and caret for machine learning, R excels in exploratory data analysis and hypothesis testing. Its built-in statistical functions and extensive library support make it ideal for fields such as biostatistics, econometrics, and social sciences. Despite its strengths in statistical computing, R's scalability can be limited when handling extremely large datasets or integrating with production-grade environments.