COP290 Assignment 1 - Subtask 1

Disha (2022CS11118)

In this subtask, it was aimed to collect data for NIFTY-50 stocks using the `jugaad-data` Python library. The goal was to fetch daily stock data for a given symbol over the last 'x' years, write this data to different file formats, benchmark the time taken, and analyze file sizes.

Methodology of Benchmark:

Time Measurement: The function captures the current time before and after the DataFrame is written to a file. The time difference provides the elapsed time taken for the write operation.

File Writing: The function facilitates the storage of the DataFrame (df) into a file, and the filename is determined based on the provided symbol. It supports various file formats, such as CSV, TXT, Binary (Pickle), and Parquet.

File Size Measurement: The function uses `os.path.getsize` to determine the size of the generated file. This metric is crucial for benchmarking the space efficiency of each file format.

Benchmark Results: The function returns a tuple containing the elapsed time and file size. These results are used for comparison and analysis to understand the performance characteristics of different file formats.

The benchmarking results revealed interesting insights into the performance of different file formats:

1. **CSV Format:**
    - Balanced trade-off between time and file size.
    - Suitable for compatibility and simplicity.
2. **TXT Format:**
    - Similar to CSV with slightly larger file sizes.
    - Faster write times compared to CSV.

3. **Parquet Format:**
    - Efficient space utilization with smaller file sizes.
    - Marginally slower write times compared to CSV and TXT.
4. **Binary Format (Pickle):**
    - Most efficient space utilization with the smallest file sizes.
    - Faster write times compared to CSV and TXT.
5. **JSON Format:**
    - Larger file sizes compared to CSV and TXT.
    - Reasonable write times, suitable for web-based systems.
6. **Excel Format (XLSX):**
    - Larger file sizes compared to CSV and TXT.
    - Reasonable write times, suitable for spreadsheet applications.
7. **SQLite Database Format:**
    - Competitive write times with reasonable file sizes.
    - Suitable for scenarios requiring relational database features.
8. **HDF5 Format:**
    - Competitive write times with compact file sizes.
    - Efficient for handling large numerical datasets.
    - Supports advanced features like compression and chunking.
    - Suitable for complex data structures and scalable solutions.