COP290 Assignment-1
Subtask-1 Report

Aneeket Yadav

January 2024

1 Introduction

In this report we discuss the performance concerning time and space requirements of different file formats used to store dataframes, which in our case is composed of historic stock data. We compared seven different file formats:

- `.txt`
- `.csv`
- `.json`
- `.md`
- `.parquet`
- `.feather`
- `.orc`
- `.bin`

The following criteria were used:

- Time taken to write the dataframe the file
- Space occupied by the file in memory
- Time taken to read the file and convert it into a pandas dataframe
- Decompression time

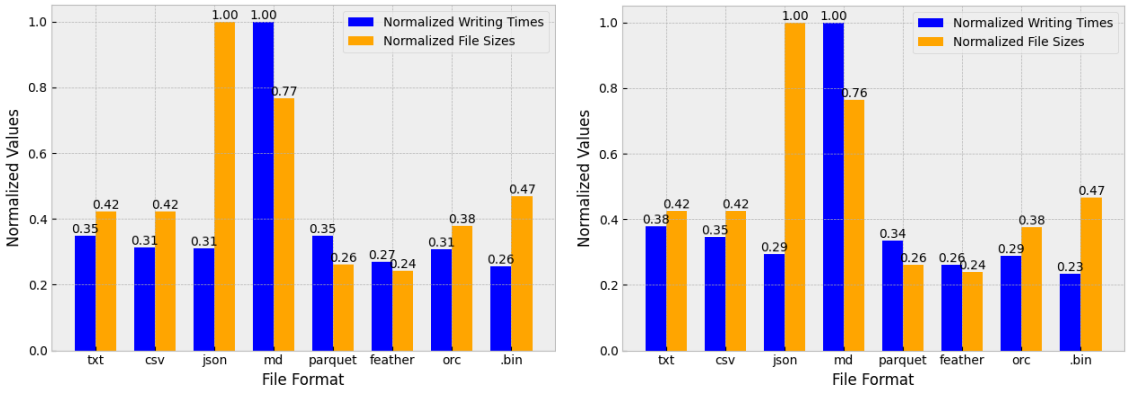
2 Observations

2.1 Writing times and Space Occupied

The graphs on the next page have been plotted to retrieve the past 25 years stock data for the symbols `SBIN`, `HDFC`, `AXISBANK` and `BAJFINANCE`.The following inferences can be made:

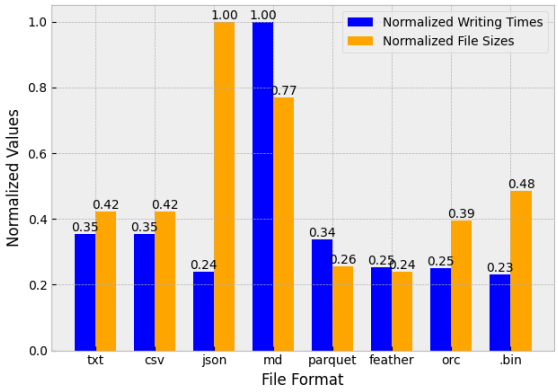
- `.bin` has the least writing time since it is a binary file. However, it is not as space efficient.
- `.feather` and `.parquet` are second best in terms of writing times as well as best in space efficiency. Either one occasionally performs slightly better than the other. They too store the data in a binary format.
- `.md` consistently perform the worst both in terms of writing times and space effciency.

- `.json` has the least writing time (even slightly better than `.parquet` and `.feather`). However, with `.md`, it least space efficient. This may be because the contents first had to be conactenated to form a datalist.
- `.csv` format which is popular in data analytics performs optimally under both criteria though it is not as efficient as `.parquet` and `.feather` formats.
- `.orc` (organised-row-column) which is lesser know has performance similar to `.csv`.

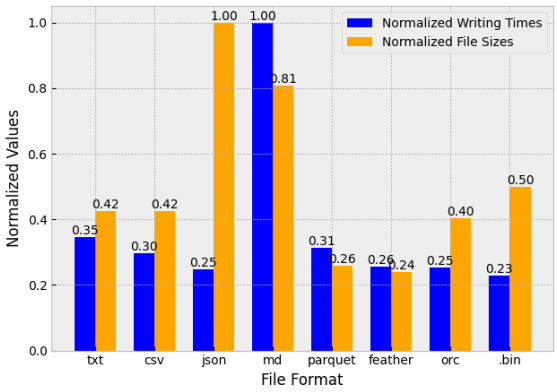


(a) `SBIN`

(b) `HDFC`



(c) `AXISBANK`



(d) `BAJFINANCE`

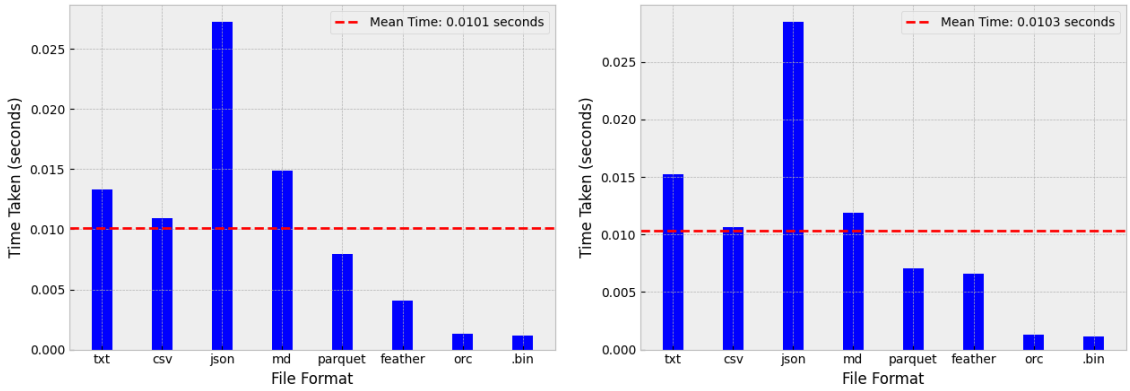
Figure 1: Double-bar graphs depicting normalised writing times and space occupied by different file formats

2.2 Reading Times

Time taken to read the contents is an important efficiency aspect of a file format used to store large amount of data. To ensure uniformity, all data was converted to panda dataframes(including `.parquet` and `.orc` tables, although it was observed that doing so using the `to_pandas()` function created negligible difference in the reading times. The following inferences were drawn:

- `.bin` has the least reading time of any format.
- `.feather` and `.orc` have the second least reading times with the latter being slightly better. Though both store the data in tabular row-column format, the difference may be attributed to better compression algorithms and encoding scheme of `.orc` format.
- `.txt`, `.csv`, `.md` formats are generally concentrated around the mean.

- `.parquet` performs slightly better than the mean but is still nowhere close to `.feather`. This may be attributed to one or more of the following differences:
  - Parquet files have additional metadata and structure compared to feather than Feather.
  - Feather supports memory mapping, which allows the data to be directly mapped to memory without the need for decompression.
  - Feather has a simpler data type system compared to Parquet.
- `.json` performs the worst. This may be due to:
  - Plain textual representation to make it human readable
  - Lack of native compression algorithm
  - Limited columnar storage



(a) `SBIN` (b) `HDFC`

(c) `AXISBANK` (d) `BAJFINANCE`

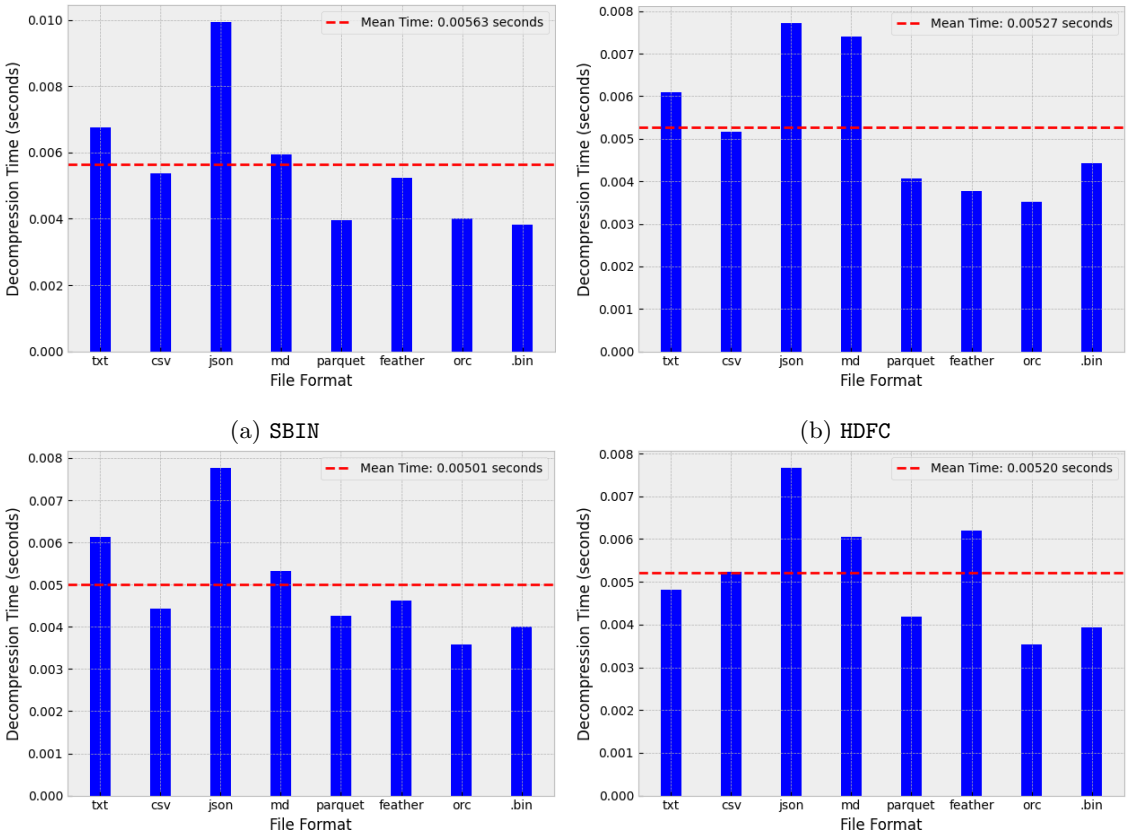
Figure 2: Bar graphs depicting reading times of different file formats

2.3 Decompression Times

Due to the large volume of data, most files are stored in a compressed format. Therefore, unpacking time must be considered during data retrieval. From the plots on the next page, the inferences are as follows:

- `.orc` has the best decompression time which may be attributed to its row-columnar storage and decompression algorithm. `.bin` is a close second in most cases.

- `.parquet`, `.feather` and `.bin` have similar optimum performances (which may vary under different circumstances the reason for which is not clear).However, `.feather` occasionally touches the mean decompression time.
- `.csv` has average performance and is only slightly worse than `.parquet`.
- `.json` and `.md` fair the worst.



(a) `SBIN` (b) `HDFC`

(c) `AXISBANK` (d) `BAJFINANCE`

Figure 3: Bar graphs depicting decompression times of different file formats

3 Conclusion

In view of the above experiments, `.feather` and `.orc` should be preferred over `.bin` even though the latter performs slightly better in most aspects. If we encounter a `.bin` file, we would typically need additional information or documentation to understand its contents and structure. On the other hand, a `..feather` file is a specific format optimized for columnar data interchange and is designed to be easily readable and writable across different programming languages, particularly with libraries like Apache Arrow and Pandas.

4 Reference

I found the following website to be useful and instructive for plotting graphs:
https://matplotlib.org/stable/users/index.html/