Before Data Science Application
- How "Big" is Big Data?

The Global Healthcare
Data Science Community

Contact Channels
UK +44 1843 609600
office@phuse.global
phuse.global

Social Media
@phusetwitta
/phusebook
/phusetube
/company/phuse

Xiaochang Liu (刘晓畅)
Data Engineer
Janssen R&D, C&SP, China
Johnson & Johnson (China) Investment Ltd

Disclaimer

- “The views and opinions expressed in the presentation and on the following slides are solely on my own, and do not necessarily reflect the views of Johnson & Johnson.”

Agenda

- Introduction to Big Data
- Big Data(frame)
- R & Python: We might not fit for Big Data
- Library Benchmarking

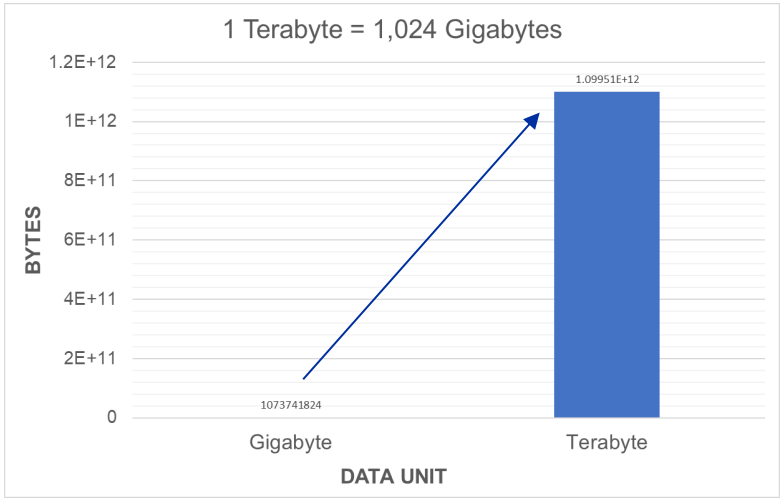
Introduction to Big Data – 3 "V"s

- **Volume**: "**High** volumes of low-density, unstructured data. For some organizations, this might be tens of terabytes of data. For others, it may be hundreds of petabytes."
- **Velocity**: "Velocity is the **fast** rate at which data is received and (perhaps) acted on."
- **Variety**: "Variety refers to the **many** types of data that are available."

- From Oracle's "What is Big Data"

Introduction to Big Data – How "Big"

(1999) **1 gigabyte** → *(2022)* **1 terabyte**



When would I feel the data are "Big"?

"**WHY** is my program running so **SLOWLY**?"

Software

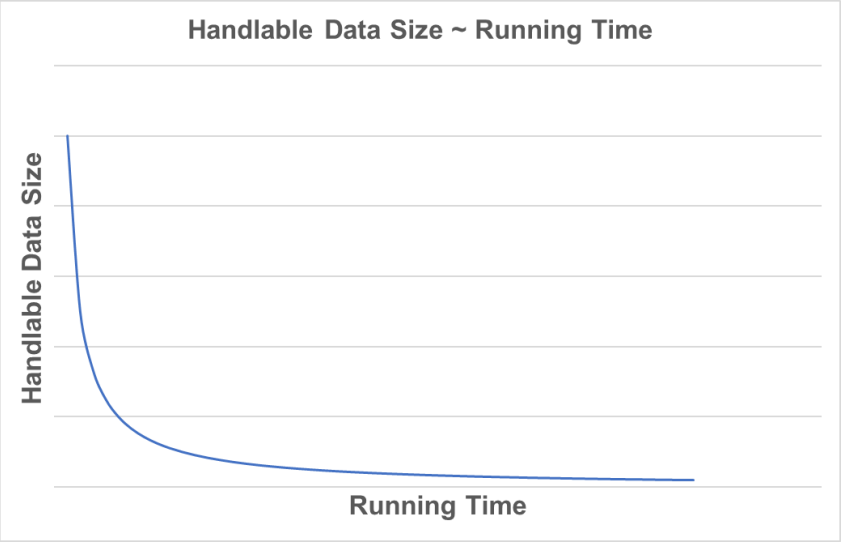
Hardware

Patience

"...alright, the data are BIG enough"

Introduction to Big Data – How to be "Big"

The **longer** time we wait, the **less** size of data we think the program can handle.



What limits the **"Big"** (What slows down our program)?

1. **Hardware:**
   - CPU
   - GPU
   - RAM
   - Storage
2. **Software:**
   - Underlying programming language
   - Programming Skills

Big Data(frame)

- **Tabular Dataset**
- **Lots of rows**

shape: (10000000, 13)

id --- i64	name --- str	address --- str	birth_date --- str	...	pet_age --- i64	phone_number --- str	height_cm --- i64	weight_kg --- i64
1	Alexander Oliver	3900 Torres Garden Suite 768 Har...	2018-04-26	...	81	234-667-2415x35374	214	86
2	Raymond Robinson	77684 Butler Isle Apt. 488 Flore...	2016-03-26	...	76	597.317.1480x2353	59	111
3	Derrick Jordan	6535 Frances Dam Suite 192 East ...	1989-04-23	...	35	712-503-8251x90940	69	98
4	Tyler Cooper	985 Cabrera Cliff Apt. 920 New J...	2018-08-01	...	78	016-991-9341x8552	186	200
...	...	...	...	...	...	...	...	...
7499997	Bryan Williams	1715 Rice Manor Apt. 916 South R...	2011-01-28	...	61	273.601.9163	215	46
7499998	Michelle Peterson	USS Wade FPO AA 75646	1999-08-30	...	94	622.334.2876x295	210	82
7499999	Sheryl Castaneda	13698 Strickland Course Suite 20...	2010-08-01	...	28	404.894.2617	110	133
7500000	Stephen White	598 Hayden Radial Apt. 362 Donna...	2010-01-16	...	82	640-493-7628x9736	219	135

- **Storage:**

1. ***.csv** (pure text)
2. *.sas7bdat (binary)
3. *.xlsx (zip of xml files)
4. *.rds/*.rdata (binary)
5. ***.feather/*.arrow** (binary)

R & Python: We might not fit for "Big" Data

R Language

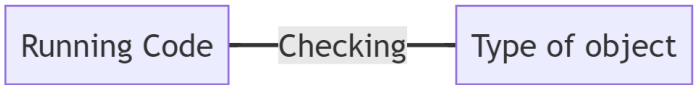
- Designed for Data analysis/Statistical Computing
- Functional programming language *(Adler, 2012)*
- Dynamically typed language
- Interpreted language *(Adler, 2012)*

Python

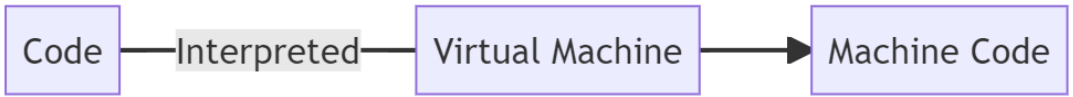
- Designed for general purpose
- Object-oriented programming language
- Dynamic typed language
- Interpreted language

What is **Dynamically typed language** and **Interpreted language**?

- **Dynamically typed language**:



- **Interpreted language**:



Reference: Adler, J., 2012. *R in a Nutshell, 2nd Edition*. [online] O'Reilly Online Learning. Available at: <https://www.oreilly.com/library/view/r-in-a/9781449358204/> [Accessed 16 October 2022].

Library Benchmarking – Libraries

Pandas (From Python)

- ✓ Built upon Python
- ✓ Designed to make data analysis easier
- ✓ Toolbox of Data Analysis

dplyr (From R)

- ✓ Provide clear and easy-to-learn grammar of data manipulation. E.g., mutate(), select()
- ✓ Combined with pipe operator (%>%) from magrittr: Super!

Polars (From Python)

- ✓ **Multi-threaded**
- ✓ Built upon Arrow2 with Rust
- ✓ Designed as lightning-fast DataFrame Library
- ✓ High performance

data.table (From R)

- ✓ **Multi-threaded**
- ✓ High performance

Library Benchmarking – Environment

If we have a laptop with this specification:

- CPU: Intel® Core™ i5-8365U CPU @ 1.60Hz
- RAM: 16 GB
- GPU: Not Available

Software environment:

- OS: Windows 10
- Python: version 3.10.4
- R: version 4.2.1
- Pandas: version 1.5.0
- Polars: version 0.14.19
- data.table: version 1.14.2
- dplyr: version 1.0.10

Library Benchmarking – Data Inspection

N of rows	Data Size	File Type	Sorted?
10,000	1.5 MB	csv	N
100,000	15.2 MB	csv	N
1,000,000	153.3 MB	csv	N
10,000,000	1.5 GB	csv	N
10,000	2.2 MB	feather	N
100,000	22.5 MB	feather	N
1,000,000	224.6 MB	feather	N
10,000,000	2.2 GB	feather	N

Each Dataset has columns:

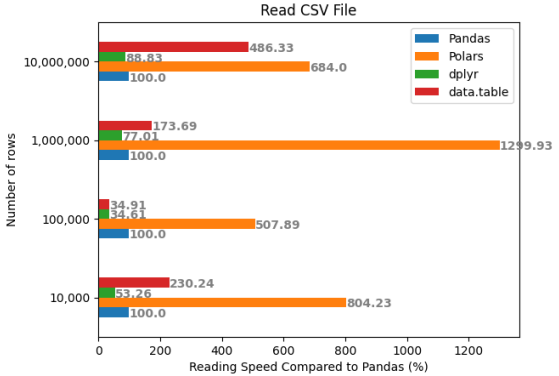
1. id
2. name
3. address
4. birth_date
5. sex
6. country
7. favorite_color
8. company
9. pet_name
10. pet_age
11. phone_number
12. height_cm
13. weight_kg

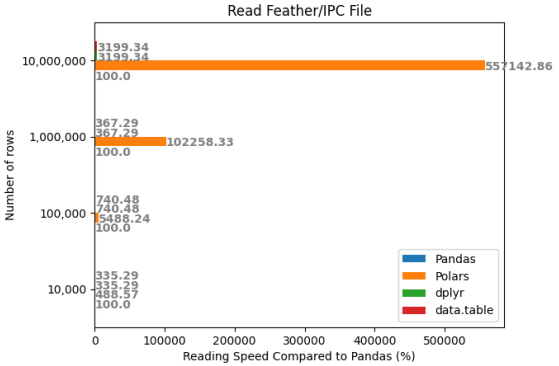
Library Benchmarking – Load Data

Task	File type	N of rows	Pandas	Polars	dplyr	data.table
Load Data	csv	10,000	0.0571s	0.0071s	0.1072s	0.0248s
	csv	100,000	0.2768s	0.0545s	0.7997s	0.7928s
	csv	1,000,000	5.8692s	0.4515s	7.6217s	3.3792s
	csv	10,000,000	87.2215s	12.7516s	98.1847s	17.9347s
	feather	10,000	0.0171s	0.0035s	0.0051s	0.0051s
	feather	100,000	0.0933s	0.0017s	0.0126s	0.0126s
	feather	1,000,000	1.2271s	0.0012s	0.3341s	0.3341s
	feather	10,000,000	15.6s	0.0028s	0.4876s	0.4876s

Note:

- Each test iterated for 10 times, and the longest running time was recorded.
- For dplyr, utils::read.csv was used to read csv file
- For both dplyr and data.table, arrow::read_ipc_file was used to read feather file



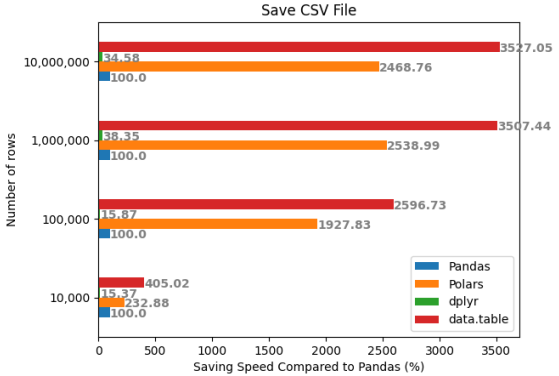


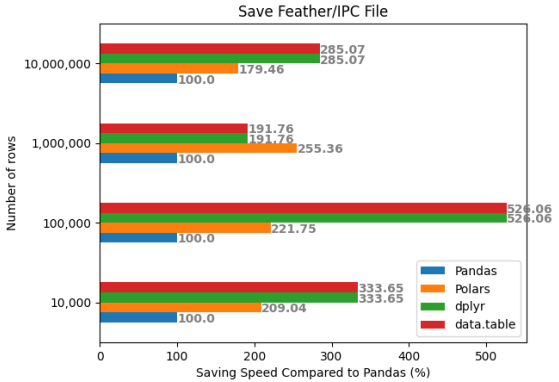
Library Benchmarking – Save Data

Task	File type	N of rows	Pandas	Polars	dplyr	data.table
Load Data	csv	10,000	0.1211s	0.052s	0.7877s	0.0299s
	csv	100,000	1.5866s	0.0823s	9.998s	0.0611s
	csv	1,000,000	18.4737s	0.7276s	48.1651s	0.5267s
	csv	10,000,000	191.9211s	7.774s	555.0487s	5.4414s
	feather	10,000	0.0347s	0.0166s	0.0104s	0.0104s
	feather	100,000	0.0989s	0.0446s	0.0188s	0.0188s
	feather	1,000,000	1.6877s	0.6609s	0.8801s	0.8801s
	feather	10,000,000	13.1897s	7.3498s	4.6269s	4.6269s

Note:

- Each test iterated for 10 times, and the longest running time was recorded.
- For dplyr, utils::write.csv was used to write csv file
- For both dplyr and data.table, arrow::write_feather was used to write feather file





Library Benchmarking – Sort Data

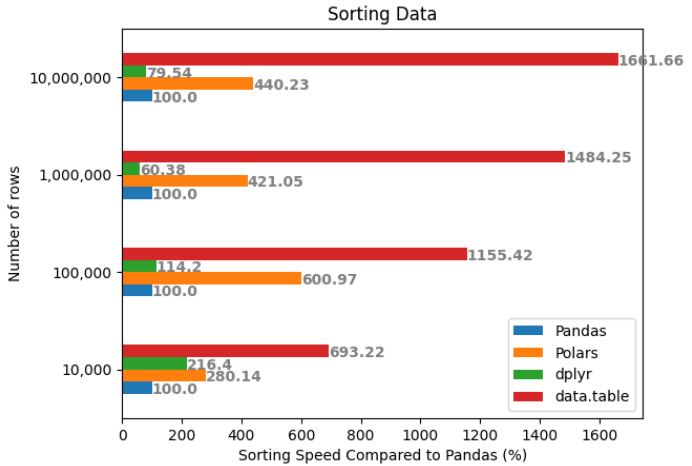
Each Dataset will be sorted by:

1. weight_kg
2. height_cm
3. birth_date
4. pet_age
5. name
6. favorite_color
7. address
8. sex
9. country
10. company
11. pet_name
12. phone_number
13. id

Note:

- Each test iterated for 10 times, and the longest running time was recorded.

Task	File type	N of rows	Pandas	Polars	dplyr	data.table
Load Data	csv	10,000	0.0409s	0.0146s	0.0189s	0.0059s
	csv	100,000	0.3732s	0.0621s	0.3268s	0.0323s
	csv	1,000,000	4.2791s	1.0163s	7.0874s	0.2883s
	csv	10,000,000	95.5053s	21.6943s	120.0727s	5.7476s



Library Benchmarking – Subset Data

Each Dataset will be subset by:

name has “Anna”

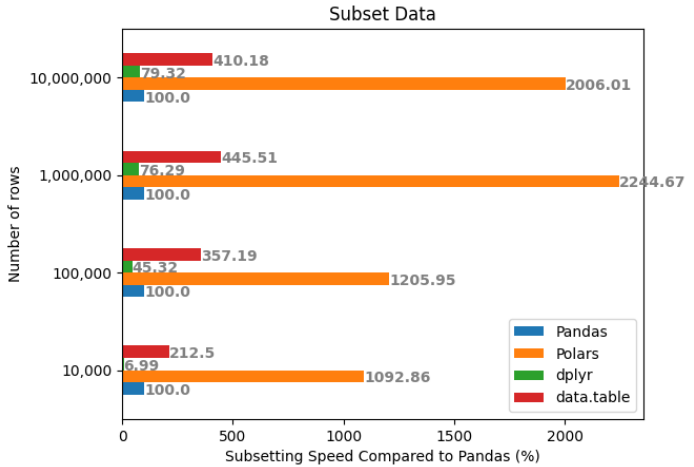
And

(153 <= height_cm <= 168 or pet_name has “Justin”)

Note:

- Each test iterated for 10 times, and the longest running time was recorded.

Task	File type	N of rows	Pandas	Polars	dplyr	data.table
Load Data	csv	10,000	0.0153s	0.0014s	0.2188s	0.0072s
	csv	100,000	0.1013s	0.0084s	0.2235s	0.02836s
	csv	1,000,000	1.578s	0.0703s	2.0684s	0.3542s
	csv	10,000,000	15.0752	0.7515s	19.0065s	3.6753s



Library Benchmarking – Group Data

Each Dataset will be grouped by:

1. name
2. pet_name
3. company

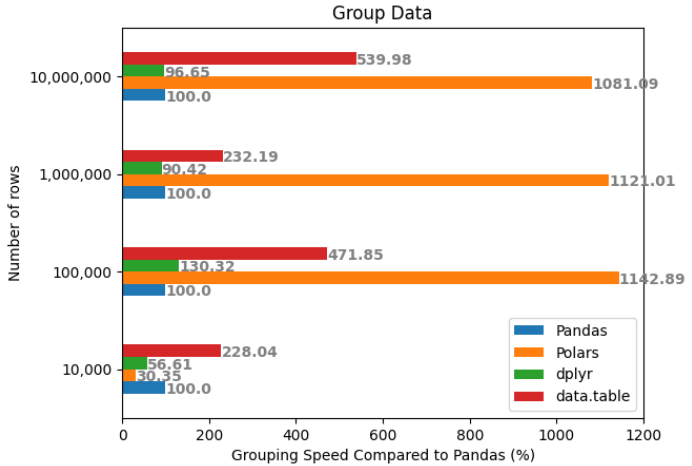
And return

1. the mean value of “height_cm”
2. the max value of “weight_kg”
3. the number of unique “birth_date” value

Note:

- Each test iterated for 10 times, and the longest running time was recorded.

Task	File type	N of rows	Pandas	Polars	dplyr	data.table
Load Data	csv	10,000	0.0224s	0.0804s	0.0431s	0.0107s
	csv	100,000	0.5063s	0.0443s	0.3885s	0.1073s
	csv	1,000,000	3.7778s	0.337s	4.1781s	1.627s
	csv	10,000,000	55.4331s	5.1275s	57.3557s	10.2658s



Summary – What should we use?

From the perspective of **performance**:

- data.table (From R)

- Polars (From Python)

- *.feather

From the perspective of **functionality/difficulty**:

- dplyr (From R)

- Pandas (From Python)

- *.csv

Thank You

Xiaochang Liu (刘晓畅)
Data Engineer
Janssen R&D, C&SP, China
Johnson & Johnson (China) Investment Ltd