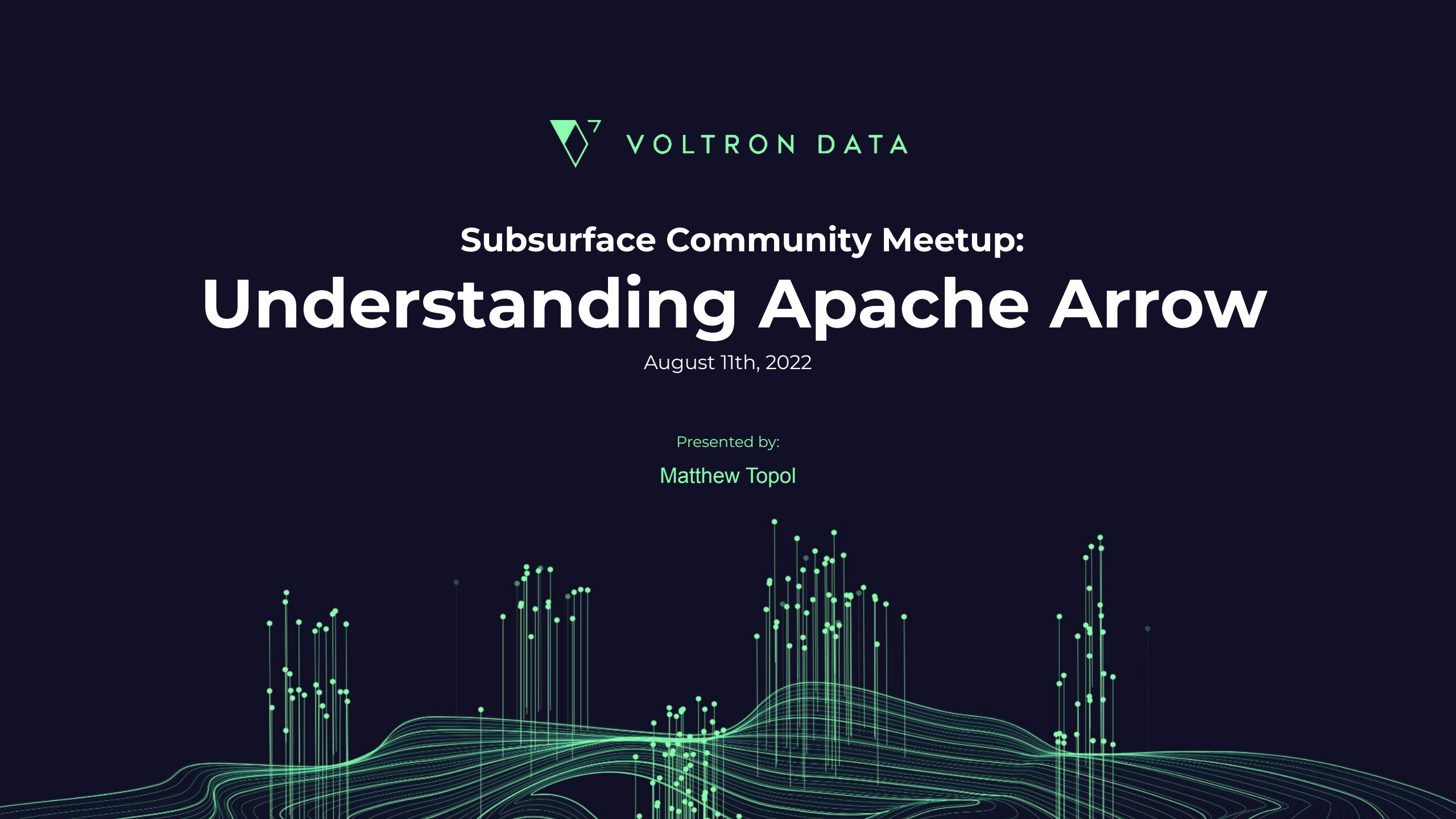

VOLTRON DATA
Subsurface Community Meetup:
Understanding Apache Arrow
August 11th, 2022
Presented by:
Matthew Topol

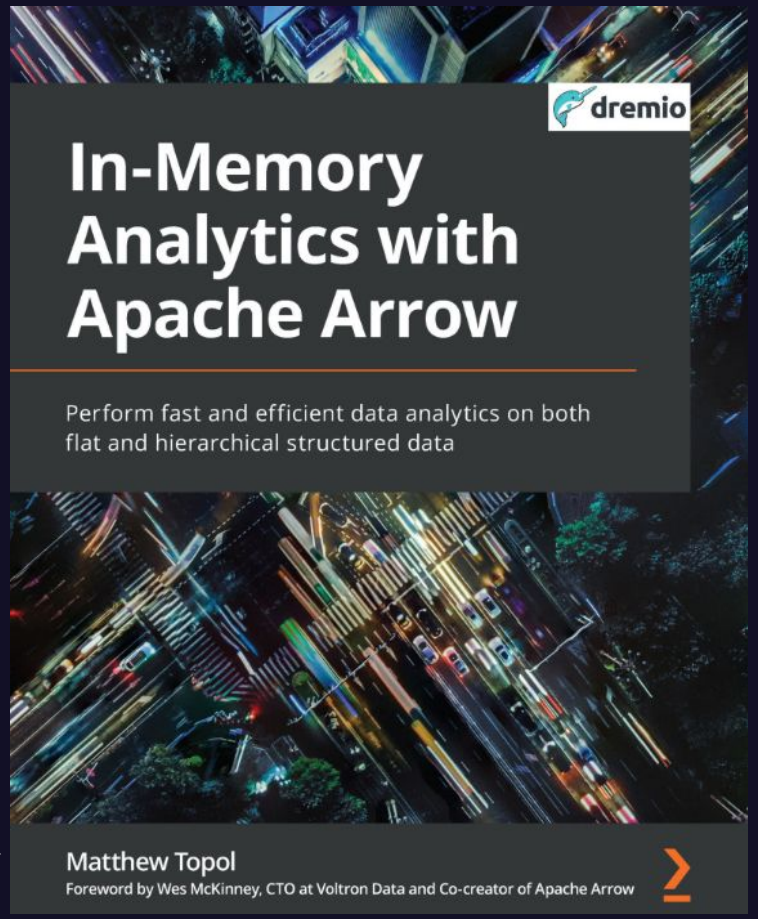
Who am I?

Email

matt@voltrondata.com

Author of

“In-Memory Analytics with Apache Arrow”



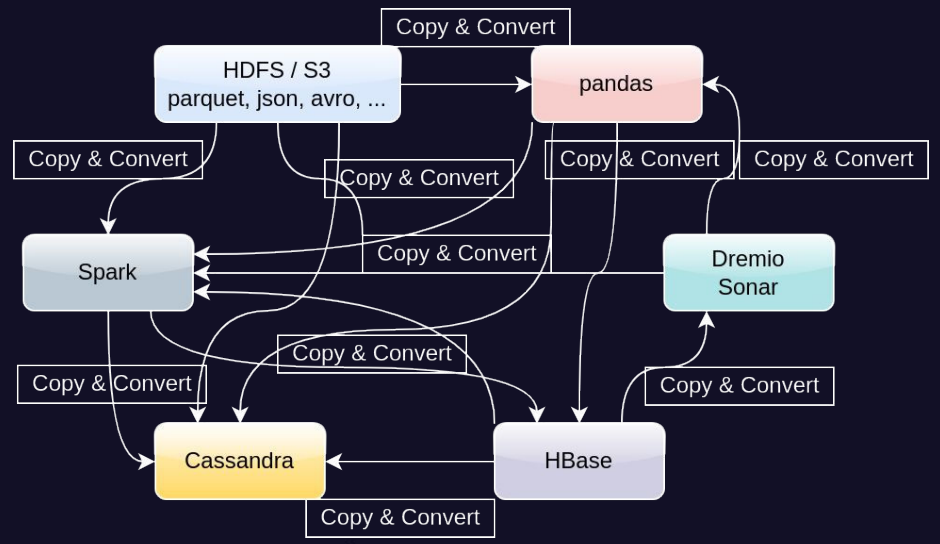
Staff Software Engineer at Voltron Data
Committer on Apache Arrow repository

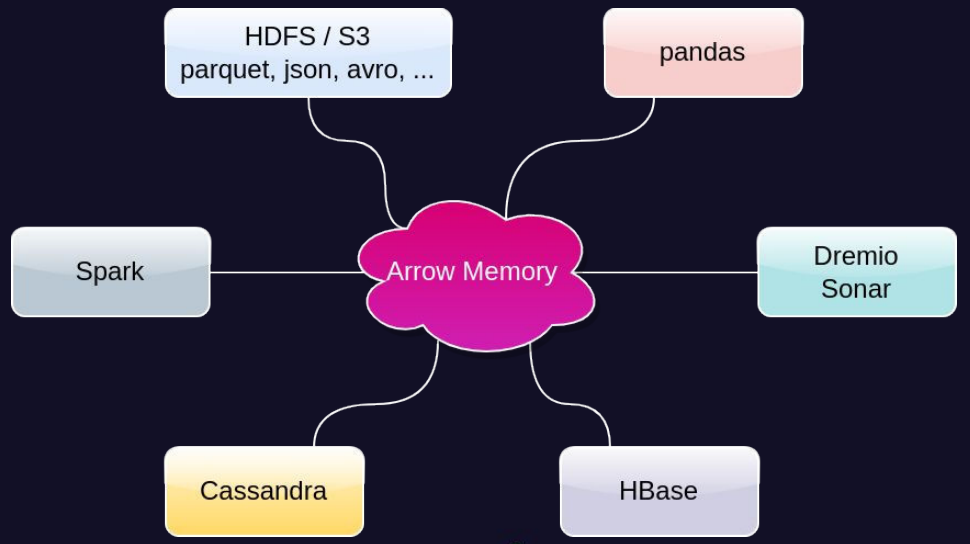
APACHE ARROW

A quick primer!

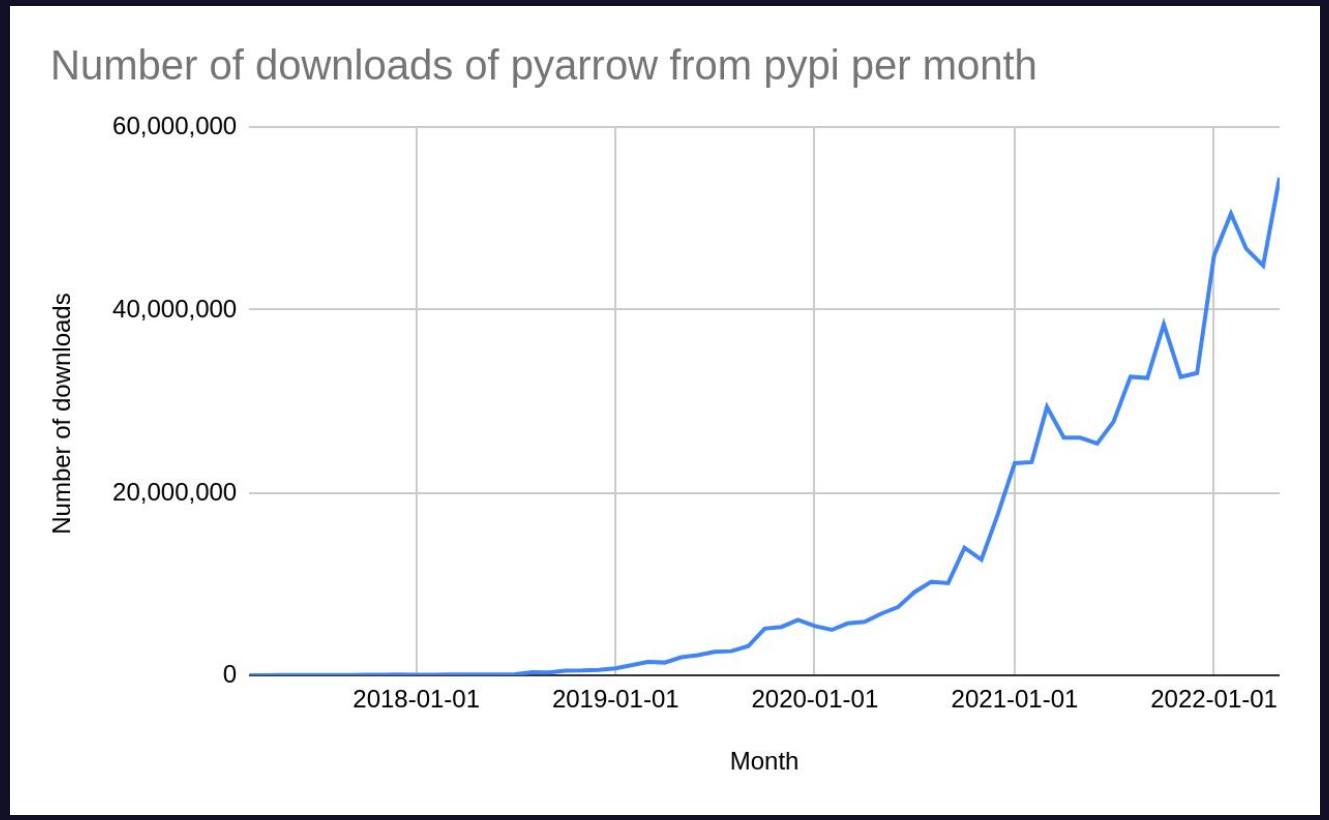
https://arrow.apache.org

- High Performance In-Memory Columnar Format
- No Data Serialization/Deserialization
- Implementations exist in many languages!





Arrow Adoption



Arrow is increasingly being adopted across the Data Science / Analytics ecosystem

What is Columnar?

Table of Data

	ARCHER	LOCATION	YEAR
ROW 1	Legolas	Mirkwood	1954
ROW 2	Oliver	Star City	1941
ROW 3	Merida	Scotland	2012
ROW 4	Lara	London	1996
ROW 5	Artemis	Greece	-600

Row Oriented Memory Buffer

Row 1	Legolas
	Mirkwood
	1954
Row 2	Oliver
	Star City
	1941
Row 3	Merida
	Scotland
	2012
Row 4	Lara
	London
	1996
Row 5	Artemis
	Greece
	-600

Arrow Columnar Memory Buffer

archer	Legolas
	Oliver
	Merida
	Lara
	Artemis
location	Mirkwood
	Star City
	Scotland
	London
	Greece
year	1954
	1941
	2012
	1996
	-600

Why Columnar?

Memory Locality
I/O
Vectorization

A **All Archers in Europe**

Only need two columns! (Archer, Location)

1. *Spin through Locations for indexes*
2. *Get Archers at those indexes*

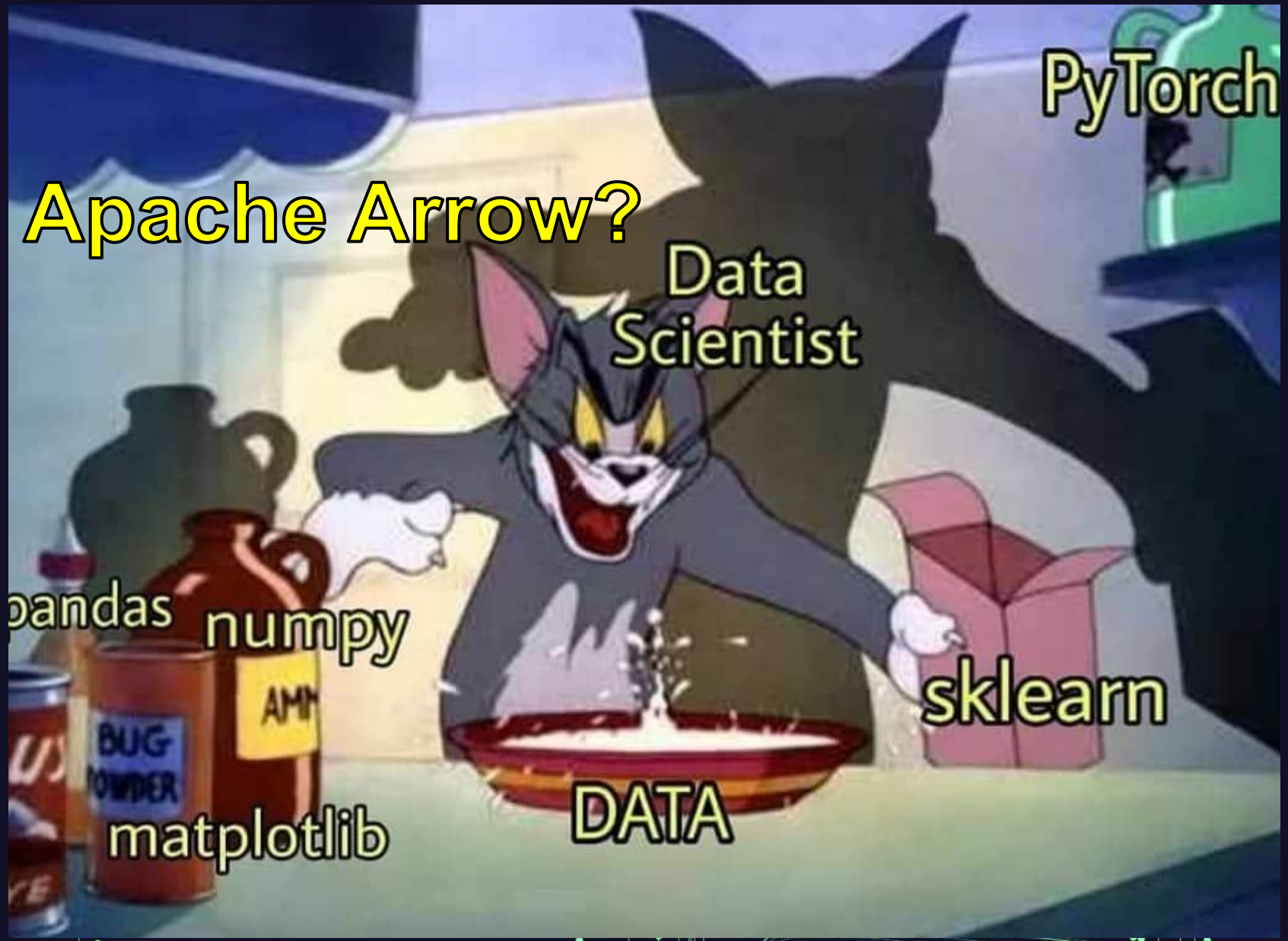
Less I/O, Lower Memory usage, Fewer page faults

B **Calculate Mean for Year column**

Only need the one column! (Year)

1. *Vectorized operations require contiguous memory*
2. *Our column is already contiguous memory!*

Significantly faster computation!


Apache Arrow?
PyTorch
Data
Scientist
pandas
numpy
sklearn
DATA
matplotlib

Interoperability

Arrow slots easily into existing popular tools

pandas

pyarrow easily converts to and from pandas Data Frames
In some cases this can be done with zero copies

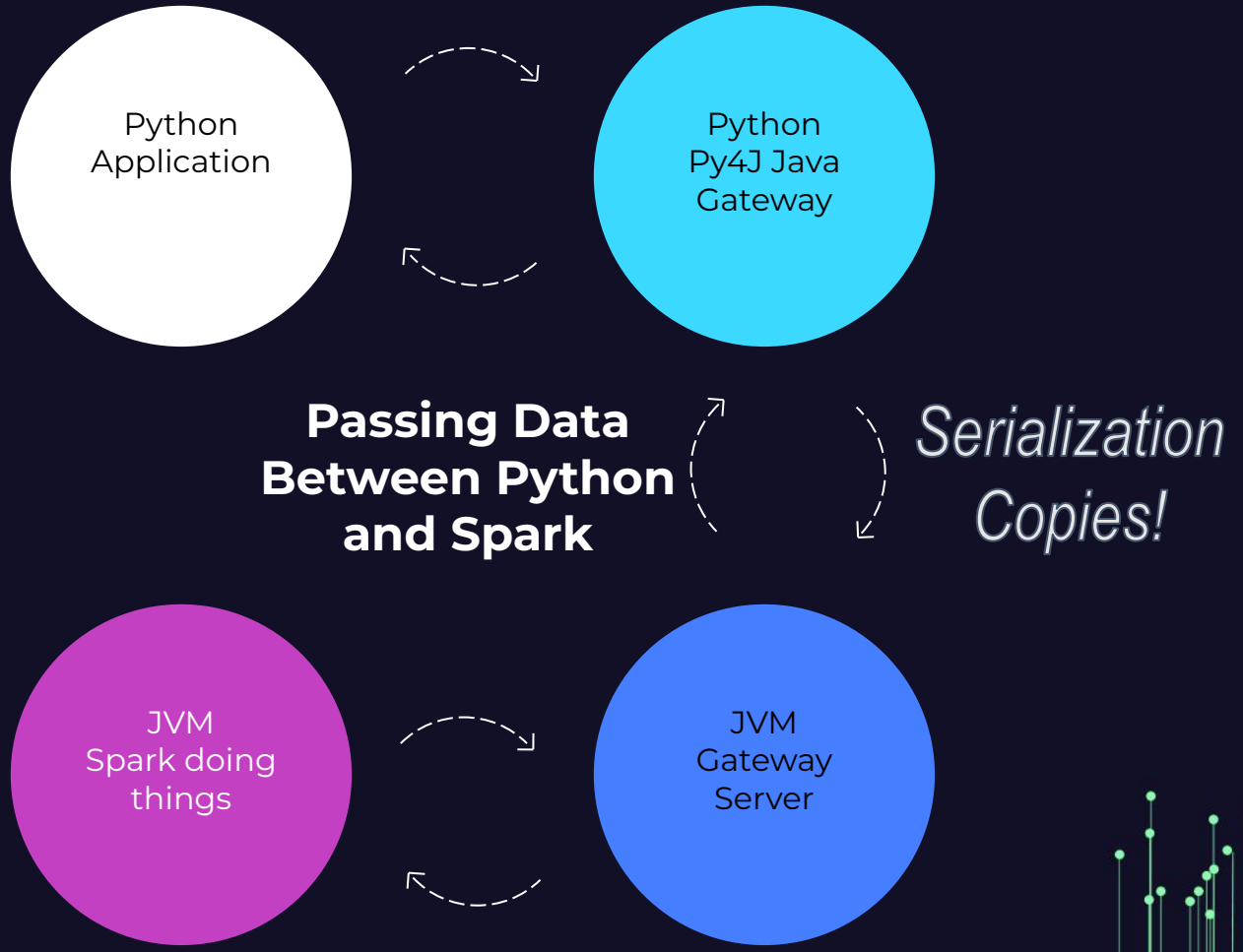
NumPy

pyarrow can easily convert between Arrow Arrays and NumPy without copying the memory

Example Case: Apache Spark

Data gets copied several times!

1. Read 4GB CSV into pandas DataFrame
2. Py4J Java Gateway serializes it (copy)
3. JVM Gateway server deserializes it (copy)
4. Spark does stuff, then sends the results back, serializing and deserializing it again.

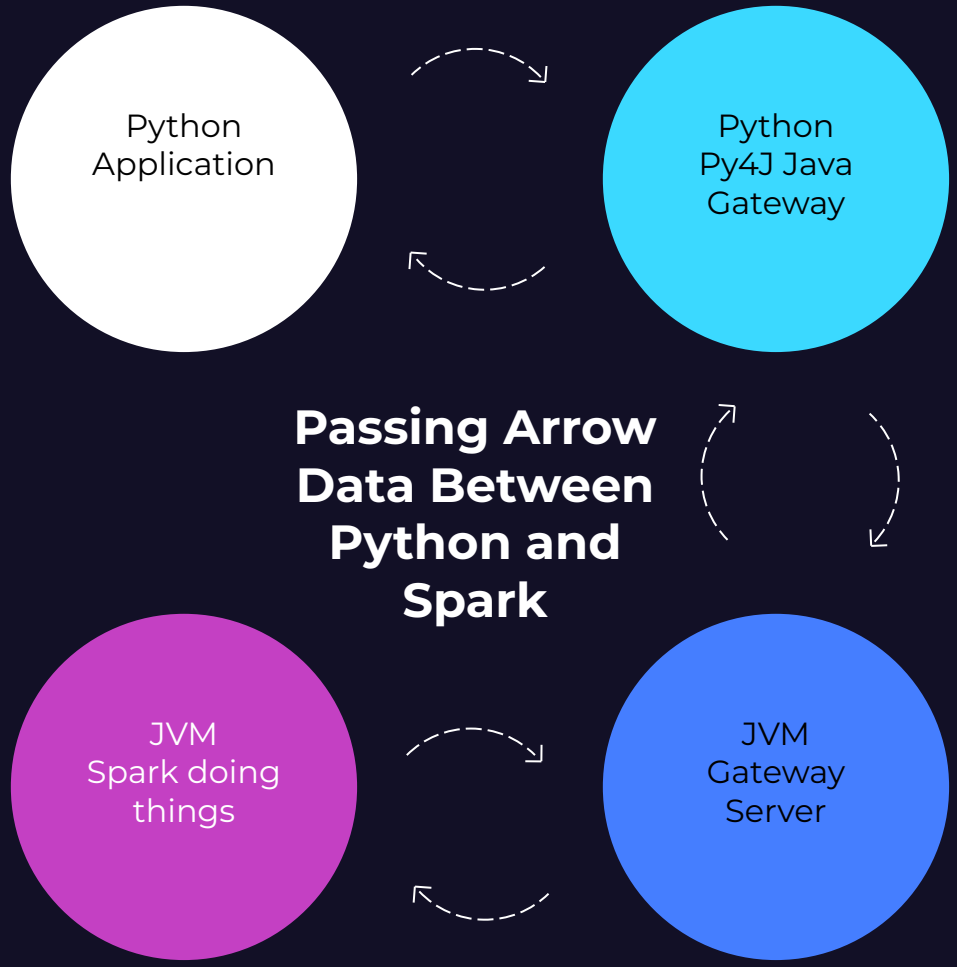


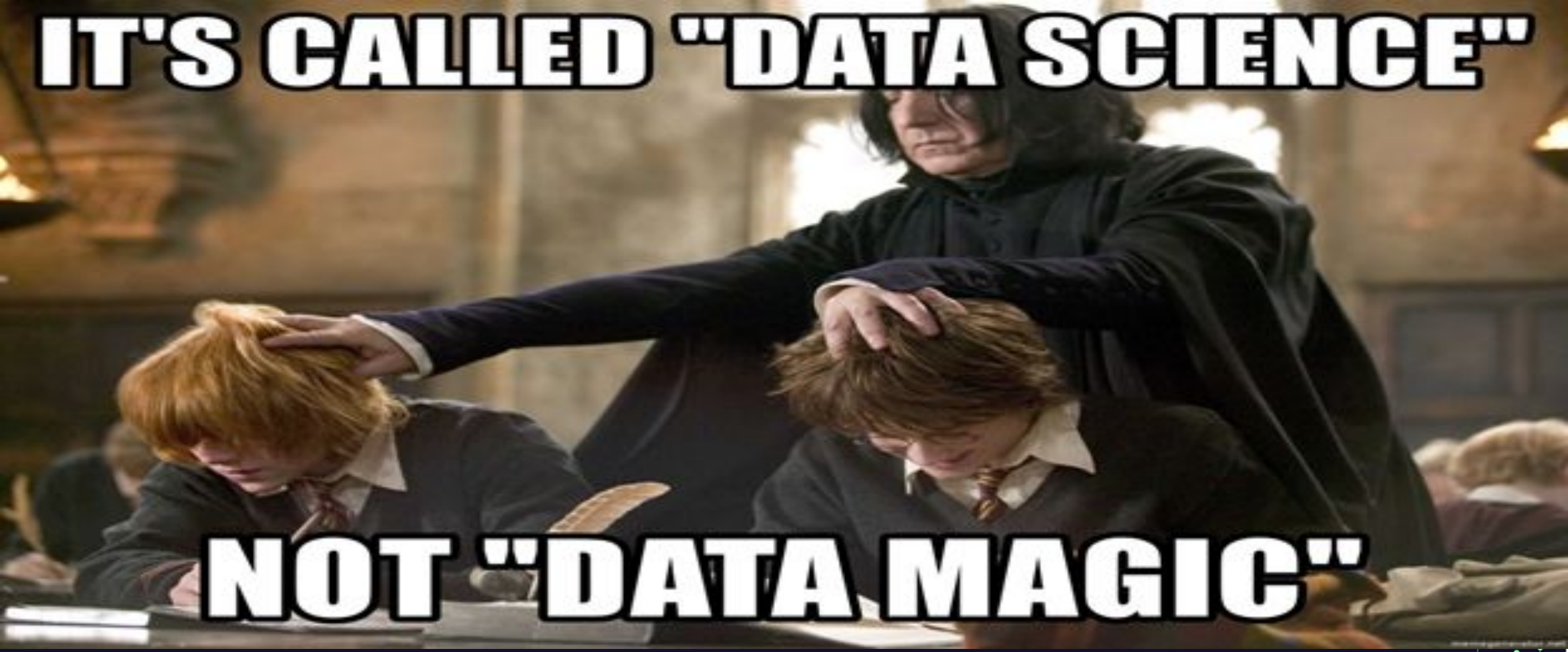
Example Case: Apache Spark

Using Arrow, no need to copy the data!

1. Read 4GB CSV into Arrow Record Batches
2. Py4J Java Gateway can send as Arrow IPC
3. JVM Gateway receives Arrow IPC
4. Spark does stuff, then sends the results back using Arrow IPC

Working Example in my book!





What use cases benefit from Apache Arrow?

Why use Arrow vs other data formats?

Let’s (Over-)Simplify!

Data Formats

- **Size** → Final size of the data representation
- **Serialize / Deserialize Speed** → Performance for converting data between the format and something that can be used in-memory for computation
- **Ease of Use** → Catch-all category: readability, compatibility, features, etc...

Data Format Categories

Long-Term or Persistent Storage

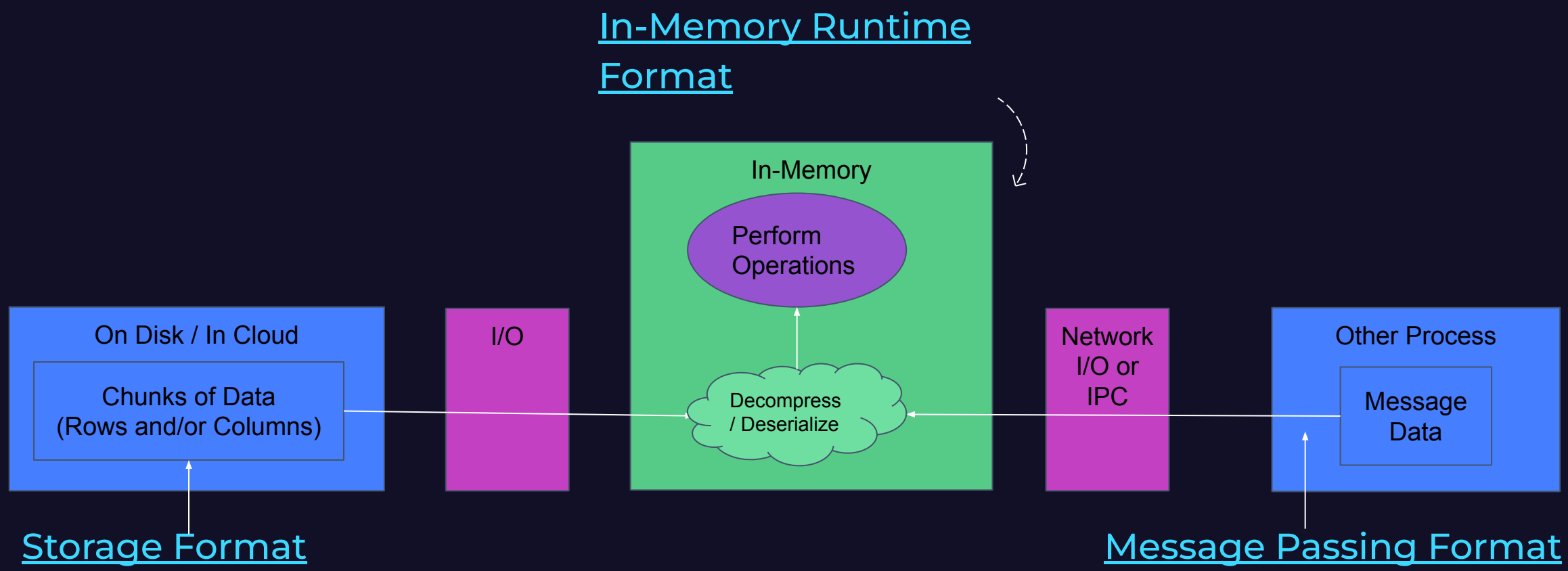
Runtime In-Memory Processing

Message Passing

Yes, these are extremely broad. That's intentional!

Data Format Categories

Relationships



Example: Memory Mapping for Efficiency

Let’s read one column from a dataset with millions of rows

1. 1.8 GB CSV file

   pandas.read_csv

2. 1.8 GB CSV file

   pyarrow.csv.read_csv

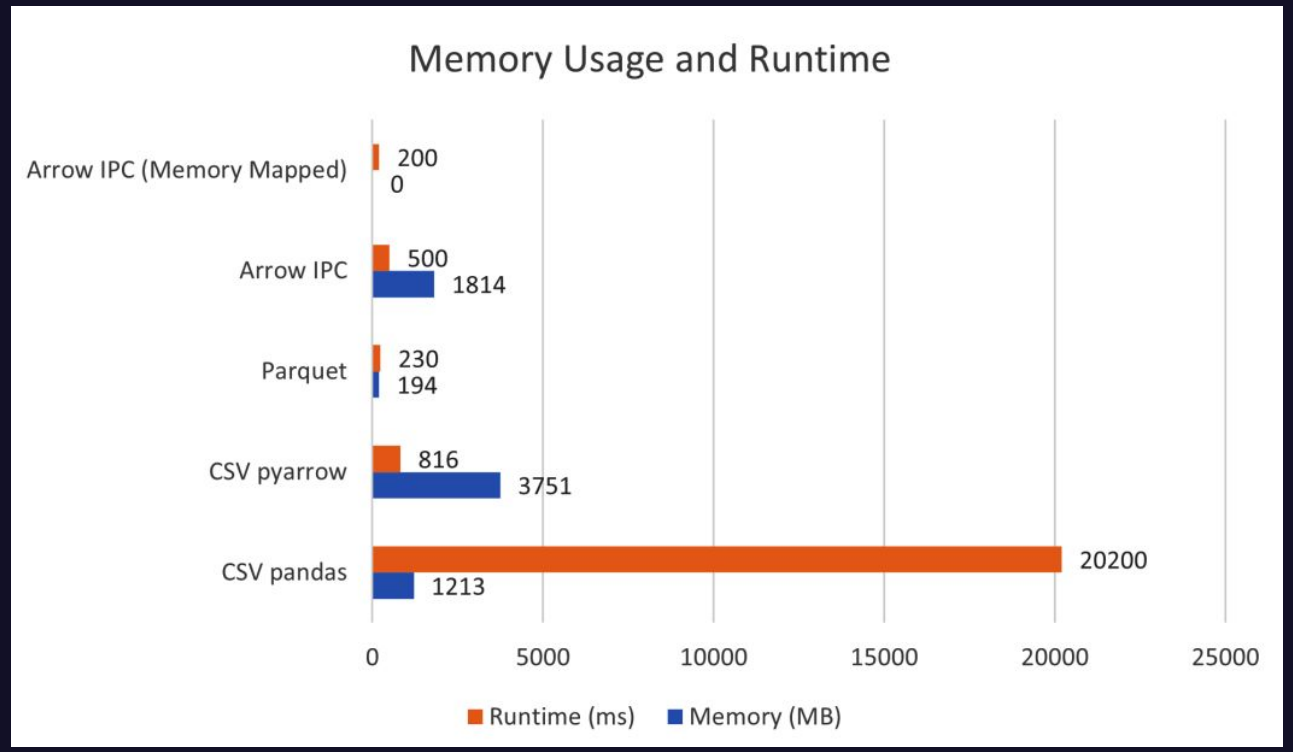
3. 286 MB Parquet file

   pyarrow.parquet

4. 1.77 GB Arrow IPC file

   mmap with pyarrow

We can measure runtime and memory usage to read the column and create a pandas Data Frame



Read 1 column

Using different formats and methods

Full code and example in my book

To Summarize!

Why should you use Arrow?

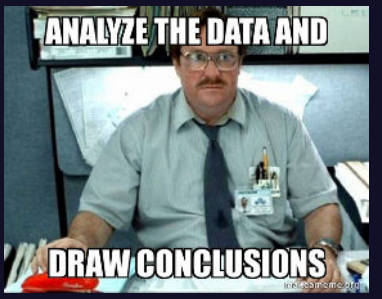
Apache Arrow

Data Transport

- Arrow supports complex types!
- Arrow IPC format is fast and efficient
- Flight / FlightSQL RPC
- Streaming Data

Tabular Analytics

- Columnar representation leads to fast computations!
- Takes advantage of Vectorization
- Includes Acero an Arrow-native compute engine
- Offers C Data API for interoperability



Q/A