

# Vaex

Out of core dataframes

**Maarten Breddels**

Freelancer / consultant

Pandas Summit @ London - 2019

# Maarten Breddels



@maartenbreddels



maartenbreddels@gmail.com



github.com/maartenbreddels




www.maartenbreddels.com

# Maarten Breddels

 @maartenbreddels

 maartenbreddels@gmail.com

 github.com/maartenbreddels


 www.maartenbreddels.com

- Ex: astronomer (working on software for big data: vaex)

# Maarten Breddels

 @maartenbreddels

 maartenbreddels@gmail.com

 github.com/maartenbreddels

 www.maartenbreddels.com

- Ex: astronomer (working on software for big data: vaex)
- Now: Freelancer / consultant / data science for Python / Jupyter

# Maarten Breddels

 @maartenbreddels

 maartenbreddels@gmail.com

 github.com/maartenbreddels

 www.maartenbreddels.com

- Ex: astronomer (working on software for big data: vaex)
- Now: Freelancer / consultant / data science for Python / Jupyter
- Like: Python, Javascript, C(++)

# Maarten Breddels

 @maartenbreddels

 maartenbreddels@gmail.com

 github.com/maartenbreddels

 www.maartenbreddels.com

- Ex: astronomer (working on software for big data: vaex)
- Now: Freelancer / consultant / data science for Python / Jupyter
- Like: Python, Javascript, C(++)
- Code: Core Jupyter-Widgets developer, authors of ipyvolume and vaex

# Maarten Breddels

 @maartenbreddels

 maartenbreddels@gmail.com

 github.com/maartenbreddels

 www.maartenbreddels.com

- Ex: astronomer (working on software for big data: vaex)
- Now: Freelancer / consultant / data science for Python / Jupyter
- Like: Python, Javascript, C(++)
- Code: Core Jupyter-Widgets developer, authors of ipyvolume and vaex
- QuantStack (Sylvain Corlay)

# Vaex

- Intro (2 minutes)
- Demo vaex (5 minutes)
- Questions / discussion





- Why am I here?

- Why am I here?
  - Author of vaex

- Why am I here?
  - Author of vaex
- What is vaex?

- Why am I here?
  - Author of vaex
- What is vaex?
  - Was: a visualization tool (Qt), 1 billion rows/second

- Why am I here?
  - Author of vaex
- What is vaex?
  - Was: a visualization tool (Qt), 1 billion rows/second
  - Now

- Why am I here?
  - Author of vaex
- What is vaex?
  - Was: a visualization tool (Qt), 1 billion rows/second
  - Now
    - A dataframe library (vaex-core, MIT License)

- Why am I here?
  - Author of vaex
- What is vaex?
  - Was: a visualization tool (Qt), 1 billion rows/second
  - Now
    - A dataframe library (vaex-core, MIT License)
      - lazy expressions / no memory copies



- Why am I here?
  - Author of vaex
- What is vaex?
  - Was: a visualization tool (Qt), 1 billion rows/second
  - Now
    - A dataframe library (vaex-core, MIT License)
      - lazy expressions / no memory copies
      - Efficient io/memory mapping/columnar storage

- Why am I here?
  - Author of vaex
- What is vaex?
  - Was: a visualization tool (Qt), 1 billion rows/second
  - Now
    - A dataframe library (vaex-core, MIT License)
      - lazy expressions / no memory copies
      - Efficient io/memory mapping/columnar storage
      - Strong emphasis on statistics on N-d grids

- Why am I here?
  - Author of vaex
- What is vaex?
  - Was: a visualization tool (Qt), 1 billion rows/second
  - Now
    - A dataframe library (vaex-core, MIT License)
      - lazy expressions / no memory copies
      - Efficient io/memory mapping/columnar storage
      - Strong emphasis on statistics on N-d grids
      - 1 000 000 000 rows/second

- Why am I here?
  - Author of vaex
- What is vaex?
  - Was: a visualization tool (Qt), 1 billion rows/second
  - Now
    - A dataframe library (vaex-core, MIT License)
      - lazy expressions / no memory copies
      - Efficient io/memory mapping/columnar storage
      - Strong emphasis on statistics on N-d grids
      - 1 000 000 000 rows/second
    - Viz tools (vaex-viz, vaex-jupyter)

- Why am I here?
  - Author of vaex
- What is vaex?
  - Was: a visualization tool (Qt), 1 billion rows/second
  - Now
    - A dataframe library (vaex-core, MIT License)
      - lazy expressions / no memory copies
      - Efficient io/memory mapping/columnar storage
      - Strong emphasis on statistics on N-d grids
      - 1 000 000 000 rows/second
    - Viz tools (vaex-viz, vaex-jupyter)
    - Hdf5 and arrow support (vaex-hdf5, vaex-arrow)

- Why am I here?
  - Author of vaex
- What is vaex?
  - Was: a visualization tool (Qt), 1 billion rows/second
  - Now
    - A dataframe library (vaex-core, MIT License)
      - lazy expressions / no memory copies
      - Efficient io/memory mapping/columnar storage
      - Strong emphasis on statistics on N-d grids
      - 1 000 000 000 rows/second
    - Viz tools (vaex-viz, vaex-jupyter)
    - Hdf5 and arrow support (vaex-hdf5, vaex-arrow)
    - ML without pipelines (in progress, vaex-ml)

- Why am I here?
  - Author of vaex
- What is vaex?
  - Was: a visualization tool (Qt), 1 billion rows/second
  - Now
    - A dataframe library (vaex-core, MIT License)
      - lazy expressions / no memory copies
      - Efficient io/memory mapping/columnar storage
      - Strong emphasis on statistics on N-d grids
      - 1 000 000 000 rows/second
    - Viz tools (vaex-viz, vaex-jupyter)
    - Hdf5 and arrow support (vaex-hdf5, vaex-arrow)
    - ML without pipelines (in progress, vaex-ml)
- Demo

*“Never do a live demo”*

-Many people



# Why am I here?

# Why am I here?

- PyData stack

# Why am I here?

- PyData stack
  - Developer: bread and butter developing tools

# Why am I here?

- PyData stack
  - Developer: bread and butter developing tools
  - Data scientist: bread and butter to use the tools

# Why am I here?

- PyData stack
  - Developer: bread and butter developing tools
  - Data scientist: bread and butter to use the tools
  - We love Python (some like C++)

# Why am I here?

- PyData stack
  - Developer: bread and butter developing tools
  - Data scientist: bread and butter to use the tools
  - We love Python (some like C++)
- Tools

# Why am I here?

- PyData stack
  - Developer: bread and butter developing tools
  - Data scientist: bread and butter to use the tools
  - We love Python (some like C++)
- Tools
  - pandas: flexible but loves making memory copies

# Why am I here?

- PyData stack
  - Developer: bread and butter developing tools
  - Data scientist: bread and butter to use the tools
  - We love Python (some like C++)
- Tools
  - pandas: flexible but loves making memory copies
  - Dask.dataframe: same issue, but distributes



# Why am I here?

- PyData stack
  - Developer: bread and butter developing tools
  - Data scientist: bread and butter to use the tools
  - We love Python (some like C++)
- Tools
  - pandas: flexible but loves making memory copies
  - Dask.dataframe: same issue, but distributes
  - Vaex: memory efficient, will lack features compared to pandas

# Why am I here?

- PyData stack
  - Developer: bread and butter developing tools
  - Data scientist: bread and butter to use the tools
  - We love Python (some like C++)
- Tools
  - pandas: flexible but loves making memory copies
  - Dask.dataframe: same issue, but distributes
  - Vaex: memory efficient, will lack features compared to pandas
  - xframe: in dev, no Python binding yet?

# Now what?

# Now what?

- Use pandas for 'medium size' data, maybe `dask.dataframe`

# Now what?

- Use pandas for 'medium size' data, maybe dask.dataframe
- Use vaex for big dataset (>100 000 000 rows)

# Now what?

- Use pandas for 'medium size' data, maybe `dask.dataframe`
- Use vaex for big dataset (>100 000 000 rows)
- Frustrations

# Now what?

- Use pandas for 'medium size' data, maybe `dask.dataframe`
- Use vaex for big dataset (>100 000 000 rows)
- Frustrations
  - Missing feature in vaex that is in pandas

# Now what?

- Use pandas for 'medium size' data, maybe `dask.dataframe`
- Use vaex for big dataset (>100 000 000 rows)
- Frustrations
  - Missing feature in vaex that is in pandas
  - Developing algorithms in C(++), passing the data as `double*` (xtensor/xframe?)

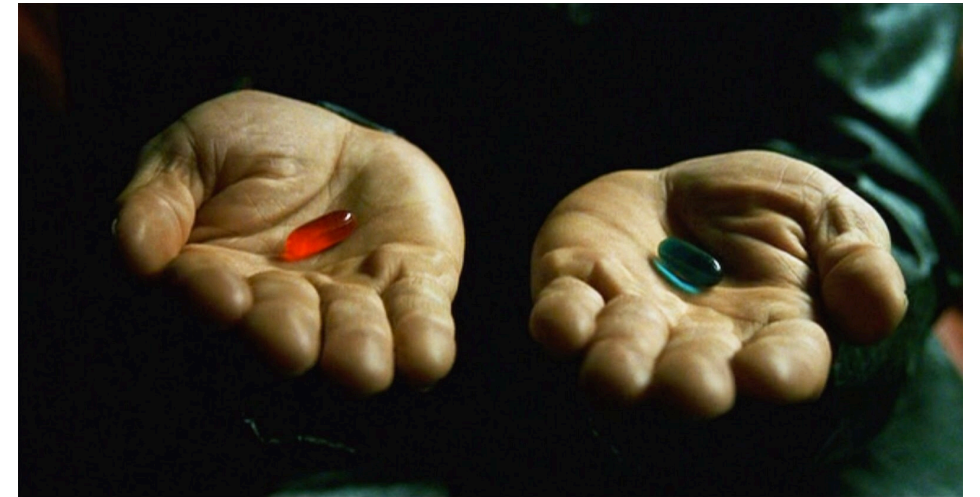


# Ways out



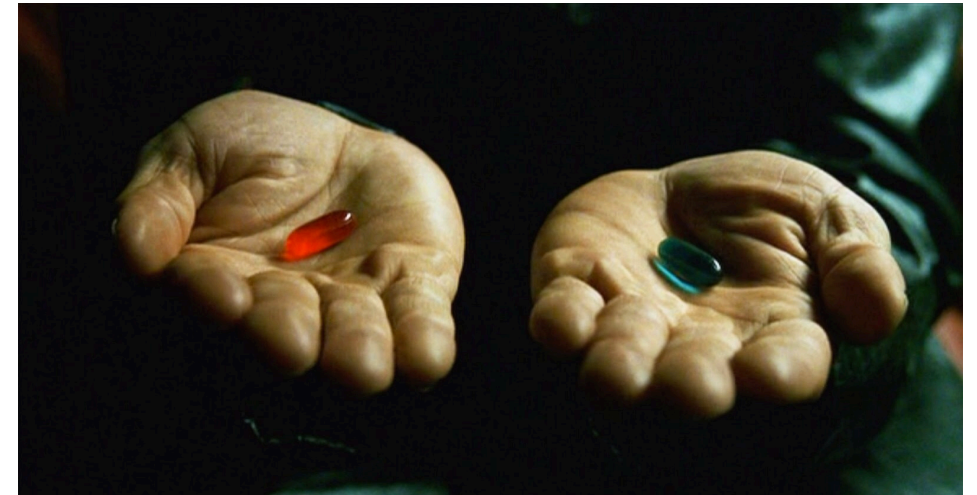
# Ways out

- Pandas2?



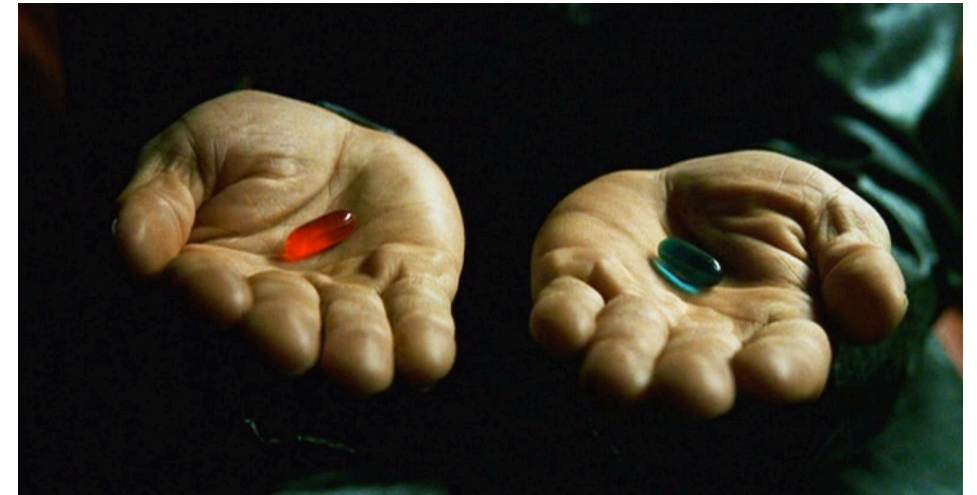
# Ways out

- Pandas2?
  - Use lessons from pandas and vaex -> 3rd library



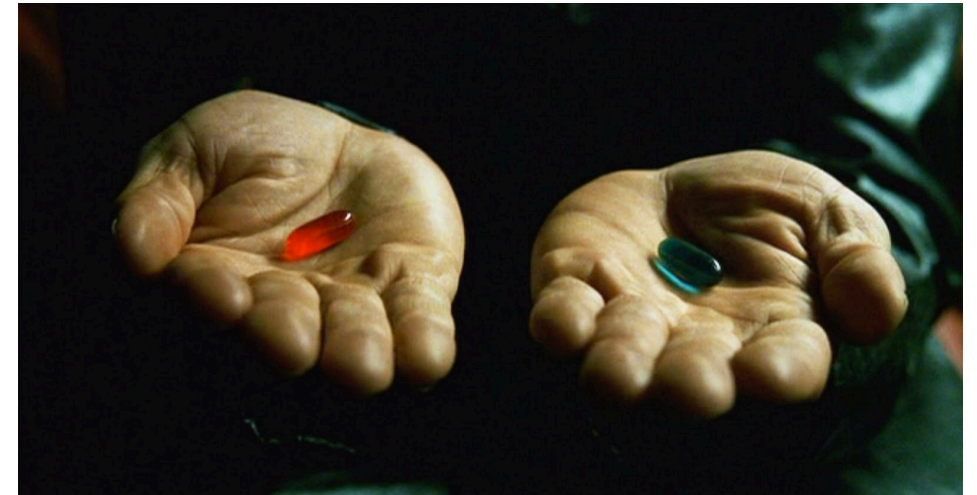
# Ways out

- Pandas2?
  - Use lessons from pandas and vaex -> 3rd library
- Upstream functionality



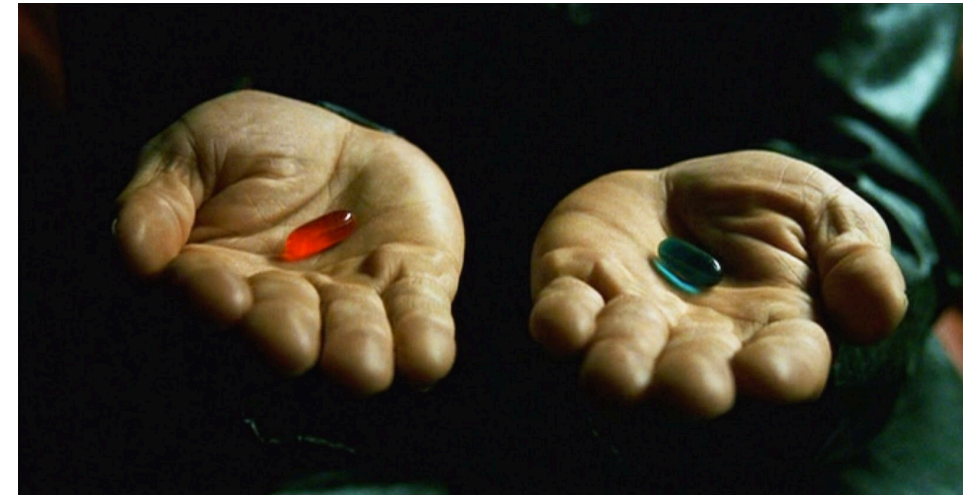
# Ways out

- Pandas2?
  - Use lessons from pandas and vaex -> 3rd library
- Upstream functionality
  - Make pandas and vaex use a common library (e.g. date time -> weekday) and arrow



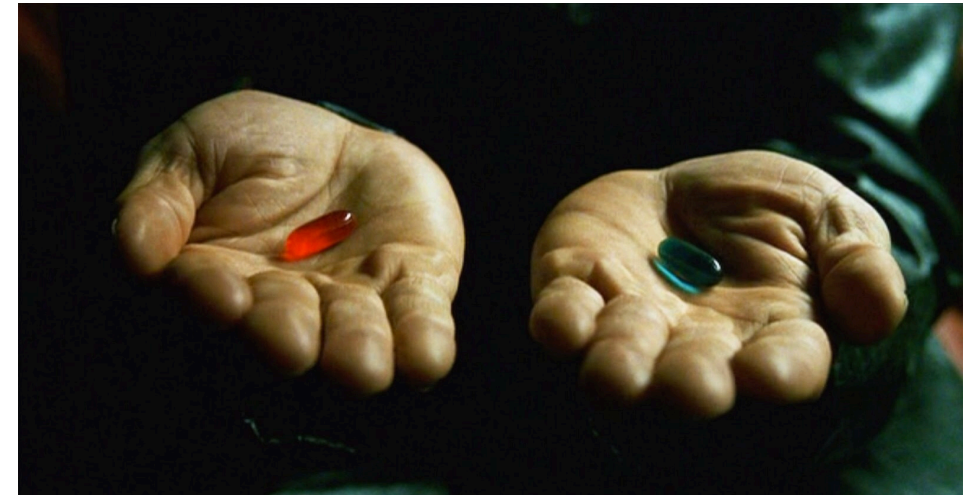
# Ways out

- Pandas2?
  - Use lessons from pandas and vaex -> 3rd library
- Upstream functionality
  - Make pandas and vaex use a common library (e.g. date time -> weekday) and arrow
- xframe



# Ways out

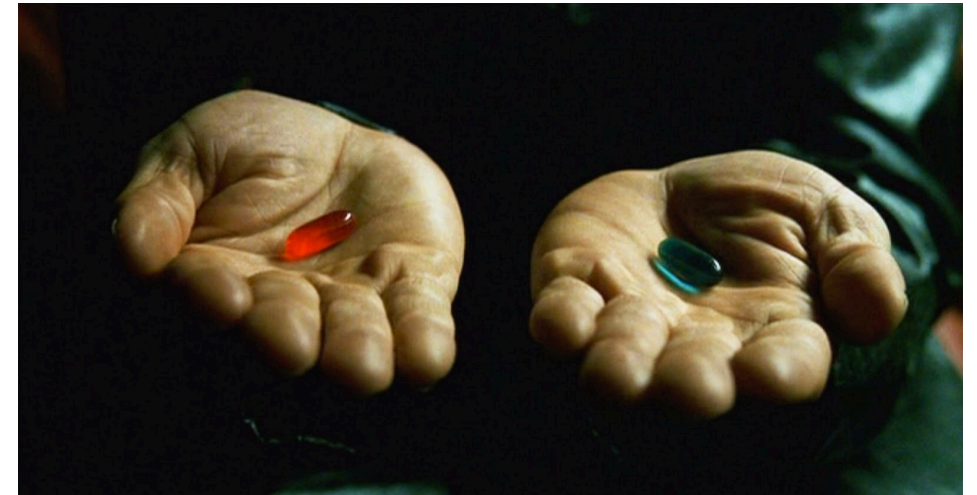
- Pandas2?
  - Use lessons from pandas and vaex -> 3rd library
- Upstream functionality
  - Make pandas and vaex use a common library (e.g. date time -> weekday) and arrow
- xframe
  - Build on top of an underlying library





# Ways out

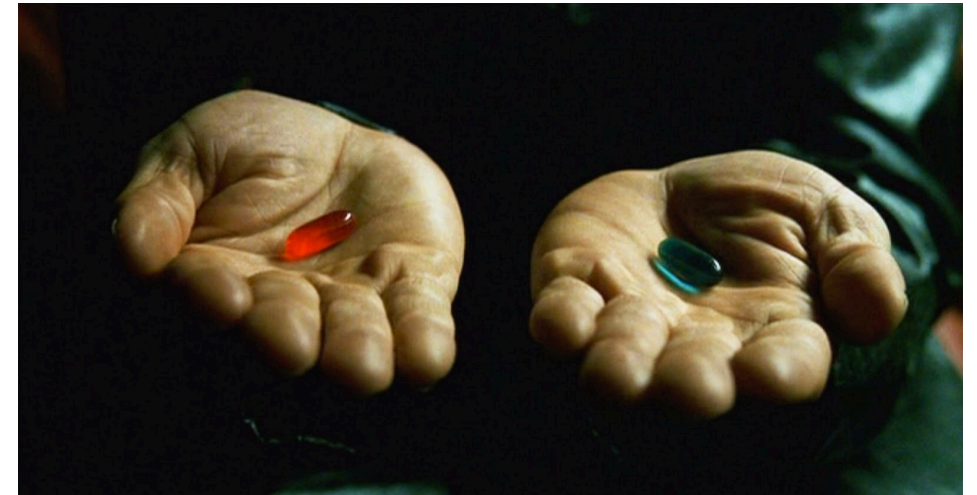
- Pandas2?
  - Use lessons from pandas and vaex -> 3rd library
- Upstream functionality
  - Make pandas and vaex use a common library (e.g. date time -> weekday) and arrow
- xframe
  - Build on top of an underlying library
  - xframe+arrow lands gives smooth access to C++ land





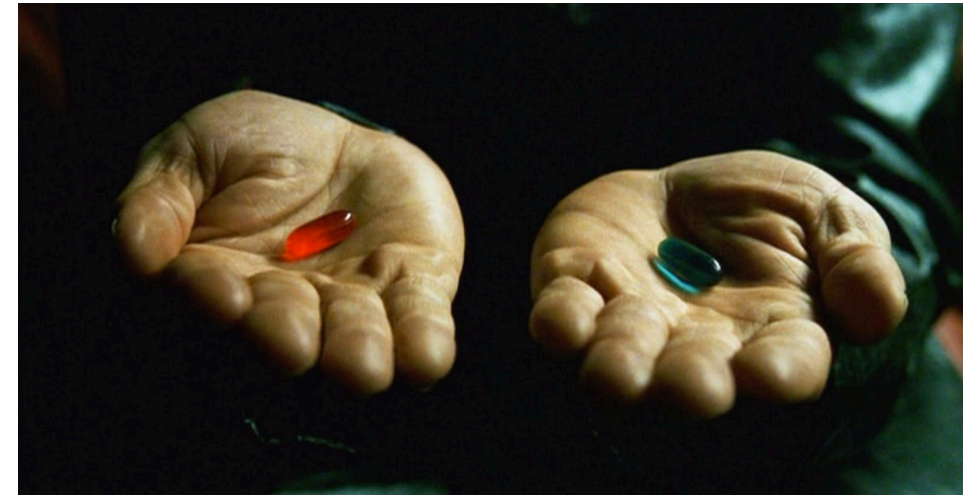
# Ways out

- Pandas2?
  - Use lessons from pandas and vaex -> 3rd library
- Upstream functionality
  - Make pandas and vaex use a common library (e.g. date time -> weekday) and arrow
- xframe
  - Build on top of an underlying library
  - xframe+arrow lands gives smooth access to C++ land
- Engine approach



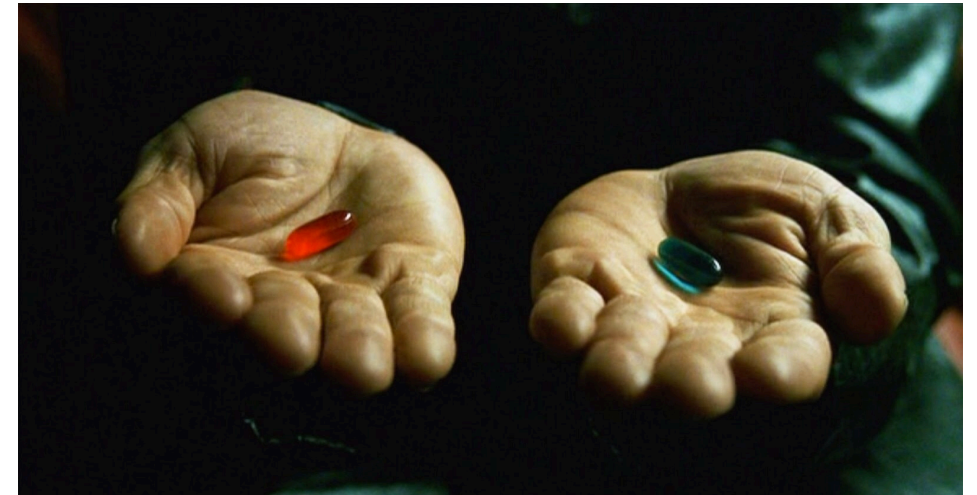
# Ways out

- Pandas2?
  - Use lessons from pandas and vaex -> 3rd library
- Upstream functionality
  - Make pandas and vaex use a common library (e.g. date time -> weekday) and arrow
- xframe
  - Build on top of an underlying library
  - xframe+arrow lands gives smooth access to C++ land
- Engine approach
  - Have a single API, different engines



# Ways out

- Pandas2?
  - Use lessons from pandas and vaex -> 3rd library
- Upstream functionality
  - Make pandas and vaex use a common library (e.g. date time -> weekday) and arrow
- xframe
  - Build on top of an underlying library
  - xframe+arrow lands gives smooth access to C++ land
- Engine approach
  - Have a single API, different engines
  - API with lowest common denominator?



**Good as it is?**

# Good as it is?

- Pandas should stay conservative (that's a good thing)

# Good as it is?

- Pandas should stay conservative (that's a good thing)
- Vaex can be more exploratory (e.g. selections/filters)

# Good as it is?

- Pandas should stay conservative (that's a good thing)
- Vaex can be more exploratory (e.g. selections/filters)
- Analogy: truck / race car

# Good as it is?

- Pandas should stay conservative (that's a good thing)
- Vaex can be more exploratory (e.g. selections/filters)
- Analogy: truck / race car





# Good as it is?

- Pandas should stay conservative (that's a good thing)
- Vaex can be more exploratory (e.g. selections/filters)
- Analogy: truck / race car



# Good as it is?

- Pandas should stay conservative (that's a good thing)
- Vaex can be more exploratory (e.g. selections/filters)
- Analogy: truck / race car



# Good as it is?

- Pandas should stay conservative (that's a good thing)
- Vaex can be more exploratory (e.g. selections/filters)
- Analogy: truck / race car
- Will the PyData stack be better with 2 (or more) libraries?





# Good as it is?

- Pandas should stay conservative (that's a good thing)
- Vaex can be more exploratory (e.g. selections/filters)
- Analogy: truck / race car
- Will the PyData stack be better with 2 (or more) libraries?
  - If so, how can we have mutual benefit from each other?



# Good as it is?

- Pandas should stay conservative (that's a good thing)
- Vaex can be more exploratory (e.g. selections/filters)
- Analogy: truck / race car
- Will the PyData stack be better with 2 (or more) libraries?
  - If so, how can we have mutual benefit from each other?
  - Split off libraries? (e.g. datetime manipulation)



# Good as it is?

- Pandas should stay conservative (that's a good thing)
- Vaex can be more exploratory (e.g. selections/filters)
- Analogy: truck / race car
- Will the PyData stack be better with 2 (or more) libraries?
  - If so, how can we have mutual benefit from each other?
  - Split off libraries? (e.g. datetime manipulation)
  - xframe+arrow for talking to C++ (to reuse the C++ algos)



# Summary

- Vaex
  - ‘alternative’ to pandas, for large # rows ( $>100\,000\,000$ )
  - No memory copies
  - Expression system
  - [github.com/vaexio/vaex](https://github.com/vaexio/vaex)
  - Next gen dataframe library