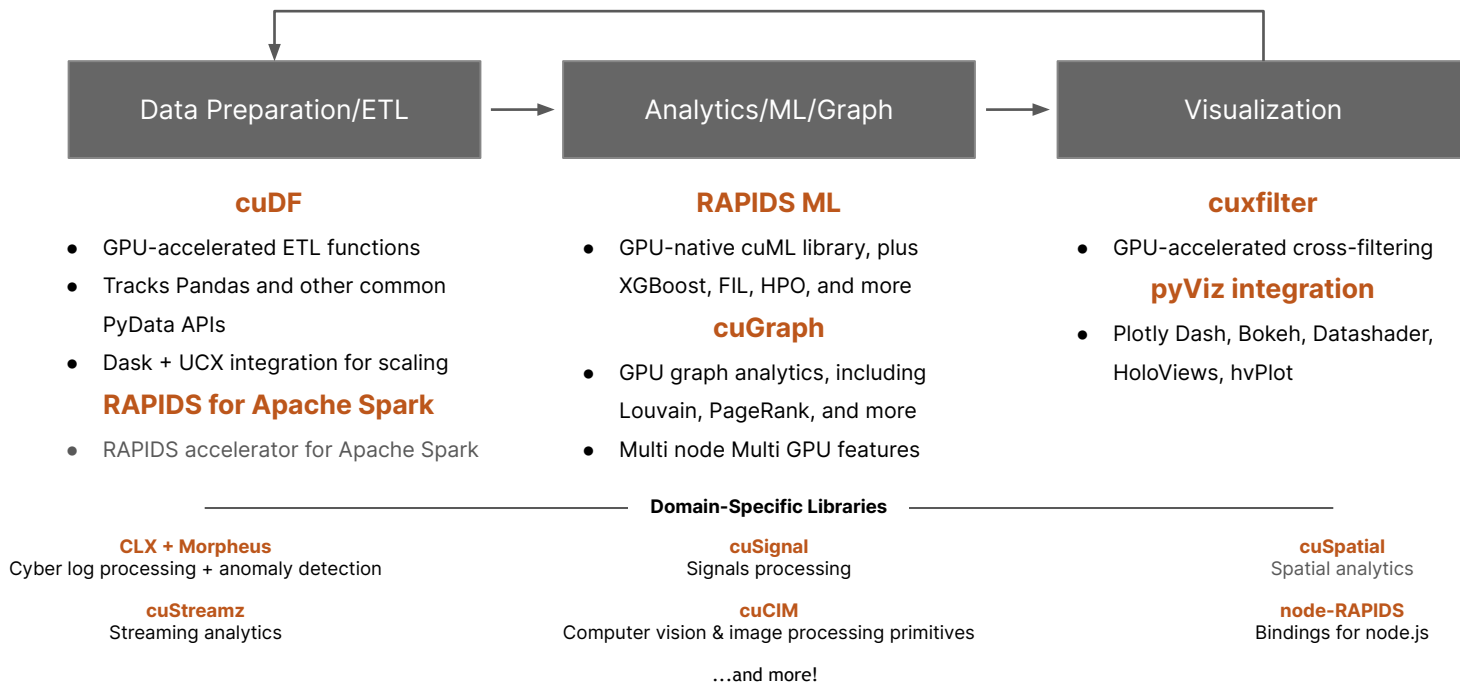# RAPIDS

The RAPIDS suite of open source software libraries gives you the freedom to execute end-to-end data science and analytics pipelines entirely on NVIDIA GPUs.
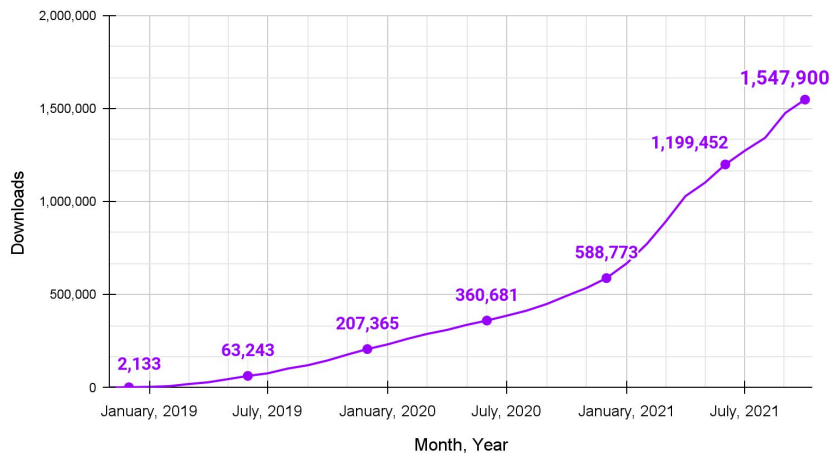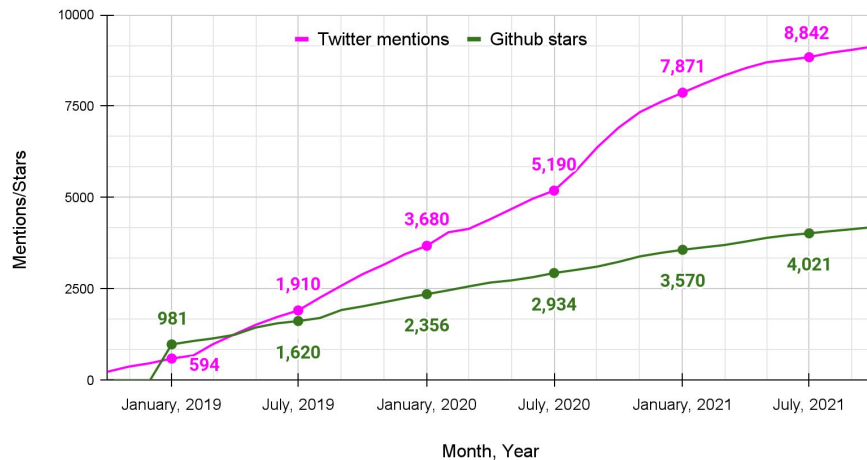
RAPIDS Product Ecosystem

Data Preparation/ETL	Analytics/ML/Graph	Visualization

cuDF

- GPU-accelerated ETL functions
- Tracks Pandas and other common PyData APIs
- Dask + UCX integration for scaling

RAPIDS for Apache Spark

- RAPIDS accelerator for Apache Spark

RAPIDS ML

- GPU-native cuML library, plus XGBoost, FIL, HPO, and more

cuGraph

- GPU graph analytics, including Louvain, PageRank, and more
- Multi node Multi GPU features

cuxfilter

- GPU-accelerated cross-filtering

pyViz integration

- Plotly Dash, Bokeh, Datashader, HoloViews, hvPlot

Domain-Specific Libraries

CLX + Morpheus
Cyber log processing + anomaly detection

cuSignal
Signals processing

cuSpatial
Spatial analytics

cuStreamz
Streaming analytics

cuCIM
Computer vision & image processing primitives

node-RAPIDS
Bindings for node.js

…and more!

RAPIDS Growth
More than **1.5M** downloads and accelerating

RAPIDS Downloads

- 2,133
- 63,243
- 207,365
- 360,681
- 588,773
- 1,199,452
- 1,547,900

RAPIDS Community Engagement

Twitter mentions — Github stars

- 594
- 1,910
- 3,680
- 5,190
- 7,871
- 8,842
- 981
- 1,620
- 2,356
- 2,934
- 3,570
- 4,021

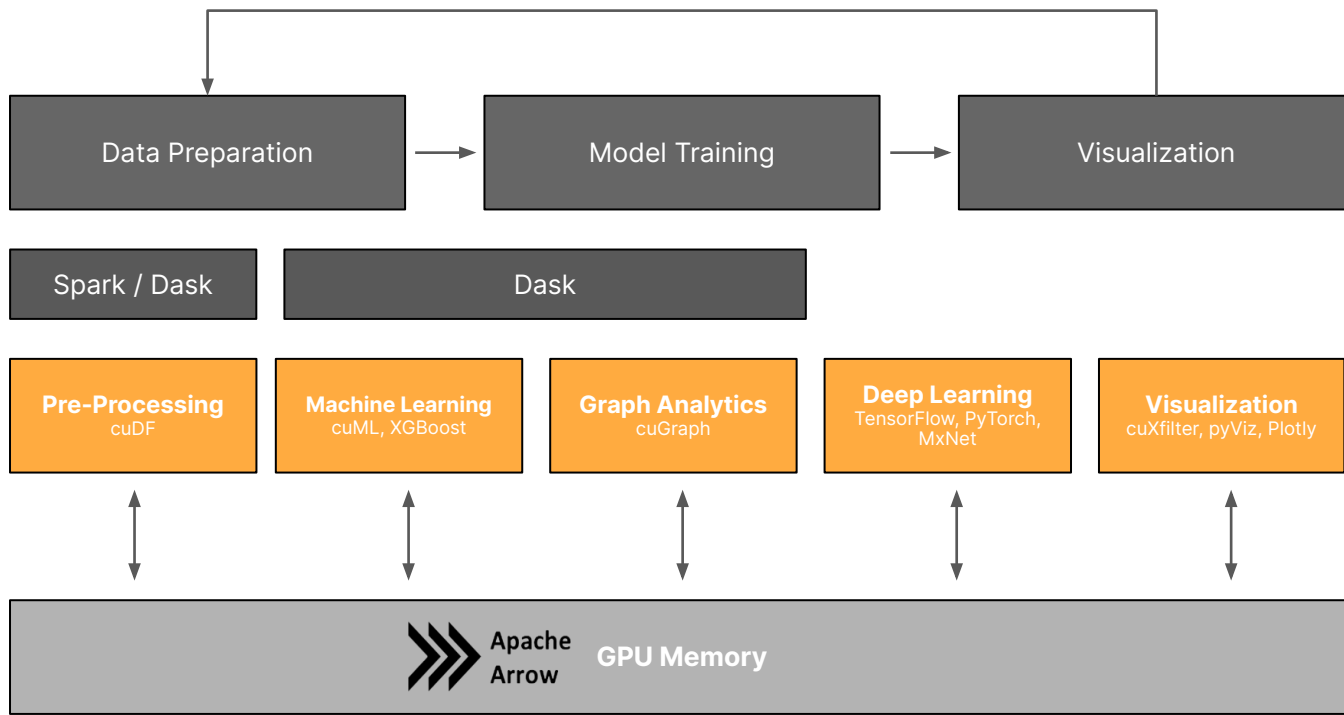# Open Source Software Has Democratized Data Science

Highly Accessible, Easy to Use Tools Abstract Complexity

Data Preparation	Model Training	Visualization

Apache Spark / Dask

Pre-Processing pandas	**Machine Learning** scikit-learn	**Graph Analytics** NetworkX	**Deep Learning** TensorFlow, PyTorch, MxNet	**Visualization** matplotlib

CPU Memory
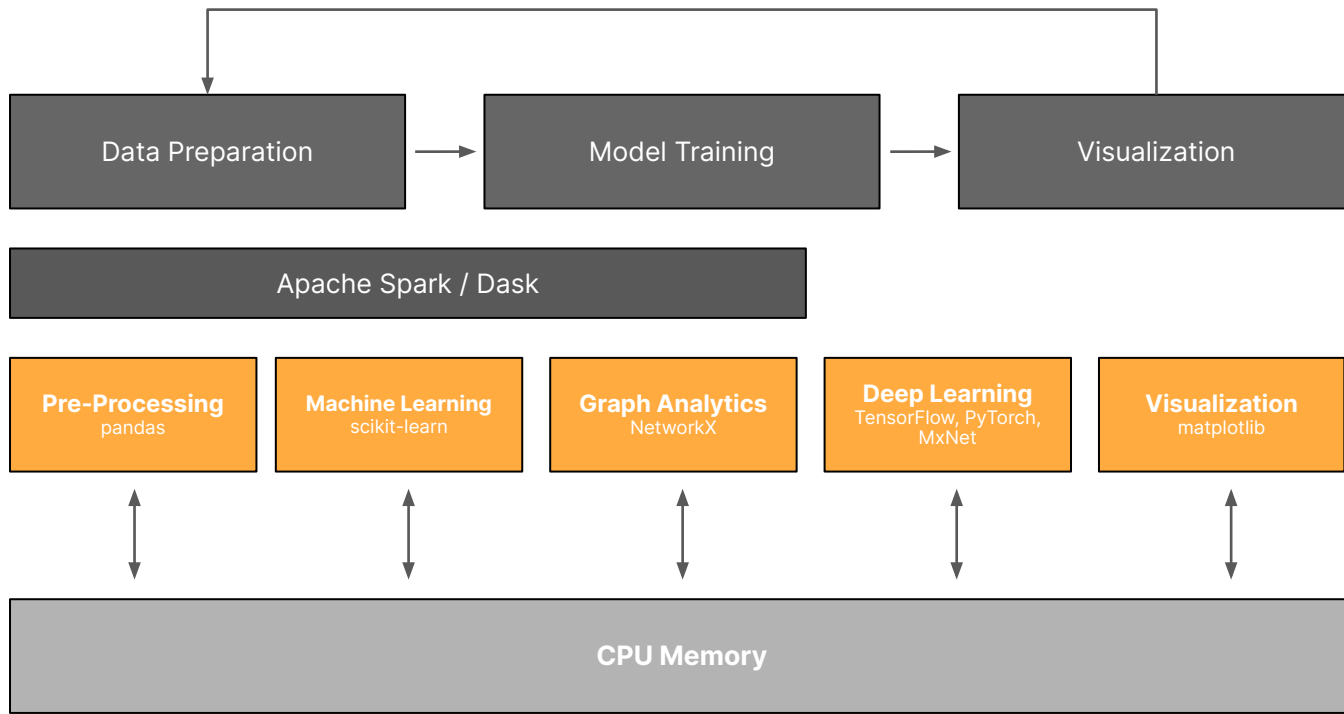
Accelerated Data Science with RAPIDS

Powering Popular Data Science Ecosystems with NVIDIA GPUs

Minor Code Changes for Major Benefits
Abstracting Accelerated Compute through Familiar Interfaces

CPU

pandas
```
>>> import pandas as pd
>>> df =
pd.read_csv("filepath")
```

CPU Spark
```
spark.sql("""
select
        order
        count(*) as
order_count
from
        orders""")
```

scikit-learn
```
>>> from sklearn.ensemble
import
RandomForestClassifier
>>> clf =
RandomForestClassifier()
>>> clf.fit(x, y)
```

NetworkX
```
>>> import networkx as nx
>>> page_rank =
nx.pagerank(graph)
```

GPU

cuDF
```
>>> import cudf
>>> df =
cudf.read_csv("filepath")
```
Average Speed-Ups: *150x*

GPU Spark
```
spark.conf.set("spark.plugins
","com.nvidia.spark.SQLPlugin")
spark.sql("""
select
        order
        count(*) as order_count
from
        orders""")
```
Average Speed-Ups: *10x*

cuML
```
>>> from cuml.ensemble
import
RandomForestClassifier
>>> cuclf =
RandomForestClassifier()
>>> cuclf.fit(x, y)
```
Average Speed-Ups: *50x*

cuGraph
```
>>> import cugraph
>>> page_rank =
cugraph.pagerank(graph)
```
Average Speed-Ups: *250x*

The Evolution of Data Processing
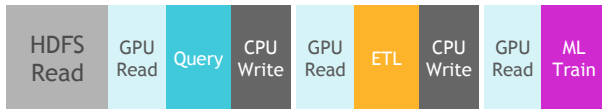## Faster Data Access, Less Data Movement

Hadoop Processing, Reading from Disk

HDFS Read	Query	HDFS Write	HDFS Read	ETL	HDFS Write	HDFS Read	ML Train

CPU-Based Spark In-Memory Processing

HDFS Read	Query	ETL	ML Train

25-100x Improvement
Less Code
Language Flexible
Primarily In-Memory

Traditional GPU Processing

HDFS Read	GPU Read	Query	CPU Write	GPU Read	ETL	CPU Write	GPU Read	ML Train

5-10x Improvement
More Code
Language Rigid
Substantially on GPU

RAPIDS

Read	Query	ETL	ML Train
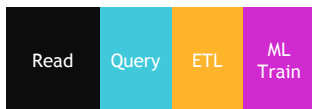
50-100x Improvement
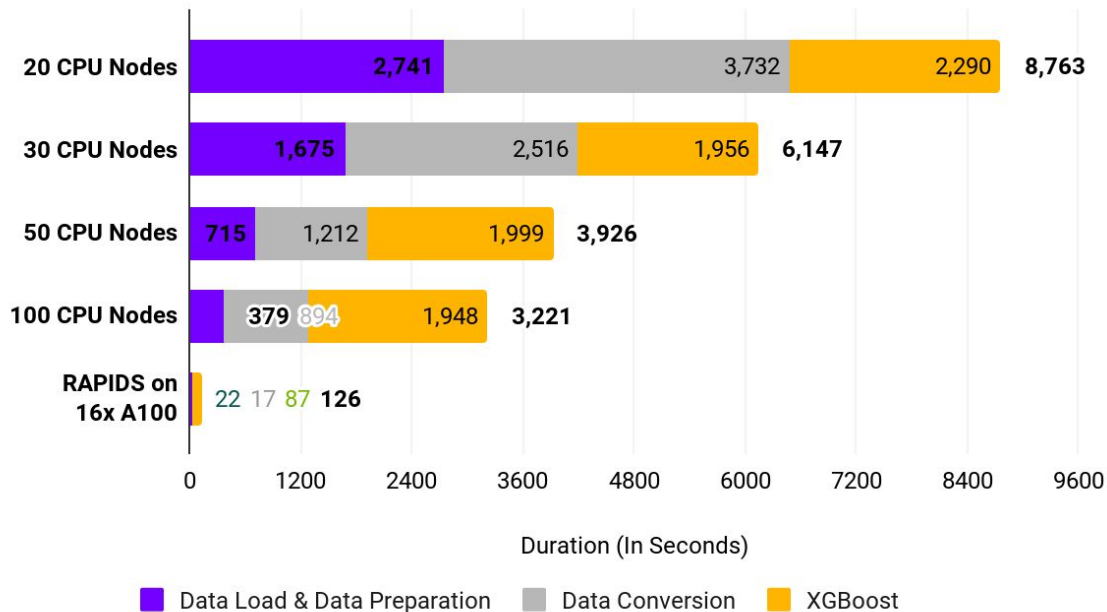Same Code
Language Flexible
Primarily on GPU

?

Why Use GPUs

GPUs are built for intensive parallel processing. As datasets continue to grow, data scientists are limited by the sequential nature of CPU compute. GPUs provide the power and parallelism necessary for today's data science.

Lightning-Fast End-to-End Performance
Reducing Data Science Processes from Hours to Seconds

RAPIDS End-to-End Workflow Runtimes

Category	Duration (In Seconds)
20 CPU Nodes	2,741 / 3,732 / 2,290 → 8,763
30 CPU Nodes	1,675 / 2,516 / 1,956 → 6,147
50 CPU Nodes	715 / 1,212 / 1,999 → 3,926
100 CPU Nodes	379 / 894 / 1,948 → 3,221
RAPIDS on 16x A100	22 / 17 / 87 → 126

Duration (In Seconds): 0, 1200, 2400, 3600, 4800, 6000, 7200, 8400, 9600

■ Data Load & Data Preparation   ■ Data Conversion   ■ XGBoost

16 A100s Provide More Power than 100 CPU Nodes

70x Faster Performance than Similar CPU Configuration

20x More Cost-Effective than Similar CPU Configuration

*CPU approximate to n1-highmem-8 (8 vCPUs, 52GB memory) on Google Cloud Platform. TCO calculations-based on Cloud instance costs.

Demo

Using RAPIDS on Hyperplane