OPERATIONALIZING MACHINE LEARNING USING GPU ACCELERATED, IN-DATABASE ANALYTICS
Why GPUs?
A Tale of Numbers

Performance
100x gains over traditional RDBMS / NoSQL / In-Mem Databases

Cores
Modern GPUs can consist of up to 3000+ cores compared to 32 in a CPU

Costs
75% reduction in infrastructure costs, licensing, staff, etc.

More with Less
Increase performance, throughput, capability while minimizing the costs to support the business
Why a GPU Database?

• Leverage Innovations in CPUs and GPUs
• Single Hardware Platform
• Simplified Software Stack
What are AI, ML, and Deep Learning?

**AI**

**ML**

Predict $y$ using function on data $x$

**Deep Learning**

- Input layer
- Hidden layer 1
- Hidden layer 2
- Output layer
No shortage of techniques and programming languages
Python and SQL cover almost all the algorithms in that scary spider and Kinetica supports all Python libraries!
ML/AI/Deep Learning Lifecycle
ML/AI/Deep Learning Lifecycle

• Create, extract, transform, and process big data: batch and streams
• Apply ML to data.
  • Model pre-processing
  • Model execution
  • Model post-processing
• Within an ecosystem of general analytics
  • Supporting a range of human and machine consumers

PRACTICALLY SPEAKING.
Typical AI Process: High Latency, Rigid, Complex Tech Stack

Enterprises struggle to make AI models available to business users.

Extracting data for AI is expensive and slow.
Kinetica: A More Ideal AI Process

API exposes custom functions which can be made available to business users.
Current Inefficient Use of Python

- Interpreted
- Single threaded
- Clean, transform
- Flow: for each member
  - Pre-process
  - Model execute
  - Post-process
Optimized SQL and Python UDF with Kinetica

- Pre-process
- Binary executable code
- Superior optimization
- Declarative SQL

- Model execute
- Only essential imperative model code
- Not relational set processing

- Post-process
- Binary executable code
- Superior optimization
- Declarative SQL
Comprehensive Solution Architecture

Major U.S Retailer

KINETICA: 10 Node Cluster

- Massive Stream Ingestion
- Apache Tomcat Applications Servers
  - Spring Endpoint oriented architecture
  - Horizontal elastic scaling
- Massive Fast Analytics

Various ETL/ELT

Fact and dimensions tables for various Use Cases
Billions of rows

Fast Streaming Projects
Fast Analytics Projects

Prompts Project

Full Model Pipeline 1

Full Model Pipeline N
Use Case Example
**MNIST: Simple Image Processing Use Case**

**A Parametric Model Python Using TensorFlow**

**Model Training**
- Set of image files stored in Kinetica Database Table
- Python UDF in Kinetica using TensorFlow

**Model Serving**
- Python UDF in Kinetica using TensorFlow
- Input = table TFModel table.
- Output = table mnist_inference_out

**Model Analytics**
- SQL!
Model Training & Inference Data Model: MPP Sharding
Thank You!

Come get your copy of the O’Reilly Book at Booth G.01!