An Open-Source Streaming Machine Learning and Real-Time Analytics Architecture

Using an IoT example

Fred Melo
@fredmelo_br

William Markito
@william_markito
Traditional Data Analytics - Limitations

- No real-time information
- ETL based
- Data-source specific
- Hard to change
- Labor intensive
- Inefficient
Stream-based, Real-Time Closed-Loop Analytics

Data Stream Pipeline

Data Lake

HDFS

In-Memory Real-Time Data

Expert System / Machine Learning

Multiple Data Sources
Real-Time Processing
Store Everything

Continuous Learning
Continuous Improvement
Continuous Adapting
A Streaming Machine Learning for IoT Example

Predictive Maintenance Scenario

Sensor Data

Real-Time

Live data becomes historical over time

Historical

Evaluates LIVE DATA

“According to historical trends, there’s an 80% chance this equipment would fail in the next 12 hours”

Smart System

Learns with HISTORICAL TRENDS

"How were the temperature and vibration sensors reading when the latest failures happened? "

Take Action
Streaming Machine Learning

- Info
- Analysis

Machine Learning

Look at past trends (for similar input)
Evaluate current input
Score / Predict
Streaming Machine Learning

Info

Analysis

Filter

[ json ]

Machine Learning
Streaming Machine Learning

Info

Analysis

Filter  Enrich

Spark  GEODE

Machine Learning
Streaming Machine Learning

Info

Analysis

Filter → Enrich → Transform →

Machine Learning
Streaming Machine Learning

Info

Analysis

ML Model

Filter ➔ Enrich ➔ Transform ➔
Streaming Machine Learning

Info

Analysis

ML Model

Filter → Enrich → Transform

Transform → Transform
Streaming Machine Learning

ML Model

In-Memory Data Grid

Update

Push

Front-end
Streaming Machine Learning

Supervised Learning Example
A Streaming Machine Learning Reference Architecture

Distributed Computing

SpringXD

Ingest → Transform → Sink

Store / Analyze

Predict / Machine Learning

Other Sources and Destinations

JMS

Fast Data

GEODE

Other Sources and Destinations

MQTT

mongoDB

Other Sources and Destinations

predict / Machine Learning

Other Sources and Destinations

MADlib

python

R
Indoors Localization - Applied Example
Trilateration and its limitations

Noisy Data

Physical Barriers

Large Overlap Areas

Moving Targets

Innacuracy

Large Overlap Areas
Particle Filters - Calculating the optimum solution

Path Estimate from Particle Filter Localization

- beacons
- estimate
- dr

Y position (m)

X position (m)

beacons
estimate
dr

APACHE GEODE
Particle Filters - Calculating the optimum solution

**Localization Algorithm**

- **EKF SLAM**
  - Prediction
  - Observation
  - Update
  - Robot Pose
  - Beacon Location

- **Particle Filter**
  - Prediction
  - Observation
  - Update
  - User Location

- **Range data**

**Prototype System**

- **Beacon 1**
- **Beacon 2**
- **Beacon 3**
- **Beacon 4**
- **Repeater**
- **Main UI**
- **Robot**
- **Human 1**
- **Human 2**
- **Monitoring PC**

* User localization based on the localization of robots and beacons

* Extended Kalman Filter Simultaneous Localization And Mapping

**Autonomous Navigation**
The Solution

1. Capture signal strength
2. Calculate distance from antenna
3. Trilaterate different sensors to predict location in real-time
4. Show on a map with live updates
Architecture Overview

Ingest

SpringXD

Groovy

Calculate Device Distance

+ Distance

Sink

Application Platform

Spring Boot

Predict Location

GUI

JSON

HTTP

Application Platform

CLOUD FOUNDARY
Geode Basic Concepts

- Cache
  - Configurable through XML, spring, Java
- Region
  - Distributed j.u.Map on steroids
  - Highly available, redundant
- Member
  - Locator, Server, Client
- Callbacks
  - Listener, Writer, AsyncEventListener, Parallel/Serial
Runs as a distributed application or as a single node
A stream is composed from *modules*. Each module is deployed to a *container* and its channels are bound to the *transport*.
Why have we selected those projects

- Iterative & Exploratory model
- Web based REPL
- Multiple Interpreters
  - Apache Geode
  - Apache Spark
  - Markdown
  - Flink
  - Python...

- Productivity
- Built-in connectors
- Cloud Agnostic
- Highly Scalable
- Easy to setup
- Streams without coding

- In-memory & Persistent
- Highly Consistent
- Extreme transaction processing
- Thousands of concurrent clients
- Reliable event model
Source code and detailed instructions available at:
https://github.com/Pivotal-Open-Source-Hub/WifiAnalyticsIoT

Follow us on GitHub!

Fred Melo
@fredmelo_br

William Markito
@william_markito
Implementing a Highly Scalable In-Memory Stock Prediction System with Apache Geode (incubating), R and Spring XD

Room: Tohotom - 14:30, Sep 30
Fred Melo, Pivotal, William Markito, Pivotal