Productionizing Spark ML Pipelines with PFA

Nick Pentreath
Principal Engineer

@MLnick

IBM CODE

About

@MLnick on Twitter & Github

Principal Engineer, IBM

CODAIT - Center for Open-Source Data & AI Technologies

Machine Learning & AI

Apache Spark committer & PMC

Author of *Machine Learning with Spark*

Various conferences & meetups

Center for Open Source Data and AI Technologies

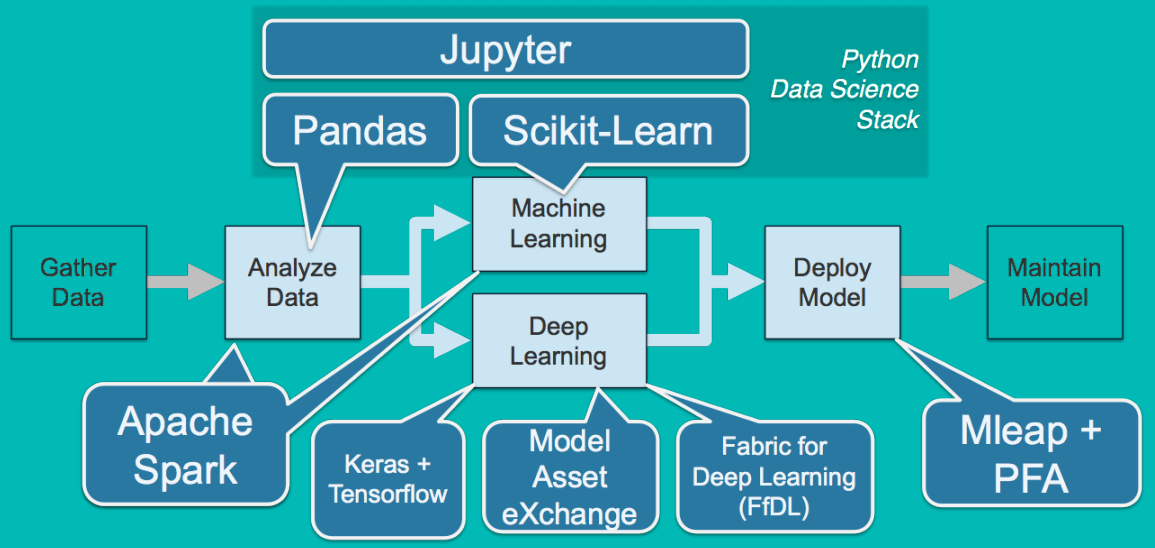
CODAIT

codait.org

CODAIT aims to make AI solutions dramatically easier to create, deploy, and manage in the enterprise

Relaunch of the Spark Technology Center (STC) to reflect expanded mission

Improving Enterprise AI Lifecycle in Open Source



Agenda

The Machine Learning Workflow

Challenges of ML Deployment

Portable Format for Analytics

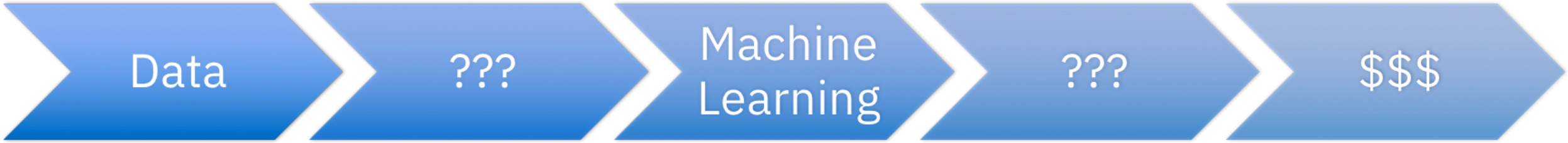
PFA for Spark ML

Performance Comparisons

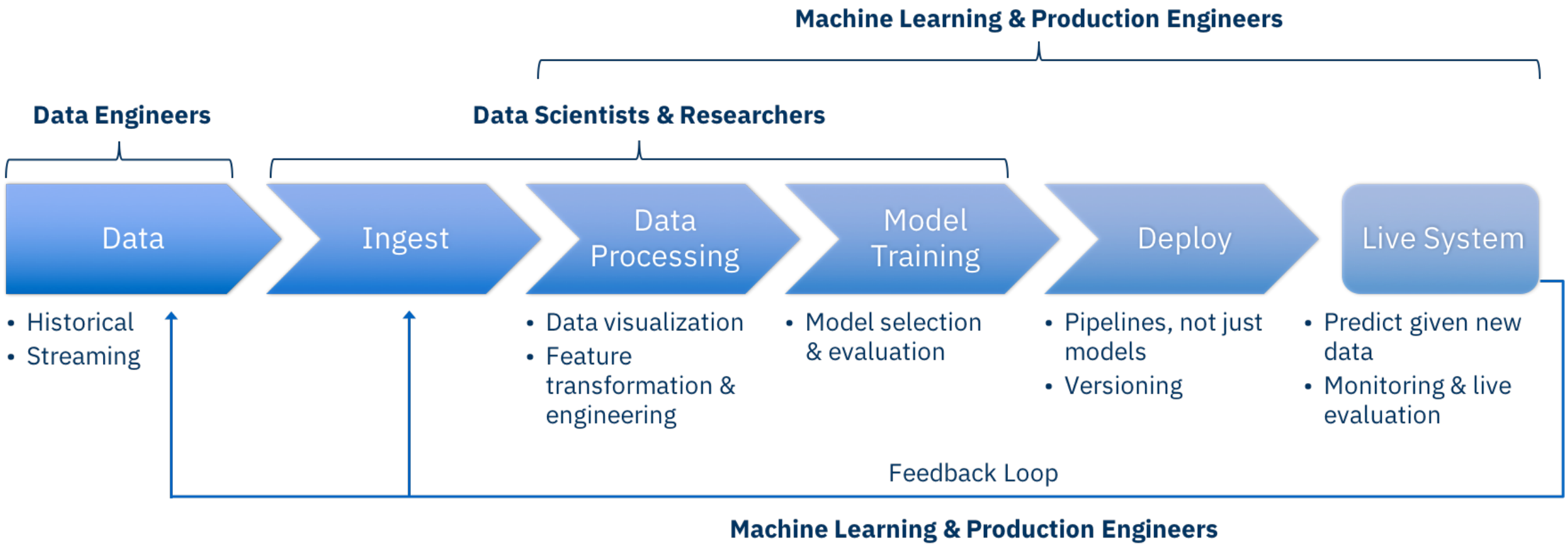
Summary and Future Directions

The Machine Learning Workflow

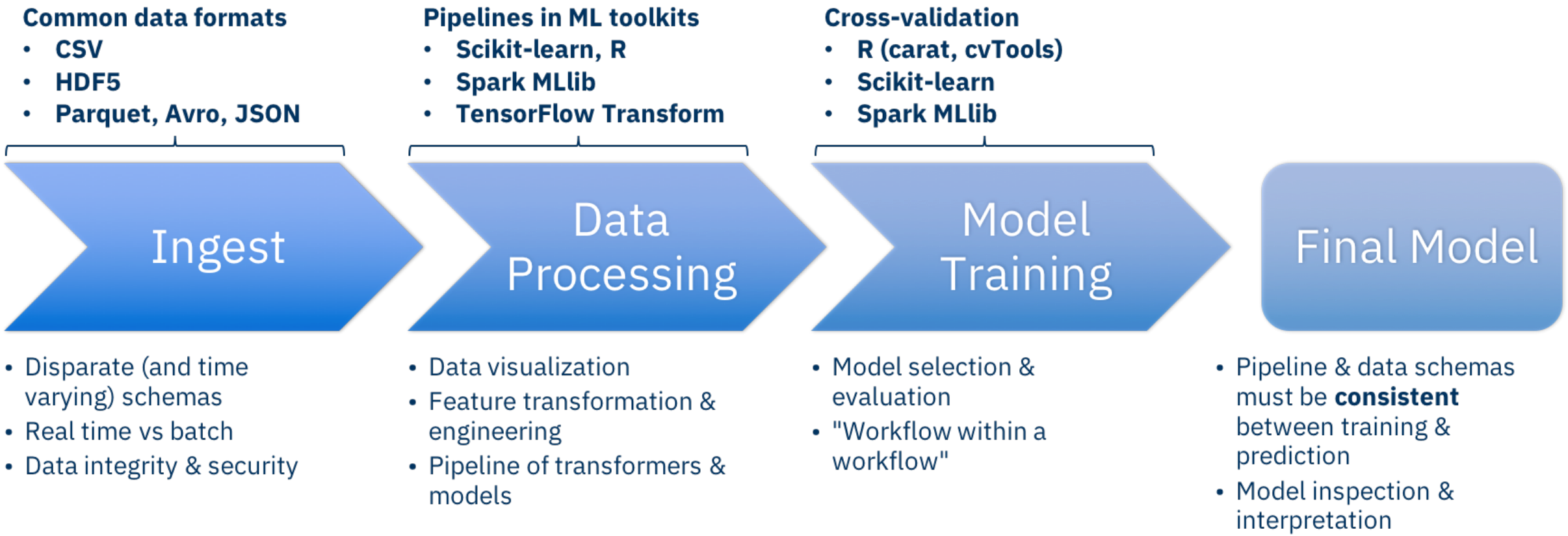
Perception



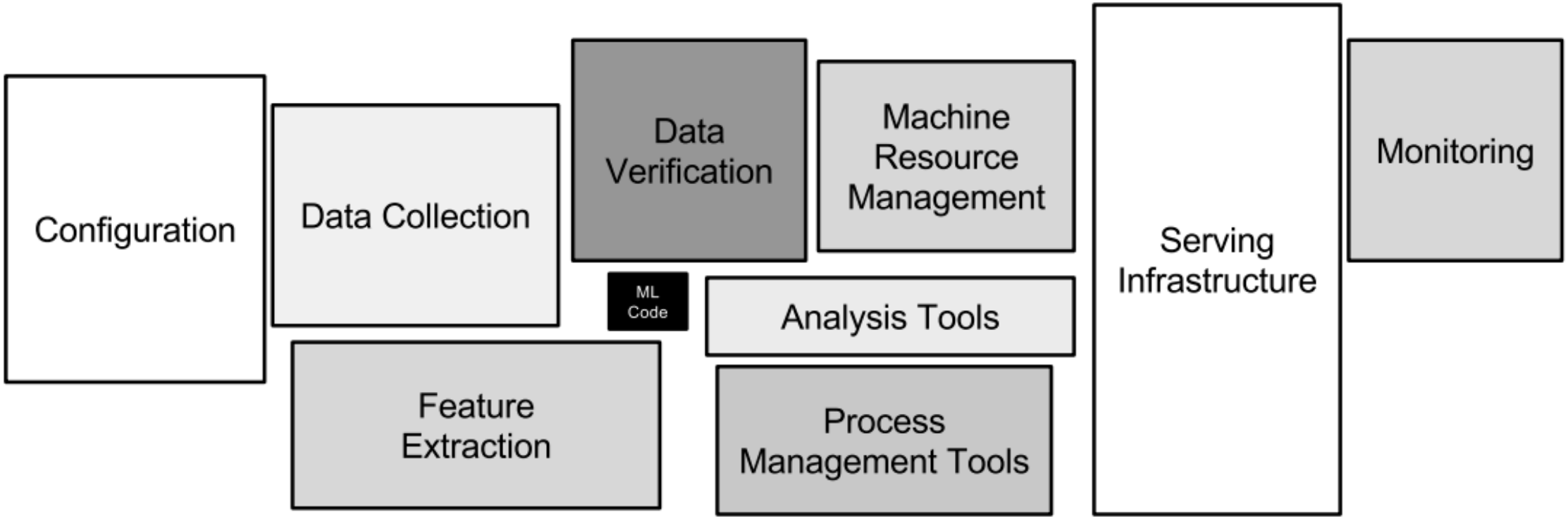
In reality the workflow spans teams ...



… and tools …



… and is a small (but critical!) piece of the puzzle



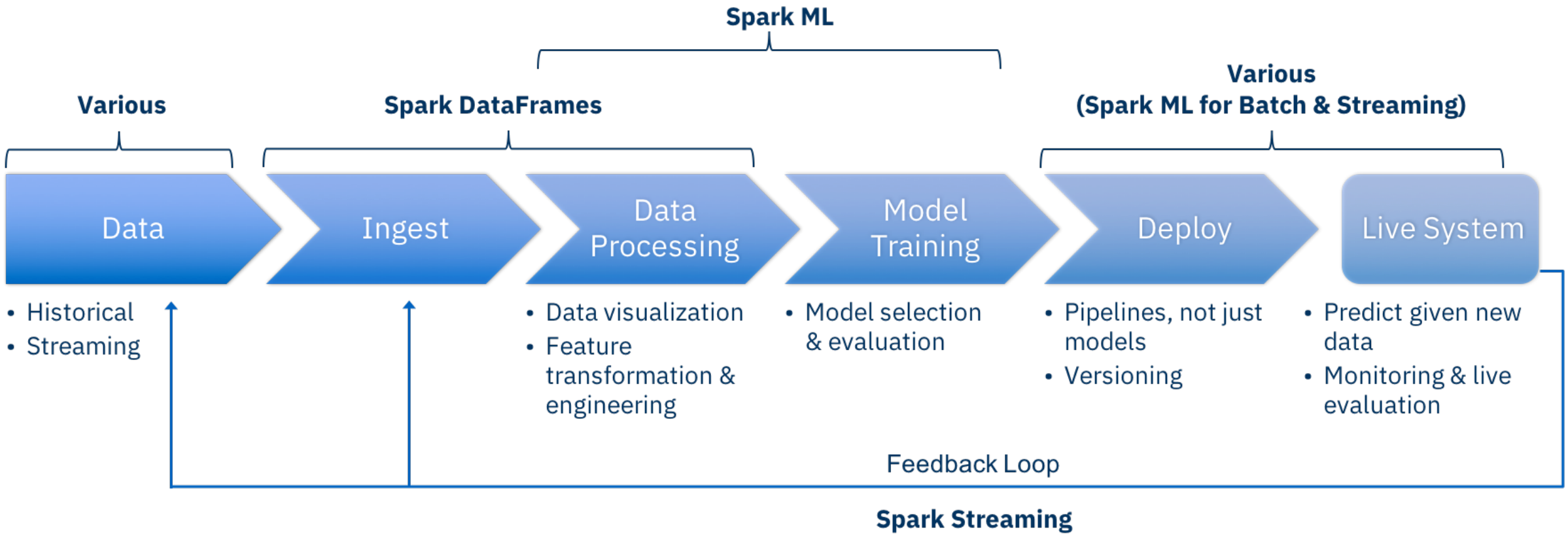
*Source: Hidden Technical Debt in Machine Learning Systems

Machine Learning Deployment

Challenges

- Need to manage and bridge many different:
  - Languages - Python, R, Notebooks, Scala / Java / C
  - Frameworks – too many to count!
  - Dependencies
  - Versions
- Performance characteristics highly variable across these dimensions
- Friction between teams
  - Data scientists & researchers – latest & greatest
  - Production – stability, control, performance
  - Business – metrics, business impact, working product!
- Proliferation of formats
- Lack of standardization => custom solutions
- Existing standards have limitations

Spark solves many problems ...



… but introduces additional challenges

- Tight coupling to Spark runtime
  - Introduces complex dependencies
  - Managing version & compatibility issues
- Scoring models in Spark is **slow**
  - Overhead of DataFrames, especially query planning
  - Overhead of task scheduling, even locally
  - Optimized for batch scoring (includes streaming "micro-batch" settings)
- Spark is **not suitable** for real-time scoring (< few 100ms latency)

- Currently, in order to use trained models outside of Spark, users must:
  - Write custom readers for Spark's native format; or
  - Create their own custom format; or
  - Export to a standard format (limited support => custom solution)
- Scoring outside of Spark also requires custom translation layer between Spark and another ML library

Everything is custom!

The Portable Format for Analytics

Overview

- PFA is being championed by the Data Mining Group (IBM is a founding member)
- DMG previously created PMML (Predictive Model Markup Language), arguably the only viable open standard currently
  - PMML has many limitations
  - PFA was created specifically to address these shortcomings

- PFA consists of:
  - JSON serialization format
  - AVRO schemas for data types
  - Encodes functions (*actions*) that are applied to inputs to create outputs with a set of built-in functions and language constructs (e.g. control-flow, conditionals)
  - Essentially a *mini functional math language* + *schema specification*
- Type and function system means PFA can be fully & statically verified on load and run by any compliant execution engine
- => portability across languages, frameworks, run times and versions

A Simple Example

- Example – multi-class logistic regression
- Specify input and output types using Avro schemas

```
{
  "name":"logistic-regression-model",
  "input":{
    "type":{
      "type":"array",
      "items":"double"
    }
  },
  "output":{
    "type":"double"
  },
```

- Specify the *action* to perform (typically on input)

```
"action":[
  {
    "a.argmax": [
      {
        "m.link.softmax": [
          {
            "model.reg.linear": [
              "input",
              {
                "cell": "model"
              }
            ]
          }
        ]
      }
    ]
  }
],
```

Managing State

- Data storage specified by *cells*
  - A cell is a named value acting as a global variable
  - Typically used to store state (such as model coefficients, vocabulary mappings, etc)
  - Types specified with Avro schemas
  - Cell values are mutable within an action, but immutable between action executions of a given PFA document
- Persistent storage specified by *pools*
  - Closer in concept to a database
  - Pools values are mutable across action executions

Other Features

- Special forms
  - Control structure – conditionals & loops
  - Creating and manipulating local variables
  - User-defined functions including lambdas
  - Casts
  - Null checks
  - (Very) basic try-catch, user-defined errors and logs
- Comprehensive built-in function library
  - Math, strings, arrays, maps, stats, linear algebra
  - Built-in support for some common models - decision tree, clustering, linear models

Aardpfark

- PFA export for Spark ML pipelines
  - `aardpfark-core`: Scala DSL for creating PFA documents
    - `avro4s` to generate schemas from case classes; `json4s` to serialize PFA document to JSON
  - `aardpfark-sparkml`: uses DSL to export Spark ML components and pipelines to PFA

```
val input = StringExpr("input")
val cell = Cell[LinearModelData](
  DenseLinearModelData(const, coeff)
)
val modelCell = NamedCell("model", cell)
val action = a.argmax(
  m.link.softmax(
    model.reg.linear(input, modelCell.ref)
  )
)

val pfa: PFADocument = PFABuilder()
  .withName("logistic-regression-model")
  .withInput[Seq[Double]]
  .withOutput[Double]
  .withCell(modelCell)
  .withAction(action)
  .pfa
```

Aardpfark - Challenges

- Spark ML Model has no schema knowledge
  - E.g. `Binarizer` can operate on numeric or vector columns
  - Need to use Avro union types for standalone PFA components and handle all cases in the action logic
- Combining components into a pipeline
  - Trying to match Spark's DataFrame-based input/output behavior (typically appending columns)
  - Each component is wrapped as a user-defined function in the PFA document
  - Current approach mimics passing a Row (i.e. Avro record) from function to function, adding fields
- Missing features in PFA
  - Generic vector support (mixed dense/sparse)

```
`type`()
.unionOf().array().items().doubleType().and()
.doubleType().endUnion()

Cast(inputExpr, Seq(asDouble, asArray))
```

Related Open Standards

PMML

- Data Mining Group (DMG)
- Model interchange format in XML with operators
- Widely used and supported; open standard
- Spark support lacking natively but 3rd party projects available: jpmml-sparkml
  - Comprehensive support for Spark ML components (perhaps surprisingly!)
  - Watch SPARK-11237
- Other exporters include scikit-learn, R, XGBoost and LightGBM

- Shortcomings
  - Cannot represent arbitrary programs / analytic applications
  - Flexibility comes from custom plugins => lose benefits of standardization

MLeap

- Created by Combust.ML, a startup focused on ML model serving
- Model interchange format in JSON / Protobuf
- Components implemented in Scala code
- Good performance
- Initially focused on Spark ML. Offers almost complete support for Spark ML components
- Recently added some sklearn; working on TensorFlow

- Shortcomings
  - “Open” format, but not a “standard”
  - No concept of well-defined operators / functions
  - Effectively forces a tight coupling between versions of model producer / consumer
  - Must implement custom components in Scala
  - Impossible to statically verify serialized model

Open Neural Network Exchange (ONNX)

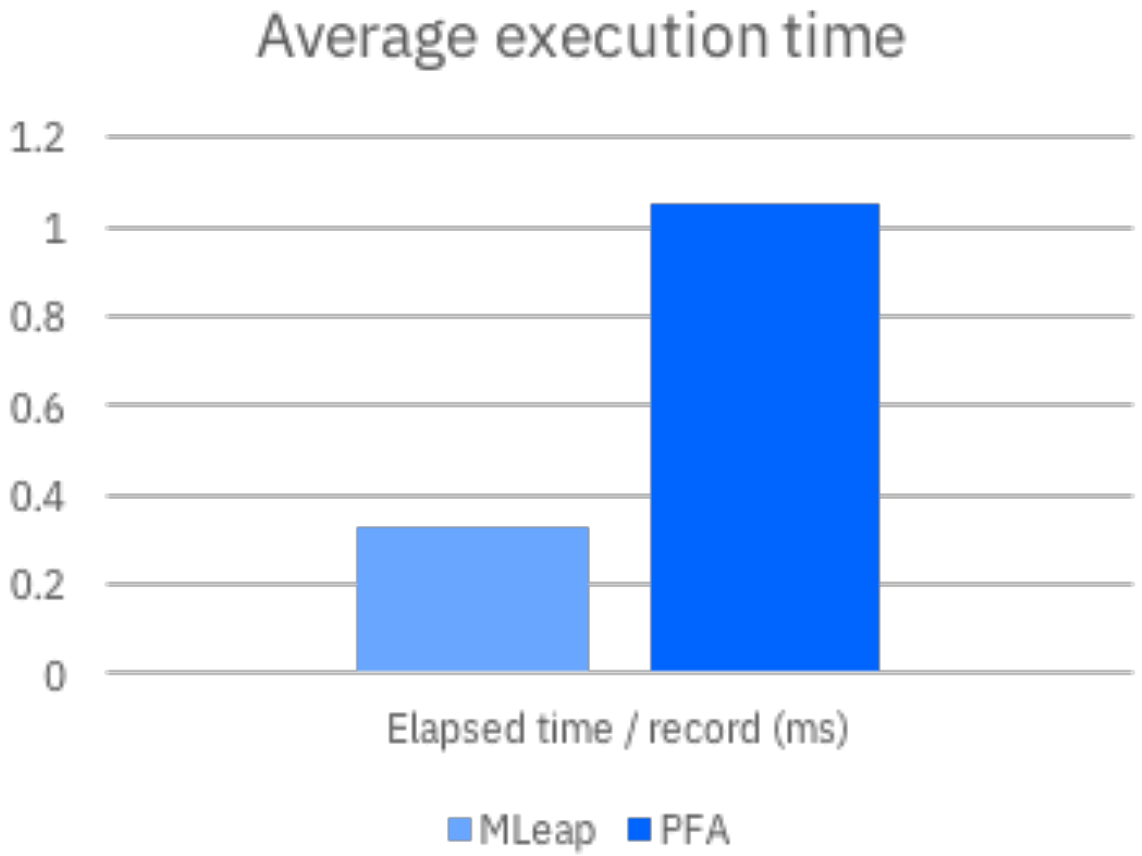
- Championed by Facebook & Microsoft
- Protobuf serialization format
- Describes computation graph (including operators)
  - In this way the serialized graph is “self-describing” similarly to PFA
- More focused on Deep Learning / tensor operations
- Will be baked into PyTorch 1.0.0 / Caffe2 as the serialization & interchange format

- Shortcomings
  - No or poor support for more “traditional” ML or language constructs (currently)
    - Tree-based models & ensembles
    - String / categorical processing
    - Control flow
    - Intermediate variables

Performance

Scoring Performance Comparison

- Comparing scoring performance of PFA with Spark and MLeap
- PFA uses Hadrian reference implementation for JVM
- Test dataset of ~80,000 records
  - String indexing of 47 categorical columns
  - Vector assembling the 47 categorical indices together with 27 numerical columns
  - Linear regression predictor
- *Note: Spark time is 1.9s / record (1901ms) - not shown on the chart*



Conclusion

Summary

- PFA provides an open standard for serialization and deployment of analytic workflows
  - True portability across languages, frameworks, runtimes and versions
  - Execution environment is independent of the producer (R, scikit-learn, Spark ML, weka, etc)
- Solves a significant pain point for the deployment of Spark ML pipelines
- Also benefits the wider ecosystem
  - e.g. many currently use PMML for exporting models from R, scikit-learn, XGBoost, LightGBM, etc.

- However there are risks
  - PFA is still young and needs to gain adoption
  - Performance in production, at scale, is relatively untested
  - Tests indicate PFA reference engines need some work on robustness and performance
  - What about Deep Learning / comparison to ONNX?
  - Limitations of PFA
  - A standard can move slowly in terms of new features, fixes and enhancements

Open-sourcing Aardpfark

- Coverage
  - Almost all predictors (ML models)
  - Many feature transformers
  - Pipeline support (still needs work)
  - Equivalence tests Spark <-> PFA
  - Tests for core Scala DSL
- Need your help!
  - Finish implementing components
  - Improve pipeline support
  - Complete Scala DSL for PFA
  - Python support
  - Tests, docs, testing it out!

https://github.com/CODAIT/aardpfark

Future directions

- Initially focused on Spark ML pipelines
  - Later add support for scikit-learn pipelines, XGBoost, LightGBM, etc
  - (Support for many R models exist already in the Hadrian project)
- Further performance testing in progress vs Spark & MLeap; also test vs PMML
- More automated translation (Scala -> PFA, ASTs etc)
- Propose improvements to PFA
  - Generic vector (tensor) support
  - Less cumbersome schema definitions
  - Performance improvements to scoring engine
- PFA for Deep Learning?
  - Comparing to ONNX and other emerging standards
  - Better suited for the more general pre-processing steps of DL pipelines
  - Requires all the various DL-specific operators
  - Requires tensor schema and better tensor support built-in to the PFA spec
  - Should have GPU support

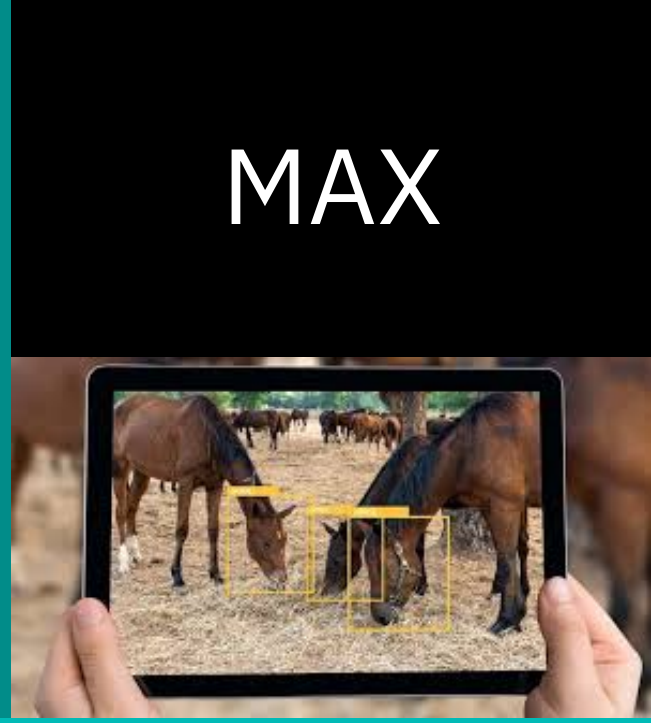
Thank you!

codait.org

twitter.com/MLnick

github.com/MLnick

developer.ibm.com/code



Sign up for IBM Cloud and try Watson Studio!

https://datascience.ibm.com/

Links & References

Portable Format for Analytics

PMML

Spark MLlib – Saving and Loading Pipelines

Hadrian – Reference Implementation of PFA Engines for JVM, Python, R

jpmml-sparkml

MLeap

Open Neural Network Exchange

IBM Sessions at Spark+AI Summit 2018 (Tuesday, June 5)

Date, Time, Location & Duration	Session title and Speaker
Tue, June 5 | 11 AM 2010-2012, 30 mins	**Productionizing Spark ML Pipelines with the Portable Format for Analytics Nick Pentreath (IBM)**
Tue, June 5 | 2 PM 2018, 30 mins	Making PySpark Amazing—From Faster UDFs to Dependency Management and Graphing! Holden Karau (Google) Bryan Cutler (IBM)
Tue, June 5 | 2 PM Nook by 2001, 30 mins	Making Data and AI Accessible for All Armand Ruiz Gabernet (IBM)
Tue, June 5 | 2:40 PM 2002-2004, 30 mins	Cognitive Database: An Apache Spark-Based AI-Enabled Relational Database System Rajesh Bordawekar (IBM T.J. Watson Research Center)
Tue, June 5 | 3:20 PM 3016-3022, 30 mins	Dynamic Priorities for Apache Spark Application's Resource Allocations Michael Feiman (IBM Spectrum Computing) Shinnosuke Okada (IBM Canada Ltd.)
Tue, June 5 | 3:20 PM 2001-2005, 30 mins	**Model Parallelism in Spark ML Cross-Validation Nick Pentreath (IBM) Bryan Cutler (IBM)**
Tue, June 5 | 3:20 PM 2007, 30 mins	Serverless Machine Learning on Modern Hardware Using Apache Spark Patrick Stuedi (IBM)
Tue, June 5 | 5:40 PM 2002-2004, 30 mins	Create a Loyal Customer Base by Knowing Their Personality Using AI-Based Personality Recommendation Engine; Sourav Mazumder (IBM Analytics) Aradhna Tiwari (University of South Florida)
Tue, June 5 | 5:40 PM 2007, 30 mins	Transparent GPU Exploitation on Apache Spark Dr. Kazuaki Ishizaki (IBM) Madhusudanan Kandasamy (IBM)
Tue, June 5 | 5:40 PM 2009-2011, 30 mins	Apache Spark Based Hyper-Parameter Selection and Adaptive Model Tuning for Deep Neural Networks Yonggang Hu (IBM) Chao Xue (IBM)

IBM Sessions at Spark+AI Summit 2018 (Wednesday, June 6)

Date, Time, Location & Duration	Session title and Speaker
Wed, June 6 | 12:50 PM	Birds of a Feather: Apache Arrow in Spark and More Bryan Cutler (IBM) Li Jin (Two Sigma Investments, LP)
Wed, June 6 | 2 PM 2002-2004, 30 mins	**Deep Learning for Recommender Systems Nick Pentreath (IBM))**
Wed, June 6 | 3:20 PM 2018, 30 mins	Bringing an AI Ecosystem to the Domain Expert and Enterprise AI Developer Frederick Reiss (IBM) Vijay Bommireddipalli (IBM Center for Open-Source Data & AI Technologies)

Meet us at IBM booth in the Expo area.