WEBINAR
Productionizing Machine Learning: From Deployment to Drift Detection

Clemens Mewald,
Director of Product Management,
Databricks

Joel Thomas,
Senior Solution Architect,
Databricks
Logistics

- We can’t hear you…
- Recording will be available…
- Slides will be available…
- Code samples and notebooks will be available…
- Submit your questions…
- Bookmark databricks.com/blog
**VISION**
Accelerate innovation by unifying data science, engineering and business

**SOLUTION**
Unified Analytics Platform

**WHO WE ARE**
- Original creators of Spark, Delta Lake, and mllflow
- 2,000+ global companies use our platform across big data & machine learning lifecycle
About Today’s Presenters

Clemens Mewald, Director of Product Management at Databricks
- MSc in Computer Science from UAS Wiener Neustadt, Austria
- MBA from MIT Sloan
- Previously Product Lead on the Google Brain Team for TensorFlow, TFX
- Product Management Lead for Data Science and ML at Databricks

Joel Thomas, Senior Solutions Architect at Databricks
- MS in Electrical Engineering from University of Houston
- Previously Principal Data Scientist at Micron
- Architect and implement data & AI solutions for customers
Outline

// ML Development Challenges

// How MLflow, Delta Lake, + Databricks tackles these

// Real Life End to End Data + ML Lifecycle

// Demo of Model Drift Detection on Databricks

// Q&As
ML Lifecycle and Challenges

Raw Data → ETL → Featurize → Train → Score/Serve
Batch + Realtime → Monitor Alert, Debug

- Production Logs
- Update Features
- Retrain

Zoo of Ecosystem Frameworks

Tuning
Deploy
Model Mgmt

Collaboration
Scale
Governance

- Feature Repository
- Experiment Tracking
- AutoML, Hyper-p. search
- Remote Cloud Execution
- Project Mgmt (scale teams)
- Model Exchange
- A/B Testing
- CI/CD/Jenkins push to prod
- Orchestration (Airflow, Jobs)
- Lifecycle mgmt.
- Data Drift
- Model Drift

datarbricks
Unifying the End-to-end Data & ML Lifecycle

Overview

Prep Data
- DELTA LAKE
- AutoML
- Hyperopt
- Distributed Training
- Runtime and Environments

Build Model
- Partnerships

Deploy/Monitor Model
- Online Serving
- Batch Scoring
- BI & Reporting

End to end ML lifecycle
Unifying the End-to-end Data & ML Lifecycle

Overview

Prep Data

Build Model

AutoML
Hyperopt
Distributed Training

Partnerships

BI & Reporting

Distributed Training

Online Serving

Batch Scoring

Runtime and Environments

End to end ML lifecycle

mlflow

Open, pluggable architecture
Reliability - High Quality Data

- Transactions and schema enforcement
- Data quality validation and expectation

Performance - Fast Queries at Scale

- Optimizations - data skipping, caching & indexing
- Compaction optimizes file layout

Open Standard - Easier Adoption

- Open source, open format; compatible with Spark API’s
- Data stays in your data lake under your control
Unifying the End-to-end Data & ML Lifecycle

Overview

- **Prep Data**
- **Build Model**
  - AutoML
  - Hyperopt
  - Distributed Training
  - Partnerships
  - Runtime and Environments
- **Deploy/Monitor Model**
  - Online Serving
  - Batch Scoring
  - BI & Reporting

Open, pluggable architecture

End to end ML lifecycle

mlflow
**ML environment and Model Training**

- **Pre-configured Environment**
- **Customization**
  - Customized Environments using Conda
  - `requirements.txt`
  - `conda.yaml`
- **Built-In AutoML and Experiment Tracking**
  - AutoML and Tracking / Visualizations with MLflow
- **Built-in Optimization for Distributed Deep Learning**
  - Distribute and Scale any Single-Machine ML Code to 1,000's of machines.
Unifying the End-to-end Data & ML Lifecycle

Overview

- Prep Data
- Build Model
- Deploy/Monitor Model

- AutoML
- Hyperopt
- Distributed Training
- Partnerships
- Runtime and Environments

- Online Serving
- Batch Scoring
- BI & Reporting

End to end ML lifecycle

mlflow
An open source framework for the complete ML Lifecycle

Simplified experiment tracking, reproducibility, and deployment, from experimentation to production

- **mlflow Tracking**: Record and query experiments: code, data, config, results
- **mlflow Projects**: Packaging format for reproducible runs on any platform
- **mlflow Models**: General model format that supports diverse deployment tools

mlflow.org  |  github.com/mlflow  |  twitter.com/MLflow
Unifying the End-to-end Data & ML Lifecycle

Overview

Prep Data

Build Model

- AutoML
- Hyperopt
- Distributed Training
- Runtime and Environments
- Partnerships

Deploy/Monitor Model

- BI & Reporting
- Online Serving
- Batch Scoring

Open, pluggable architecture

End to end ML lifecycle

mlflow
Model Deployment and Monitoring

- **Model Format**
  - Flavor 1
  - Flavor 2

- **Simple model flavors usable by many tools**

- **ML Libraries**
  - mlflow
  - TensorFlow
  - Apache Spark
  - H2O.ai
  - PyTorch
  - Keras
  - mleap

- **In-Line Code**
- **Containers**
- **Batch & Stream Scoring**
- **Cloud Inference Services**
What happens after Deployment?

Heraclitus (paraphrased):
“Change is the only constant in life”
Types of change to worry about in Machine Learning

Concept Drift

- Statistical properties of target variable change (i.e. what you are trying to predict changes)
- E.g. fraud detection
Types of change to worry about in Machine Learning

Concept Drift
- Statistical properties of target variable change (i.e. what you are trying to predict changes)
- E.g. fraud detection

Data Drift
- Statistical properties of input variables change
- E.g. seasonality, personal preferences, trends change
Types of change to worry about in Machine Learning

Concept Drift
- Statistical properties of target variable change (i.e. what you are trying to predict changes)
- E.g. fraud detection

Data Drift
- Statistical properties of input variables change
- E.g. seasonality, personal preferences, trends change

Upstream Data Changes
- Encoding of a feature changes (e.g. switch from Fahrenheit to Celsius)
- Features are no longer being generated (leads to missing values)
Ways to detect and protect against changes

Monitor

Training Data

► Schema & distribution of incoming data
► Distribution of labels

Requests & Predictions

► Schema & distribution of requests
► Distribution of predictions
► Quality of predictions
Ways to detect and protect against changes

Monitor

Training Data
- Schema & distribution of incoming data
- Distribution of labels

Requests & Predictions
- Schema & distribution of requests
- Distribution of predictions
- Quality of predictions

Intervene

Correct Data Problems
- Detect and correct upstream data problems early
- Clean up erroneous labels

Retrain / Calibrate Model
- In cases where there is concept or data drift, keep models fresh by retraining them
Your Machine Learning Models become stale

Model Staleness over time

Model Quality

Time
Your Machine Learning Models become stale

Model Staleness over time

Refreshing models over time
Real-Life Examples of Drift

- **IoT**
  - Sensor readings can change over time (that’s why sensors need to be recalibrated)
  - Sensor readings may change with seasonality

- **Retail**
  - Inventory changes over time
  - Customer preferences change over time
  - Regional changes and trends

- **Media & Entertainment**
  - Content changes every second
  - Today’s patterns may not apply tomorrow
From Data Prep to Production: Real Life IoT Example
Process and Prepare Data at Scale

Bronze
- Raw data
  - Lots and lots of small files on BLOB storage

Streaming
- Data enriched with originating filepath, derived data type and year, month, date (for partitioning)
- Checkpointed for fault tolerance & file listings
- ~150k records per second

Silver
- Single Delta Table
  - Consistent schema for ML/Analytics
  - 70 Billion rows of data
  - Query performance improvement of 5.08 hours to 30.13 minutes following an OPTIMIZE

Gold
- Time series resampled & interpolated
- Feature transpose
- Feature reduction

From Data Prep to Production: Real Life IoT Example
Process and Prepare Data at Scale
From Data Prep to Production: Real Life IoT Example
Train, Deploy, and Monitor at Scale

Training data
160k sensors / 60 billion rows

Trained models
100k

Predictions
9 million in 1 hour on a 4 node cluster

Model Training
Train an sklearn model per sensor via a Pandas UDF

Model Inference
Apply an sklearn model per sensor via a Pandas UDF

Model Monitoring
Compare predicted value to rolling average via a Pandas UDF

Log params, metrics & model to MLflow
Select & load model from MLflow

Gold feature tables
DELTA LAKE™

Training data
160k sensors / 60 billion rows

Trained models
100k

Predictions
9 million in 1 hour on a 4 node cluster

Model Training
Train an sklearn model per sensor via a Pandas UDF

Model Inference
Apply an sklearn model per sensor via a Pandas UDF

Model Monitoring
Compare predicted value to rolling average via a Pandas UDF

Log params, metrics & model to MLflow
Select & load model from MLflow

Gold feature tables
DELTA LAKE™

From Data Prep to Production: Real Life IoT Example
Train, Deploy, and Monitor at Scale

Training data
160k sensors / 60 billion rows

Trained models
100k

Predictions
9 million in 1 hour on a 4 node cluster

Model Training
Train an sklearn model per sensor via a Pandas UDF

Model Inference
Apply an sklearn model per sensor via a Pandas UDF

Model Monitoring
Compare predicted value to rolling average via a Pandas UDF

Log params, metrics & model to MLflow
Select & load model from MLflow

Gold feature tables
DELTA LAKE™
From Data Prep to Production: Real Life IoT Example

1) Training at Scale

```
models = df.groupby("SensorId").apply(train_sklearn)
```
From Data Prep to Production: Real Life IoT Example

2) Deploying at Scale

1 line of code to apply UDF to $n$ new sensor readings

```
predictions = df.groupby("SensorId").apply(apply_sklearn_model)
```

Write out our predictions to a Delta table

```
predictions.write.format("delta")
.mode("append").partitionBy("SensorId").save(predictions_path)
```
From Data Prep to Production: Real Life IoT Example

2) Deploying at Scale

Sensor 2  Sensor 1  Sensor 3
Data     Data     Data
Model 2  Model 1  Model 3

Sensor 1  Predict 1  Predict 1
Sensor 2  Predict 2  Predict 2
Sensor 3  Predict 3  Predict 3
...
...
...

Predictions:

July: Predict 1, Predict 2, Predict 3, ...
August: Predict 1, Predict 2, Predict 3, ...

mlflow Model 1 Model 2 Model 3 ...

databricks ML Runtime

Sensor 1
Sensor 2
Sensor 3
...

Databricks

Delta Lake
From Data Prep to Production: Real Life IoT Example

3) Monitoring at Scale

```
# Write out our notifications to a table
notifications.write.format("delta")
  .mode("append").partitionBy("SensorId").save(notifications_path)
```

Predict Trend 1
Predict Trend 2
Predict Trend 3

Alert 1
Alert 2
...
Demo
Who wants a perfect cake?

Parameters like temperature and duration have an impact on quality of cake.
Glassware Manufacturing Dataset

Data streaming from IoT sensors
- Temperature
- Pressure
- Duration

These parameters in ideal combination gives good quality product.

**Goal:** Predict product quality inline to be used for additional manual inspection

Synthesized dataset to showcase model drift
Deployment to Drift Detection - a Typical Workflow

1. To understand the data, we start with EDA (Exploratory Data Analysis)
2. Using historical data, we explore various modeling methods, tune its hyperparameters, and identify our best model
3. All the experiment runs are tracked using MLflow and we tag the best model for production use
4. While scoring in a streaming pipeline, production model is accessed from MLflow
5. Model is stable for first ‘x’ days
KPI to Monitor Model Quality

Model Drift KPIs:

- KPIs and its margin depends on the model and business problem
- Sometimes more than 1 KPI maybe needed at times to capture behavior changes
Deployment to Drift Detection - a Typical Workflow

- After ‘y’ days, we see model drift occur, as identified by tracking KPIs
- This triggers re-training process
Deployment to Drift Detection - a Typical Workflow

• Once again, we explore various modeling methods, tune its hyperparameters, and identify our new best model
• The new model is tagged as current production model in MLflow
• We once again observe that KPIs are back within acceptable range
• Over time, based on business demands, it may be needed to update KPIs and its acceptable limits
Deployment to Drift Detection - a Typical Workflow
Summary - Deployment to Drift Detection

Model Monitoring KPI over time

1st model train data 1st model inference 2nd model learnt and applied

1st drift if model not updated 2nd drift

% in population

0 20 40 60 80 100


Date

accurate_prediction Accurate
inaccurate
Model Drift Demo: Source Code

https://github.com/joelcthomas/modeldrift
Summary

- The only constant is change. Concept drift, data drift, and upstream data problems all affect your ML pipeline.

- Monitoring data and ML model performance is critical.

- In many cases, your ML model needs to adapt to changes.

- Databricks provides an end-to-end platform to deploy, monitor, and automatically retrain your ML models.
Coming soon

**mlflow** Model Registry
Tagging, sharing and versioning models in the MLflow tracking server

- Register any model in MLflow Model format
- Deploy to serving systems
- Add metadata (e.g. tags/notes) and track model creators and users

---

**SPARK+AI SUMMIT EUROPE**
15 - 17 October 2019 | Amsterdam
New product announcements around Delta Lake, MLflow, etc.
Sign up at [databricks.com/sparkaisummit](https://databricks.com/sparkaisummit)
Q&A

Get Started - databricks.com/try
Contact us - databricks.com/contact