Seamless MLOps with Seldon and MLflow

Adaptive. Intelligent.
Agile. Intuitive.
Alive. Inspiring.
About me

Adrian Gonzalez-Martin
Machine Learning Engineer

agm@seldon.io
@kaseyo23
github.com/adriangonz
About Seldon

We are hiring!
seldon.io/careers/
Outline

➔ Why is MLOps hard?
➔ Training with MLflow
➔ Serving with Seldon
➔ Demo!
Why is MLOps hard?
What do we mean by MLOps?

Notebooks (by themselves) **don’t scale!**
What do we mean by MLOps?

And training is **not** the end goal!

**DISCLAIMER:** This is just a high-level overview!
What do we mean by MLOps?

There is a larger ML lifecycle

DISCLAIMER: This is just a high-level overview!
What do we mean by MLOps?

“MLOps (a compound of “machine learning” and “operations”) is a practice for collaboration and communication between data scientists and operations professionals to help manage production ML lifecycle.” [1]

Why is MLOps hard?

Managing the ML lifecycle is hard!

- Wide heterogeneous requirements
- Technically challenging (e.g. monitoring)
- Process needs to scale up across every ML model!
- Organizational challenge
  - The lifecycle needs to jump across many walls
Why is MLOps hard?

Organizational challenge

This is what DevOps tried to solve
Why is MLOps hard?

Organizational challenge

This is what MLOps has to solve

Data Engineering

Data Science

SW / ML Engineering

DevOps
How can we make MLOps better?

Breaking up siloes

→ Automate, automate, automate!
→ Measure and monitor everything
→ “Shift-left” on responsibilities, e.g.
  ◆ Data scientist to “own” training pipelines
  ◆ Data scientists “own” production models
→ We need tooling to allow this while keeping the infrastructure “hidden”
How can we make MLOps better?

What tools do we have available?

Awesome production machine learning

This repository contains a curated list of awesome open source libraries that will help you deploy, monitor, version, scale, and secure your production machine learning.

Quick links to sections in this page

How can we make MLOps better?

How can we make MLOps better?

Machine Learning Life Cycle

How can we make MLOps better?

We’ll focus on **training** and **serving**
Training with MLFlow
What is MLflow?

➔ Open Source project initially started by Databricks
➔ Now part of the LFAI

“MLflow is an open source platform to manage the ML lifecycle, including experimentation, reproducibility, deployment, and a central model registry.” [4]

What is MLflow?

1. Run experiment and track results

2. "Log" trained model

2.1. Trained model goes into a persistent store

3. Model registry keeps track of model iterations

MLflow Project

MLflow Tracking

MLflow Registry

Model V1

Model V2

Model V3

S3, Minio, DBFS, GCS, etc.
What is MLflow?

» MLflow Project
  ◆ Defines environment, parameters and model’s interface.

» MLflow Tracking
  ◆ API to track experiment results and hyperparameters.

» MLflow Model
  ◆ Snapshot / version of the model.

» MLflow Registry
  ◆ Keeps track of model metadata
How does MLflow work?

**MLproject** file and training

```yaml
name: mlflow-talk
conda_env: conda.yaml
entry_points:
  main:
    parameters:
      alpha: float
      l1_ratio: {type: float, default: 0.1}
    command: "python train.py \{alpha\} \{l1Ratio\}"
```

```
$ mlflow run ./training -P alpha=0.5
$ mlflow run ./training -P alpha=1.0
```
How does MLflow work?

**MLmodel** snapshot

```
MLmodel

artifact_path: model
flavors:
    python_function:
        data: model.pkl
        env: conda.yaml
        loader_module: mlflow.sklearn
        python_version: 3.6.9
sklearn:
    pickled_model: model.pkl
    serialization_format: cloudpickle
    sklearn_version: 0.19.1
run_id: 5a6be5a1ef844783a50a6577745dbdc3
utc_time_created: '2019-10-02 14:21:15.783806'
```
How does MLflow work?
Serving with Seldon
What is Seldon Core?

→ Open Source project created by Seldon

“An MLOps framework to package, deploy, monitor and manage thousands of production machine learning models” [6]

What is Seldon Core?

1. Containerise
2. Deploy
3. Monitor

From model binary
Or language wrapper
Into fully fledged microservice

Simple Seldon Core Inference Graph

API (REST, gRPC)

Model A

Complex Seldon Core Inference Graph

API (REST, gRPC)

Model B
Model A
Model C

Multi Armed Bandit
Feature Transformation
Outlier Detection
Explanation

Direct traffic to the most optimal model
Key features to identify outlier anomalies (Fraud, KYC)
Why is the model doing what it’s doing?

Outlier Detection
Key features to identify outlier anomalies (Fraud, KYC)

Why is the model doing what it’s doing?
What is Seldon Core?

Cloud Native

➔ Built on top of Kubernetes Cloud Native APIs.
➔ All major cloud providers are supported.
➔ On-prem providers such as OpenShift are also supported.

How does Seldon Core work?

SeldonDeployment Pod

`kubectl apply -f deployment.yaml`

Downloads model artifacts

Orchestrator requests between components

init-container

input-transformer

my-model

classifier

model-combiner

orchestrator

deployment.yaml

apiVersion: machinelearning.seldon.io/v1
kind: SeldonDeployment
metadata:
  name: example-model
spec:
  name: example
  predictors:
    - componentSpecs:
      - spec:
        containers:
          - image: model:0.1
            name: my-model
          - image: transformer:0.1
            name: input-transformer
          - image: combiner:0.1
            name: model-combiner
  graph:
    name: input-transformer
    type: TRANSFORMER
    children:
      - name: model-combiner
        type: COMBINER
        children:
          - name: my-model
            type: MODEL
          - name: classifier
            implementation: MLFLOW_SERVER
            modelUri: gs://seldon-models/mlflow/model-a
  name: default
  replicas: 1
What is Seldon Core?

**SeldonDeployment CRD** to manage ML deployments

- Abstraction for Machine Learning deployments: **SeldonDeployment CRD**
  - Simple enough to deploy a model only pointing to stored weights
  - Powerful enough to keep full control over the created resources
- Pre-built *inference servers* for a subset of ML frameworks
  - Ability to write custom ones
- A/B tests, shadow deployments, etc.
- Integrations with *Alibi explainers*, *outlier detectors*, etc.
- Tools and integrations for *monitoring*, *logging*, *scaling*, etc.
How does Seldon Core work?

Inference Servers

→ Pre-packaged servers for common ML frameworks.

How does Seldon Core work?

Monitoring

→ Seldon integrates with **Prometheus** for metrics
→ Out of the box: **memory**, **CPU**, **latency**, etc.
→ **Custom metrics** are also supported

How does Seldon Core work?

Monitoring

→ Seldon leverages KNative for (more advanced) **async monitoring pipelines**
  
  ◆ **Outlier detection** (through Alibi Detect)
  ◆ **Drift detection** (through Alibi Detect)
  ◆ **Custom metrics**

How does Seldon Core work?

Monitoring

→ Seldon leverages KNative for (more advanced) async monitoring pipelines

How does Seldon Core work?

Monitoring

How does Seldon Core work?

Auditability

→ We can leverage a similar pattern to log each prediction request

How does Seldon Core work?

Auditability

→ And / or some attributes of each instance (e.g. outliers)

How does Seldon Core work?

Auditability

How does Seldon Core work?

Advanced Deployment Models

- **A/B Tests**
  - We'll see this in the demo!

- **Shadow Deployments**

Demo!

https://github.com/adriangonz/mlflow-talk
Demo

Wine e-commerce

➔ We want to predict wine quality for new wines
➔ We want to listen to feedback from customers
### Wine quality dataset

<table>
<thead>
<tr>
<th>Fixed Acidity</th>
<th>Volatile Acidity</th>
<th>Citric Acid</th>
<th>...</th>
<th>Sulphates</th>
<th>Alcohol</th>
<th>Quality</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>0.27</td>
<td>0.36</td>
<td></td>
<td>0.45</td>
<td>8.8</td>
<td>6</td>
</tr>
<tr>
<td>6.3</td>
<td>0.3</td>
<td>0.34</td>
<td></td>
<td>0.49</td>
<td>9.5</td>
<td>7</td>
</tr>
<tr>
<td>8.1</td>
<td>0.28</td>
<td>0.4</td>
<td></td>
<td>0.44</td>
<td>10.1</td>
<td>1</td>
</tr>
<tr>
<td>7.2</td>
<td>0.23</td>
<td>0.32</td>
<td></td>
<td>0.4</td>
<td>9.9</td>
<td>2</td>
</tr>
<tr>
<td>7.2</td>
<td>0.23</td>
<td>0.32</td>
<td></td>
<td>0.4</td>
<td>9.9</td>
<td>5</td>
</tr>
</tbody>
</table>

...
Demo

ElasticNet

→ Linear regression with L1 and L2 regularisers.

→ Two hyperparameters: \( \{a, b\} \)

\[
y = x^T \hat{\beta} \\
\hat{\beta} = \arg \min_{\beta} \|y - X\beta\| + a \|\beta\|_1 + b \|\beta\|_2^2
\]
Demo

Training

1. run experiment and track results

2. "log" trained model

2.1. trained model goes into a persistent store

MLflow Tracking

Wine Project

Model A

Model B

GCS
Demo

Hyperparameter setting?

➔ Train two versions of ElasticNet
➔ It’s not clear which one would be better in production
➔ Deploy both and do A/B test based on user’s feedback

Which one is best?
Demo

Serving

SeldonDeployment CR

A/B Test Router

model-a

model-b

GCS

Model A

Model B

mlflow

mlflow

Router
Demo

Feedback

→ Reward signal based on “proxy metric”, e.g.
  ◆ Sales
  ◆ User rating
Demo

Feedback

$\Rightarrow$ We can build a **rough reward signal** using the **squared error**

$$R(x_n) = \begin{cases} \frac{1}{(y_n-f(x_n))^2}, & y_n \neq f(x_n) \\ 500, & y_n = f(x_n) \end{cases}$$
Seldon Analytics
Demo

https://github.com/adriangonz/mlflow-talk
Thanks!

agm@seldon.io
@kaseyo23
github.com/adriangonz

We are hiring!
seldon.io/careers/