SageMaker In-Person Workshop

Notes from AWS SageMaker Workshop in January 2023.

Intro

Wednesday 25/01/2023

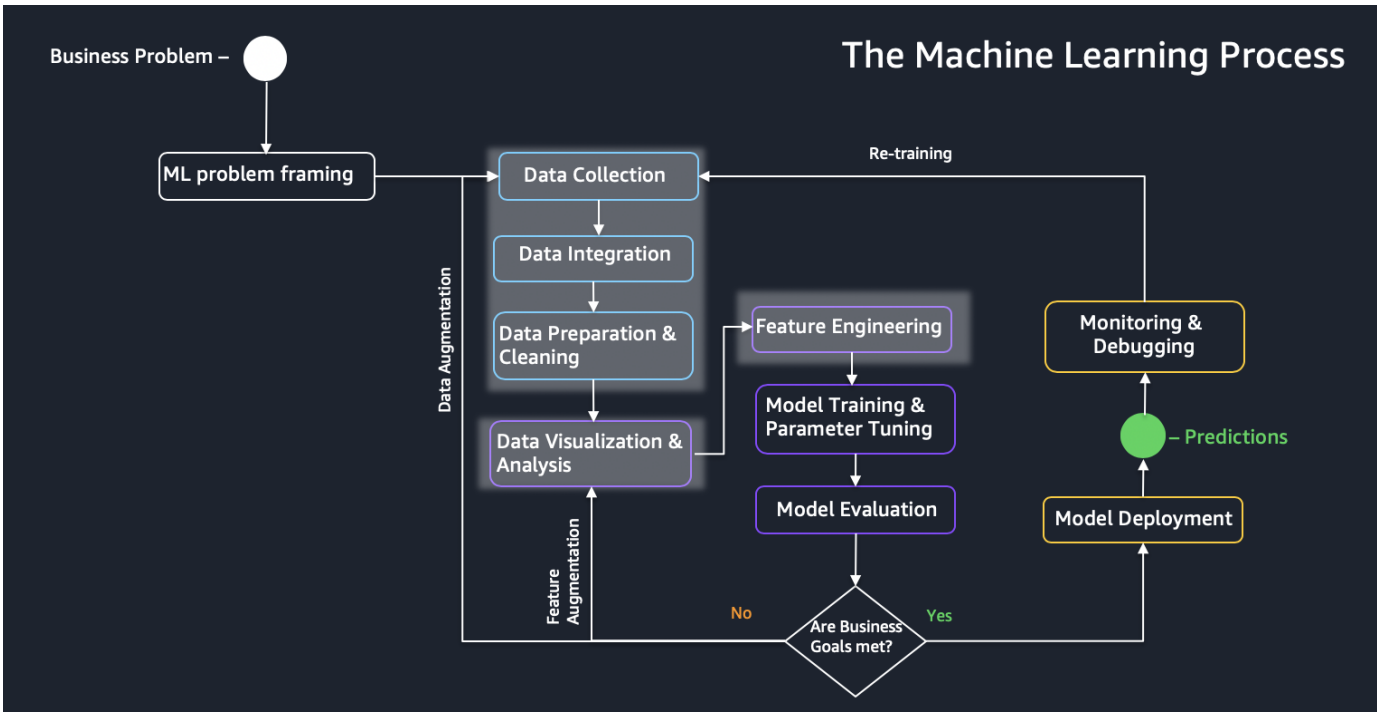
Time (GMT)	Subject	Comments
13:00-13:45	Welcome and introduction to SageMaker	presentation
13:45-14:15	Feature engineering in SageMaker, focusing on processing jobs.	presentation
14:15-14:30	Connect to the AWS accounts	lab
14:30-15:00	Hands on lab: Processing jobs	lab
15:00-15:15	break	
15:15-15:45	Training in SageMaker with a focus on script mode. It includes an overview of SageMaker deployments	presentation
15:45-16:30	Hands on lab: training script mode + 15 min extra in case of delays	lab
16:30-16:45	Session closing + Q&A and next steps	open session

Friday 27/01/2023

Time (GMT)	Subject	Comments
13:00-13:30	Welcome back, recap of previous day and potential Q&A	presentation
13:30-14:00	SM Experiments	presentation
14:00-14:30	Hands on SageMaker Experiments	lab
14:30 - 15:00	MLOps: Pipelines and model registry	presentation
15:00-15:15	break	
15:15 - 16:00	MLOps: Pipelines and model registry	lab
16:00 - 17:00	Collaborate between users and latest SageMajer features	presentation + demo
16:15-17:00	Next steps discussion, Q&A and architecture discussion	open session

- Amazon SageMaker Studio
  - Multiple domains can be created

Data Preparation



Different ways of preparing data

- SageMaker Data Wrangler (no-code/low-code)
  - built-in data preparation capability in notebooks
- SageMaker processing
  - scalable data processing workloads
- SageMaker feature store
  - for storing sharing and retrieving ML featured (real time or batch)

Data preparation

Options for data prep:

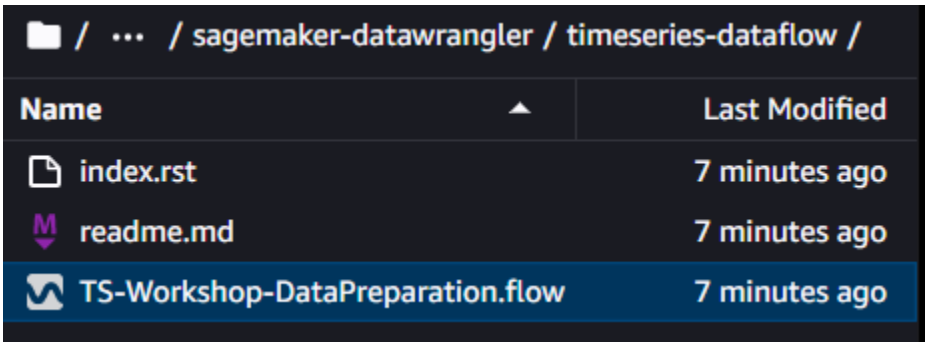
A. Data Wrangler

B. SageMaker studio notebooks (for experimenting, good for interactivity functionaility given by individual notebook cells)

C. Processing jobs (once experiments accomplished, just check output)

A. Data Wrangler

`.flow` files



- cleanse and explore data
- visualise and understand data
- enrich data

B. SageMaker Processing

- data processing and model evaluation tool
- custom scripts for feature engineering
- take script+ take data store output used for building model or feature store
- recommendation:
  - provide script using lirbaries like pandas, numpy
  - into an already provided container
  - if doesn't work: provide own container (more complex, more maintenance)
- can leverage security and compliance feature functionality provided
- automatically created/configured/terminated instance

C. Processing Jobs

https://catalog.us-east-1.prod.workshops.aws/workshops/63069e26-921c-4ce1-9cc7-dd882ff62575/en-US/lab1/option3

- `from sagemaker.processing import ProcessingInput, ProcessingOutput`
- job needs:
  - container:
    - instance type (defines number of cpu cores, memory)
    - number of instances
  - code:
    - python script (`.py`)
  - locations:
    - input data path in container
    - output data path in container
- need to adapt where data is collected from, and where it's saved to, sagemaker takes care of rest:
  - manually upload data in S3 bucket if it's local using `Session`
  - convert steps from notebook cells to a pythpn script using `%%writefile` magic command
- automatically saved back to S3 at the end of the job, so that they can be collected either in a pipeline or in a notebook
- notebooks VS processing job:
  - notebooks = developing using sample of dataset
  - processing = scaling using whole dataset
- Framework processors: sklearn, pyspark, tensorflow, xgboost, etc.

Can see summary of jobs from the console (outcome pass/fail, metadata, …)

Example notebook: https://github.com/aws-samples/amazon-sagemaker-immersion-day/blob/master/processing_xgboost.ipynb

Notebook cloned and run/modified during the hands-on lab:

D. FeatureStore

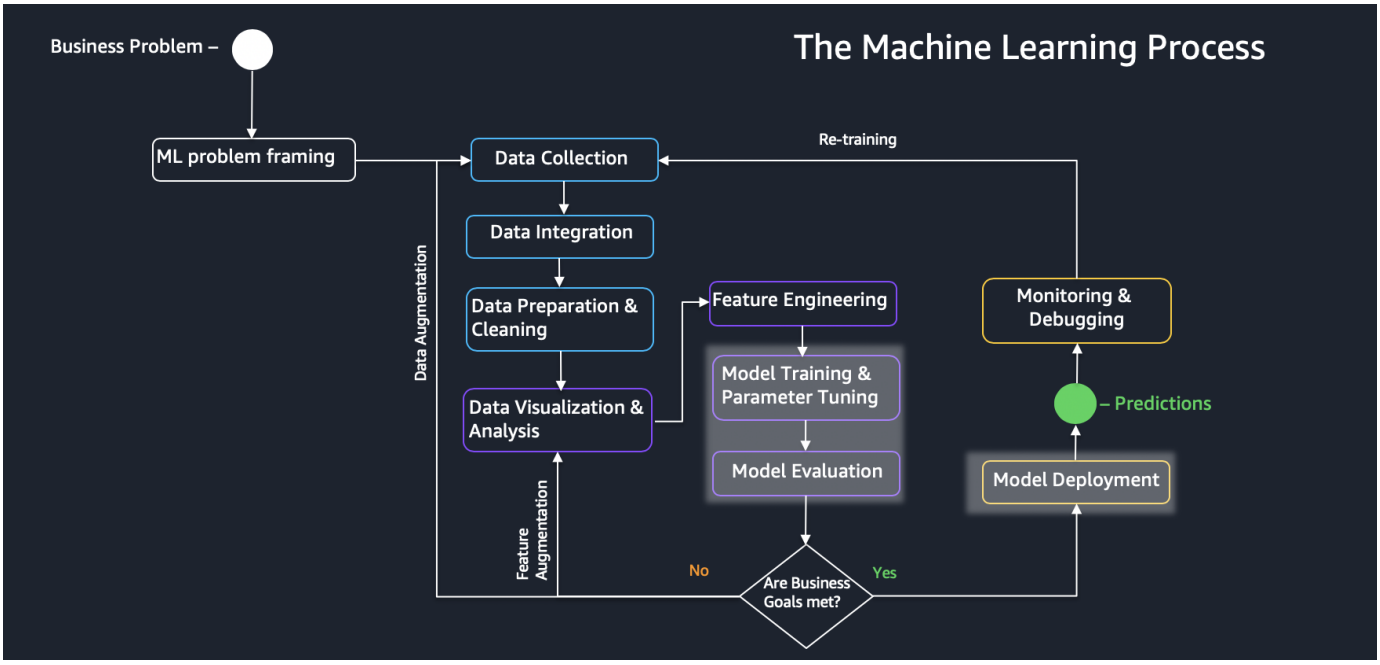
Pros:

- feature processing can be a Processing job (e.g., DataWrangler)
- keep meta data of features
- feature re-use

Online VS Offline feature store:

- online: real-time predictions
- offline: for training and batch predictions

Training



Pros: fully managed by SM, distributed, automatically scaled, secured

Keeping same containers increases efficiency as faster to spin up

PS: container refers to Docker containers

- container is a box that contains everything needed to train model (libraries, algorithms, data if not on S3)

- box can be moved around easily

1. Training Options

Multiple training options exist. See 3 options below:

A. Built-in

= *No control*

Model options that already exist:

- xgboost
- matrix factorisation
- regression
- pca
- k-means clustering
- etc. *(17 total)*

all built-in in SageMaker, no ML coding required

B. Own Script

= *Partial control*

- If none of built-in options match, then use own script
- SageMaker builds container, just need script with algorithm
- e.g., extend container, then place extra libraries (such as TF, Keras, Sklearm, PyTorch) with specific version in container, then train

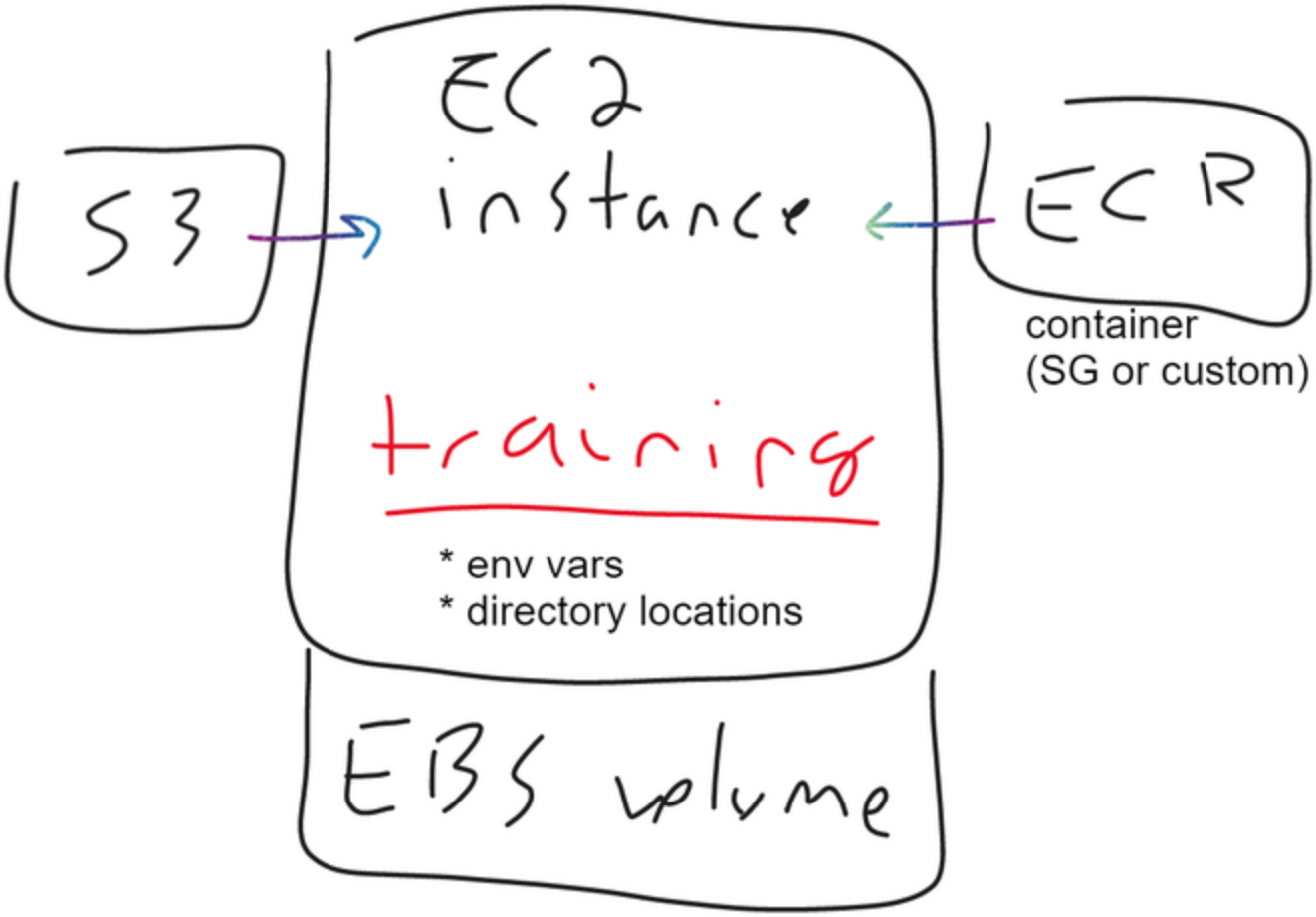
C. Own Container

= *Full control*

2 options:

- extend an existing contain from B
  - extend then add additional libraries
- full custom container, no framework works

2. Script mode



1. copy data from S3 EBS volume of EC2 training instance (where training will occur)
2. container (SM or custom) sits in Elastic container registry EC2 training instances
3. sagemaker train using the:
    a. different params set for algorithm in script
    b. environment variables
    c. saves output in defined location (fitted model, outputs) [in EBS volume]
        - in `/opt/ml/X` where X are sub directories such as `/input` `/model` `/code` `/output`
        - `/opt/ml/code` contains training code
        - turn off EC2 training instances to avoid unnecessary extra costs

> Can have multiple instances within same cluster and distribute it across multiple cores to speed up process.

3. Deployment Options

4 types of deployment options offered in SM:

Real-time

Real-time inference

- low latency
- on all the time
- can scale, but pay continuously
- used for detection of an event e.g., fraud
- A/B testing

Serverless inference

- auto-scaling
- faster than real-time

- used when have intermittent traffic (e.g., more in specific time, follow flow of traffic)
- pay only when running, but workload needs to support cold starts

Asynchronous inference

- real-time

Batch transform

- for large datasets
- higher throughput

> Need to save model/data/output in S3 so it can be later collected.

SageMaker Experiments

For organising, tracking and comparing ML models

Many different stuff to do in an ML pipeline: Experiments brings everything together

Capabilities

- track at scale (params, metrics across models)
- organise by team/goal
- visualise experiments and compare them
- metrics/logging

How to use

- setup experiments from notebooks (as well as SM Training/Autopilot/Pipelines)
- visually identify best models based on performance
- ensure models are reliable/stable

Concepts

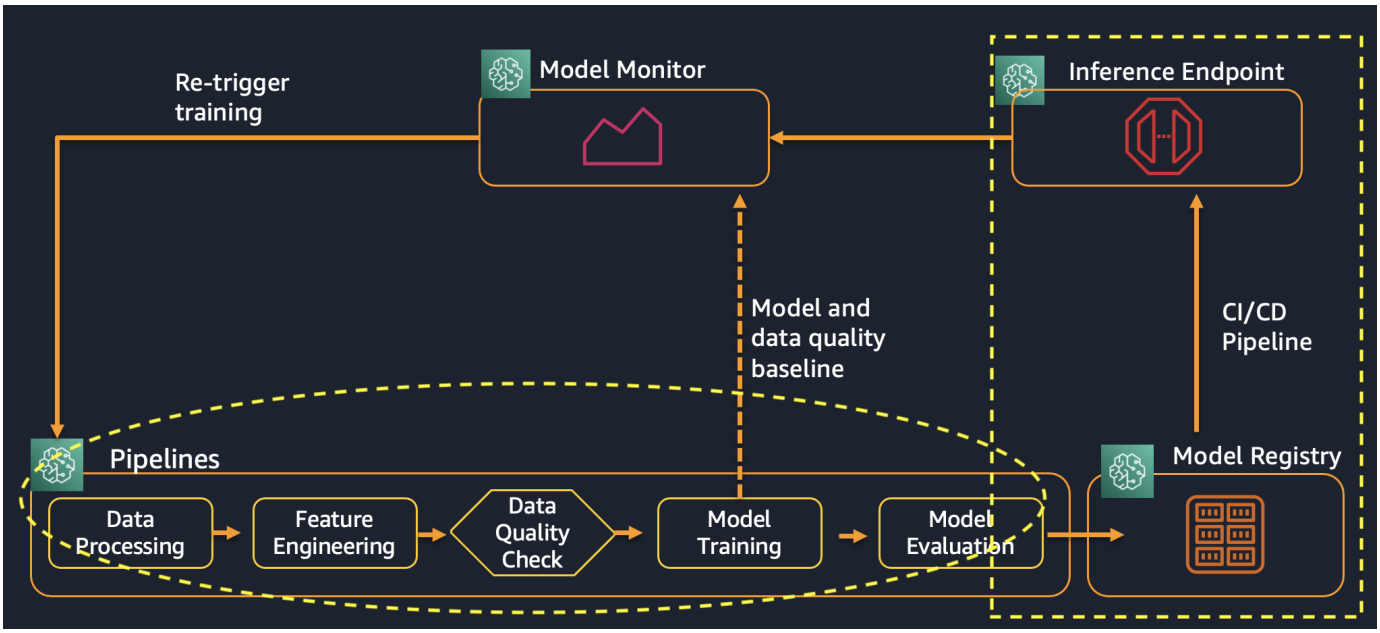
- Experiment = collection of runs (initialiase run in training loop, include experiment name unique to AWS account)
- Run = consists of a pipeline all inputs / parameters / configs / results for 1 iteration of model training. Initialise an experiment run for tracking a training job with `Run()`

Usage

- `from sagemaker.experiments.run import Run`
- in notebook, can:
  - create an experiment and run
    - `with Run(<exp_name>, <run_name>) as run:`
  - log parameters
    - `run.log_parameters()`
  - log metrics
    - `run.log_metrics()`
  - log artifacts
    - `run.log_file()`
    - `run.log_artifacts()`
  - log charts
    - `run.log_confusion_matrix()`
    - auc, roc, etc.

> Need `role` as it's used to know where to collect data/script/instances from in container and get correct authentication.

MLOps



Requirements

To produce something from the code, need more than just the ML code:

- config
- data collection
- data verification
- feature extraction
- machine resource management
- analysis tool
- process management tools
- serving infrastructure
- monitoring

MLOps combines of all this together and automates it

Customer challenges / ML Lifecycle / Personas

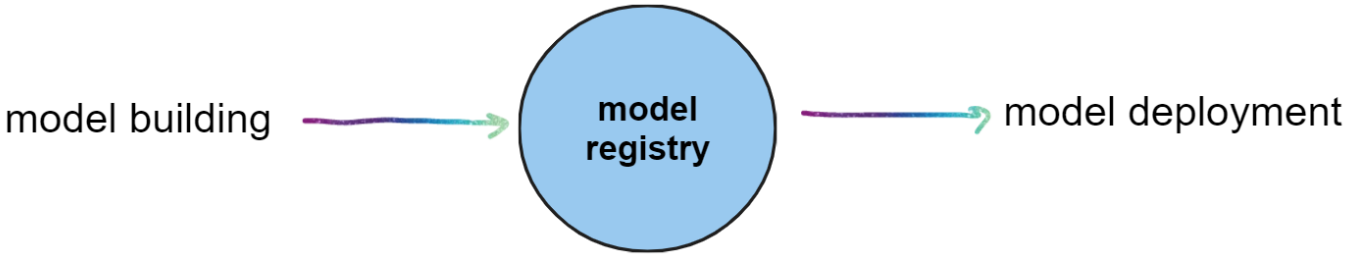
People	Operations	Technology
business analysists, data scientists, ML engineers, data engineers, IT	business decision making, KPI evaluation, QA, model building/validating/versioning, deployment, code QA, code testing, CI/CD dev,data ingestion/prep, …	sagemaker, notebooks, code commits, code pipeline, ECR, Lambda functions, step functions (or SM pipelines), …

MLOps = all these work together

⚠️ Step functions VS SM pipelines (step functions has better compatibility with other services, but isn't free)

4 phases of MLOps Model Maturity

1. Initial establishing experimentation environment, testing in SM and notebooks.
2. Repeatable building a Pipeline, standardise across teams. Going from research notebooks to ML pipeline and automation. Using tools such as CI/CD, save final model versions to model registry.
3. Reliable Once model accepted and behaves as expected by stakeholders, needs to be tested in a mirror production environment (not actual prod env). Involves testing, monitoring and multi-account deployment.
4. Scalable Once model reliable, can be deployed into production.

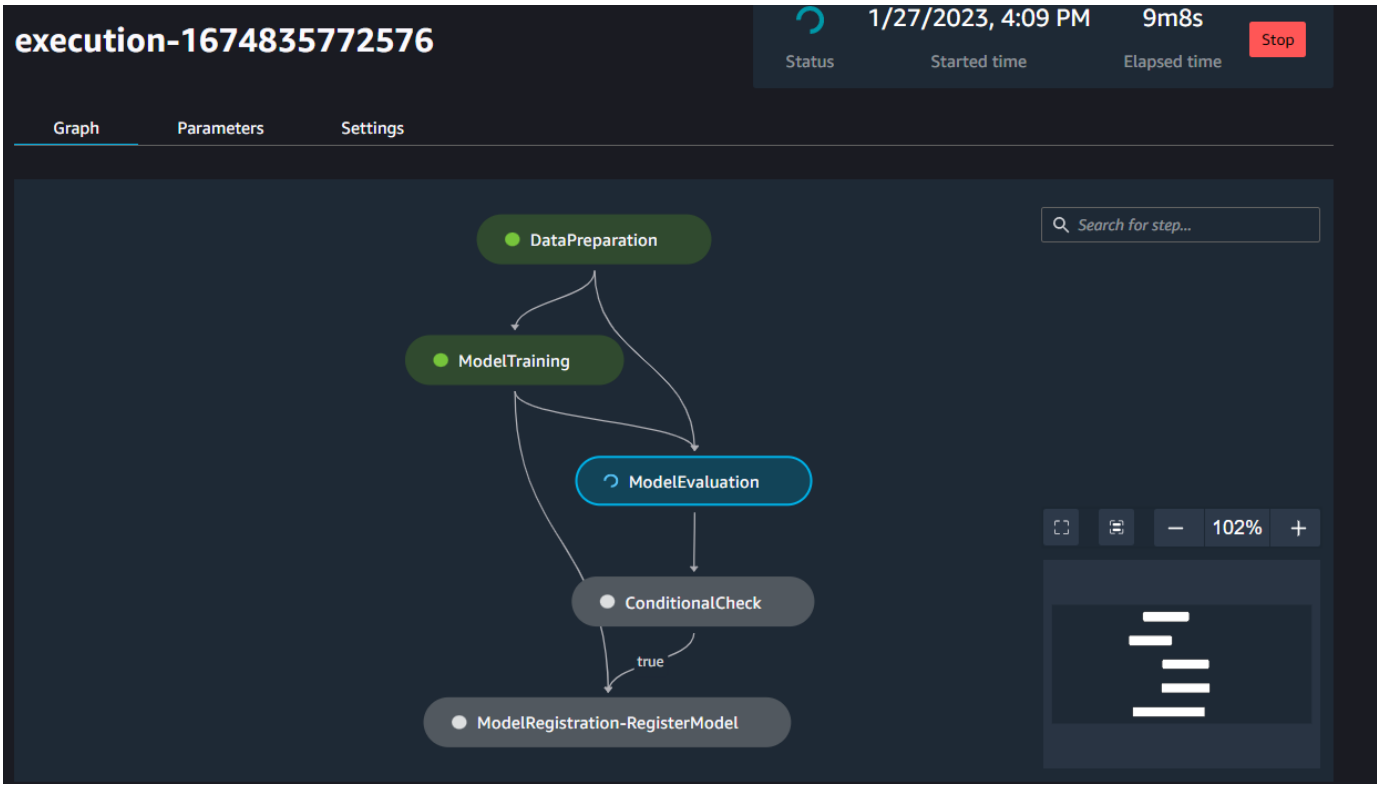


Tip: Use local mode for testing scripts without using cluster instances (which takes time to spin up and is billed).

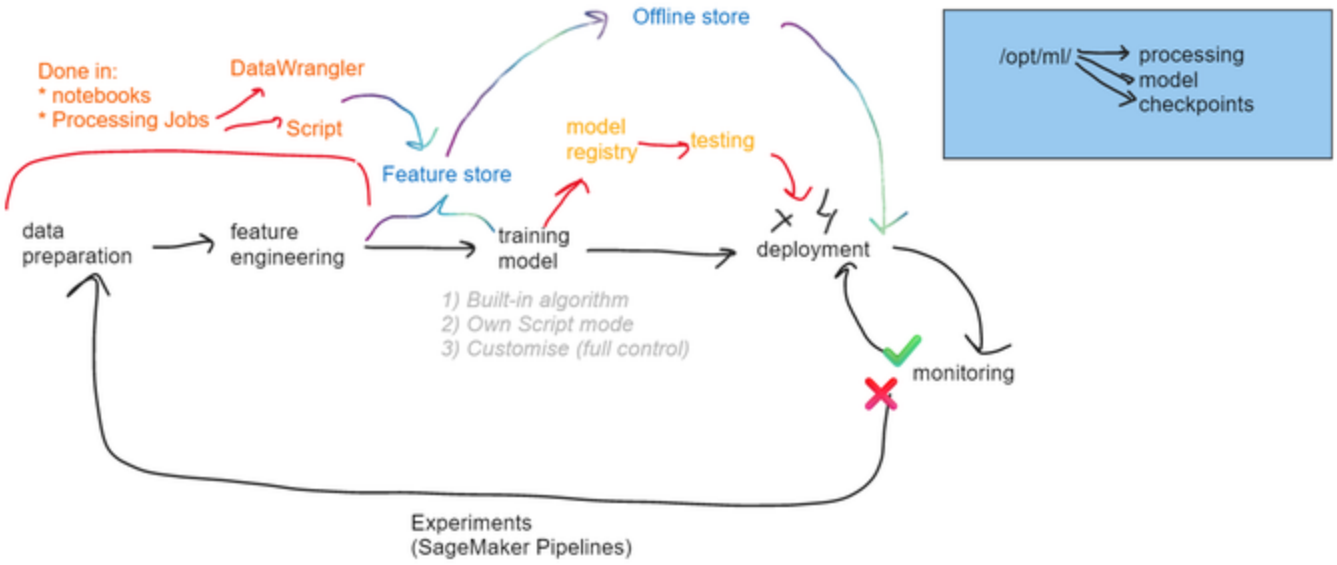
Stitching together

SageMaker Pipelines allows to stitch the following together:

Lab:
https://catalog.us-east-1.prod.workshops.aws/workshops/7acdc7d8-0ac0-44de-bd9b-e3407147a59c/en-US/module2



Summary



Resources

Github repos:

- https://github.com/aws-samples/amazon-sagemaker-immersion-day
- https://github.com/aws/amazon-sagemaker-examples
- https://github.com/aws-samples/mlops-amazon-sagemaker

Documentation: https://sagemaker-examples.readthedocs.io/en/latest/

Blog posts:

- https://aws.amazon.com/blogs/machine-learning/mlops-foundation-roadmap-for-enterprises-with-amazon-sagemaker/
- https://medium.com/@pandey.vikesh/move-from-local-jupyter-to-amazon-sagemaker-part-1-7ef14af0fe9d

Slides: X

Repo for this workshop: https://github.com/Adamouization/SageMaker-Training

File	Modified
image-20230125-141324.png	Jan 25, 2023 by Adam Jaamour
planning-20230125-130117.png	Jan 25, 2023 by Adam Jaamour
image-20230125-143929.png	Jan 25, 2023 by Adam Jaamour
image-20230125-145212.png	Jan 25, 2023 by Adam Jaamour
image-20230125-145242.png	Jan 25, 2023 by Adam Jaamour
processing_xgboost.html	Jan 25, 2023 by Adam Jaamour
image-20230125-154219.png	Jan 25, 2023 by Adam Jaamour
image-20230125-155332.png	Jan 25, 2023 by Adam Jaamour
image-20230127-134956.png	about 3 hours ago by Adam Jaam
image-20230127-154154.png	about an hour ago by Adam Jaam
image-20230127-155231.png	about an hour ago by Adam Jaam

image-20230127-160323.png — 45 minutes ago by Adam Jaamour

image-20230127-161905.png — 29 minutes ago by Adam Jaamour

Drag and drop to upload or browse for files

Download All

Version	Date	Comment
Current Version (v. 17)	**Jan 27, 2023 16:39**	**Adam Jaamour**
v. 16	Jan 27, 2023 15:50	Adam Jaamour
v. 15	Jan 27, 2023 15:48	Adam Jaamour
v. 14	Jan 27, 2023 15:34	Adam Jaamour
v. 13	Jan 27, 2023 15:34	Adam Jaamour
v. 12	Jan 27, 2023 13:56	Adam Jaamour
v. 11	Jan 27, 2023 13:52	Adam Jaamour
v. 10	Jan 25, 2023 16:59	Adam Jaamour
v. 9	Jan 25, 2023 16:11	Adam Jaamour
v. 8	Jan 25, 2023 15:33	Adam Jaamour
v. 7	Jan 25, 2023 15:02	Adam Jaamour
v. 6	Jan 25, 2023 14:40	Adam Jaamour
v. 5	Jan 25, 2023 14:36	Adam Jaamour
v. 4	Jan 25, 2023 14:23	Adam Jaamour
v. 3	Jan 25, 2023 14:21	Adam Jaamour
v. 2	Jan 25, 2023 14:13	Adam Jaamour
v. 1	Jan 25, 2023 13:02	Adam Jaamour