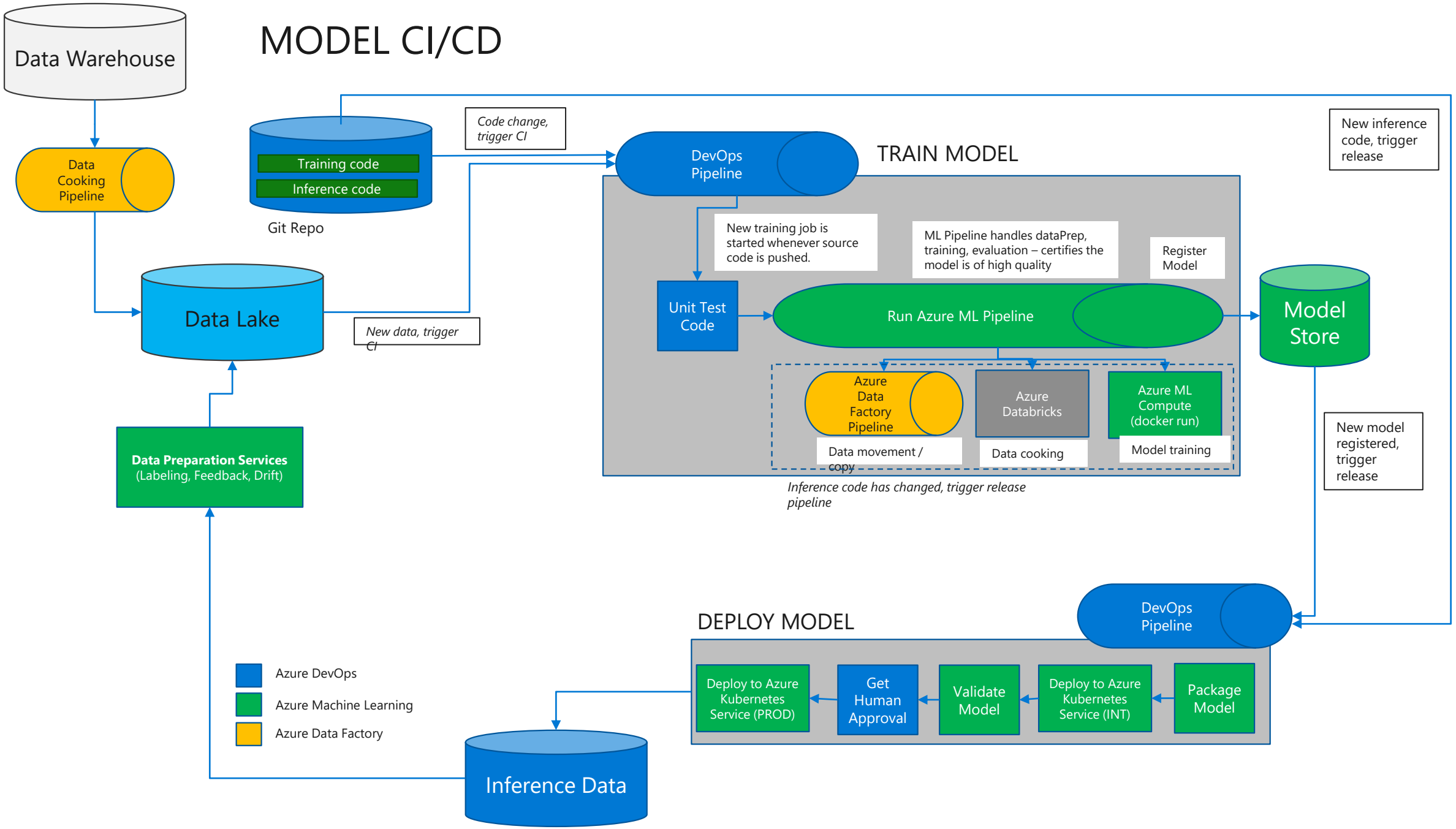
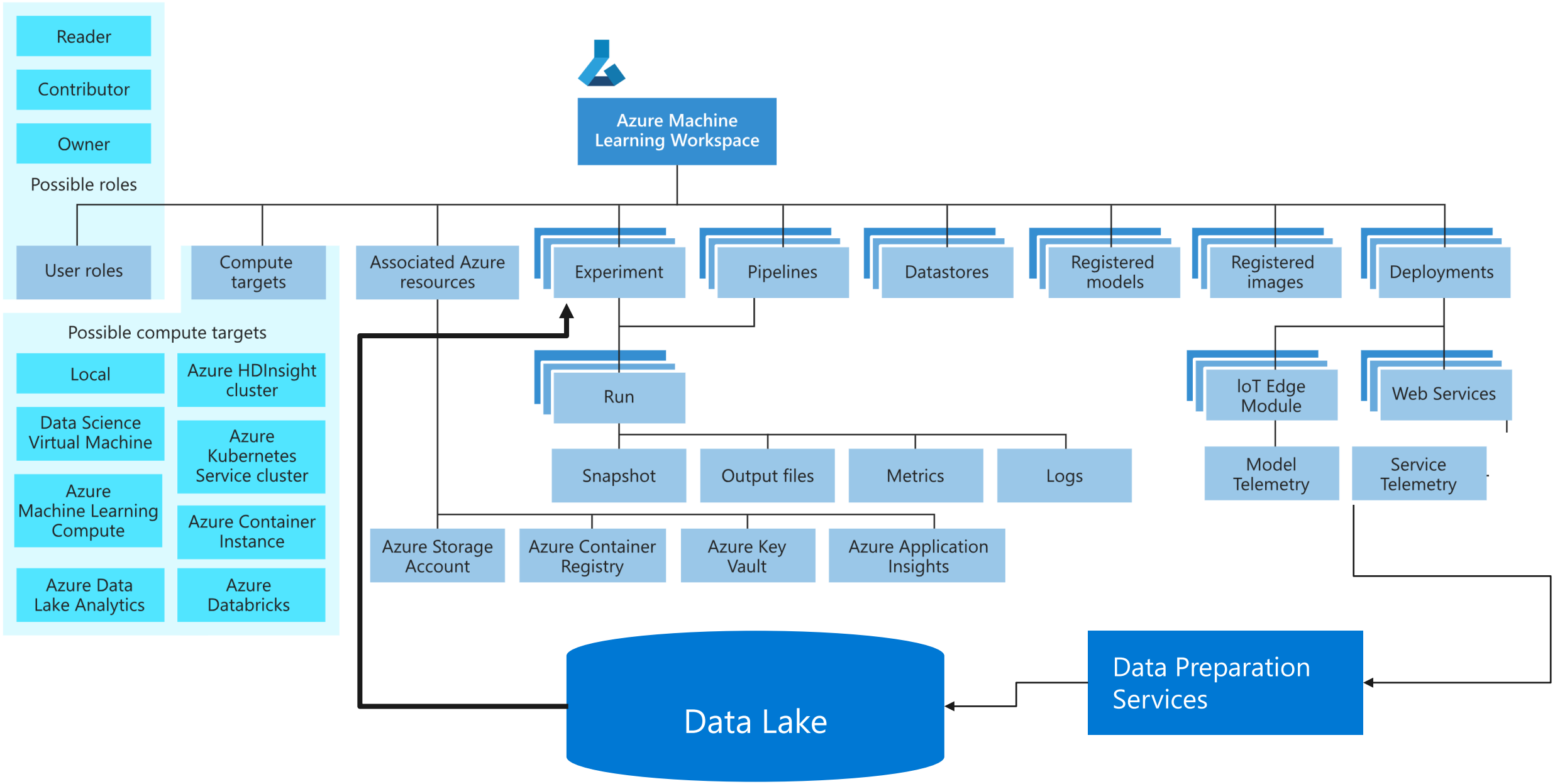
MODEL CI/CD


Data Warehouse
Data Cooking Pipeline
Training code
Inference code
Git Repo
Code change, trigger CI
DevOps Pipeline
TRAIN MODEL
New inference code, trigger release
New training job is started whenever source code is pushed.
ML Pipeline handles dataPrep, training, evaluation – certifies the model is of high quality
Register Model
Unit Test Code
Run Azure ML Pipeline
Model Store
Data Lake
New data, trigger CI
Azure Data Factory Pipeline
Azure Databricks
Azure ML Compute (docker run)
Data movement / copy
Data cooking
Model training
Inference code has changed, trigger release pipeline
New model registered, trigger release
Data Preparation Services (Labeling, Feedback, Drift)
DevOps Pipeline
DEPLOY MODEL
Deploy to Azure Kubernetes Service (PROD)
Get Human Approval
Validate Model
Deploy to Azure Kubernetes Service (INT)
Package Model
Azure DevOps
Azure Machine Learning
Azure Data Factory
Inference Data


Reader
Contributor
Owner
Possible roles
Azure Machine Learning Workspace
User roles
Compute targets
Associated Azure resources
Experiment
Pipelines
Datastores
Registered models
Registered images
Deployments
Possible compute targets
Local
Azure HDInsight cluster
Data Science Virtual Machine
Azure Kubernetes Service cluster
Azure Machine Learning Compute
Azure Container Instance
Azure Data Lake Analytics
Azure Databricks
Run
IoT Edge Module
Web Services
Snapshot
Output files
Metrics
Logs
Model Telemetry
Service Telemetry
Azure Storage Account
Azure Container Registry
Azure Key Vault
Azure Application Insights
Data Lake
Data Preparation Services

Choose the right pipeline for the right task

Scenario	Primary Persona	Azure Offering	OSS Offering	Canonical pipe	Differentiator
Model orchestration	Data scientist	Machine Learning Pipelines	Kubeflow pipelines	Data -> model Training/ retraining	Distribution, caching, code-first, reuse
Data orchestration	Data engineer	Data Factory Pipelines	Apache airflow	Data -> data Data movement/ transform	Strongly typed movement / data-centric activities, **activity first**
Code & app orchestration	App developer / ops	DevOps Pipelines	Jenkins	Code -> app/svc Validation/ deployment	Most open / flexible activity support, approval queues, phases with gating

Data Science DevOps Pain Points

Activities

- Experimentation
- Modeling
  - Versioning
  - Lineage
  - Conversion
  - Export
  - Quantization
- Inferencing
- Retraining
- A/B Testing
- Etc.

Pain Points

- Need to solve ML problem quickly.
- ML stack might be different from rest of the application stack.
- Lots of Glue code.
- Testing accuracy of ML model.
- ML code is not always version controlled.
- Hard to reproduce models
- Integrating model into application can take weeks
- Need to re-write featurizing and scoring code multiple times (in different languages)
- Want to start using customer data to build models
- Hard to track breaking changes

Model Registry

Goals:

- Track important metadata on a model
  - **tags, versions** used for discovery – what do I have?
  - **runtime info** - model framework, operators leveraged, memory footprint, …
  - **key metrics –** how has my model evolved over time?
  - **model lineage** – what was used to create my model (code / compute / data / other models)?
  - **upstream consumption** - where has it been deployed? (services, ML pipelines, IoT devices)… **is it healthy?**
- Make models easy to discover, consume, customize
  - Determine supported image types and deployment targets for a given model
  - Promote the best model from my experiment
  - Explore / fork the experiment used to create a model (customize or retrain)

Syntax (CLI)

```
az ml model register {{modelType}} -n modelName -f modelFile(s) [modelFlavor specific metadata]
```

Example model types: pytorch, pyspark, sklearn

Examples:

- `az ml model register tensorflow -n mymodel -f model.pb --model-type inceptionv3`
- `az ml model promote -n promotedModelName –experiment-id experimentId -m mybestmodel`
- `az ml model update –n mymodel:latest --all`