AI/ML Architecture - how it all fits together

Maciej Kępa

Roadmap

Data Community

01 AI/ML architecture - how it all fits together

Understand the full ML stack from data sources to production, including roles, cloud reference architectures, layers, tools, and a simple ETL + model demo.

02 Data Preparation - practical foundations

Learn practical techniques for building ML data pipelines, cleaning data, using feature stores, validating datasets, and preparing a dataset in PySpark

03 Feature Engineering – the art of extracting value from data

Discover how to create effective features, handle different types of data, apply encoding and normalization, monitor feature drift, and track features with Spark + MLflow.

04 ML Algorithms - the classical approach

Get familiar with core algorithms like regression, trees, and gradient boosting, understand classification, regression, clustering, their pros and cons, and try MLlib demos in Databricks.

05 Model Training in Practice

Learn how to split data, perform hyperparameter tuning, scale training, and train models with code and metrics using MLflow.

06 Deep Learning - leveling up

Understand neural network basics, choose between TensorFlow and PyTorch, leverage GPU/TPU scaling, and build a simple CNN with visualization.

07 ML Pipelines - automation and CI/CD

Build repeatable ML workflows with orchestration tools, CI/CD pipelines, model registry/versioning, and a Databricks pipeline demo.

08 MLOps - manage your ML solution

Explore ML lifecycle management with tracking, model monitoring, data drift handling, canary deployments, A/B testing, and monitoring demos.

09 Generative AI and LLMs - the new wave of technology

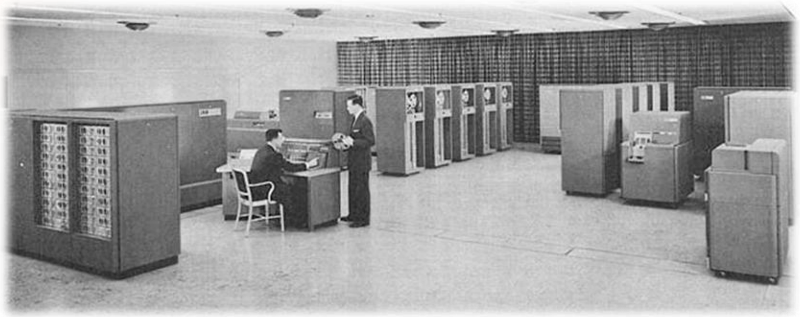
Learn how LLMs differ from classical ML, understand transformers, RAG, AI agent orchestration, and try a simple RAG demo in Azure OpenAI.

Agenda

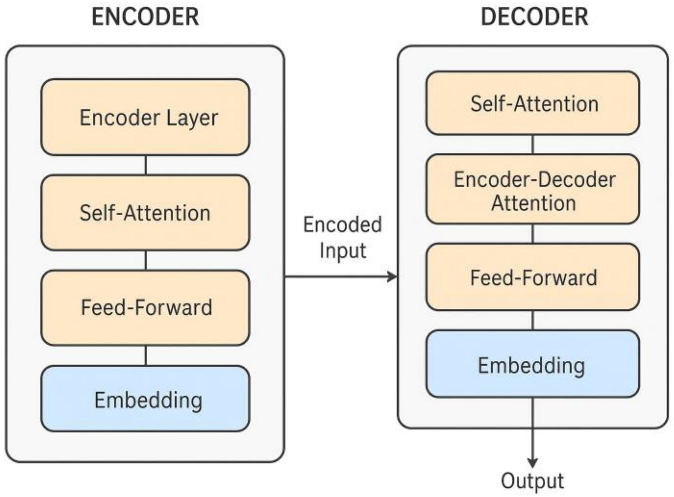
1. Introduction
2. What an AI/ML system is made of?
3. Roles and responsibilities
4. Cloud architecture
5. Demo
6. System design best practices

AI history

Long, long ago...

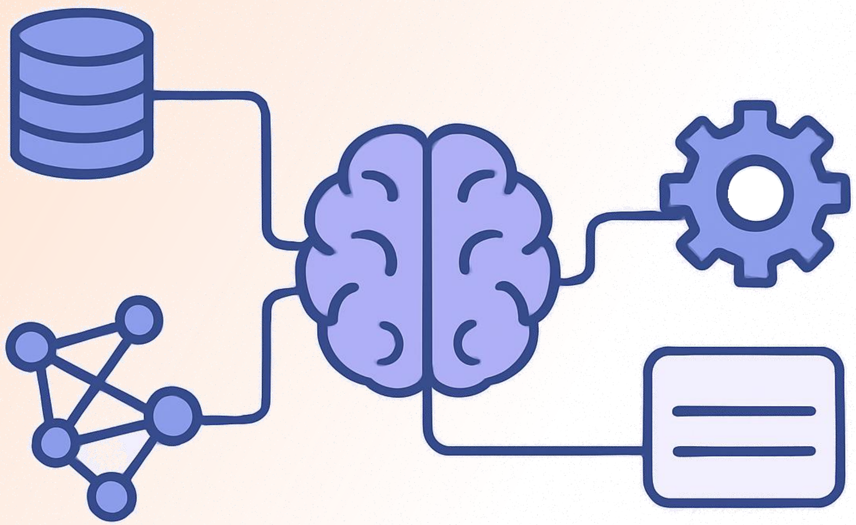


1950s



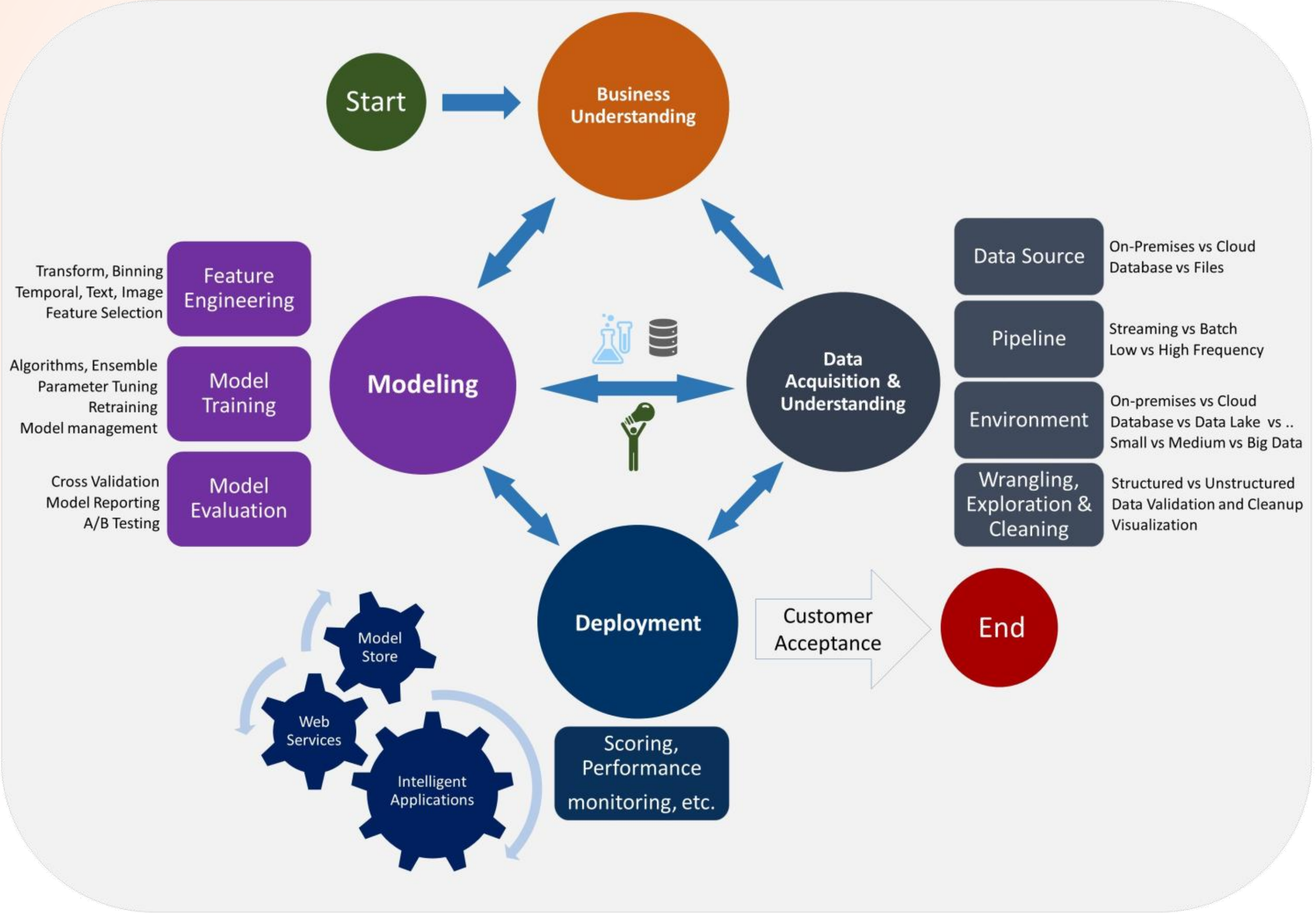
2017

Completely **A**utomated **P**ublic **T**uring test to tell **C**omputers and **H**umans **A**part.



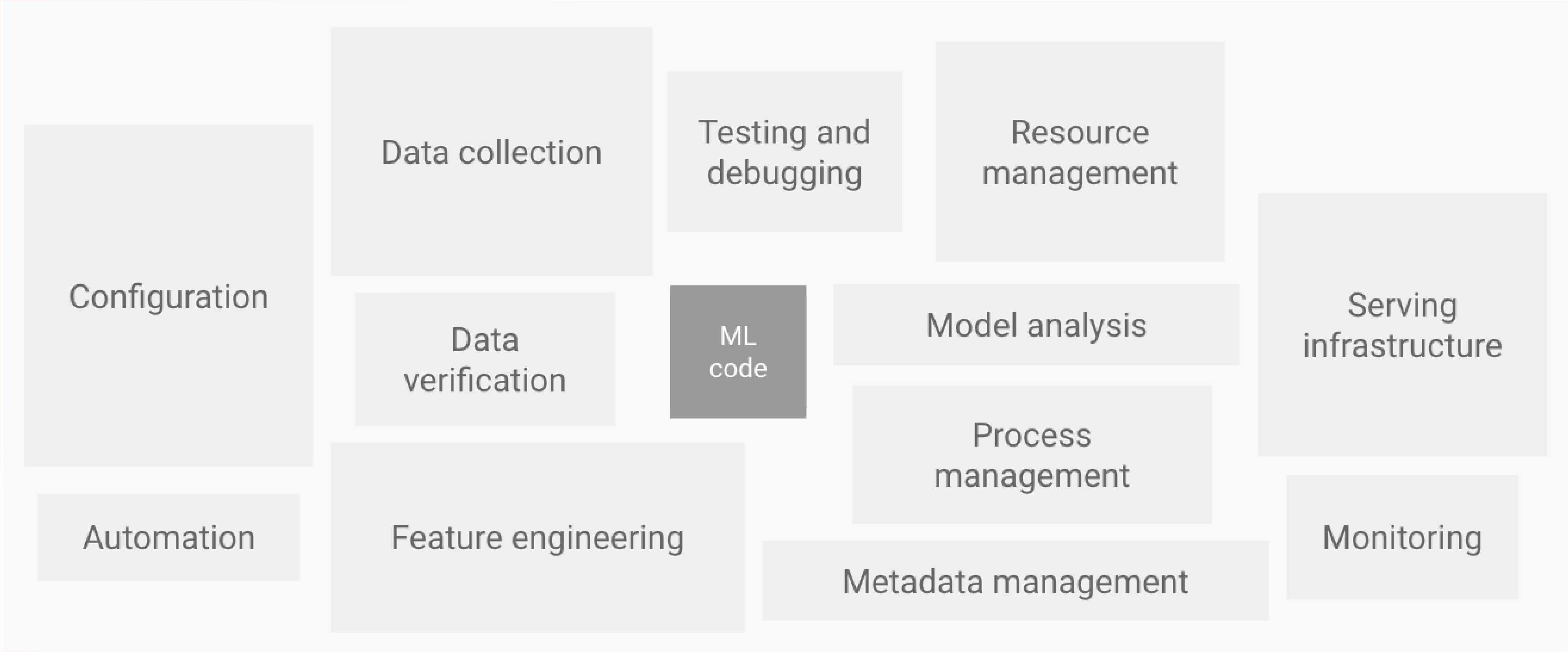
What an AI/ML system is made of?

AI project lifecycle



Source: https://github.com/Azure/Microsoft-TDSP/blob/master/Docs/README.md

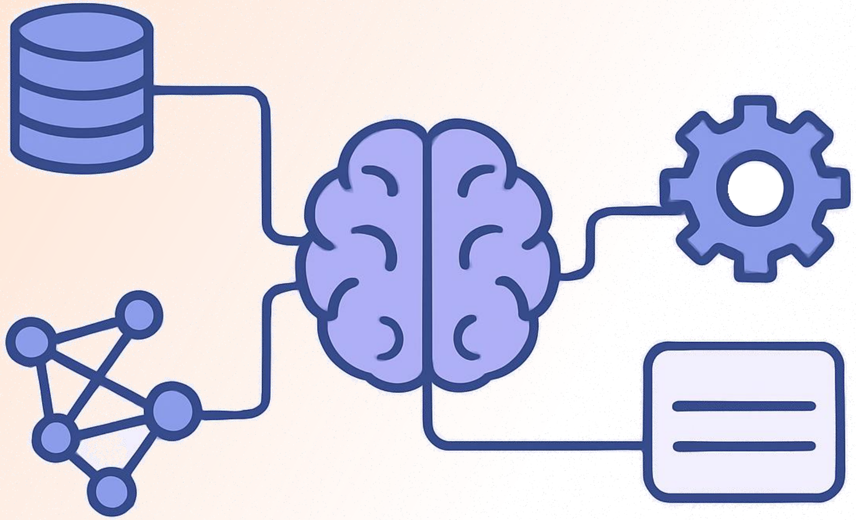
AI project areas



Source: https://cloud.google.com/architecture/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning

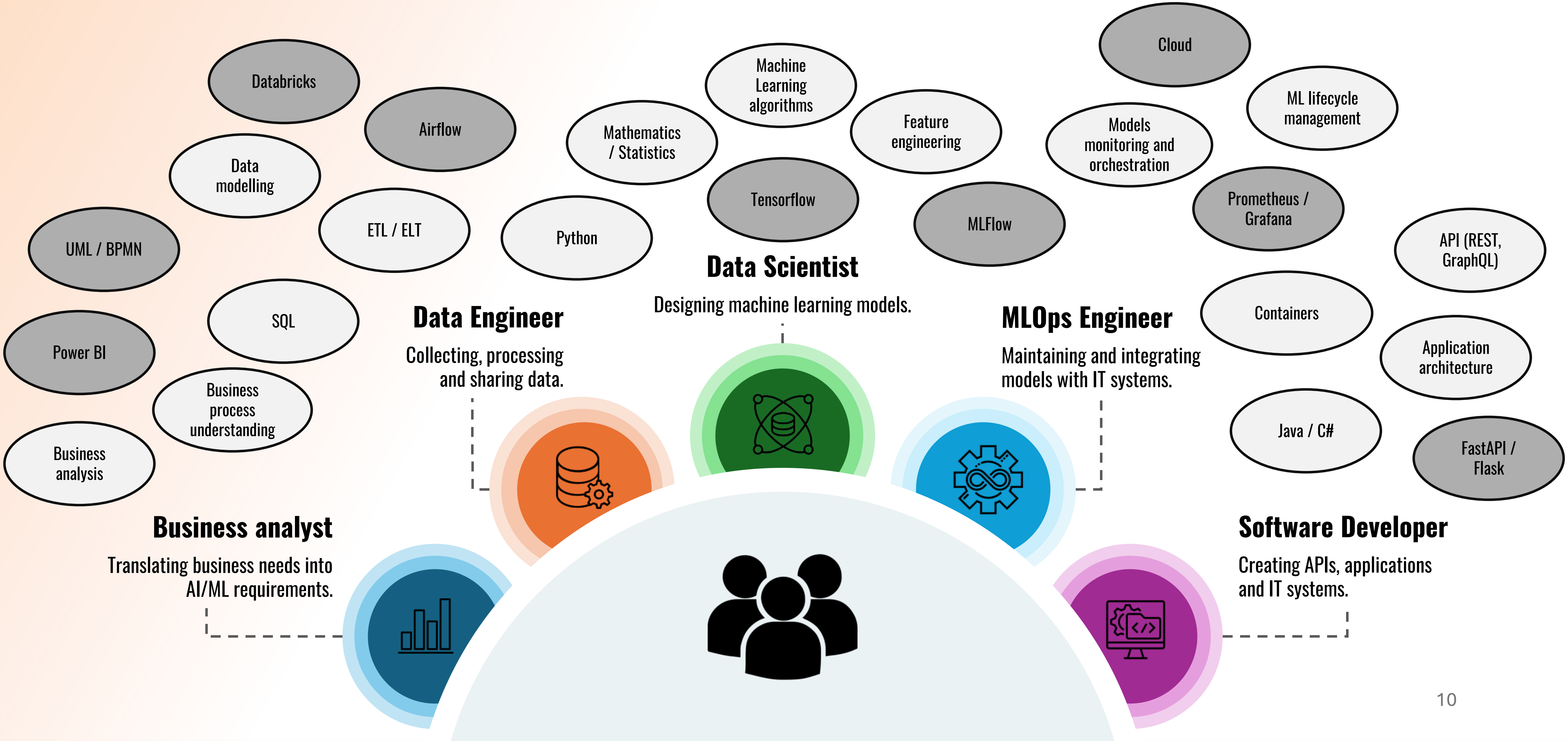
Key layers

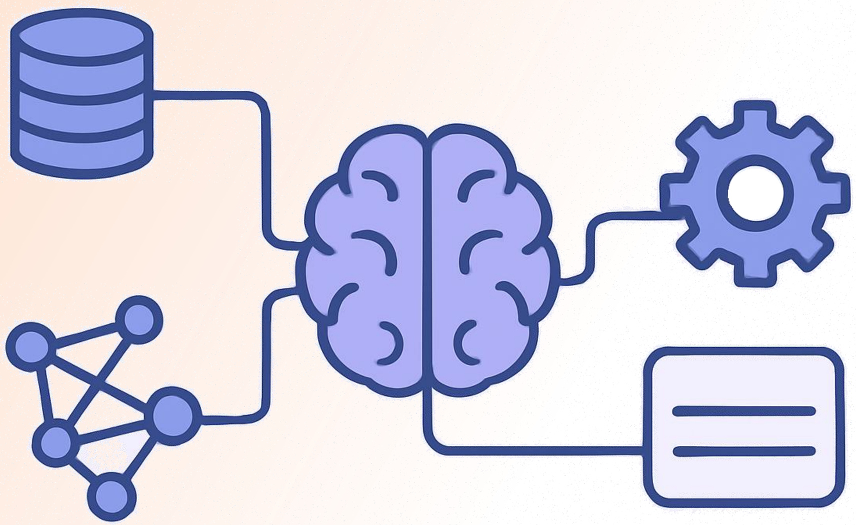
Layer	Purpose	Typical Components / Tools	Key Responsibilities	Artifacts
Data Layer	Collects, stores, and prepares data for modeling.	Data Lake (Azure Data Lake, GCS, S3), Data Warehouse (Synapse, BigQuery), Feature Store (Databricks Feature Store, Feast).	Ingestion from raw sources, cleaning, transformation, feature creation, and validation (Great Expectations).	Raw and processed datasets, feature tables, data validation reports, ETL notebooks, schema definitions.
Model Layer	Focuses on training, tracking, and versioning ML models.	Databricks, MLflow, Azure ML, TensorFlow, PyTorch, scikit-learn.	Model development, experiment tracking, hyperparameter tuning, model registration, and reproducibility.	Model training notebooks, experiment logs, metrics, model binaries, registered models.
Deployment Layer	Delivers trained models to production and maintains operational quality.	CI/CD pipelines (GitHub Actions, Azure DevOps), Model Registry, APIs, Monitoring (Azure Monitor, Prometheus).	Model packaging and deployment, version control, A/B testing, drift monitoring, continuous retraining.	Deployment scripts, inference APIs, dashboards, monitoring reports, retraining jobs.



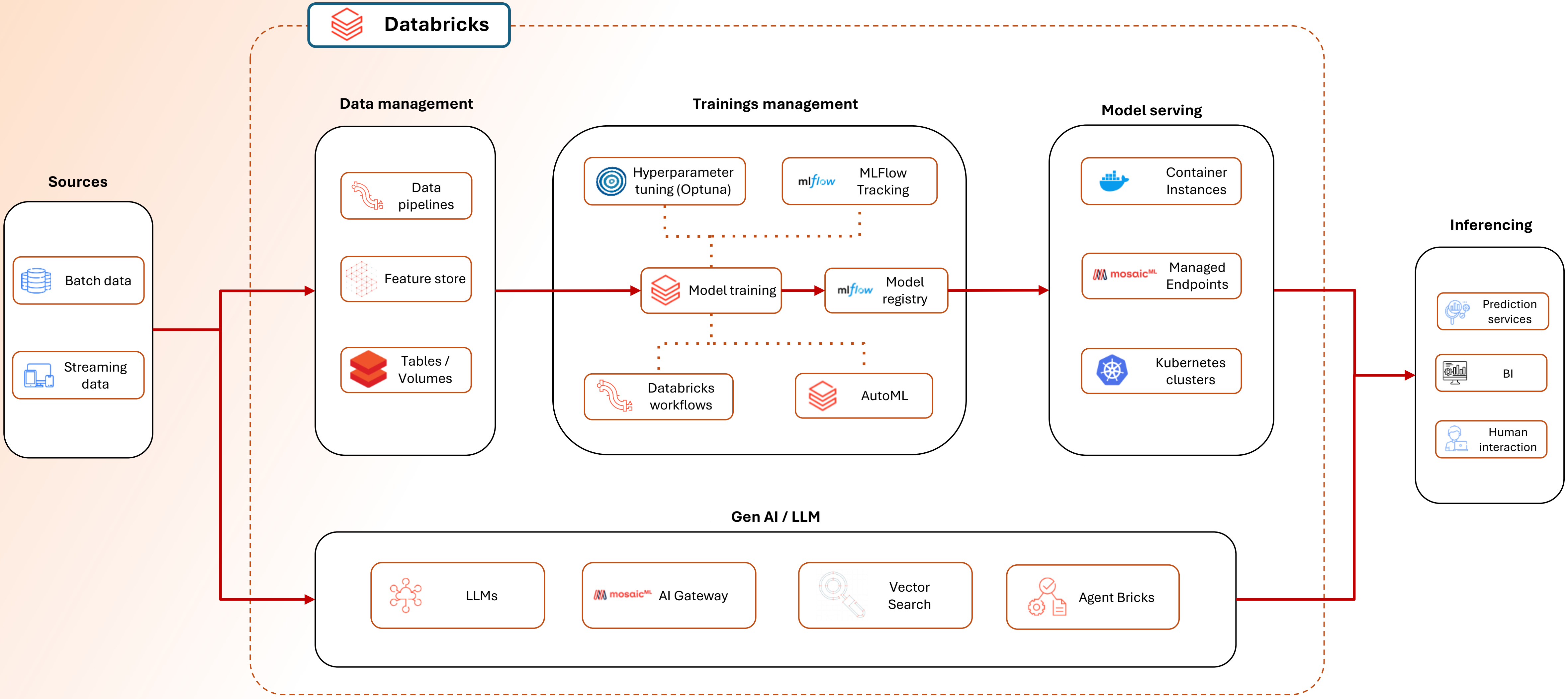
Roles and responsibilities

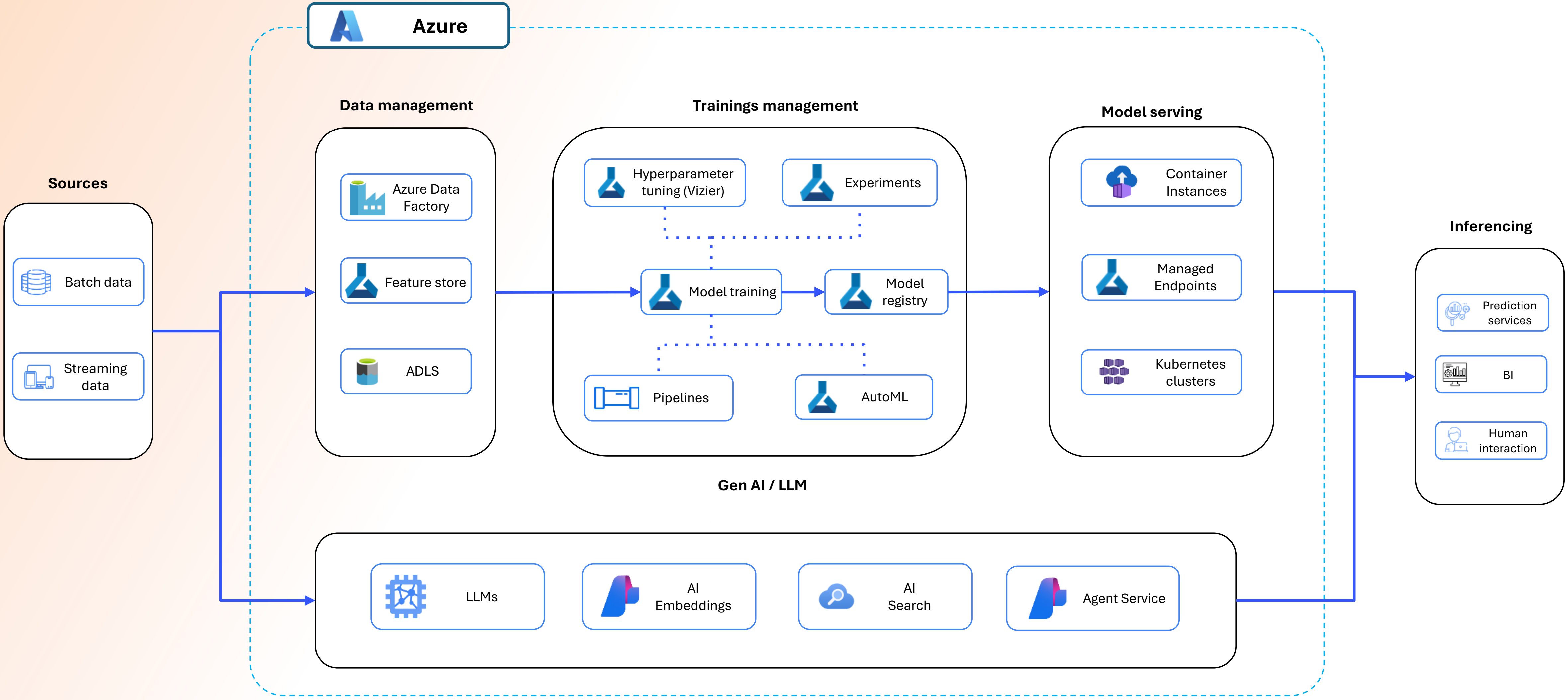
Division of roles in AI/ML teams



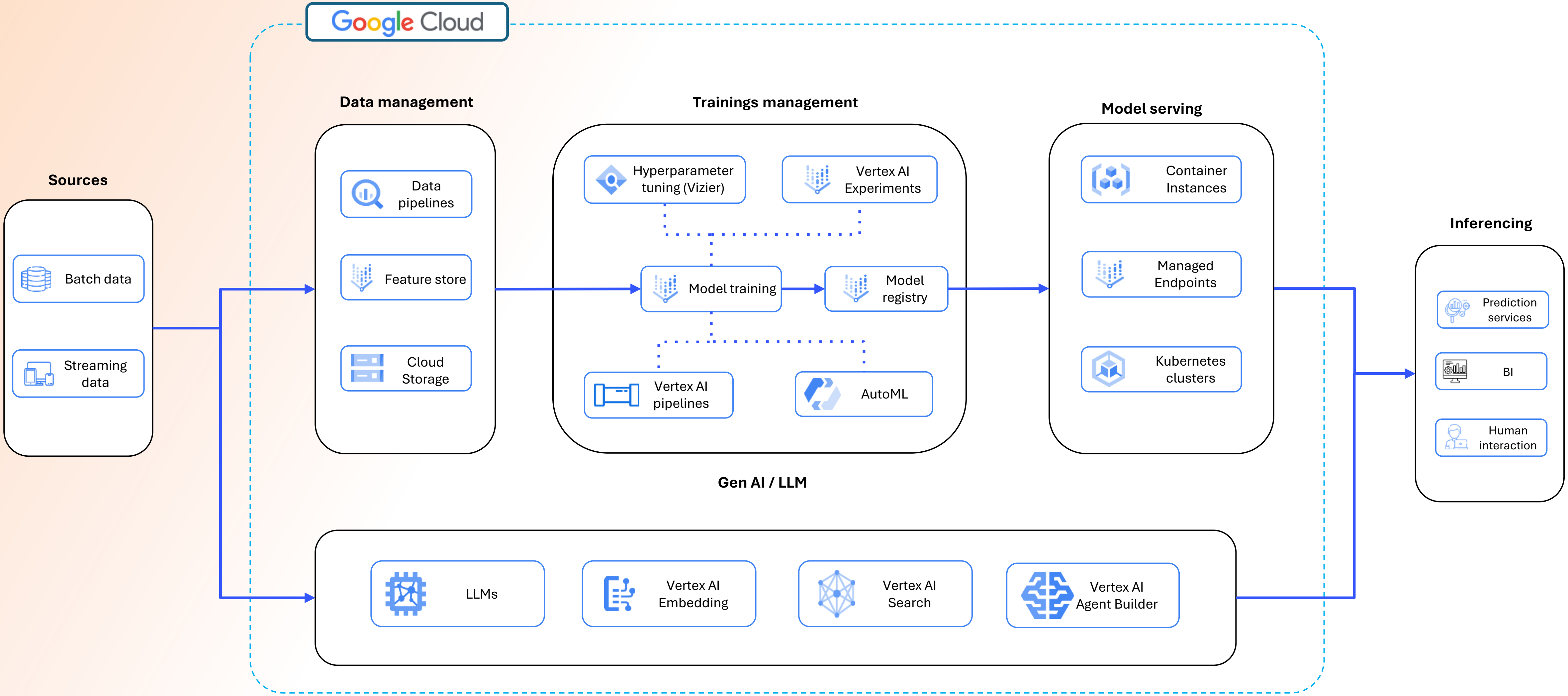


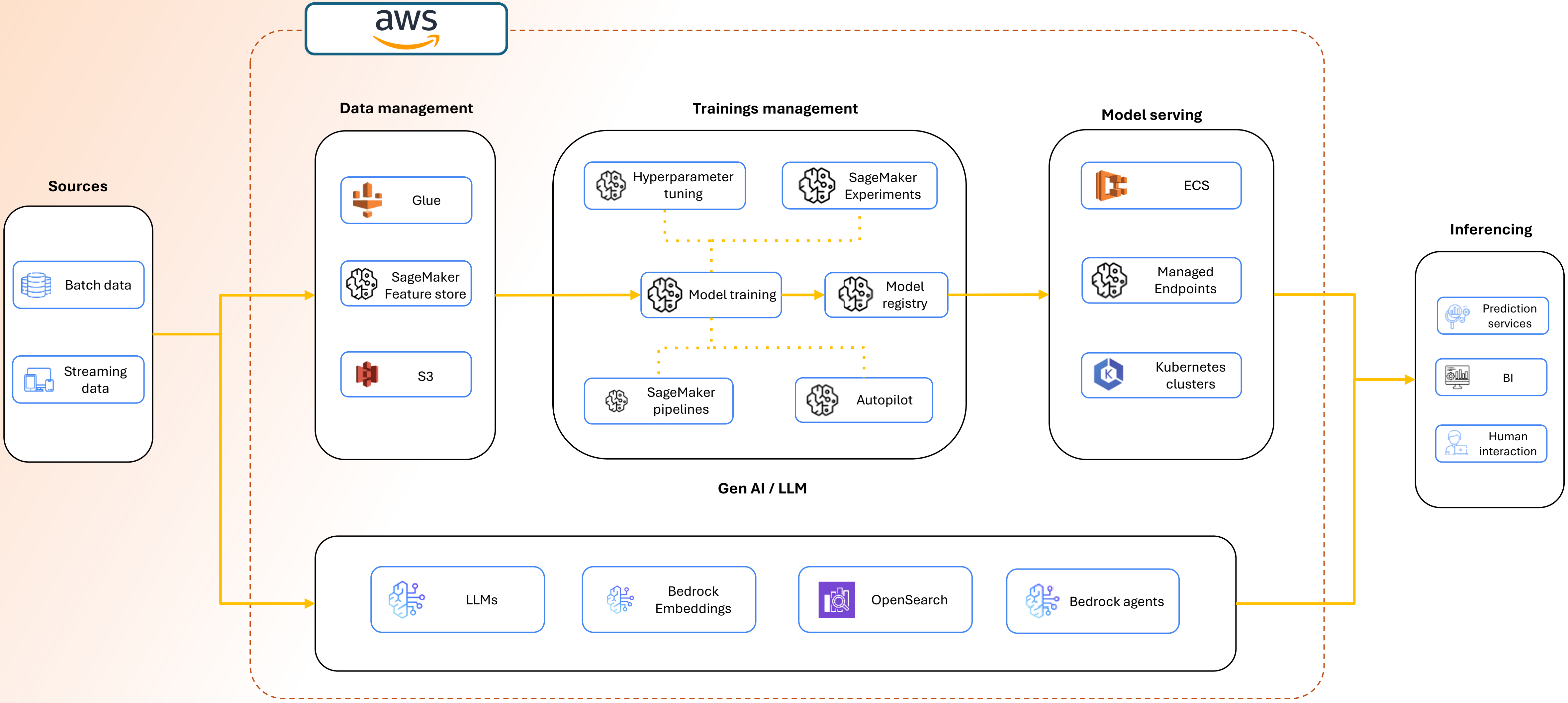
Cloud Architecture


Databricks
Sources
Batch data
Streaming data
Data management
Data pipelines
Feature store
Tables / Volumes
Trainings management
Hyperparameter tuning (Optuna)
mlflow
MLFlow Tracking
Model training
mlflow
Model registry
Databricks workflows
AutoML
Model serving
Container Instances
mosaicML
Managed Endpoints
Kubernetes clusters
Inferencing
Prediction services
BI
Human interaction
Gen AI / LLM
LLMs
mosaicML
AI Gateway
Vector Search
Agent Bricks

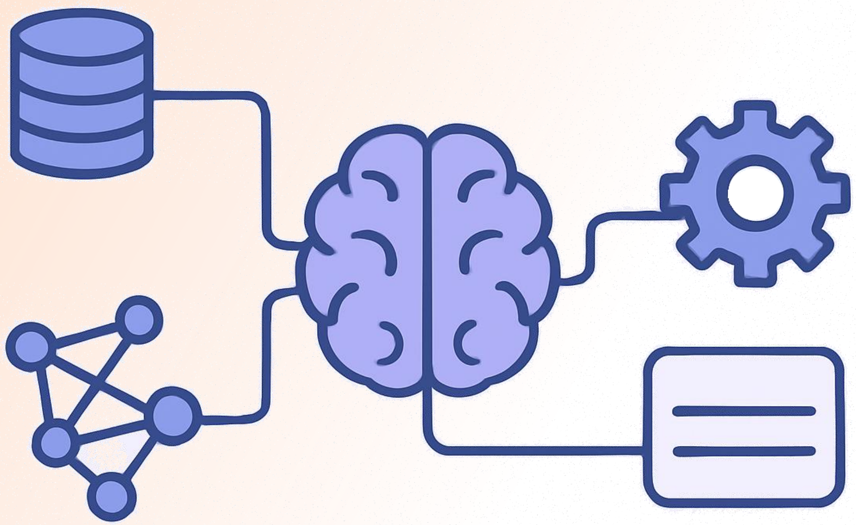


Azure
Sources
Batch data
Streaming data
Data management
Azure Data Factory
Feature store
ADLS
Trainings management
Hyperparameter tuning (Vizier)
Experiments
Model training
Model registry
Pipelines
AutoML
Model serving
Container Instances
Managed Endpoints
Kubernetes clusters
Inferencing
Prediction services
BI
Human interaction
Gen AI / LLM
LLMs
AI Embeddings
AI Search
Agent Service


Google Cloud
Sources
Batch data
Streaming data
Data management
Data pipelines
Feature store
Cloud Storage
Trainings management
Hyperparameter tuning (Vizier)
Vertex AI Experiments
Model training
Model registry
Vertex AI pipelines
AutoML
Model serving
Container Instances
Managed Endpoints
Kubernetes clusters
Inferencing
Prediction services
BI
Human interaction
Gen AI / LLM
LLMs
Vertex AI Embedding
Vertex AI Search
Vertex AI Agent Builder


aws
Sources
Batch data
Streaming data
Data management
Glue
SageMaker Feature store
S3
Trainings management
Hyperparameter tuning
SageMaker Experiments
Model training
Model registry
SageMaker pipelines
Autopilot
Model serving
ECS
Managed Endpoints
Kubernetes clusters
Inferencing
Prediction services
BI
Human interaction
Gen AI / LLM
LLMs
Bedrock Embeddings
OpenSearch
Bedrock agents

DEMO



System design best practices

Takeaways

- **Don't overengineer from day one:** start with a simple, working system, add the more advanced features later (if you really need them).
- **If its not tracked, it doesn't exist:** every experiment, model version, and hyperparameter - tracking is never optional.
- **Understand your problem:** AI projects fail due to too high expectations and problem misunderstanding
- **Clean data beats fancy models:** A simple model on good data will outperform any complex architecture trained on garbage - every single time.
- **Choosing your cloud provider matters less than you expect:** the tools differ; the principles don't. Pick one and focus on building value, not migrating clouds.

Thank you!

Data Community

Contact:

https://www.linkedin.com/in/maciej-kepa

https://github.com/maciejkepa/ai-ml-in-practice