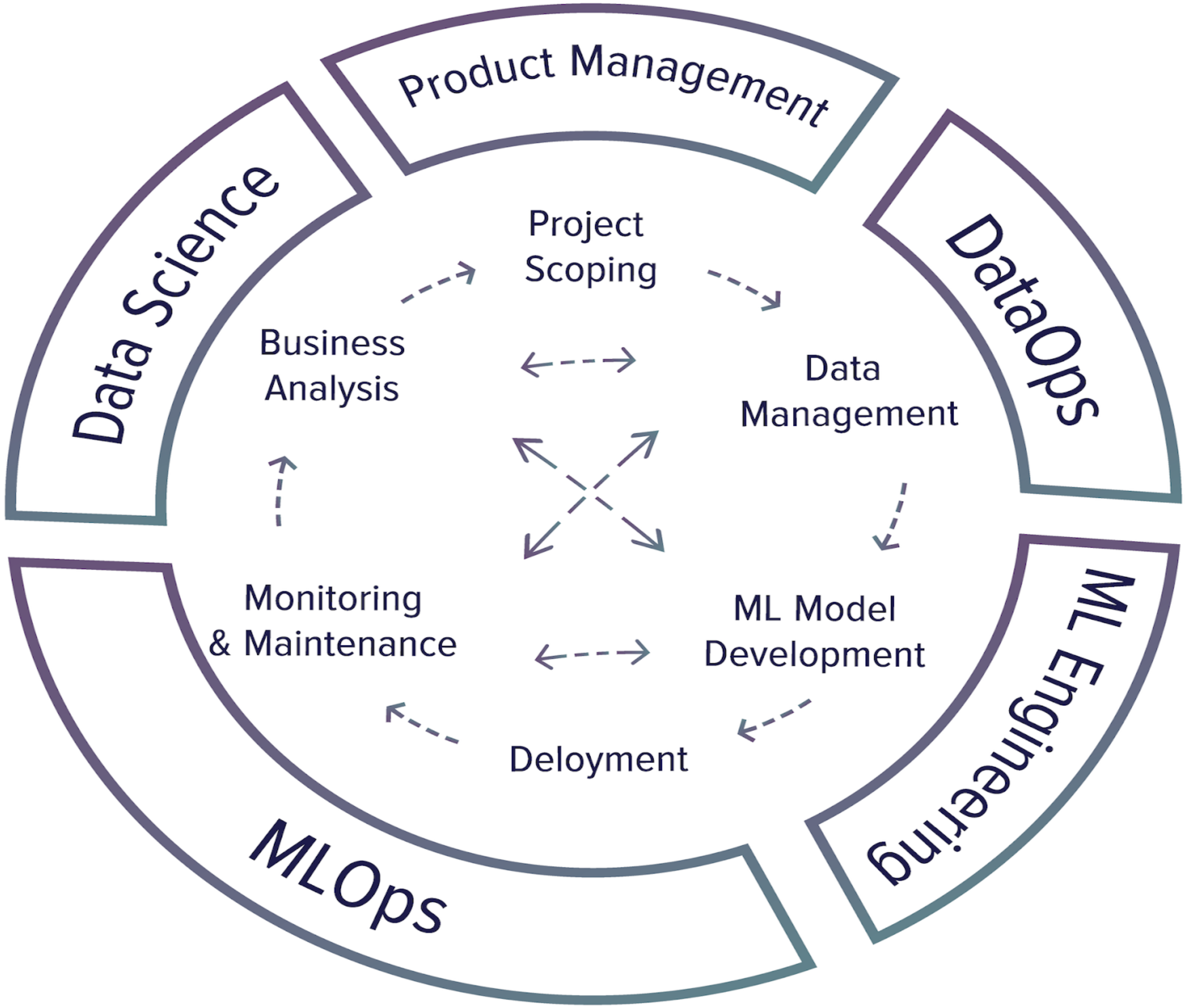
MLOps

K Kiran Kumar–
INSOFE
Senior Assistant Professor

ML in Production



https://stanford-cs329s.github.io/

What is a Productionized model

A **productionized model** is a **prediction service** that makes predictions on new (unseen) data. It should support **automated** training, testing, and deployment. It should be scalable, available, and if the model is online it should be able to enrich feature vectors using a feature store.

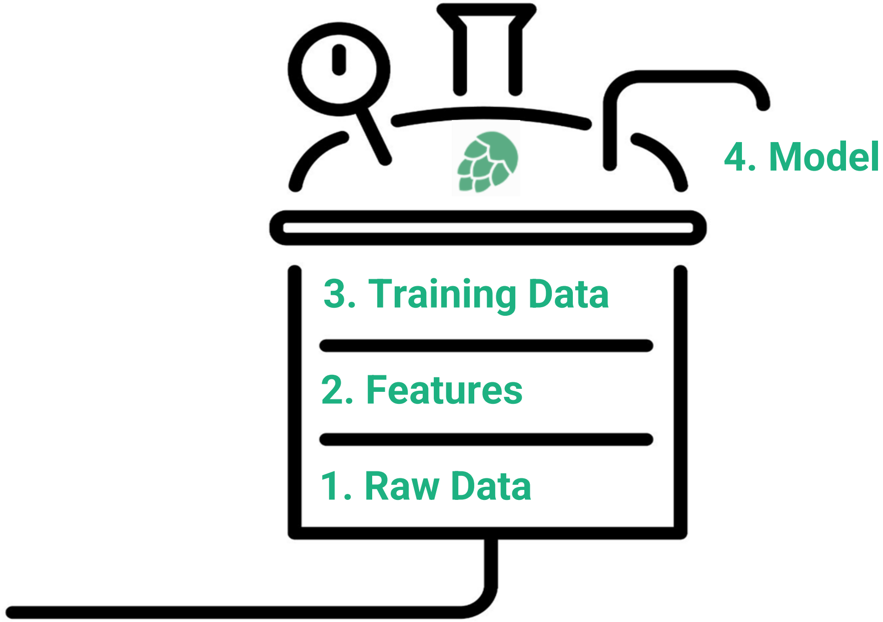
If the model is a bus - the "bus service" is "prediction service".

What is MLOps

The Continuous Delivery Foundation’s SIG-MLOps defines MLOps as “the extension of the DevOps methodology to include Machine Learning and Data Science assets as first class citizens within the DevOps ecology”.

CREDITS : HOPS AI

MLOps is not the chat, it’s the chat Bhandar



4. Model

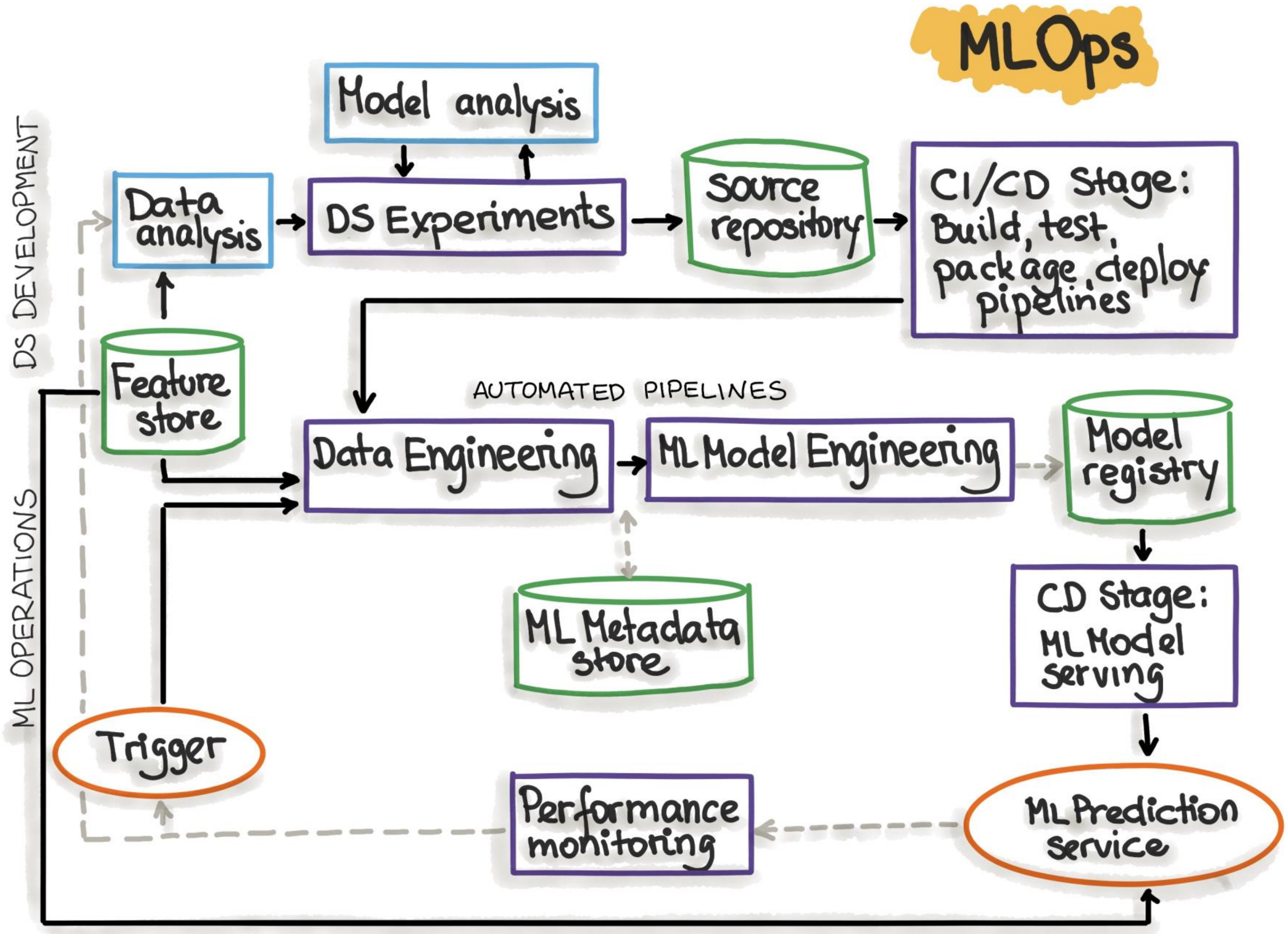
3. Training Data

2. Features

1. Raw Data

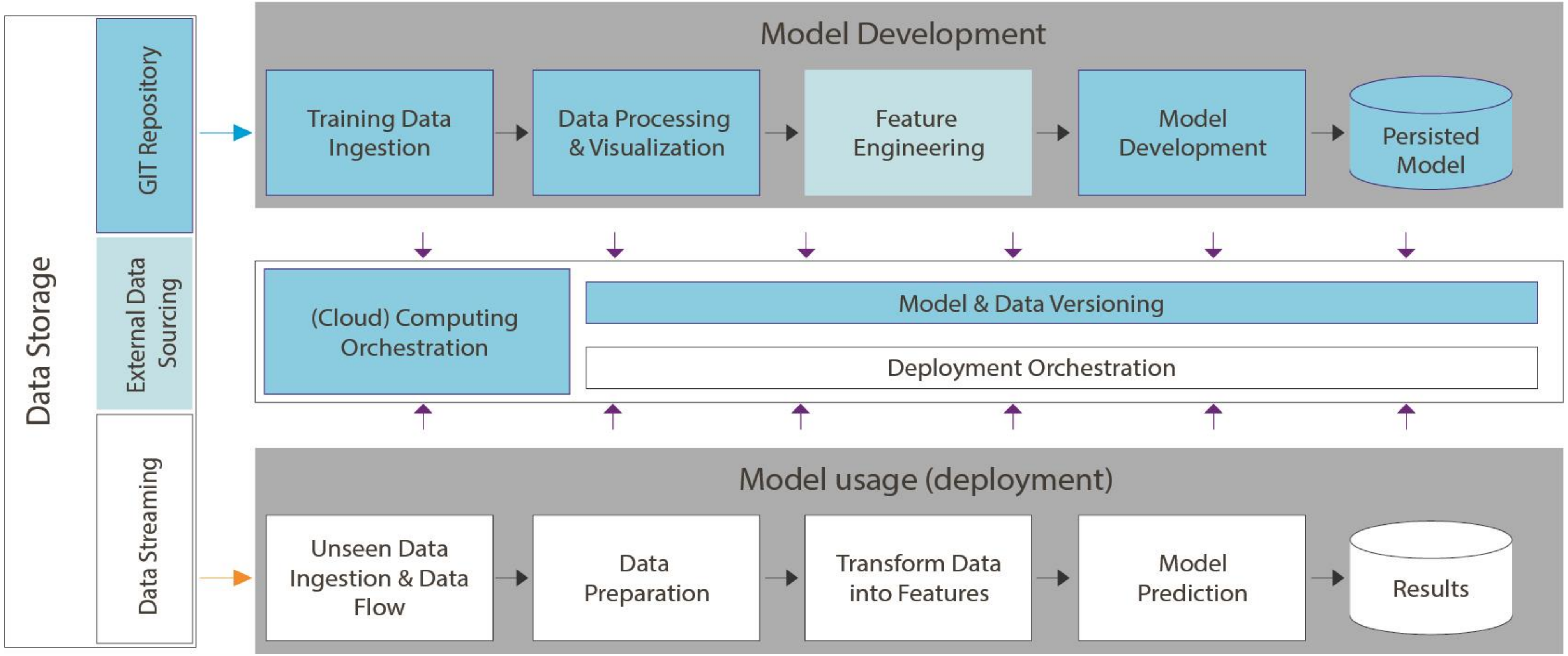
CREDITS : HOPS AI

MLOps



https://ml-ops.org/#gettingstarted

Model development & usage process



The Need for MLOps

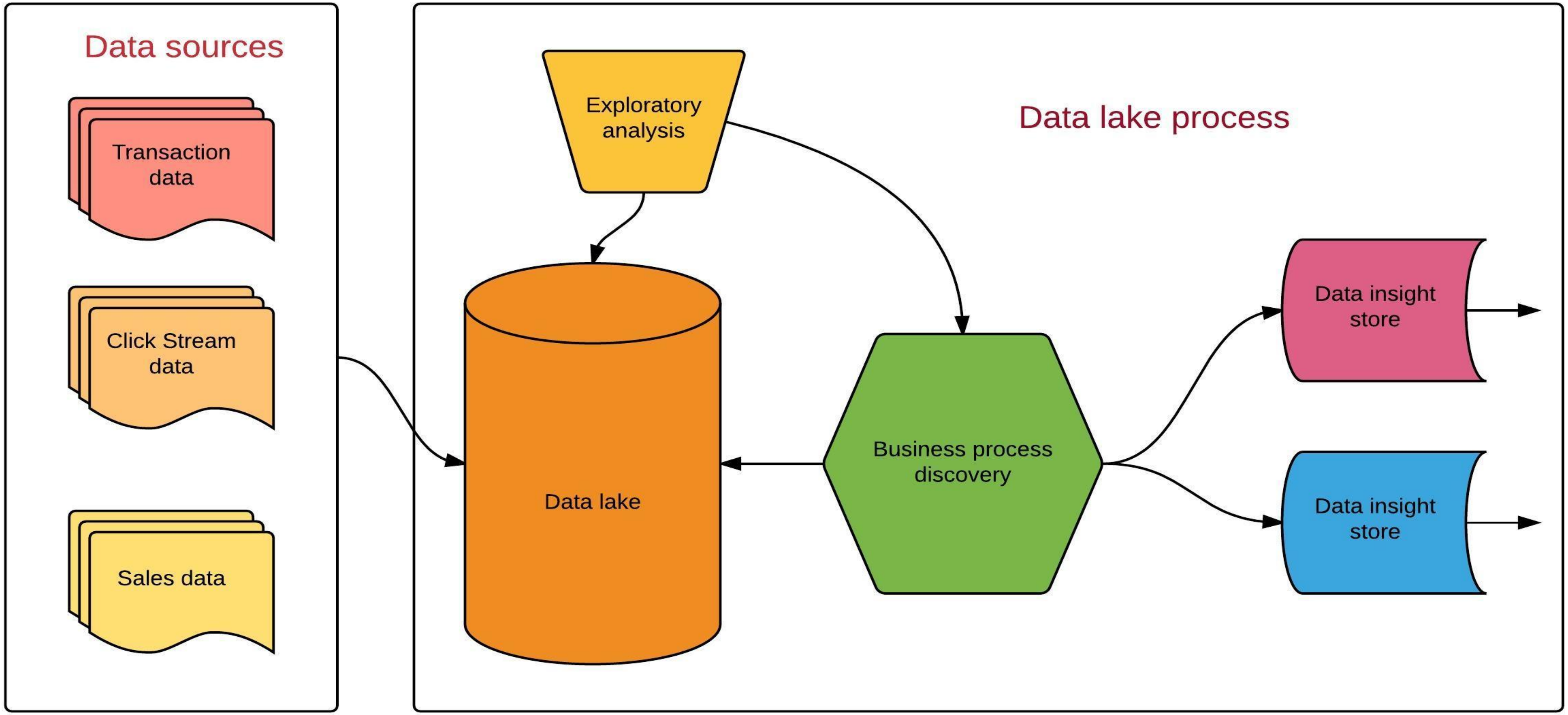
- Data is constantly changing
  - Need to constantly update training data and models
- Multiple teams need to coordinate to put models in production
  - Data Engineers -> Data Scientists -> ML Engineers
- Infrastructure support needed
  - Feature engineering -> model training -> model serving
  - Orchestration support for pipelines
  - Data and code versioning

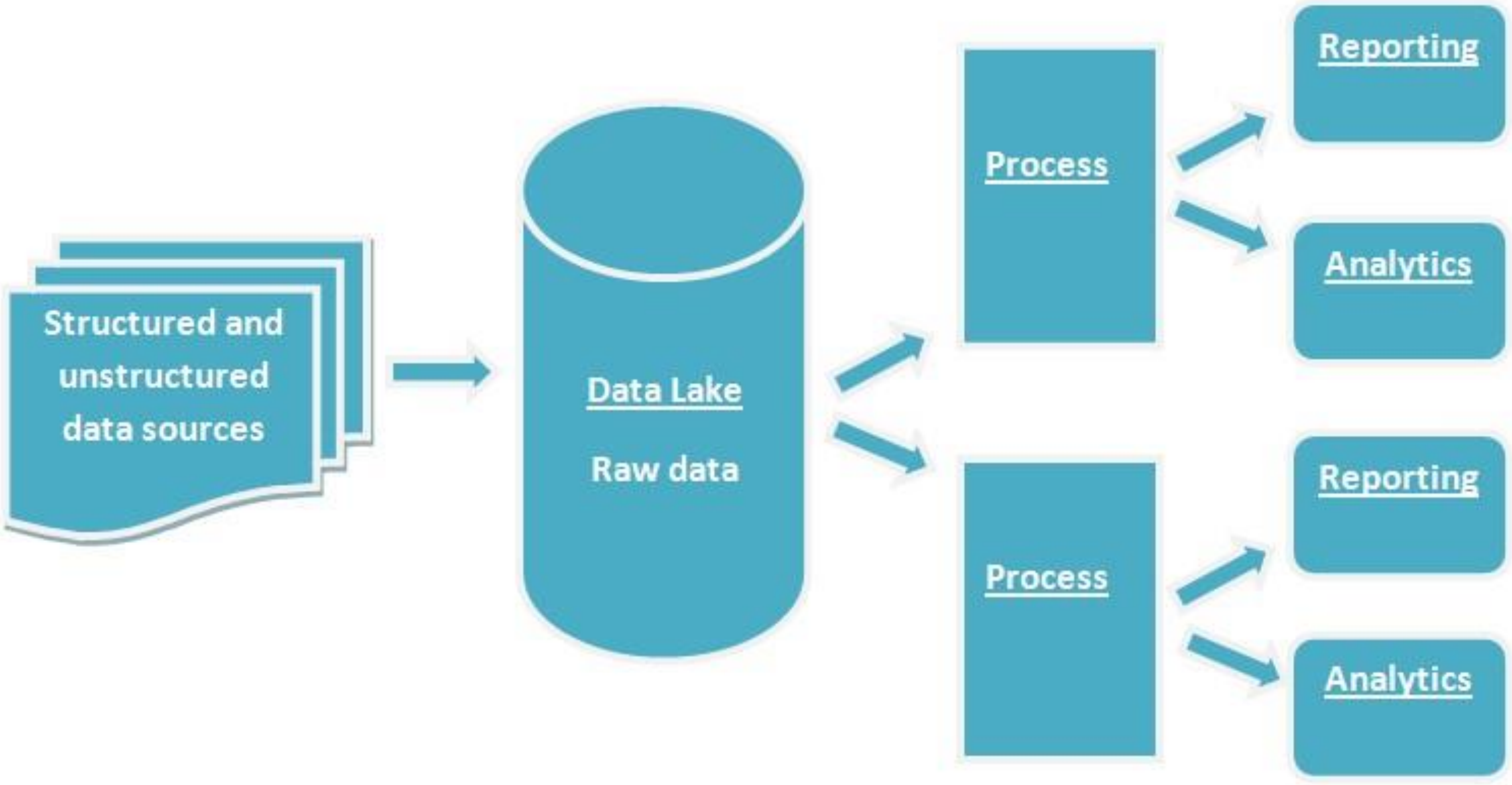
CREDITS : HOPS AI

Project Setup

- Goals
- User Experience
- Performance constraints
- Evaluation
- Personalization
- Project Constraints

Loading Upstream data into Data Lake

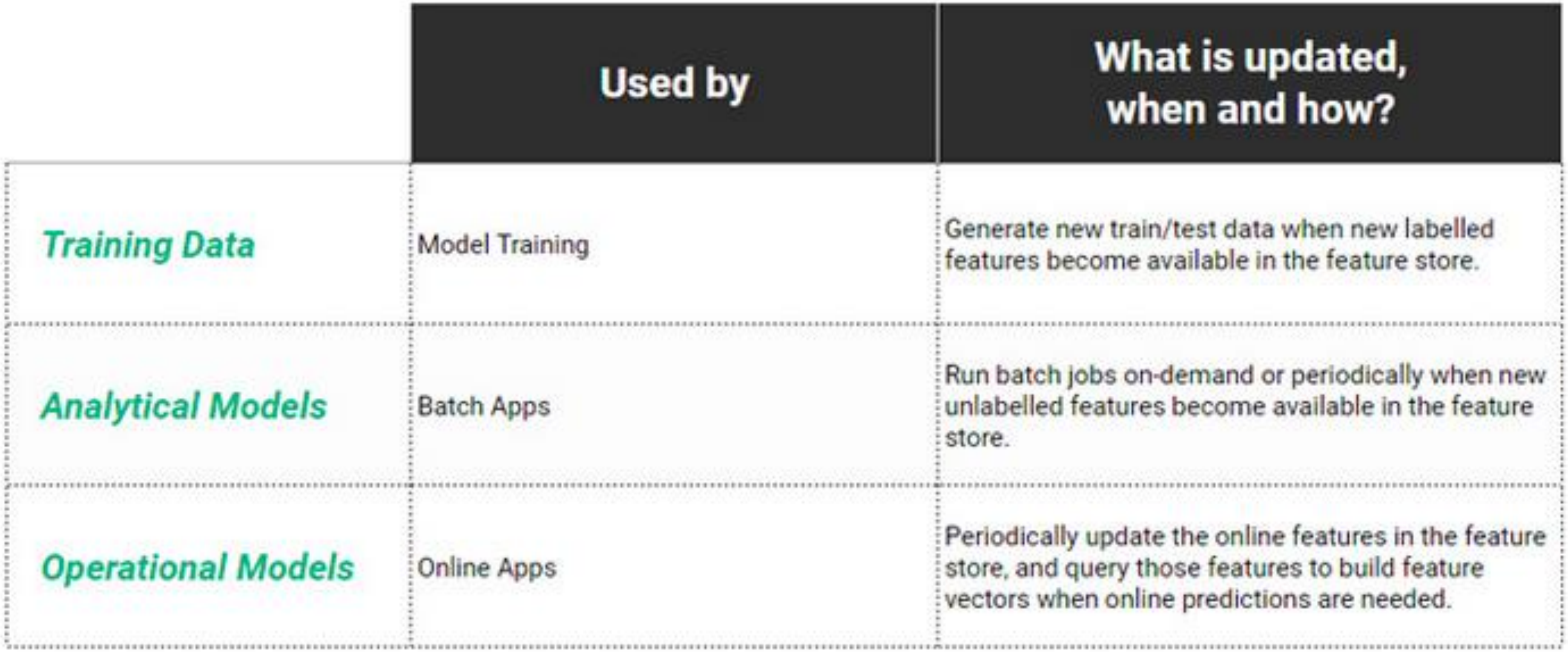



Structured and unstructured data sources
Data Lake
Raw data
Process
Reporting
Analytics
Process
Reporting
Analytics

Feature Store

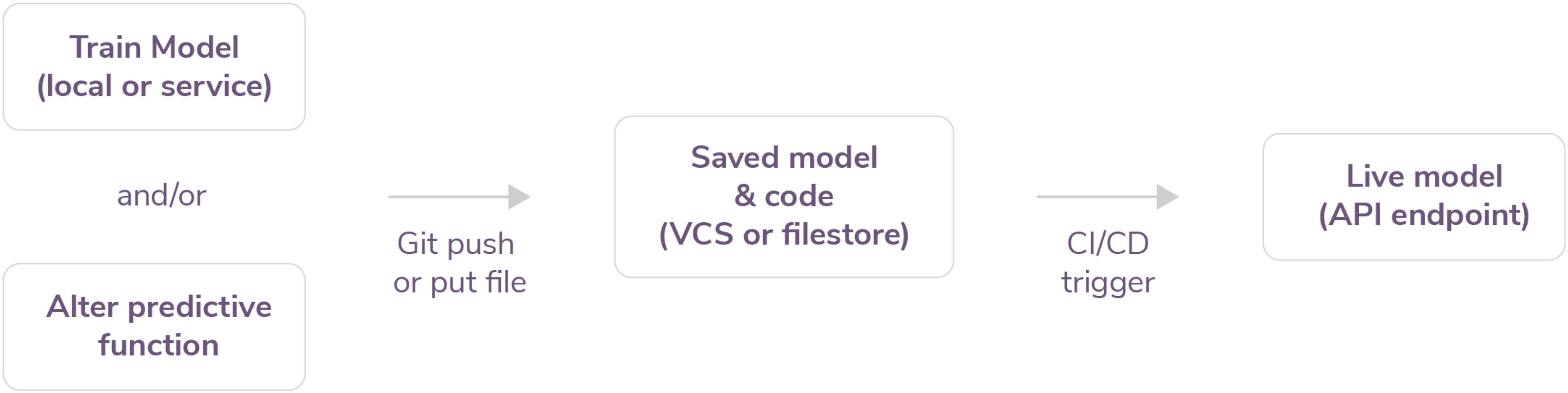
The feature store is the data warehouse for Data Science - it is a central vault for storing documented, curated, and access-controlled features that can be used across many different models. The feature store ingests data from the Enterprise’s many different sources after transforming, aggregating, and validating the data.

Feature Store



	Used by	What is updated, when and how?
Training Data	Model Training	Generate new train/test data when new labelled features become available in the feature store.
Analytical Models	Batch Apps	Run batch jobs on-demand or periodically when new unlabelled features become available in the feature store.
Operational Models	Online Apps	Periodically update the online features in the feature store, and query those features to build feature vectors when online predictions are needed.

CI/CD Pipeline



Train Model
(local or service)

and/or

Alter predictive
function

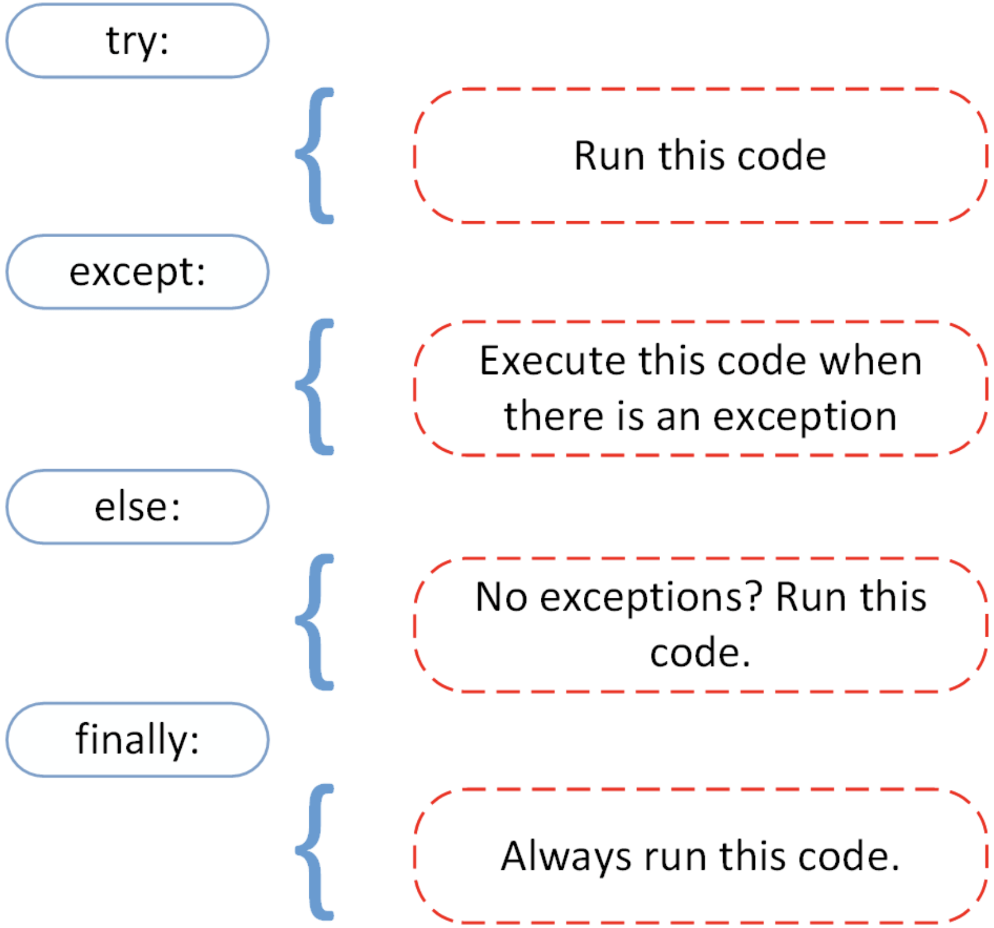
Git push
or put file

Saved model
& code
(VCS or filestore)

CI/CD
trigger

Live model
(API endpoint)

Error Handling and Exception handling



try:

Run this code

except:

Execute this code when there is an exception

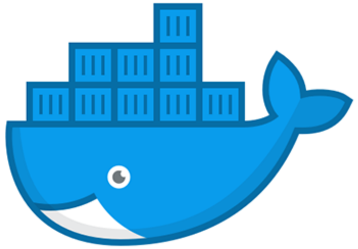
else:

No exceptions? Run this code.

finally:

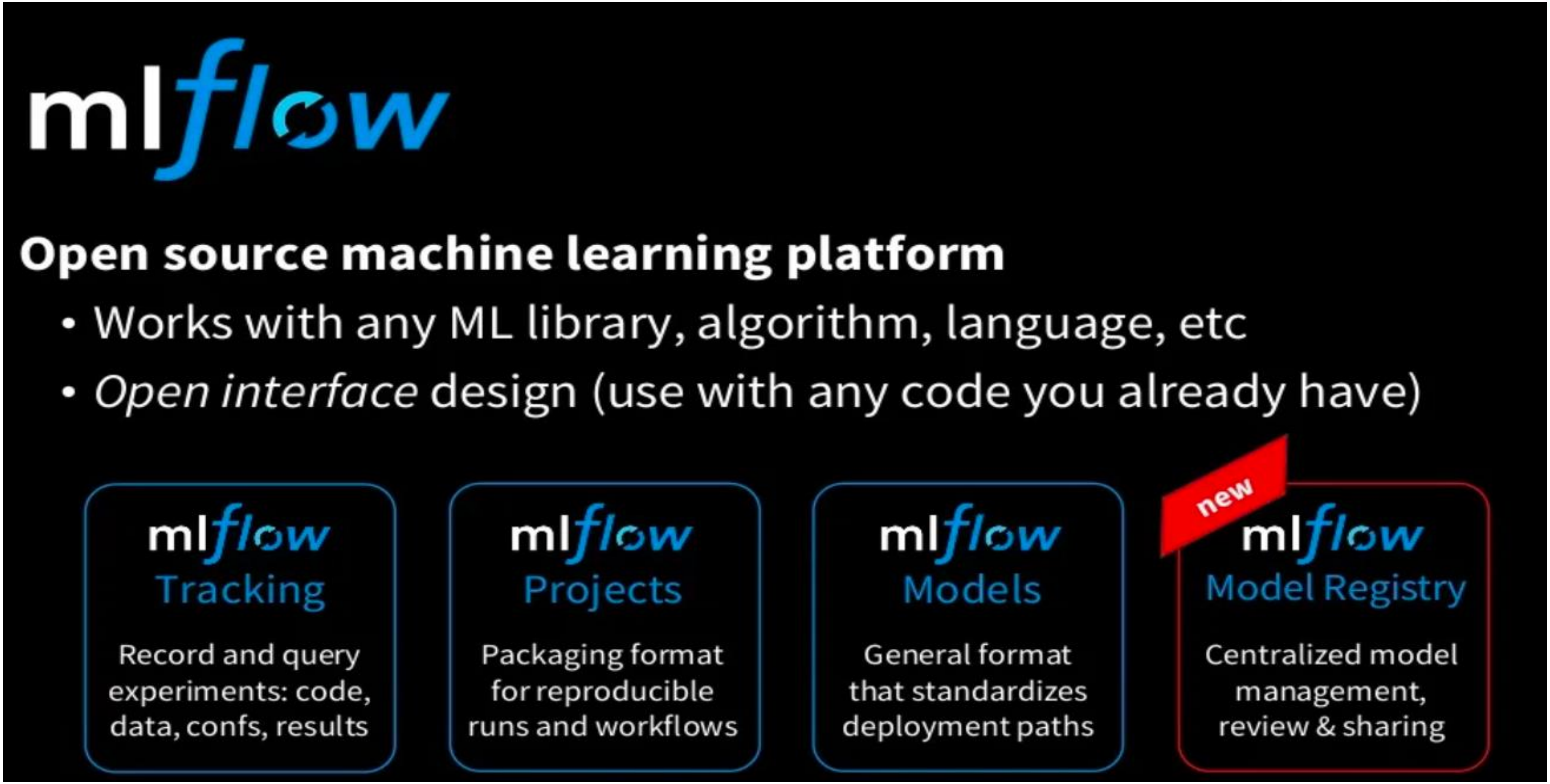
Always run this code.

CI/CD Tools



Machine Learning Model Management

Features	DVC	mlFlow	Sacred
Language Support	Universal using Shell	Python, R, Java , REST	Python
Control	DAG using scripts	Python package	Annotations
Storage Backend	File Storage	File Storage	MongoDB + others
License	Apache 2	Apache 2	MIT
Notebook support	Yes	Yes	Only Dry-run
Model Storage	Yes	As Artefacts	Only Artefacts
Input Data Storage	Yes	Only Artefacts	Yes
Parameter Tracking	Yes	Yes	Yes



mlflow

Open source machine learning platform

- Works with any ML library, algorithm, language, etc
- *Open interface* design (use with any code you already have)

mlflow Tracking
Record and query experiments: code, data, confs, results

mlflow Projects
Packaging format for reproducible runs and workflows

mlflow Models
General format that standardizes deployment paths

new

mlflow Model Registry
Centralized model management, review & sharing

THANK YOU