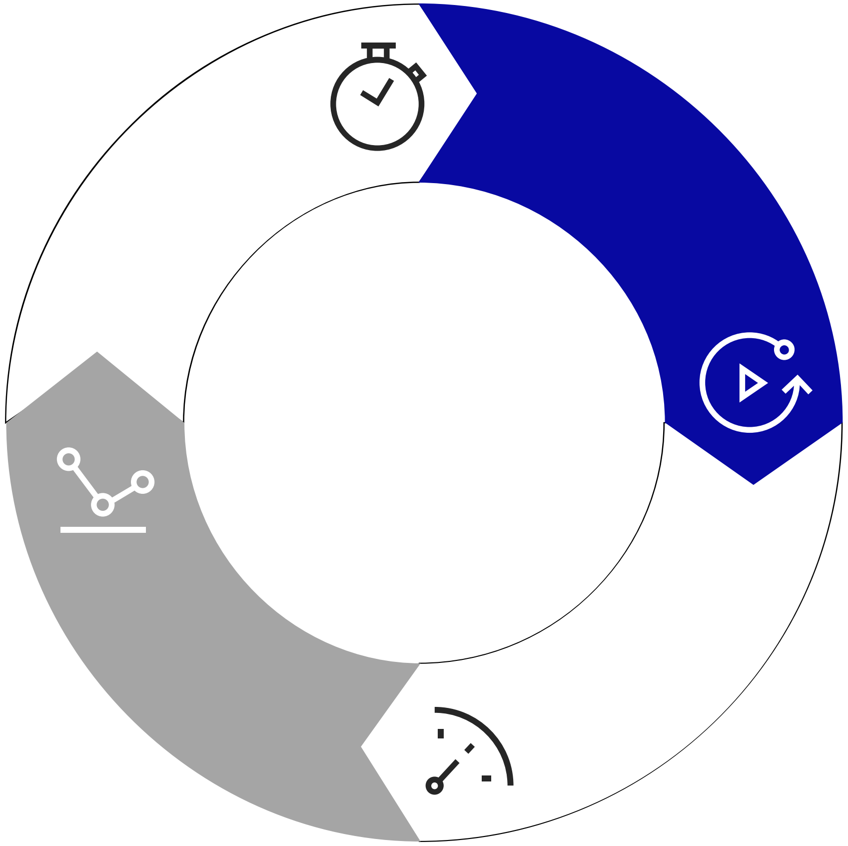
MLOps and LLM Observability

Karthikeyan Vaiyapuri

01

Overview

Topics



01 Evaluating model performance and metrics

02 Monitoring during training and inference

03 Model deployment basics with Flask/Streamlit

04 Hands-on lab: Deploy a simple model

02

Introduction to Model Validation Techniques

Introduction to Model Validation Techniques

What is model validation?

- Assessing how well your model generalizes to unseen data
- Prevents overfitting and underfitting

Common Techniques:

- **Hold-out validation:** Split data into train/test sets (e.g., 80/20)
- **K-fold cross-validation:** Rotate through K train/test splits for robust estimates
- **Stratified sampling:** Maintains class proportions in splits

Goal: Ensure your model performs well on real, unseen data.

03

Evaluating Model Performance and Metrics

Evaluating Model Performance and Metrics

Classification

(Metrics)

- **Accuracy:** Overall correctness of predictions
- **Precision/Recall/F1:** Quality and coverage of positive predictions
- **AUC-ROC:** Probability curve for distinguishing classes
- **Confusion Matrix:** Breakdown of true/false positives/negatives

Regression

(Metrics)

- R^2 Score: Proportion of variance explained by model
- MAE, RMSE: Average prediction errors

Best Practice

Always report multiple metrics and visualize them for every model.

04

Interpreting Model Metrics

Why does metric choice matter?

> Different applications value metrics differently

Precision in fraud detection

Recall matters in health Diagnosis

How to flag issues?

High accuracy + low recall

Model misses many positives (false negatives)

High recall + low precision

Model creates many false alarms (false positives)

What’s a “good” score?

Depends on dataset, business context, and risk tolerance

05

Monitoring Metrics During Training and Inference

Training Phase Monitoring

01

Loss curves

Watch for convergence or divergence

02

Accuracy over epochs

Detect over/underfitting

03

Early stopping

Halt training if validation loss increases

04

Time per epoch/batch

Performance monitoring

Inference (Production) Monitoring

Prediction latency

How fast are results returned?

Error rates and input distributions

Detecting data/model drift

Ongoing evaluation

Periodically validate with new data

06

Introduction to Model Deployment

Goal

Make model available for real-world use

01

Why deploy?

Enable predictions via APIs or user-facing applications

Integrate into business processes

Deployment Methods

01

REST API (Flask/FastAPI)

02

Web app (Streamlit/Gradio)

03

Batch prediction

07

Deploying with Flask – RESTAPI Approach

Flask

01

Lightweight Python web framework

How it works

Model loaded as a Python object

Receives HTTP requests with input data

Returns predictions as JSON

Sample Flask API Structure

01

Example Code

```
@app.route('/predict', methods=['POST'])
def predict()features = request.json['features']
prediction = model.predict([features]).tolist()
return jsonify({'prediction': prediction})
```

02

Use Case

Easy integration with applications and microservices

08

Deploying with Streamlit – Interactive Web App

Deploying with Streamlit – Interactive Web App

01 Streamlit:

Tool for turning Python scripts into shareable web apps

02 Why Streamlit?

- Easily Display predictions, charts, explanations
- Instant web UIs with minimal code

Sample Streamlit UI

Example Code

```
import streamlit as st
input_val = st.number_input('Enter a value')
if st.button('Predict')pred = model.predict([[input_val]])
st.write(f"Prediction: {pred[0]}")
```

Use Case:

Demos, prototyping, data science reporting

09

Comparing Deployment Patterns

Flask

01

	Flask	Streamlit
Type	REST API	Web App
Users	Developers, other systems	End-user,business stakeholders
Interactivity	API calls only	Full GUI, sliders, charts
Best for...	Production integration	Demos, rapid prototyping

"

10

Lab Exercise – Deploy a Model

Objective

01

Make trained model accessible through Flask API and Streamlit app

Lab Steps

Prepare your trained model

Pickle/joblib save from Day 1 or Day 2

Flask Deployment

Write a /predict endpoint
Test with curl or Postman

Streamlit Deployment

Create input form for user features
Display prediction and basic metrics

Test & Validate

Try several prediction examples
Confirm outputs are correct and responsive

Thanks

Karthikeyan Vaiyapuri