Iris Classification Infrastructure

General Information

To demonstrate experience with machine learning infrastructure will need to solve the **Iris classification problem**. The goal is to create pipelines for data extraction, transformation, training, and inference on a Kubernetes cluster.

Problem

Iris classification is a classic problem for data science: you can find additional information here. As a reference, we will use a working example from the official Tensorflow guide (https://www.tensorflow.org/tutorials/customization/custom_training_walkthrough).

Requirements

1. Need to deploy Kubernetes cluster. For simplicity, you can use minikube, kind, or microk8s.
2. Need to have 2 separated solutions - training pipeline and inference pipeline
3. Need to configure a remote storage, that will be available from the cluster(it can be mounted to the containers as volume)
4. Need to configure resources monitoring (Prometheus + Grafana). According to your judgment include the most important and relevant metrics from the underlying Kubernetes nodes and pods.
5. Need to "wrap" the code from the TensorFlow guide into docker containers for the following tasks:

Task	Description	Output
Dataset downloading pipeline	Docker container to download the iris dataset into the storage from previous step	iris_training.csv and iris_test.csv are in storage
Training pipeline	Wrap "Creating a model using Keras" and "Train the model" chapters code into a docker container	keras model in storage
Optimization pipeline	Wrap "Create an optimizer" chapter code into a docker container	keras model in storage
Evaluation pipeline	Wrap "Evaluate the model's effectiveness" chapter code into a docker container and implement a script to write an evaluation report to the text file	evaluation.txt in storage
Prediction cluster	Deploy API for iris prediction in ai_services namespace. Requirements: It can be simple python api (flask, fastAPI or any other framework) or a tensorflow server https://www.tensorflow.org/tfx/tutorials/serving/rest_simple (chapter "Serve your model with TensorFlow serving") Need to add more than 1 instance for fault tolerance (replicas > 1)	API is available inside the kubernetes

Extra options (optional)

- You can use any frameworks/tools that you like to create cluster, pipelines, and service.
- If you have experience with ML Flow, you can deploy ML flow cluster and log all pipeline steps/evaluation metrics to ML flow in addition to a text file
- If you have any additional experience related to the infrastructure (benchmarks, helm specs, Traefik configuration and so on) feel free to demonstrate it - it will be a strong plus.

Output

1. Archive with Kubernetes specs, python code, docker files, and outputs (model and evaluation)
2. Brief description of the infrastructure, pipelines, and log with an example of API usage (for example when you send a request to working infrastructure)
3. Cover letter: describe how you could solve a similar problem in a production project for sentiment analysis task (you may add extra steps to the pipeline if needed)

Validation

To score the solutions we will use the following criteria:

- Working solution with all pipelines and service
- Code quality, comments, and readable internal structure
- Cover letter: understanding of machine learning in production
- Extra options: any deep/specific knowledge is a plus