Math-Topic Classification with MLflow

AIN-3009 — MLOps Term-Project Report

Firas Elbayoumi — 2100503

May 17, 2025

Contents

1 Introduction

This project implements a complete MLOps workflow that classifies any mathematical question into one of eight well-defined topics. Beyond mere model training, it tracks every experiment end-to-end with MLflow, automates hyper-parameter search, and packages the winning checkpoint as a reproducible MLflow artifact. The workflow further demonstrates real-world operability by deploying the model as an Azure ML online endpoint and wiring a lightweight FastAPI front end for interactive use.question to one of eight topics, using data from the Kaggle competition *Classification of Math Problems by Kasut Academy*.[1] The work is steered by three objectives:

1. **Efficiency** — exploit 4-bit QLoRA so that large language models (LLMs) can be fine-tuned on a single P100.
2. **Reproducibility** — capture every experiment with MLflow and promote the best run through a governed model registry.
3. **Operability** — expose an Azure ML online endpoint and provide a minimal web front end suitable for educational platforms.

The delivered system achieves **0.9114 F1-micro** on the private leaderboard (7^{th}/341).

2 Dataset

The competition supplies the training and test corpora in plain-text CSV files: 21 600 labeled questions and 5 400 unlabeled questions, each row containing a single problem statement. Over 97 % of questions are fewer than 500 characters ($\approx$ 200 tokens), making them well within the context window of modern small LLMs. A stratified 70 %/30 % train/validation split is used with random seed 42 so every topic is represented proportionally.

Table 1: Eight target classes provided by the competition.

ID	Topic
0	Algebra
1	Geometry and Trigonometry
2	Calculus and Analysis
3	Probability and Statistics
4	Number Theory
5	Combinatorics and Discrete Math
6	Linear Algebra
7	Abstract Algebra & Topology

[1] Competition URL recorded in the hyperlink.

3 Methodology

3.1 Pipeline Overview

Figure 1 summarizes the lifecycle—starting with raw data and ending with a live endpoint and Kaggle submission.

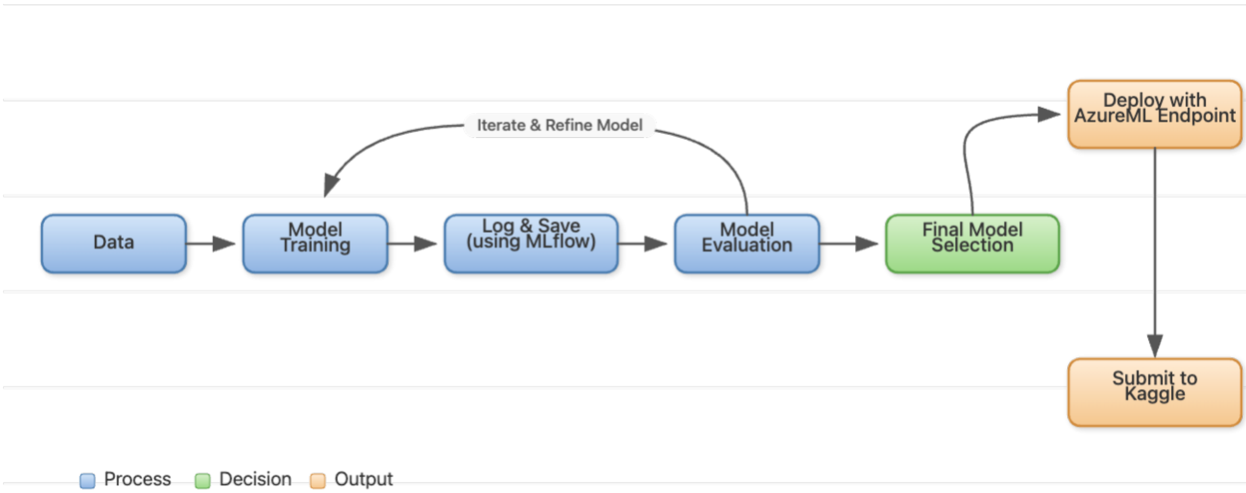

Figure 1: Full ML lifecycle: data ingestion, training, MLflow tracking, evaluation, model selection, Azure ML deployment, and final Kaggle submission.

1. **Data** — Load the text questions and perform minimal normalization (lower-casing and whitespace cleanup).
2. **Model Training** — Fine-tune a 4-bit quantized LLM with LoRA adapters; objective = cross-entropy.
3. **Log & Save** — MLflow records all parameters, metrics, and model artifacts (checkpoints, tokenizer) during training.
4. **Model Evaluation** — Validation loss and F1 are computed each epoch; early stopping prevents over-fitting.
5. **Final Model Selection** — The run with highest validation F1 is promoted in the MLflow Model Registry.
6. **Deploy with Azure ML Endpoint** — The chosen model is served through Azure MLflow workspace.
7. **Submit to Kaggle** — Predictions on the public test set are uploaded for leaderboard scoring.

3.2 Model Choice

Phi-4 Mini Reasoning (3.8 B) was selected for its strong reasoning capacity and manageable parameter count. 4-bit QLoRA compression reduces GPU memory to ≈ 5.3 GB, enabling batch size 16 on a P100.

3.3 QLoRA Configuration

- *Quantization*: NF4 integers with double-quant strategy.
- *Adapter rank*: 8 (empirically optimal).
- *Target modules*: attention and MLP projections.
- *Optimizer*: `paged_adamw_32bit`; weight decay 0.01; linear scheduler.

3.4 Training Environment

All experiments ran inside Kaggle Notebook sessions backed by a single NVIDIA Tesla P100 GPU (16 GB VRAM), 13 GB system RAM, and one vCPU. The software stack was CUDA, PyTorch, Hugging Face Transformers , and bitsandbytes. Thanks to 4-bit QLoRA quantization, the full 3.8 B-parameter model plus adapters, optimizer state, and a batch size of 16 comfortably fit within the 16 GB GPU memory limit.

4 Hyper-parameter Tuning

A grid search explored LoRA rank $\{4, 8, 16\}$ and learning rate $\{1 \times 10^{-4},\ 2 \times 10^{-5},\ 1 \times 10^{-5}\}$.

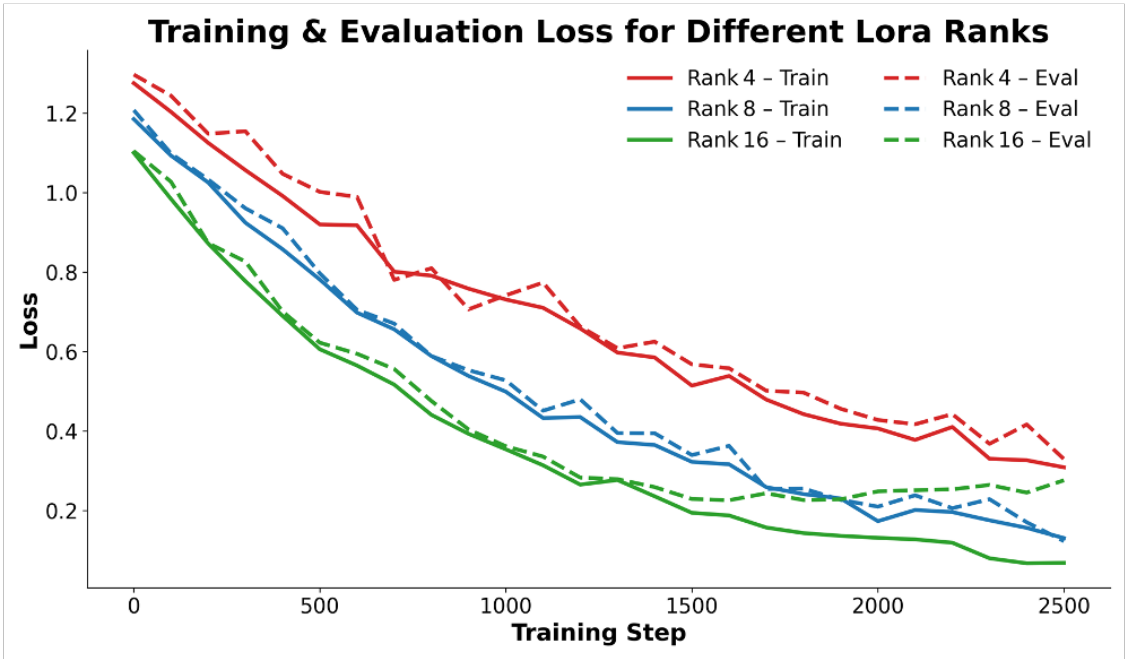

Figure 2: Training and validation loss versus steps for LoRA ranks 4, 8, 16. Rank 8 converges fastest while retaining good generalization.

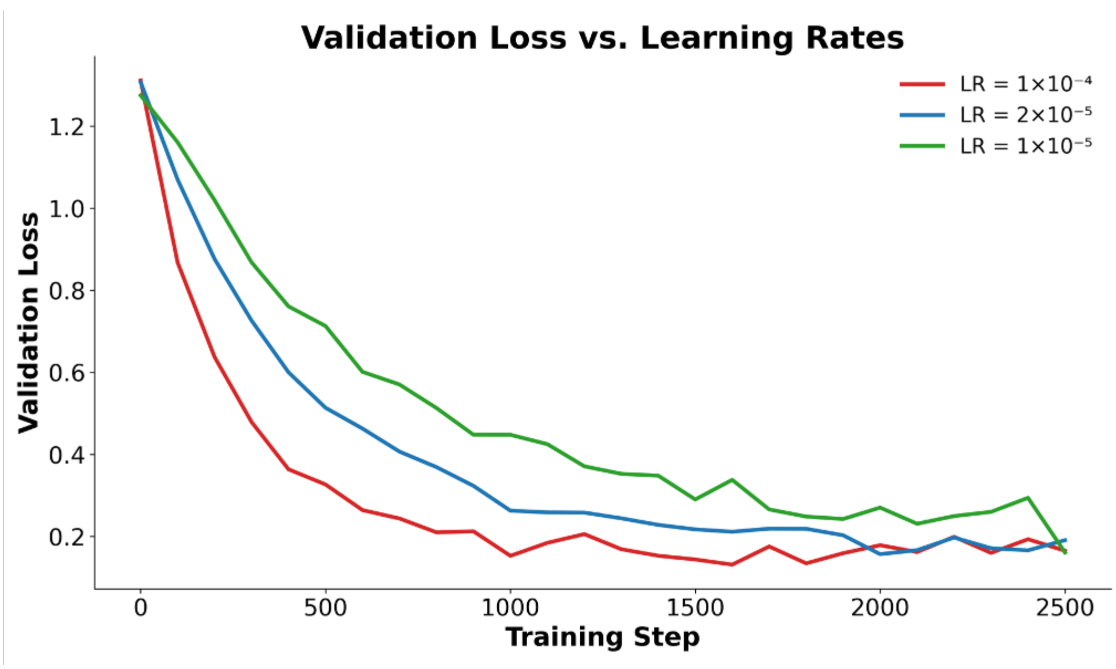

Figure 3: Validation loss trajectories for three learning rates; curves almost overlap, indicating robustness to the LR choice.

5 Results

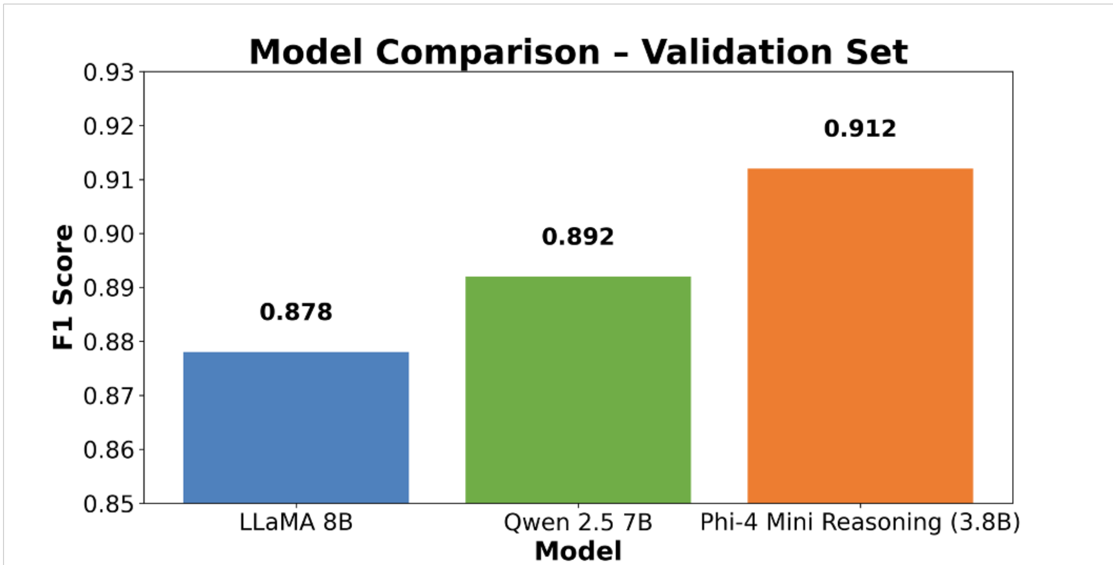

Figure 4: Validation F1 comparison—Phi-4 Mini Reasoning outperforms LLaMA-8B and Qwen-2.5-7B under the identical fine-tuning recipe.

- **Best run**: Phi-4 Mini Reasoning + LoRA rank 8 + LR 1×10^{-4}
- **Validation F1**: 0.912
- **Private-test F1**: 0.9114 ($7^{\text{th}}/341$)

6 Deployment

Model registry → Azure ML. The production run is exported by MLflow and registered on Azure ML; `conda.yaml` and a model signature are auto-generated.

Endpoint. The 3.8 B model exceeded the free CPU quota, so a distilled *Qwen-2.5-0.5 B QLoRA adapter* was merged and deployed, sacrificing ≈ 2 % F1 but meeting cost constraints. The endpoint exposes `/score` and is secured by a managed identity.

Frontend. A lightweight FastAPI UI (Figure 5) calls the endpoint, displays topic probabilities, and is containerized, pushed to Azure Container Registry, and served on Azure Web App (`B1` plan).

Live demo → `ain3009-webapp.azurewebsites.net`
Source code → `github.com/FerasOo/math-topic-classifier`

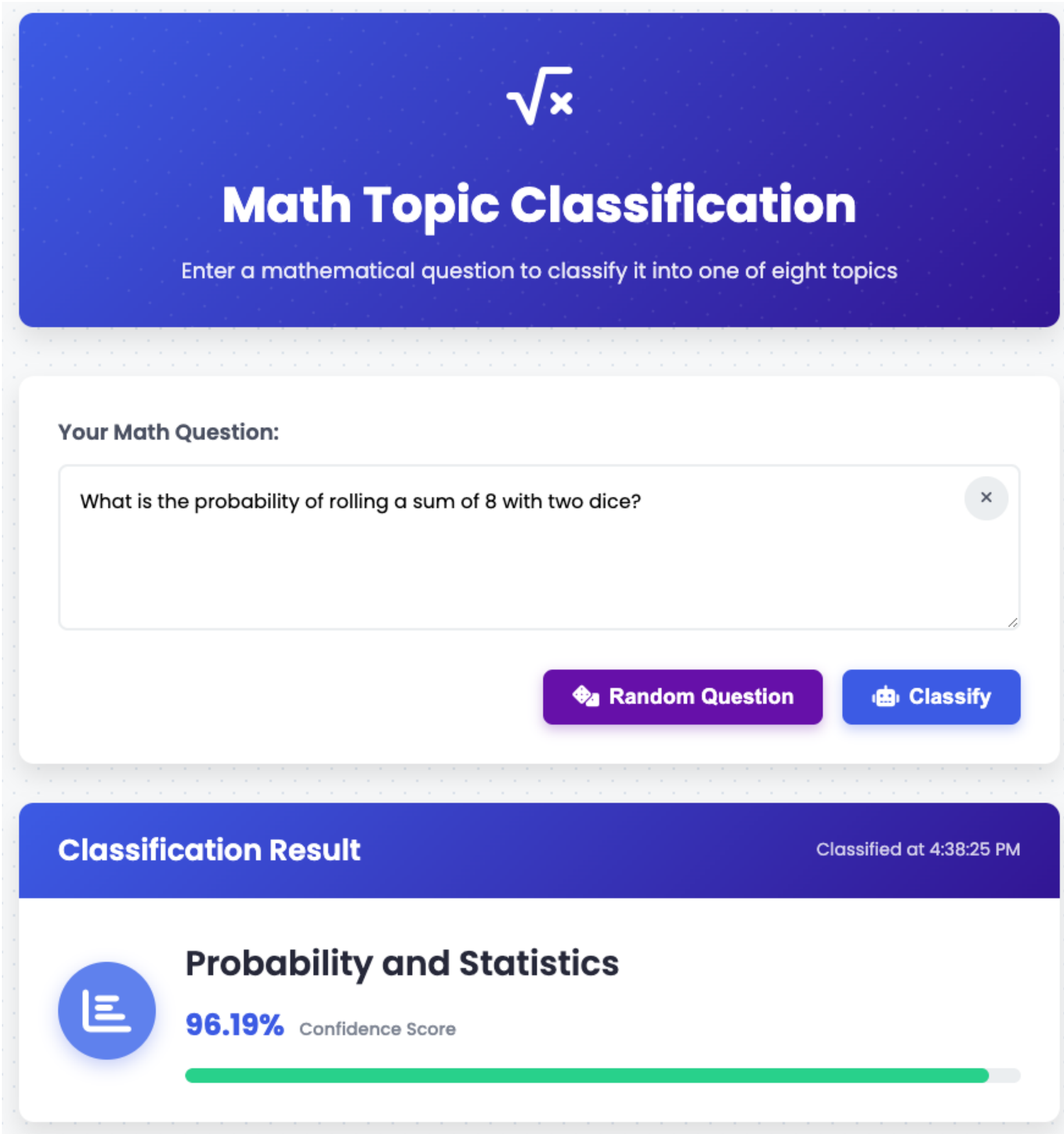

Figure 5: Screenshot of the deployed FastAPI frontend querying the Azure ML endpoint.

7 Difficulties & Lessons Learned

1. **MLflow → Azure ML Compatibility**. Local runs packaged cleanly, but Azure required explicit Python, CUDA, and `bitsandbytes` versions before acceptance.

2. **Resource Constraints**. GPU inference was beyond the course subscription, necessitating deployment of the smaller 0.5 B model—highlighting the importance of artifact-size budgeting early in the project.