DAY – 9

SUPERVISED MACHINE LEARNING & GOOGLE AI STUDIO – END-TO-END WORKFLOW

On Day 9, we shifted our focus towards Supervised Machine Learning and its integration into the Google AI Studio environment. The objective was to gain a solid understanding of how prompt engineering, coding assistance, and model evaluation can be combined to build and refine ML workflows.

We specifically studied an end-to-end ML pipeline, where Gemini AI assisted us in multiple phases including dataset preparation, code optimization, and evaluation.

1. DATASET PREPARATION FOR FINE-TUNING

We began by preparing a supervised dataset required for training and fine-tuning a machine learning model. The steps included:

- Creating input-output pairs for a prediction task.
- Formatting the dataset into a structure suitable for Gemini's model understanding (JSON, CSV).
- Ensuring label clarity and feature uniformity to reduce bias and improve generalization.

This preparation was essential for simulating a fine-tuning scenario in Google AI Studio.

2. SUPERVISED MODEL: TRAIN-TEST SPLIT & FINE-TUNING WORKFLOW

Using the prepared dataset, we explored how to perform train-test splitting, which is critical in supervised learning to validate model performance. From there, we moved on to fine-tuning a model, where Gemini assisted in:

a. Prompt Engineering for Model Training

- Designing task-specific prompts that help generate relevant training and evaluation code.
- Using Gemini's intelligence to auto-suggest prompts based on input goals.

b. Code Optimization with Gemini

- Leveraging Gemini to clean, optimize, and explain the ML code generated during the fine-tuning process.
- Ensuring the code follows Python best practices and reduces computational overhead.

c. Prompt Engineering for Debugging

- Providing error messages to Gemini and getting detailed responses on:
    - Why the error occurred
    - Suggestions for fixing it
    - Directly editable, corrected code blocks

This significantly accelerated the debugging process and improved our understanding of error resolution using AI.

3. AI-BASED CODE EVALUATION

As a final step, we tested the generated and optimized code using the dataset, and evaluated the output with the help of AI. Key observations included:

- Gemini was able to validate model logic, suggest corrections, and identify inefficiencies.
- The responses were context-aware, meaning Gemini understood the dataset structure, model objective, and desired output.
- This built confidence in using AI not just for coding but also for reviewing and validating ML pipelines.

CONCLUSION

Day 9 was a deeply technical and insightful session where we explored the practical application of prompt engineering in machine learning workflows. Through Google AI Studio and Gemini, we were able to:

- Prepare a supervised dataset for fine-tuning
- Train and test models with AI-generated code
- Debug and evaluate the models intelligently using prompts

This end-to-end experience made it clear how AI tools like Gemini can become integral to automated model development, especially for students, data scientists, and developers building scalable ML systems.