UDLAP®

Emiliano Mixali Tofas Rodríguez – 175413

Second Evaluation Summary Reports

Artificial Intelligence - DSI5022

November 30, 2025

Autumn 2025

Machine Learning Course Summary Report

1. Main Concepts Learned

The course curriculum spanned foundational data processing to advanced Large Language Model (LLM) architectures and responsible AI practices. Key takeaways include:

- Feature Engineering & Data Preparation:
    - Scaling: Techniques like Linear scaling (mapping to 0-1) and Z-score scaling (using standard deviations) are vital, though Z-score is often preferred for handling outliers. Log scaling is specifically useful for power-law distributions.
    - Handling Data: Strategies include binning for clustered data, encoding categorical data into vectors, and handling sparse features where values are predominantly zero.
    - Pipeline Integrity: The importance of detecting Training-Serving skew (schema or feature skew) to ensure production data matches training data.
- Model Architecture & Optimization:
    - Regularization: Using regularization or early stopping to prevent overfitting by penalizing complexity or halting training before convergence.
    - Neural Networks: The use of ReLU activation functions to mitigate the vanishing gradient problem and Softmax for multi-class classification.
    - LLMs: Understanding Transformers (Encoders/Decoders), Self-attention mechanisms, and Embeddings to project high-dimensional data into lower-dimensional space.
- Responsible AI (Fairness):
    - Understanding various biases, such as Selection bias (coverage/sampling gaps) and Automation bias (favoring automated results regardless of error rate).
    - Metrics for evaluation including Demographic Parity and Equality of Opportunity.

2. Example Application: Fairness in Income Prediction

Exercise: *Addressing Bias and Fairness in ML Models* (fairness_income.ipynb) Goal: Predict if an individual earns >$50k using the ACSIncome dataset.

I implemented a binary classification model and utilized the TensorFlow Model Remediation library to address demographic disparities.

- The Issue: Initial evaluation using Fairness Indicators revealed a disparity in the False Negative Rate (FNR). The model was disproportionately failing to identify high-earning females compared to males.
- The Solution: I applied MinDiff, a remediation technique that penalizes the model during training for distributional differences in predictions between the sensitive group (female) and non-sensitive group (male).
- Result: The remediated model significantly reduced the FNR gap between groups while maintaining overall accuracy (AUC), successfully demonstrating "Equality of Opportunity".

## 3. Personal Reflection

The module on Fairness was the most intellectually stimulating portion of the course. The concept of Counterfactual Fairness—stipulating that changing a sensitive attribute while keeping other features constant should yield the same prediction—is a powerful standard for ethical AI.

However, working through the notebooks triggered a specific concern regarding technical debt. I noticed the environment setup required pinning tensorflow==2.15 and fairness-indicators==0.46.0. Seeing these older versions raises a fear regarding the longevity and maintenance of these remediation tools. In a rapidly evolving field, relying on libraries that aren't aggressively maintained creates a risk that these crucial fairness tools might become incompatible with modern production pipelines. It highlights that responsible AI is not just about math, but also about maintaining the infrastructure to support ethical checks over time.

Course Summary: AI Python for Beginners

1. Main Concepts Learned

This course provided a foundational bridge between standard Python programming and practical AI application. The curriculum progressed from core syntax to integrating Large Language Models (LLMs) into software.

- Python Fundamentals: Mastery of essential programming building blocks including variables, data types (strings, integers, floats, booleans), and control flow mechanisms (conditional if/else statements).
- Data Structures & Logic: Utilization of Lists for ordered storage and Dictionaries for key-value data management, paired with for and while loops to automate repetitive tasks.
- Modular Code: Defining custom Functions to encapsulate logic, improve code reusability, and handle variable scopes effectively.
- AI Integration (LLMs): The defining feature of the course was learning to use the OpenAI API. Key concepts included:
    - Setting up API keys and environment variables.
    - Understanding the "Chat Completion" format (System vs. User roles).
    - Parsing LLM responses to extract usable data for Python applications.

2. Example Application: LLM-Powered Chat Loop

Exercise: *Building an Interactive AI Helper* (Derived from Lesson_7 and Lesson_8 concepts)

One of the core implementations involved creating a script that moves beyond simple print statements to meaningful AI interaction. I implemented a function, often styled as get_llm_response(), which acted as a wrapper for the API call.

- Implementation:
    - I utilized the openai library to connect to a model (e.g., GPT-3.5/4).
    - I wrote a helper function that accepts a user prompt as an argument and returns the model's text string.
    - The Application: This function was integrated into a while loop, creating a continuous terminal-based chat interface where I could query the AI for code debugging or creative writing, and it would generate responses in real-time within the notebook environment.

3. Personal Reflection

Overall, I found this course to be a very accessible entry point into AI engineering.

- Difficulties: I encountered no real difficulties; the progression was logical, and the coding challenges were well-scoped for a beginner level. Considering I’m a software engineer this was a walk in the park.
- Most Interesting Aspect: The standout experience was the ability to query a Large Language Model directly from the codespace environment. Moving interaction from a web browser (like ChatGPT) to a code cell gave me a tangible sense of control over the AI, highlighting how easily intelligence can be embedded into standard Python scripts.