--- Logical Execution Order Analysis ---

Based on dependencies and typical data science workflow:

1. Data Loading: data.py

2. Preprocessing: preprocessing.py (depends on model.py)

3. Model Definition: model.py

4. EDA: eda.py (depends on data.py)

5. Training/Evaluation: train.py (depends on data.py, preprocessing.py, model.py)

6. Nested CV: nested\_cv.py (depends on data.py, preprocessing.py, model.py)

7. Early Risk: early\_risk.py (depends on data.py)

8. Segmentation: segmentation.py (depends on data.py, preprocessing.py)

9. Concepts/Causal Analysis: concepts.py (depends on data.py)

10. Sequence Models: sequence\_models.py (potential dependency on data.py, preprocessing.py)

11. Dashboard/Reporting: dashboard.py (depends on generated tables/figures, likely data.py)

Proposed Logical Workflow Sequence:

1. data.py (Load data)

2. model.py (Define model - needed for preprocessing pipeline)

3. preprocessing.py (Build preprocessing pipeline - depends on model)

4. eda.py (Perform EDA - depends on data)

5. train.py (Train model - depends on data, preprocessing, model)

6. nested\_cv.py (Perform nested cross-validation - depends on data, preprocessing, model)

7. segmentation.py (Perform segmentation - depends on data, preprocessing)

8. early\_risk.py (Analyze early risk - depends on data)

9. concepts.py (Perform causal analysis - depends on data)

10. dashboard.py (Run Streamlit dashboard)

cd C:\Users\MyName\Documents\Github\10-Aug-25

**# Generate EDA plots**

**python -m src.eda**

**python -m src.train --csv-path student-mat.csv**

**# Generate model performance metrics and explanations**

**python -m src.train --csv-path student-mat.csv --model-type random\_forest --param-grid default**

**# Generate fairness metrics**

**python -m src.train --csv-path student-mat.csv --group-cols sex**

**# Generate nested CV results**

**python -m src.nested\_cv**

**# Generate concept importance**

**python -m src.concepts**

**# Run Dashboard**

**streamlit run dashboard.py**

pip install captum

From the root of your repository, run this in your shell or PowerShell:

pip install -r requirements.txt

If you’re using **Anaconda**, you’d first activate your environment:

conda activate your\_env\_name

pip install -r requirements.txt

If you want to ensure everything is fresh and compatible, you can also do:

python -m pip install --upgrade pip

pip install -r requirements.txt

Do you want me to also give you the **command for creating a fresh virtual environment** before installing requirements? That would help avoid the CLI mismatch you’ve been getting.

Make an educator-facing dashboard integration (Keep old/existing dashboard separate)

Create 2 new versions of dashboard (learning analytics dashboard)

One dashboard teacher facing and another student facing dashboard

Transfer learning

I want to apply transfer learning. Apply deep learning on OULAD. then apply on small model i.e. UCI student dataset. map some or more OULAD and UCI student dataset.

Deep learning

Apply more extensive deep learning on oulad dataset

Explore dataset and look where and what type of deep learning could be applied

Transfer learning-2

Logically map oulad dataset features with uci student data. Perform extensive transfer learning

Explore Process mining on oulad dataset