Data Pipeline Assignment

Teammates: Amitesh Tripathi, Deepak Udayakumar, Dinesh Sai Pappuru, Rohit Kumar Gaddam Sreeramulu, Sneha Amin

Dataset Information:

The PMC-Patients Dataset is an extensive and unique resource designed for clinical decision support in healthcare research. This dataset contains 167,000 patient summaries derived from PubMed Central (PMC) case reports, annotated with over 3.1 million relevance scores for patient-article relationships and 293,000 similarity annotations between patients. These annotations provide a foundation for clinical retrieval tasks and relational analyses, supporting both benchmarking and model evaluation.

Dataset Purpose and Tasks:

The dataset primarily supports Retrieval-based Clinical Decision Support (ReCDS) benchmarks with two core tasks:

- Patient-to-Article Retrieval (PAR): Matching patient summaries to relevant research articles.
- Patient-to-Patient Retrieval (PPR): Identifying similar patient cases based on shared characteristics.

Data Format and Attributes:

The PMC-Patients dataset is provided in a CSV file format (PMC-Patients.csv), which includes several important columns:

- patient_id: Continuous numerical ID for each patient.
- patient_uid: Unique identifier for each patient, combining the PubMed ID (PMID) with an additional index to distinguish multiple patients within one article.
- PMID: Unique identifier of the source article for each patient summary.
- file_path: Path to the XML file containing the original case report.
- title: Title of the source article.
- patient: Textual summary of the patient's case.
- age: Tuple list detailing patient age, with values and units (e.g., year, month).
- gender: Coded as 'M' (Male) or 'F' (Female).
- relevant_articles: Dictionary with relevant article PMIDs as keys and relevance scores (1 or 2) as values.
- similar_patients: Dictionary linking similar patient_uids with similarity scores (1 or 2).

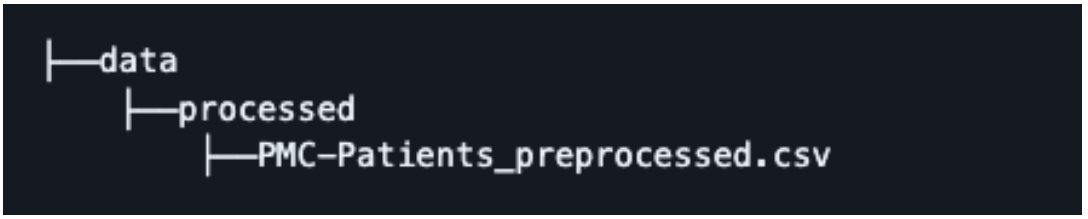
Folder Structure:
Link to readme file:
https://github.com/theamiteshtripathi/PatientInsight/blob/main/docs/data_pipeline_README.md

- data/: Stores raw and processed data files used throughout the project.
- scripts_location/: Contains source code for data processing and analysis tasks.
- tests/: Includes unit tests to ensure code functionality and reliability.
- airflow/: Holds Airflow DAG definitions for orchestrating data workflows.

Data Pipeline Components

1. **Downloading Data :** The download.py script is responsible for downloading the PMC-Patients dataset. It loads AWS credentials from environment variables, checks for the existence of the data/raw directory, and then downloads and saves the dataset in CSV format.

2. **Data Preprocessing:** The preprocess.py script is designed to clean and transform the raw PMC-Patients dataset, preparing it for analysis. It selects essential features (e.g., patient ID, age, gender), converts complex age data into a unified "age in years" format, and maps gender to binary values. Additionally, it ensures that missing values in relevant articles and similar patients fields are handled appropriately. Finally, the preprocessed data is saved in the data/processed directory for downstream tasks.



3. **Unit Testing:** The test_data_download.py and test_data_preprocess.pyscripts are unit tests designed to verify the functionality of data ingestion and preprocessing components in the pipeline. The test_data_download.py script tests that download_pmc_patients_dataset successfully downloads the dataset and saves it to the specified directory. Meanwhile, test_data_preprocess.py checks that

preprocess_pmc_patients correctly preprocesses the dataset, saves it to the processed directory.

4. **Data Version Control (DVC):** DVC is used in this pipeline to manage and version data files, providing reproducibility and efficient storage management.

    - **DVC Files:** The PMC-Patients.csv.dvc PMC-Patients.csv.dvc file in data/raw/ tracks the raw dataset, and PMC-Patients_preprocessed.csv.dvc in data/processed/ tracks the preprocessed data. Each DVC file records metadata, including the MD5 hash (md5), file size (size), and path
    - **Remote Storage:** These DVC-tracked files are uploaded to an Amazon S3 bucket, allowing scalable and secure storage. By using DVC with remote storage, the pipeline can pull or push large datasets as needed without duplicating storage locally.
5. **Statistics Generation:** Thegenerate_stats.py script is designed to create a statistical summary report for the preprocessed PMC-Patients dataset. It uses ydata-profiling (formerly pandas-profiling) to generate an HTML report based on the data, which includes insights into data distribution, missing values, and other statistics.

6. **Airflow DAG for Data Pipeline Orchestration:** The data pipeline is orchestrated using Apache Airflow, which enables automated scheduling, dependency management, and monitoring of tasks. The Airflow Directed Acyclic Graph (DAG) defined here coordinates the steps required to download, preprocess, and notify upon completion of the PMC-Patients dataset processing.

    - **DAG Configuration:** The DAG, named patient_insight_dag.py, includes several configuration settings. default_args specify parameters such as the owner, start date, email notifications for failure, and retry policies.
    - **Tasks and Workflow:**
        - **Download Task:** The download_data task calls the download_pmc_patients_dataset function to retrieve the raw dataset and save it to data/raw.
        - **Preprocess Task:** The preprocess_data task executes preprocess_pmc_patients, transforming the downloaded data and saving the output in data/processed.
        - **Statistics Generation Task:** This executes the generate_stats,designed to create a statistical summary report for the preprocessed PMC-Patients dataset.
        - **Email Notification Task:** The send_email task uses send_custom_email to send a notification when the DAG completes. The TriggerRule.ALL_DONE rule ensures that the email is sent regardless of the previous tasks' success or failure.
    - **Task Dependencies:** Task dependencies are defined to ensure a logical flow. download_task runs first, followed by preprocess_task, and then email_task.

This order guarantees that preprocessing and notification only occur after the dataset is successfully downloaded.

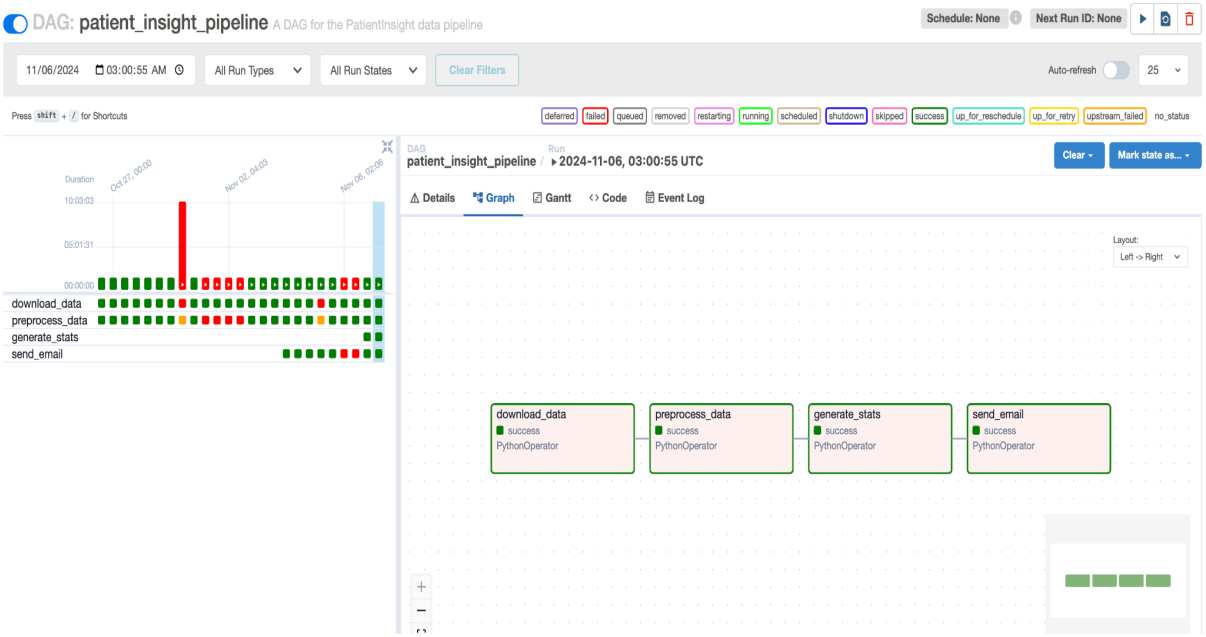


Fig 1: Graph

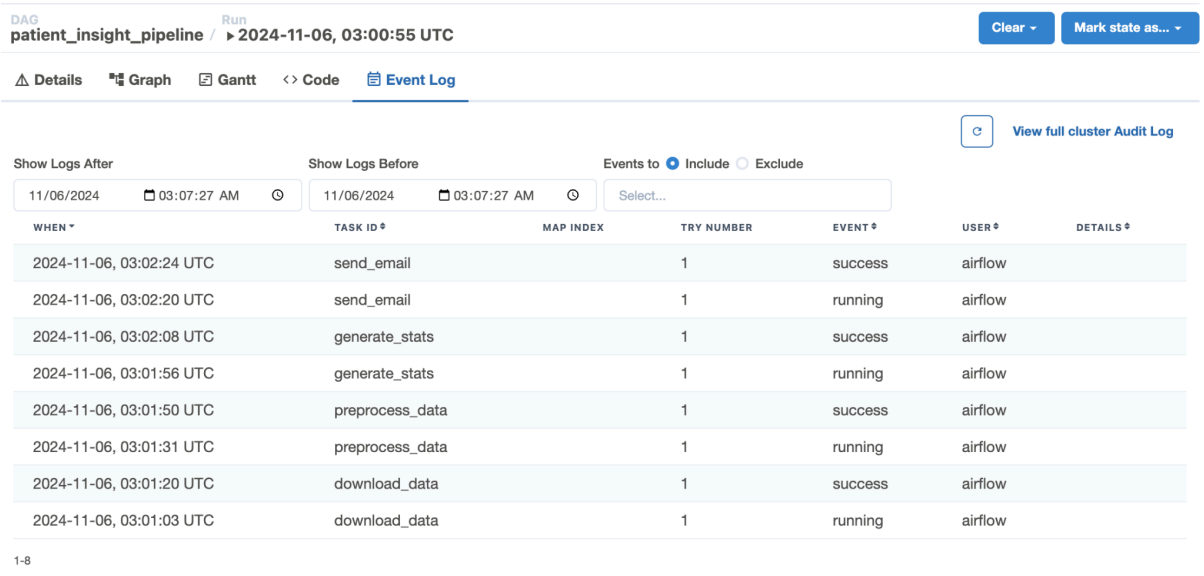


Fig 2: Event Log

- The Event Log shows the record of each task's execution status, showing timestamps and sequential task progression from download_data to send_email, which is useful for tracking task performance and debugging.

- **Alerts:** Using Airflows alert system to manage alerts. We used a new gmail account to set up mail alerts, and all the mails are directed using this email id mlopsgroup11@gmail.com. Team members get a notification if any anomalies are detected, or if the DAG runs successfully.

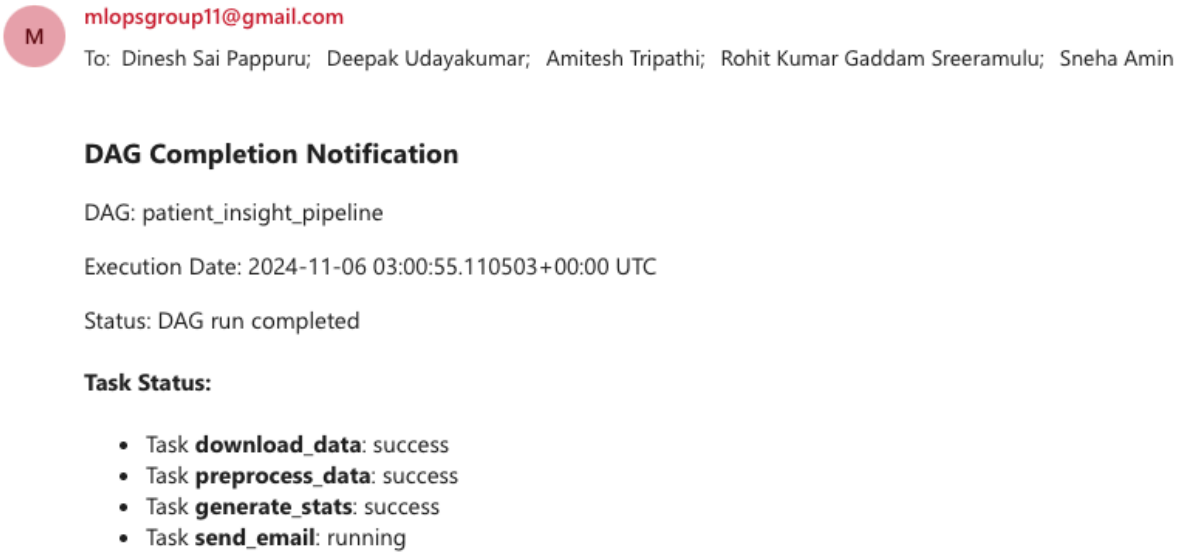


Fig 3: Email Alert