**Data Pipeline Documentation**

Overview

This DAG automates the data pipeline process, including data ingestion, validation, transformation, storage, versioning, and model building. The pipeline is designed to run on-demand with manual triggering.

Dependencies

The following dependencies are installed and configured:

* Apache Airflow (with required operators)
* Python (>=3.6)
* AWS CLI (for S3 integration)
* DVC (for data versioning)
* MLflow (for model tracking)

DAG Components

1. Import Required Modules

The script imports necessary modules from Airflow, including DAG, PythonOperator, and BashOperator, along with runpy for script execution.

1. Define Python Functions

Each function corresponds to a specific task in the pipeline and executes a Python script located in the /opt/airflow/dags/ directory:

* data\_ing() - Runs data\_ingestion.py to ingest raw data.
* aws\_s3\_upload() - Uploads raw data to AWS S3.
* data\_validation() - Runs validation checks on the ingested data.
* data\_preparation() - Prepares data for transformation.
* data\_transformation\_storage() - Transforms and stores processed data.
* dvc\_data\_versioning() - Implements data versioning using DVC.
* building\_model() - Triggers model training and evaluation.
* read\_files() - Reads files from the specified storage.

1. Define Default Arguments

The DAG includes default parameters:

* owner: Specifies the owner of the DAG.
* start\_date: Defines the DAG's start date.
* retries: Set to 0 to prevent automatic retries.
* depends\_on\_past: Set to False to ensure tasks do not depend on previous runs.

1. DAG Definition

The DAG is defined with:

* dag\_id: "test\_222"
* schedule\_interval: None (manual trigger only)
* catchup: False (prevents backfilling past runs)

1. Task Definitions

The DAG consists of multiple tasks defined using PythonOperator:

* data\_ingestion
* s3\_upload
* file\_read
* merged\_data\_validation
* merged\_data\_preparation
* merged\_data\_transformation\_storage
* data\_versioning
* model\_building

1. Task Dependencies

The pipeline follows this execution order:

data\_ingestion >> s3\_upload >> file\_read >> merged\_data\_validation >> merged\_data\_preparation >> merged\_data\_transformation\_storage >> data\_versioning >> model\_building

Expected Outcome

Upon successful execution:

1. Data is ingested, validated, transformed, and stored.
2. Processed data is uploaded to AWS S3.
3. Versioning is managed through DVC.
4. A machine learning model is trained and evaluated.

Troubleshooting

1. Airflow DAG not running: Ensure the Airflow scheduler is active.
2. AWS S3 upload failures: Check AWS credentials and permissions.
3. DVC versioning issues: Verify the DVC remote storage configuration.
4. Model training failures: Ensure required ML libraries are installed.