MLOPS

Name: Ans Zeshan Shahid

Roll Number: 20I-0543

Department: SE

Course: MLOPS

Submitted to

Sir Sami Ullah

Submission Date

May 12th, 2024

Documentation of Data Preprocessing Steps and DVC Setup

Data Preprocessing Steps:

Data preprocessing is a crucial step in the data analysis pipeline, aimed at cleaning and transforming raw data into a format suitable for analysis. In this project, we employed several preprocessing techniques to prepare the extracted text data for further analysis.

1. **Data Extraction:**

   We utilized web scraping techniques to extract data from two prominent news websites, dawn.com and bbc.com. The extraction process involved gathering links from the landing pages and extracting titles and descriptions from articles displayed on their homepages.

2. **Data Transformation:**

   Once the raw data was obtained, we performed various transformations to clean and format the text appropriately. This included:

   - **Removing HTML tags:** We used regular expressions to eliminate HTML tags from the extracted text, ensuring that only the raw text content remained.
   - **Removing punctuation:** Another preprocessing step involved removing punctuation marks from the text data. This was achieved using regular expressions to enhance the text's readability and consistency.
   - **Lowercasing:** To maintain consistency, we converted all text to lowercase. This step is essential for text analysis tasks as it prevents the model from treating words with different cases as distinct entities.
   - **Stopwords Removal:** Stopwords are common words that do not carry significant meaning in text analysis. We utilized NLTK's stopwords corpus to remove such stopwords from the text data, focusing on words that are more informative for analysis.

3. **Tokenization:**

   We employed NLTK's word_tokenize function to tokenize the text data, splitting it into individual words or tokens. Tokenization is a fundamental preprocessing step that enables further analysis, such as sentiment analysis or topic modeling.

DVC Setup:

Data Version Control (DVC) is a powerful tool used to track and manage versions of datasets, ensuring reproducibility and collaboration in data science projects. In this project, we implemented DVC to track versions of the processed data and maintain a structured workflow.

1. **Initialization:**

   We initialized a DVC project to establish version control for our data. This involved setting up a DVC repository and linking it to our project directory.

2. **Data Versioning:**

   After preprocessing the data and storing it in CSV files, we utilized DVC to track versions of the processed data. Each time the data was updated or modified, we committed the changes to DVC, creating a new version of the dataset. This facilitated easy rollback to previous versions and ensured reproducibility of our analysis.

3. **Metadata Versioning:**

   In addition to tracking data versions, we also versioned the metadata associated with each DVC push to the GitHub repository. This included documentation of the preprocessing steps, ensuring that all changes and updates to the data were accurately recorded and documented.

Workflow and Challenges

Workflow

1. **Data Extraction:**

   We utilized web scraping techniques to extract data from two prominent news websites, dawn.com and bbc.com. The extraction process involved gathering links from the landing pages and extracting titles and descriptions from articles displayed on their homepages.

2. **Data Transformation:**

   Once the raw data was obtained, we performed various transformations to clean and format the text appropriately. This included removing HTML tags, punctuation marks, and stop words, as well as converting text to lowercase and tokenizing it for further analysis.

3. **Data Storage and Version Control (DVC):**

   After preprocessing the data, we stored the processed data in CSV files. Additionally, we implemented Data Version Control (DVC) to track versions of the processed data and maintain a structured workflow. This involved initializing a DVC project, committing changes to DVC after each data transformation, and versioning metadata against each DVC push to the GitHub repository.

4. **Apache Airflow DAG Development:**

To automate the processes of extraction, transformation, and storage, we developed an Apache Airflow DAG script. The DAG included tasks for data extraction from dawn.com and bbc.com, as well as tasks for writing the extracted data to CSV files. Task dependencies were established to ensure that data extraction tasks were completed before writing data to CSV files.

Challenges Encountered

1. **Website Structure Changes:**

One of the main challenges encountered during implementation was dealing with changes in the structure of the target websites. Websites frequently update their HTML structure, which can affect the effectiveness of web scraping techniques. We addressed this challenge by regularly monitoring the websites and updating our scraping code accordingly.

2. **Task Dependencies and Error Management:**

Another challenge was managing task dependencies and error handling within the Apache Airflow DAG. Ensuring that data extraction tasks were completed before proceeding to data transformation tasks required careful configuration of task dependencies. Additionally, implementing effective error management strategies to handle unexpected issues was essential for maintaining the reliability of the workflow.