Muhammad Nibras Aamir
21i-2683
DS-M
Assignment # 2

Report

To prepare and process the JSON data for insertion into a MySQL database, I undertook several preprocessing and cleaning steps to ensure the data's integrity and usefulness. The process involved loading the data, normalising it to a suitable format for analysis, renaming columns to match the database schema, cleaning specific columns, and processing text data to enhance its quality. Below are the detailed steps and the challenges encountered, along with the resolution strategies.

Data Loading and Normalisation

Loading JSON Data: The first step involved loading the JSON file into a Pandas DataFrame. JSON data can vary in structure, particularly with nested elements, which necessitates normalisation to create a flat table suitable for SQL databases.

Normalisation: Used pd.json_normalize() to flatten the JSON structure, making it easier to transform into a tabular format.

Column Renaming and Cleaning

Renaming Columns: The column names in the DataFrame were renamed to match the expected column names in the SQL database. This is crucial for direct mapping and insertion into the database.

Cleaning Specific Columns: Some columns contained prefixed information that was not needed (e.g., 'productId: ASIN1234' was cleaned to 'ASIN1234'). I used a lambda function to split and keep the desired part.

Price Column: The price column contained dollar signs and commas, which I removed to convert the values into a float data type. This step ensures that numerical operations can be performed on the price data.

Text Data Preprocessing

Text Cleaning: The review texts and product descriptions contained unnecessary punctuation, URLs, and were not in a uniform case. I performed cleaning to remove these elements, keeping only alphanumeric characters and converting texts to lowercase. This standardisation helps in textual analysis and reduces the dimensionality of the data.

Stop Words Removal and Lemmatization: To further refine the text data, I removed stop words (common words that do not contribute much meaning) and applied lemmatization,

which reduces words to their base form. This process improves the relevance of text data for natural language processing tasks.

Image Downloading

Image Downloading: A function was created to download images from URLs present in the DataFrame. The images were saved locally, which could be useful for applications requiring direct access to these images. Handling errors and timeouts was a challenge, especially with broken or inaccessible URLs. I implemented try-except blocks to manage these exceptions gracefully.

Data Insertion into MySQL Database

Database Connection and Insertion: Using the pymysql library, I established a connection to the MySQL database and executed INSERT statements for each row of the cleaned DataFrame. This step required careful mapping of DataFrame columns to the corresponding database columns.

Challenges and Resolutions

Handling Nested JSON: The structure of the JSON file presented a challenge, particularly with nested elements. The pd.json_normalize() function was instrumental in flattening these structures for easier processing.

Text Data Cleaning: Dealing with diverse text data required a combination of regular expressions and NLP techniques to clean and standardize the data. Testing various regex patterns and adjusting the NLP pipeline ensured effective cleaning.

Image Downloading Errors: Some image URLs were broken or led to timeouts. Implementing a robust error handling mechanism ensured the process could continue smoothly without interruption.

Schema

The database schema was designed to accommodate the cleaned and processed data. It consists of five tables:

```
CREATE DATABASE IF NOT EXISTS amazon;
USE amazon;

create table amazon(
   amazon_id varchar(10) primary key,
   URL varchar(100),
   manufacturer varchar(100),
   title varchar(300),
   ASIN varchar(20)
   );
```

```
ALTER TABLE amazon
ADD FOREIGN KEY (ASIN) REFERENCES product(ASIN);


create table product(
   ASIN varchar(20) primary key,
   gender varchar(20),
   rating float,
   price float,
   description text,
   brandNAme varchar(255),
   colors varchar(2500),
    cat_id varchar(10),
    image_id varchar(10),
    rev_id varchar(10)
   );


ALTER TABLE product
ADD FOREIGN KEY (cat_id) REFERENCES category(cat_id);
ALTER TABLE product
ADD FOREIGN KEY (image_id) REFERENCES images(image_id);
ALTER TABLE product
ADD FOREIGN KEY (rev_id) REFERENCES reviews(rev_id);

create table category(
   cat_id varchar(10) primary key,
   category varchar(60),
   subCategory varchar(60));

create table images(
   image_id varchar(10) primary key,
   color_images varchar(5000),
   images varchar(1500));

create table reviews(
   rev_id varchar(10) primary key,
   review_texts text);
```