**Real-time Stock market streaming and analysis**

# **Stock prices streaming**

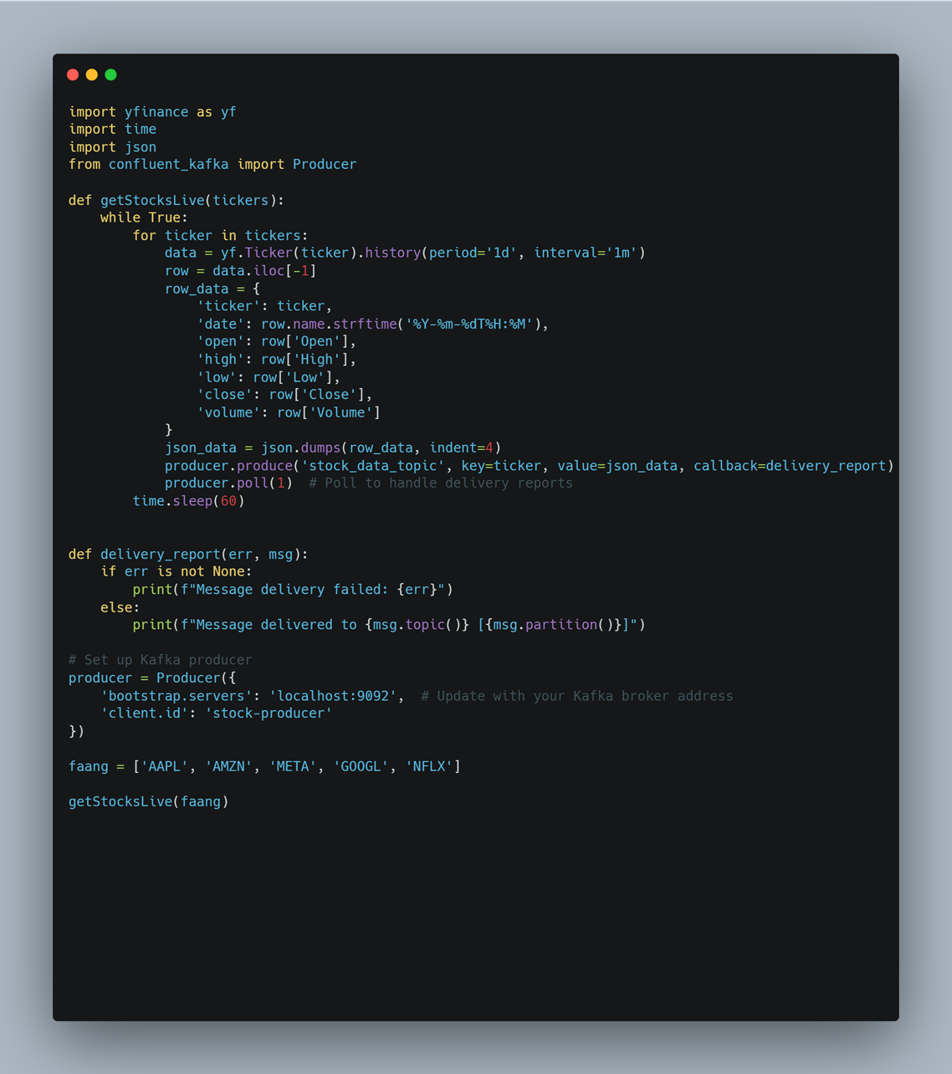
streaming FAANG stock prices from Yahoo Finance every minute with Kafka, sending it to Spark for processing then storing it in Cassandra for querying and in MongoDB for analysis

requirements:

* Zookeeper
* Kafka
* pyspark
* Cassandra
* MongoDB

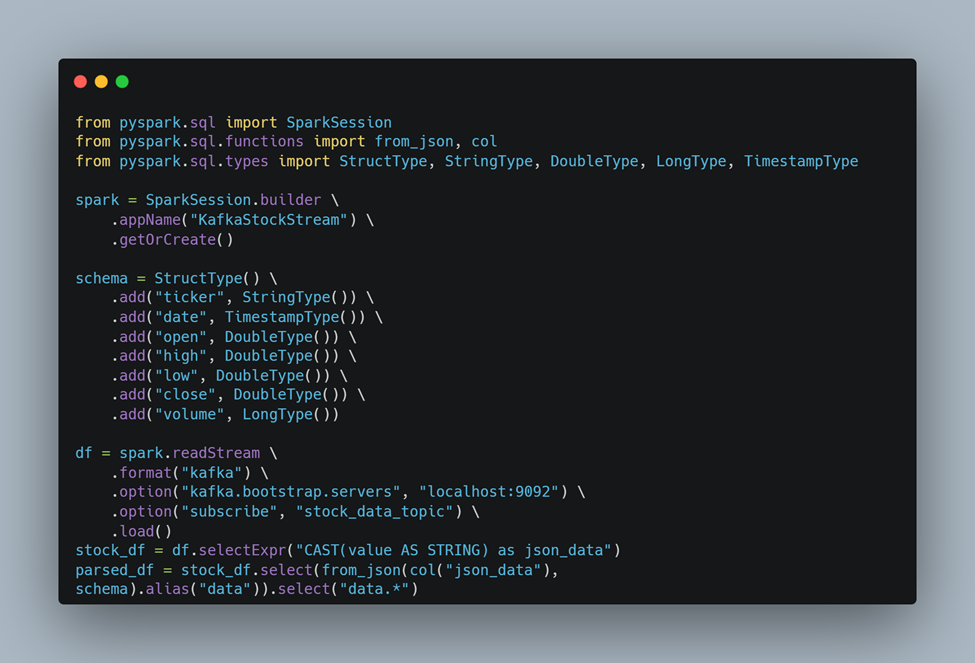
step 1: Loading data

I created a python script that fitches stock prices every minute and send it to a kafka topic via a kafka producer

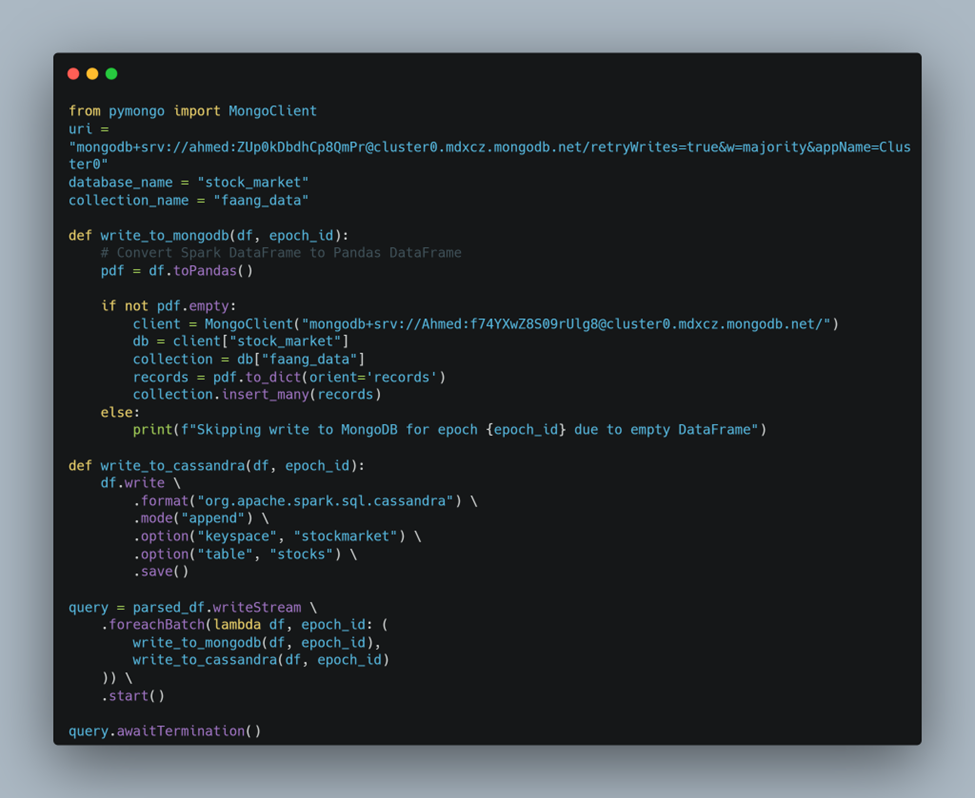


step 2: data processing:

* subscribe to the kafka topic and continuously consume stock data using structured streaming processing the data in micro batches
* define the data schema to ensure proper parsing of the data
* parse JSON data into spark dataframe based on the defined schema

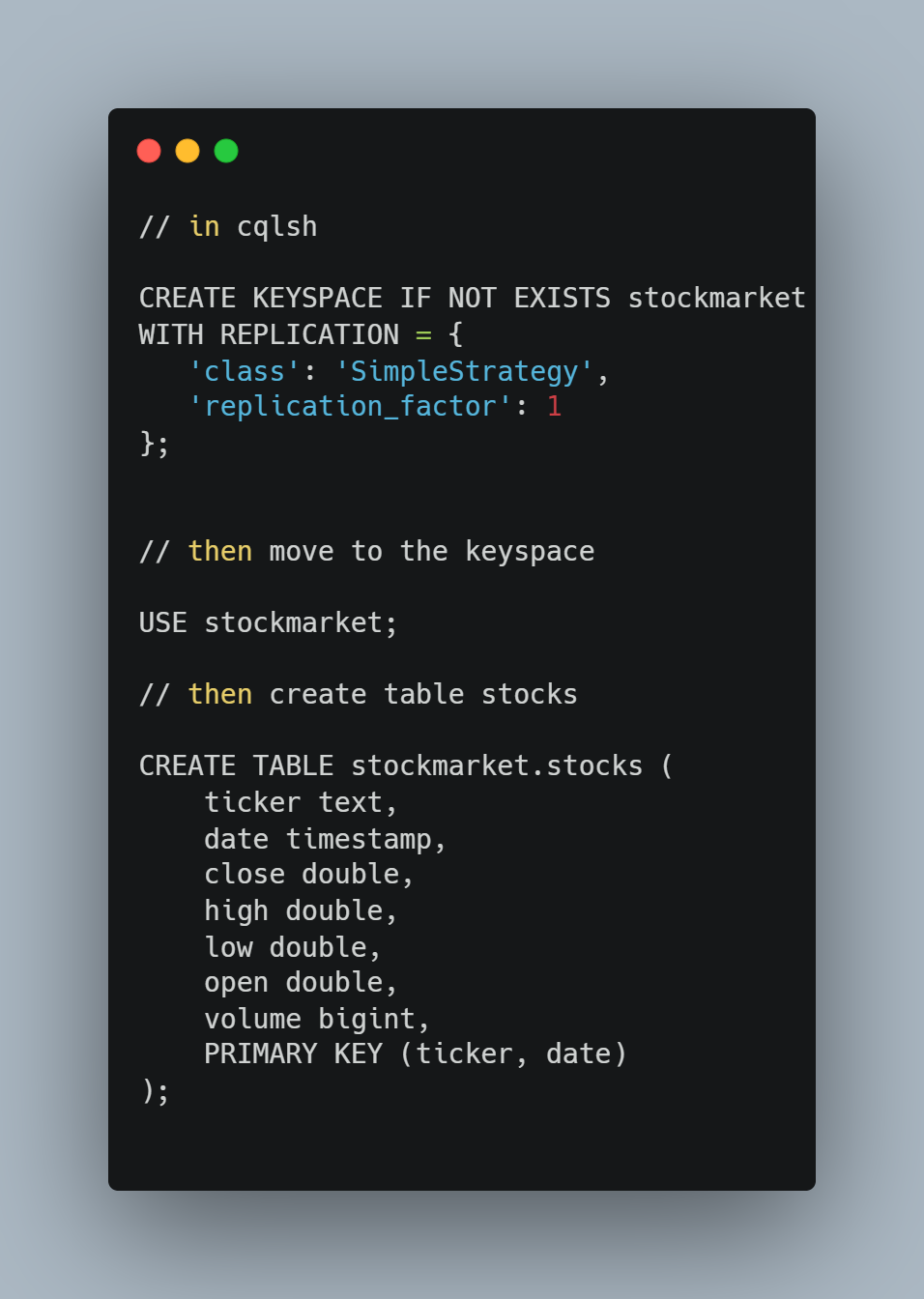


* process each micro batch and append it to cassandra and mongoDB



step 3:

create Cassandra database



step 4:

running the pipeline

* starting zookeeper and kafka

| $ZOOKEEPER\_HOME/bin/zkServer.sh start |
| --- |

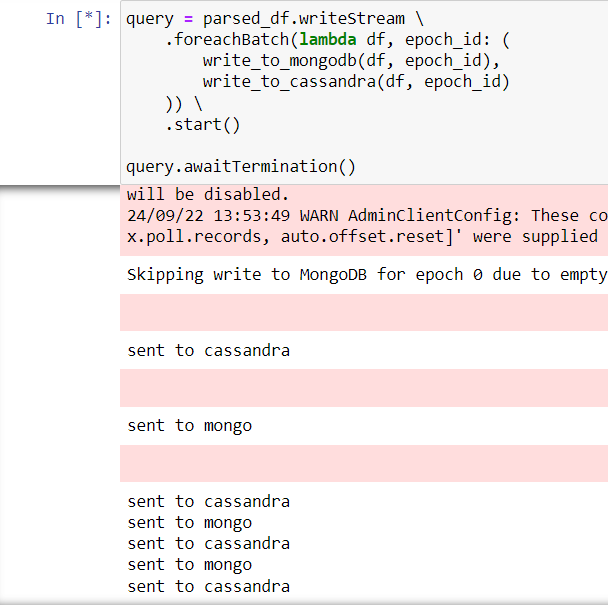
| $KAFKA\_HOME/bin/kafka-server-start.sh $KAFKA\_HOME/config/server.properties |
| --- |

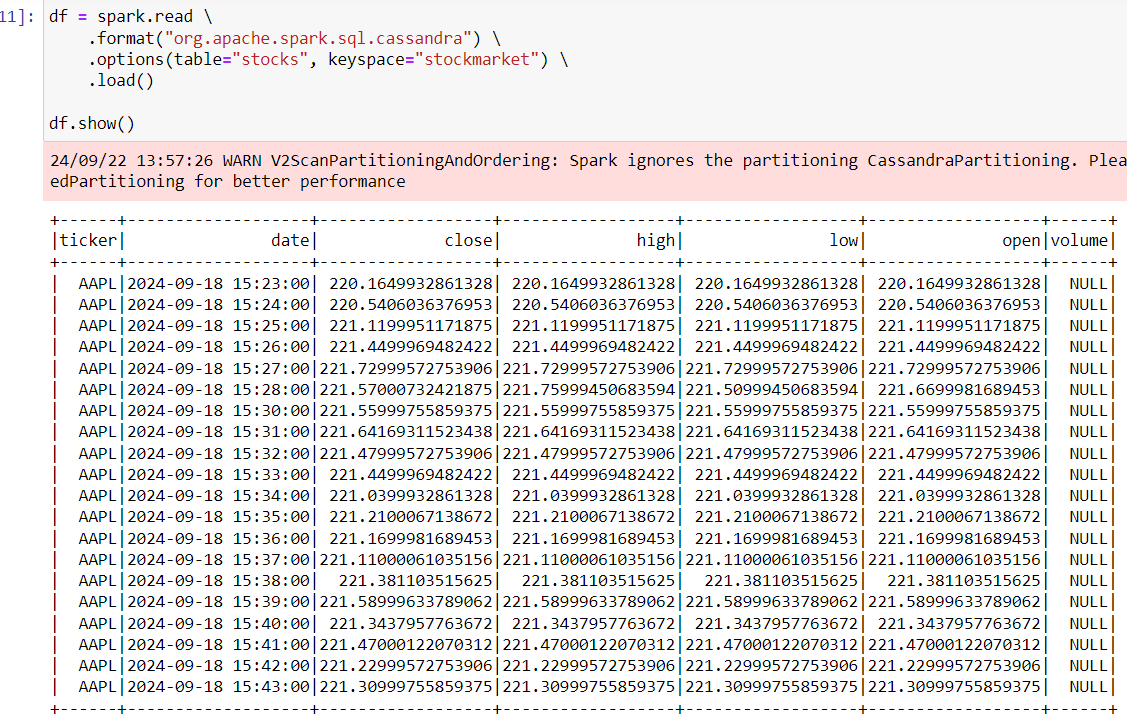
* start cassandra docker container
* starting pyspark with kafka and cassandra connectors

| pyspark --packages org.apache.spark:spark-sql-kafka-0-10\_2.12:3.5.2,com.datastax.spark:spark-cassandra-connector\_2.12:3.1.0 |
| --- |

* start the streaming in spark, start the kafka producer

| python3 liveProducer.py |
| --- |





**Stock news streaming**

stream FAANG stock-related news via flume to HDFS, then from HDFS to Spark to perform sentiment analysis using Hugging Face's transformers library, and save the results to MongoDB using PySpark Streaming.

**Prerequisites**

* PySpark
* Hugging Face transformers library
* MongoDB (with PyMongo)
* HDFS (Hadoop Distributed File System)
* Flume

**FAANG News Fetching and CSV Conversion Script (script.py)**

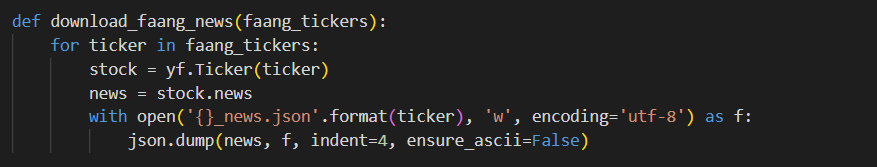
This script automates the process of downloading FAANG stock news using the yfinance library, converting the JSON data into CSV format, and scheduling it to run every 5 minutes. The resulting CSV is stored in a designated directory to be consumed by Apache Flume for further processing.

* 1. **Imports**

The script imports several key libraries:

* yfinance: For fetching stock data and news.
* requests, json, csv: For handling API requests, JSON parsing, and CSV writing.
* schedule: For scheduling tasks at fixed intervals.
* glob, datetime, time: For file management, adding timestamps, and managing the execution time.
  1. **FAANG News Downloading**

This function downloads the latest news for each FAANG stock (Apple, Amazon, Meta, Google, Netflix) and stores the data in JSON format with the ticker symbol as the file name.



**1.3. JSON to CSV Conversion**

This function converts all JSON news files to a single CSV file (merged\_news1.csv) with fields: *title, relatedTickers, publisher, and date*. A timestamp (*current\_date*) is added to each record.

A screen shot of a computer code

Description automatically generated

**1.4 Scheduled Task**

The script schedules the task to run every 5 minutes, where it downloads the FAANG news and converts it to CSV format.

* Csv file generated from script A computer screen with white and yellow text

  Description automatically generated

A screenshot of a computer

Description automatically generated

1. **Flume Configuration File (spooldir.conf)**

This configuration file sets up Apache Flume to monitor a directory for new CSV files and stream them to HDFS for storage.

* 1. *Agent Setup*

The configuration defines an agent (agent1) with one source (spoolSrc), one channel (memChannel), and one sink (hdfsSink).

A screenshot of a computer

Description automatically generated

*2.2. Source Configuration (Spool Directory Source)*

**Type**: The source is a spool directory, which reads new CSV files placed in the /home/student/flume/input folder.

**Check Interval**: Flume checks for new files every 60 seconds.

**File Handling**: Files are processed one at a time (batchSize = 1) and remain in the directory after processing, with the suffix .COMPLETED.

A screenshot of a computer

Description automatically generated

*2.3. Channel Configuration (Memory Channel)*

The memory channel has a capacity of 10,000 events and can process 1,000 events per transaction.

A screenshot of a computer

Description automatically generated

*2.4. Sink Configuration (HDFS Sink)*

The sink writes data to HDFS in text format under the /user/student/. Files are rolled every 300 seconds (rollInterval), and no size or record count-based roll is enforced (rollSize and rollCount are set to 0).

A screenshot of a computer

Description automatically generated

* Flume and python script run in parallel

A screen shot of a computer

Description automatically generated

* Show any random file in hdfs after flume transfer file from input dir to hdfsA screen shot of a computer

  Description automatically generated

1. **FAANG Stock News Streaming and Sentiment Analysis (Jupyter Notebook)**

This notebook is part of a pipeline to stream FAANG stock-related news, apply sentiment analysis, and store the results in MongoDB.

**3.1. Spark Session Initialization**

The notebook creates a Spark session for processing streaming data from HDFS.

A screenshot of a computer

Description automatically generated

**3.2. Sentiment Analysis Setup**

A pre-trained model from Hugging Face’s transformers library is initialized to analyze the sentiment of news articles.

A screenshot of a computer

Description automatically generated

**3.3. Filtering FAANG Stocks**

This function filters news articles related to FAANG stocks, ensuring only relevant news is processed.

A screen shot of a computer code

Description automatically generated

**3.4. Reading Streaming Data from HDFS**

Define the schema for incoming data and reads streaming CSV files from HDFS and filters them based on related FAANG stock tickers.

A white background with red and blue text

Description automatically generated

**3.5. Applying Sentiment Analysis**

This function applies sentiment analysis to each news article, labeling it as positive, negative, or neutral.

A close-up of a computer screen

Description automatically generated

**3.6. Apply filtering and transformation in data**

A close-up of a computer code

Description automatically generated

**3.7. Writing Results to MongoDB**

The analyzed data is saved to MongoDB in a database named stock\_news and a collection named faang\_news.

A screen shot of a computer code

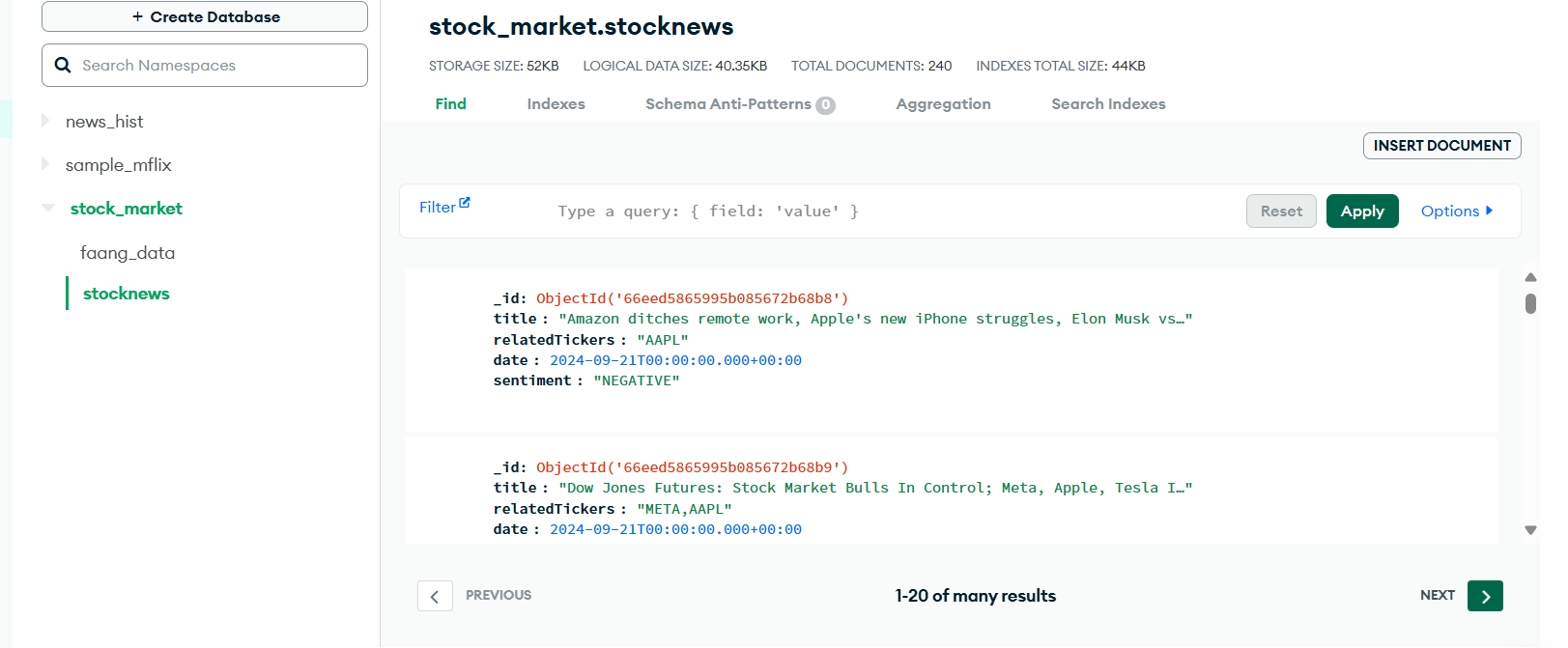
Description automatically generated

**3.8.Set up streaming query to write to MongoDB**

A screenshot of a computer

Description automatically generated

* Data in Mongodb



FAANG Stock Historical News Processing and Sentiment Analysis .

This document details the steps involved in processing historical FAANG stock news, performing sentiment analysis, and storing the results in MongoDB. The following sections explain each part of the codebase, focusing on downloading historical news data, processing it with PySpark, and visualizing the results. Visual aids are provided to support the explanation.

**1. Prerequisites**

The following libraries and technologies are required to run the pipeline:

- PySpark: For distributed data processing.

- Hugging Face Transformers: For sentiment analysis.

- MongoDB: For storing the results of the sentiment analysis.

- HDFS (Hadoop Distributed File System): To store large volumes of news data.

- Flume: To transfer data from the local directory to HDFS.

**2. Downloading Historical Data**

The first step in the pipeline involves downloading the historical FAANG news data from GDELT. The data is in .csv.zip format, so after downloading, the zip files are extracted.

**2.1. Imports**

Key libraries used in the downloading script:

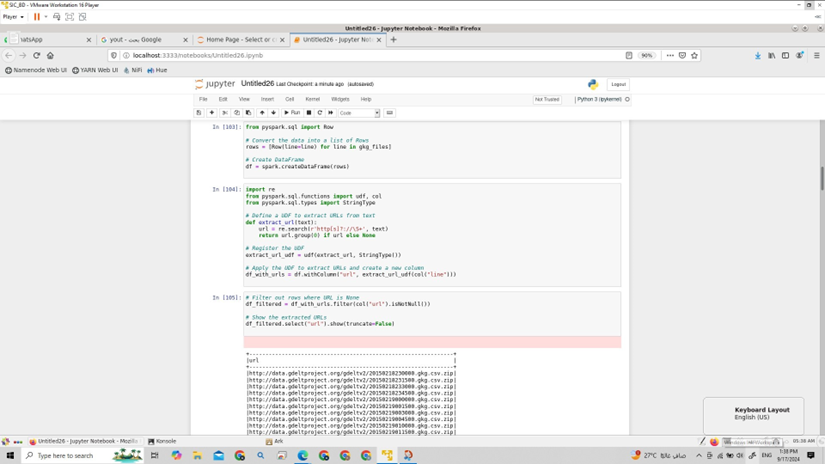
- requests: To download the zip files from GDELT.

- zipfile: To handle zip file extraction.

- os: For file handling and directory management.

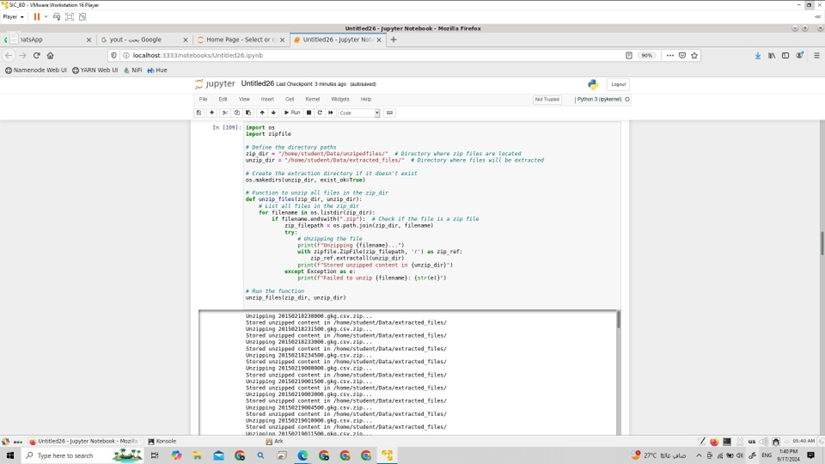
**2.2. Downloading URLs**

In the first step, we download URLs of GKG (Global Knowledge Graph) files from GDELT. The PySpark job filters the list to capture only .gkg.csv.zip files. The filtered URLs are then saved to a file for later downloading.



The image above shows how the script lists the URLs of the .gkg.csv.zip files from the GDELT project.

**2.3. Unzipping the Data**

Once the URLs are downloaded, the script proceeds to download the zip files and unzip their contents. Each file is saved into a designated directory for processing. The directory structure is created if it doesn't already exist.

The image above displays how the zip files are extracted and stored in the appropriate directory.

**3. Processing Historical FAANG News with PySpark**

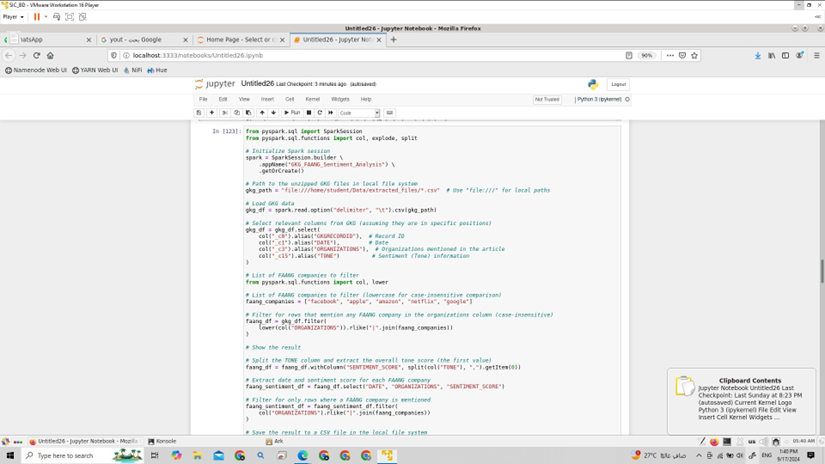
Once the data is downloaded and extracted, it is loaded into PySpark for processing. The processing includes filtering relevant FAANG news, extracting key fields, and applying sentiment analysis.

**3.1. Loading the Data**

The .csv files are loaded into PySpark DataFrames using the Spark SQL library. The relevant columns are selected from the GKG data, which include the GKG record ID, date, organizations mentioned in the article, and sentiment score.

**3.2. Filtering FAANG Companies**

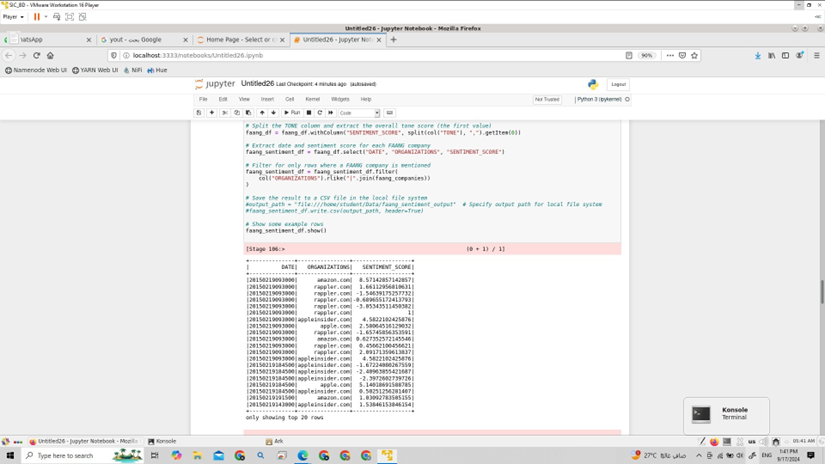
A predefined list of FAANG companies is used to filter only the news articles that mention Facebook, Apple, Amazon, Netflix, or Google. This ensures that only the relevant news articles are processed.



The image above shows how the data is filtered to include only FAANG companies.

**3.3. Extracting Sentiment Scores**

After filtering, the sentiment score is extracted from the 'Tone' column. The sentiment score is calculated for each article to determine whether it is positive, negative, or neutral.



The image above shows an example of the sentiment scores calculated for the FAANG companies.

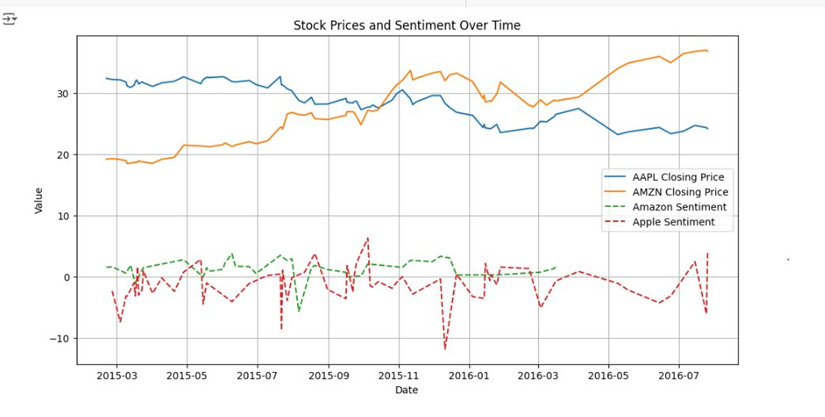
**4. Storing Results in MongoDB**

The sentiment analysis results are stored in MongoDB. A collection is created under the 'news\_historical' database to store the articles along with their sentiment scores. The data is inserted into MongoDB one record at a time, with a short delay to prevent overloading the database.

The image above shows the records being saved to MongoDB.

**5. Visualizing FAANG Stock Sentiment**

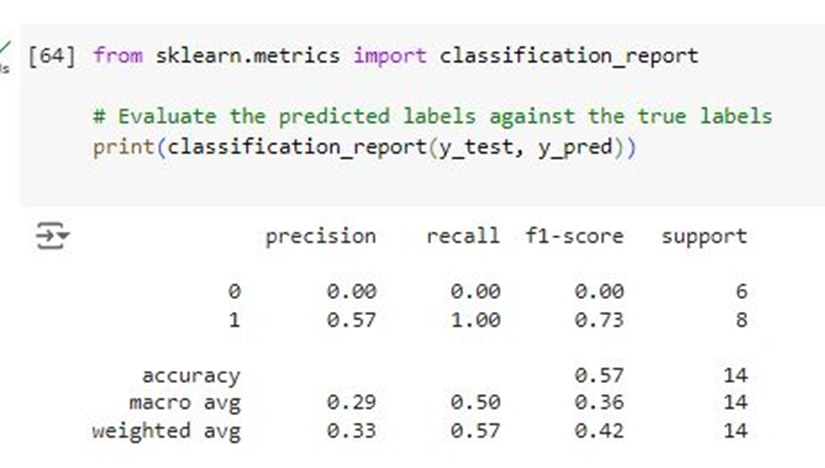
The final step in the pipeline is to visualize the sentiment trends over time. A Jupyter notebook is used to plot the sentiment score trends of FAANG companies over time. The results are displayed alongside stock prices to observe any correlations.



The image above shows how the stock prices and sentiment trends are plotted for further analysis.

**6. Evaluating Accuracy of Sentiment Model**

The sentiment analysis model's performance is evaluated using an LSTM (Long Short-Term Memory) model. The classification report generated shows the precision, recall, and F1-score for the model. This provides insights into how well the model is predicting sentiment.



The image above shows the classification report generated for the LSTM model.

## Visualizations and Charts

**1. Sentiment Landscape Across Key Market Players**

* **Description**: A horizontal bar chart representing the sentiment distribution (positive and negative) for key stock market players. The tickers represented include well-known companies such as Netflix (NFLX), Amazon (AMZN), Apple (AAPL), Meta (META), and Google (GOOG).
* **Data Source**: Extracted from articles related to stock tickers, where each article has a sentiment label (positive/negative) and mentions specific companies.
* **Purpose**: To visualize how each stock ticker is perceived in terms of sentiment, highlighting any strong positive or negative coverage. It gives a snapshot of market sentiment for these companies over a specific time range.
* **Key Insights**:
  + The green bars represent **positive** sentiment.
  + The blue bars represent **negative** sentiment.
  + This chart helps investors and analysts understand the overall market sentiment surrounding these companies.

A graph of a number of numbers

Description automatically generated with medium confidence

**2. Top Related News Articles by Tickers**

* **Description**: A table that lists stock tickers (e.g., AAPL, META) along with the titles of news articles related to these tickers. Additionally, the table provides a count of how often a sentiment is associated with the ticker.
* **Data Source**: Extracted from articles where each article mentions specific companies and is classified as having positive or negative sentiment.
* **Purpose**: To provide a quick view of the most popular news articles associated with stock tickers and their related sentiment. This table allows for a more detailed drill-down into the specific stories driving market sentiment.
* **Key Insights**:
  + Each row represents a ticker, its related news headline, and the sentiment count for that ticker.
  + This table allows users to quickly see which companies are receiving the most attention and the nature of the sentiment in the articles.

A screenshot of a web page

Description automatically generated

**3. Average Sentiment Distribution by Ticker**

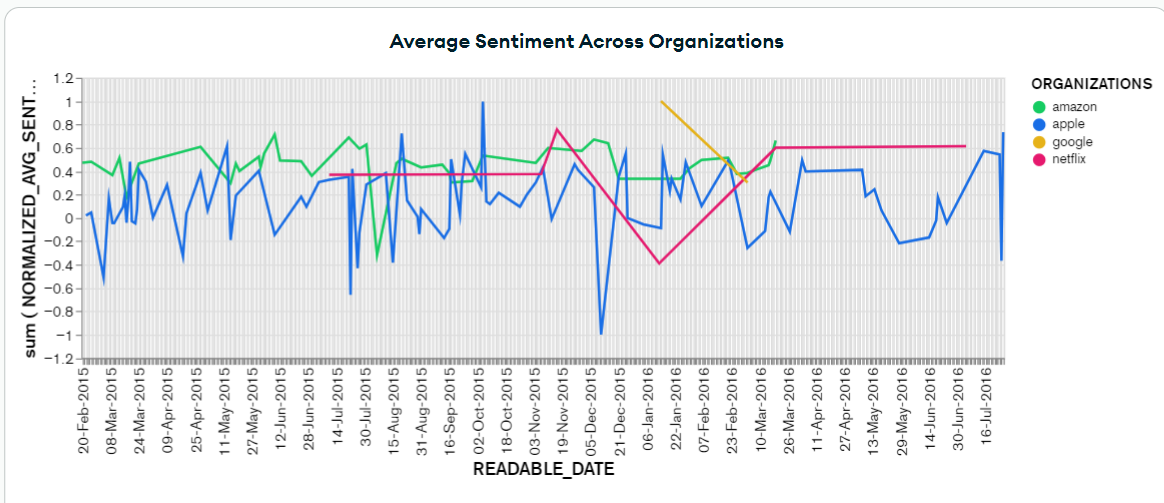
* **Description**: A bar chart or bar chart that shows the average sentiment distribution (positive, negative) across all articles mentioning a particular ticker. Each sector or bar is proportional to the sentiment associated with the ticker.
* **Data Source**: Similar to previous charts, this data is sourced from articles mentioning companies and their respective sentiments.
* **Purpose**: To provide a high-level view of sentiment trends across tickers, helping investors understand the overall tone of the news coverage each company receives.
* **Key Insights**:
  + Highlights the overall balance of positive and negative sentiment for each company.
  + Allows for quick visual comparison of how each company is perceived by the media.

A graph with green rectangles

Description automatically generated

**4. Sentiment Over Time**

* **Description**: A line chart that tracks sentiment trends (positive/negative) for stock tickers over time. This helps identify how sentiment shifts over days, weeks, or months.
* **Data Source**: Article sentiment data aggregated over time, possibly broken down by days or weeks.
* **Purpose**: To help analysts track shifts in public or market sentiment for specific companies, correlating this data with stock price movements or other market events.
* **Key Insights**:
  + Detects periods of increased negative or positive sentiment.
  + Highlights important events or shifts in perception that may affect stock prices.



**8. Candlestick Chart: FAANG Stock Price Movements**

* **Description**: A **Candlestick Chart** is used to represent the price movement of stocks over a specific time period. It visually displays the opening, closing, high, and low prices for each time interval. The candlesticks are color-coded to indicate whether the stock price increased (green) or decreased (red) during that interval.
* **Data Source**: This data comes from stock market pricing data for FAANG stocks (Facebook (Meta), Apple, Amazon, Netflix, Google) over a chosen time range. Data points include opening price, closing price, highest price, and lowest price for each time period (e.g., day, week).
* **Purpose**: The candlestick chart is widely used by traders and analysts to understand price fluctuations and trends in stock performance. It helps in identifying patterns such as bullish or bearish trends, reversals, and consolidations.
* **Key Insights**:
  + **Green Candlesticks**: Indicates that the stock closed higher than it opened (price increase).
  + **Red Candlesticks**: Indicates that the stock closed lower than it opened (price decrease).
  + **Wicks (Shadows)**: The thin lines extending from the top or bottom of the candlestick represent the highest and lowest prices during the period.
  + This chart helps traders spot patterns and predict future price movements based on historical data.

A graph of stock prices

Description automatically generated

**9. Trading Volume Over Time**

* **Description**: A **line chart** or **bar chart** that tracks the total trading volume for each FAANG stock over time. The volume indicates the number of shares traded for a particular stock during each time interval (e.g., day, week).
* **Data Source**: Collected from stock exchange data, the trading volume represents the total quantity of shares exchanged between buyers and sellers during the specified period.
* **Purpose**: This chart helps investors and analysts understand the liquidity and activity level of a particular stock. Higher trading volumes can indicate increased interest or activity, often triggered by major news events, earnings reports, or market sentiment shifts.
* **Key Insights**:
  + **Spikes in Volume**: Can indicate periods of high interest, often due to important news or earnings releases.
  + **Low Volume Periods**: Suggest less activity and interest in the stock, which could mean fewer traders are engaging with that stock during that period.
  + **Correlating Volume with Price**: By overlaying volume data with a price chart, analysts can understand whether the price movement is supported by strong trading activity (e.g., price increase with high volume suggests bullish sentiment).

A graph showing a number of different colored lines

Description automatically generated