FED Announcements

Brian F, Jonah N, Kyle H, Haley J

Overview

- FED communication
  - Statements
  - Press Conferences
  - Intermeeting transcripts
  - Economic projections
  - Speeches

- Effect on the market
  - Different indices instead of individual stocks

How do financial markets process monetary policy information from central bank communications?

Process

Scrape Announcement Files - FED Website has historicals

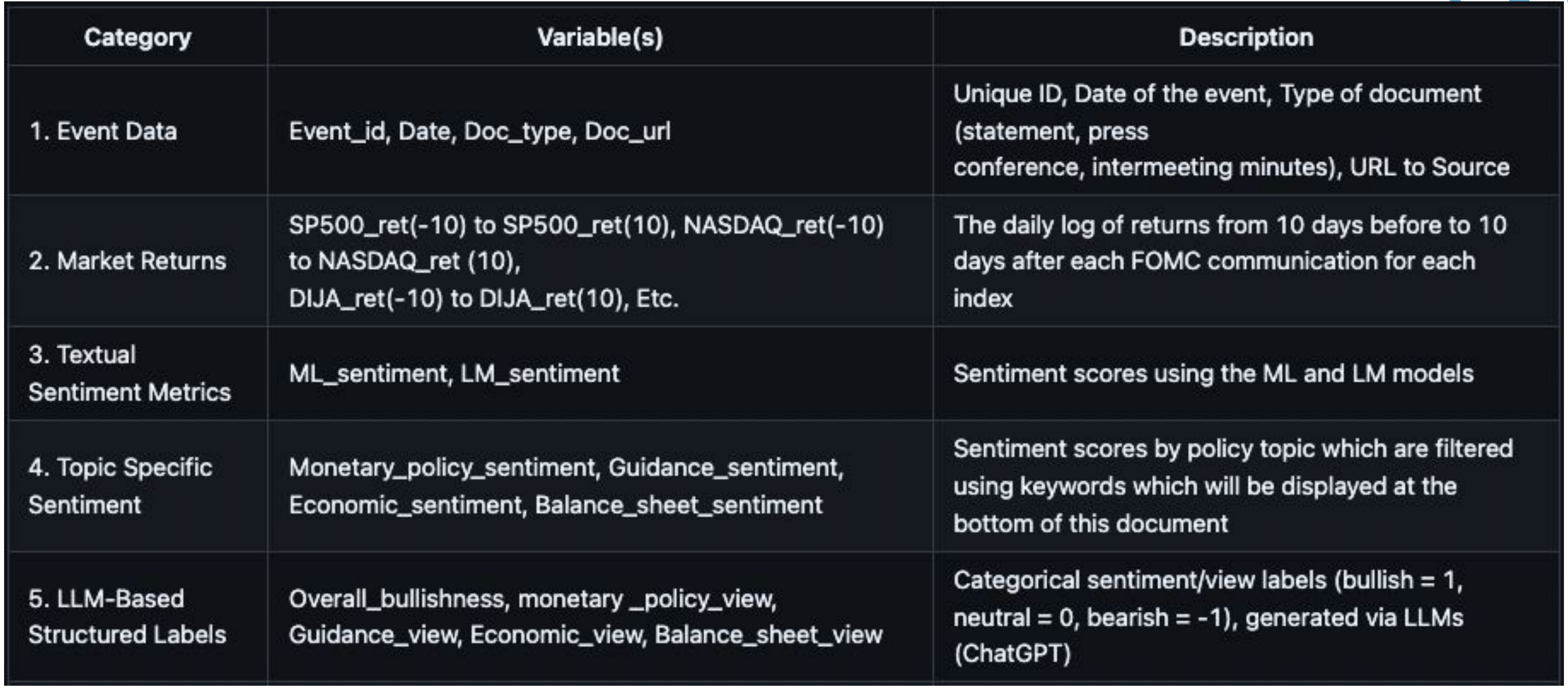
Get market returns - Pulled indices info from Yahoo Finance

Sentiment Analysis - LM and ML models from mid-term, and contextual sentiment word lists

LLM-based analysis - OpenAI API implementation and prompt building

Dashboard - Interactive Streamlit dashboard

Building the Dataset

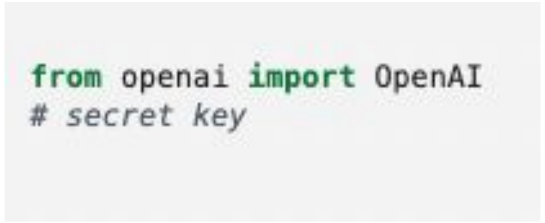


Category	Variable(s)	Description
1. Event Data	Event_id, Date, Doc_type, Doc_url	Unique ID, Date of the event, Type of document (statement, press conference, intermeeting minutes), URL to Source
2. Market Returns	SP500_ret(-10) to SP500_ret(10), NASDAQ_ret(-10) to NASDAQ_ret (10), DIJA_ret(-10) to DIJA_ret(10), Etc.	The daily log of returns from 10 days before to 10 days after each FOMC communication for each index
3. Textual Sentiment Metrics	ML_sentiment, LM_sentiment	Sentiment scores using the ML and LM models
4. Topic Specific Sentiment	Monetary_policy_sentiment, Guidance_sentiment, Economic_sentiment, Balance_sheet_sentiment	Sentiment scores by policy topic which are filtered using keywords which will be displayed at the bottom of this document
5. LLM-Based Structured Labels	Overall_bullishness, monetary _policy_view, Guidance_view, Economic_view, Balance_sheet_view	Categorical sentiment/view labels (bullish = 1, neutral = 0, bearish = -1), generated via LLMs (ChatGPT)

ChatGPT Analysis

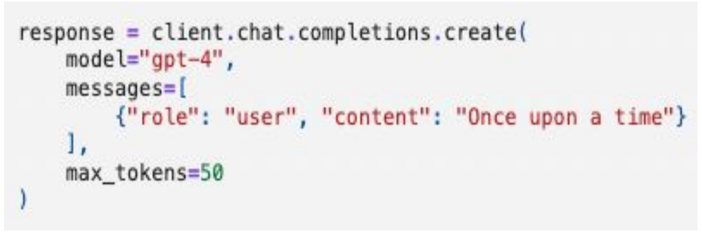
- Imported OpenAI api into jupyter labs
- Must pip install OpenAI
- Cost some money to use the model

Step 1 - Import OpenAI and load in your secret key



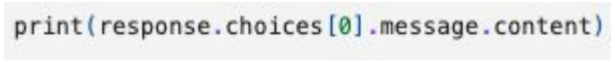
```python
from openai import OpenAI
# secret key
```

Step 2 - Create the prompt

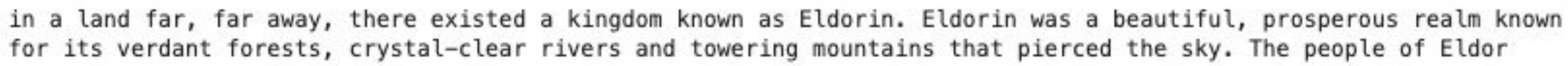


```python
response = client.chat.completions.create(
    model="gpt-4",
    messages=[
        {"role": "user", "content": "Once upon a time"}
    ],
    max_tokens=50
)
```

Step 3 - Get Output

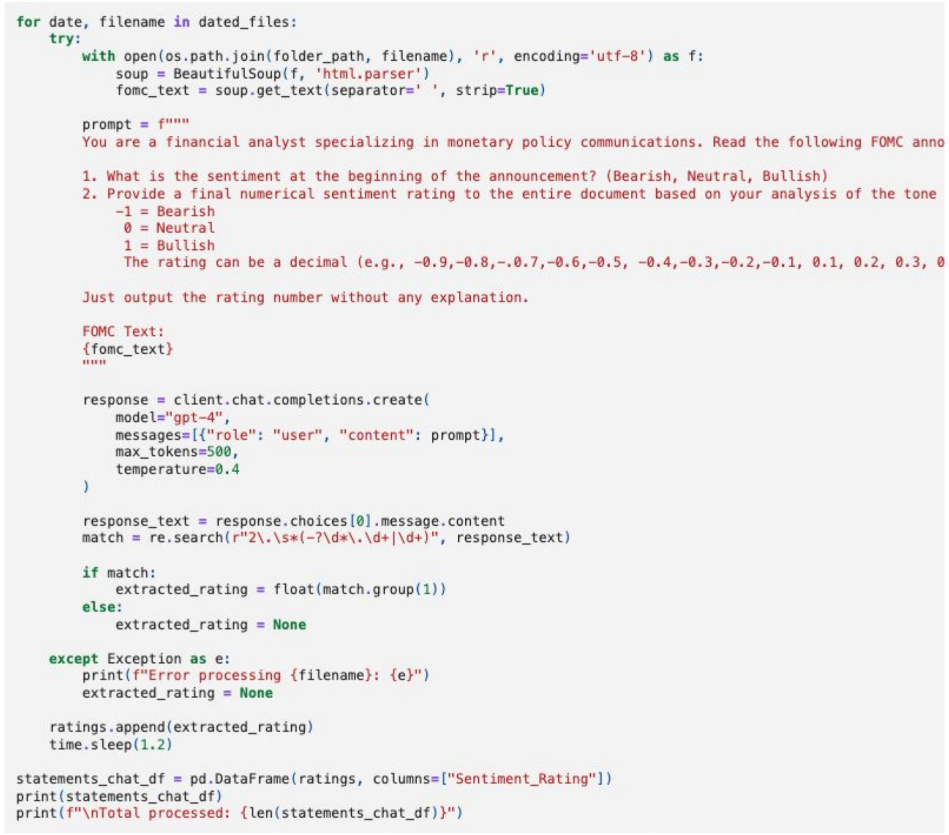


```python
print(response.choices[0].message.content)
```

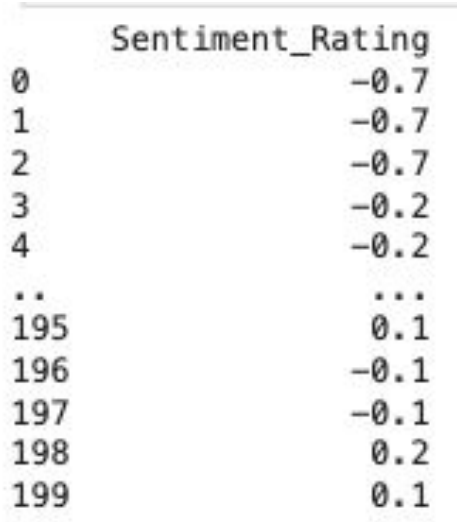


```
in a land far, far away, there existed a kingdom known as Eldorin. Eldorin was a beautiful, prosperous realm known
for its verdant forests, crystal-clear rivers and towering mountains that pierced the sky. The people of Eldor
```

ChatGPT Analysis Continued……



```
for date, filename in dated_files:
    try:
        with open(os.path.join(folder_path, filename), 'r', encoding='utf-8') as f:
            soup = BeautifulSoup(f, 'html.parser')
            fomc_text = soup.get_text(separator=' ', strip=True)

        prompt = f"""
        You are a financial analyst specializing in monetary policy communications. Read the following FOMC anno

        1. What is the sentiment at the beginning of the announcement? (Bearish, Neutral, Bullish)
        2. Provide a final numerical sentiment rating to the entire document based on your analysis of the tone
            -1 = Bearish
             0 = Neutral
             1 = Bullish
             The rating can be a decimal (e.g., -0.9,-0.8,-.0.7,-0.6,-0.5, -0.4,-0.3,-0.2,-0.1, 0.1, 0.2, 0.3, 0

        Just output the rating number without any explanation.

        FOMC Text:
        {fomc_text}
        """

        response = client.chat.completions.create(
            model="gpt-4",
            messages=[{"role": "user", "content": prompt}],
            max_tokens=500,
            temperature=0.4
        )

        response_text = response.choices[0].message.content
        match = re.search(r"2\.\s*(-?\d*\.\d+|\d+)", response_text)

        if match:
            extracted_rating = float(match.group(1))
        else:
            extracted_rating = None

    except Exception as e:
        print(f"Error processing {filename}: {e}")
        extracted_rating = None

    ratings.append(extracted_rating)
    time.sleep(1.2)

statements_chat_df = pd.DataFrame(ratings, columns=["Sentiment_Rating"])
print(statements_chat_df)
print(f"\nTotal processed: {len(statements_chat_df)}")
```



```
     Sentiment_Rating
0                -0.7
1                -0.7
2                -0.7
3                -0.2
4                -0.2
..                ...
195               0.1
196              -0.1
197              -0.1
198               0.2
199               0.1
```

Let’s check out the dashboard………

https://fed-announcements.streamlit.app/