Bank of England Employer Project
Team: BEACON 02 June 2025

Final Report

Predictive Analysis of the Impact of Bank of England Speeches on the Economy

Chris Buck, Germán de la Fuente, Kelvin Dhondee, Kate Jacobs, Harshdeep Kohli, Victoria Pogonyaylova

Table of Contents

Background and context

The Bank of England (BoE) plays a vital role in maintaining financial stability, with communication being a key tool (Pastorella, 2023). BoE speeches aim to provide transparency and reassurance, particularly during periods of uncertainty.

The objective of this project is to deliver evidence-based insights into BoE communication effectiveness, specifically regarding the relationship between speech sentiment and economic indicators.

Findings can help improve BoE communication strategies to enhance clarity, trust, and policy transmission.

Problem Statement: To what extent does the sentiment of central bank communications help predict financial market movements, and how can sentiment be incorporated into predictive models?

See additional questions and problem-solving framework in Appendix 1.

Project process overview

Project stages:

1. **Setup and planning**, including definition of scope and research on relevant academic work and core economic indicators (see Appendix 2)
2. **Data acquisition**, including additional BoE speeches and economic indicators
3. **Data cleaning** and EDA
4. **Sentiment model assessment**/selection
5. **Predictive modelling**, including linear regression, gradient boosting, and SARIMAX
6. **Synthesis of findings** into final presentation

Python was our core analysis tool for the following reasons:

- All-in-one tool with extensive libraries (e.g. for web scraping, cleaning, modelling, visualisation)
- NLP/sentiment-analysis abilities (core to project objectives)
- Advanced analytical and ML capabilities
- Team familiarity
- Reproducibility

Process and technical approach

Data scraping (BoE recent speeches)

To get the most recent speeches data, we performed web scraping on www.bankofengland.co.uk website. This allowed us to obtain the speech date, title, author, and links to each speech's content (see Appendix 3).

165 additional speeches were compiled this way, extending the timeline up to 25/03/2025.

Data cleaning - master dataset

The process involved (see details in Appendix 4):

- correcting governor attributions by aligning author names with official BoE terms
- resolving ambiguities between individuals with similar names
- standardising location references
- removing duplicates and misattributed speeches
- producing a final verified dataset of 1365 UK speeches (1998-2025) suitable for sentiment analysis

A new column of speech identifier 'custom_ref' was introduced for optimisation of processing time and simplification of custom datasets' engineering.

Data collection and cleaning - economic indicators

The process involved (see details in Appendix 5):

- identifying relevant indicators and retrieving datasets from reputable sources
- renaming columns for uniqueness and consistency to facilitate merging
- splitting datasets with annual, quarterly, and monthly values in one column into three separate dataframes
- transforming quarterly dataframes into monthly ones to facilitate merging
- changing the date column to period data type to match indicator frequency

- creating a forward-filled daily dataframe for bank rates to add these values to every speech row
- merging indicators with speech sentiment in preparation for predictive analyses

Sentiment model selection

We assessed several sentiment models (see *sentiment_testing_results.ipynb*):

- **General purpose:** TextBlob, VADER
- **Financial lexicon:** Adapted Loughran-McDonald (LM) Dictionary provided by BoE
- **Financial pre-trained ML models:** FinBERT, CentralBankRoBERTa (Pfeifer and Marohl 2023)
- **Generative LLMs:** ChatGPT, Claude

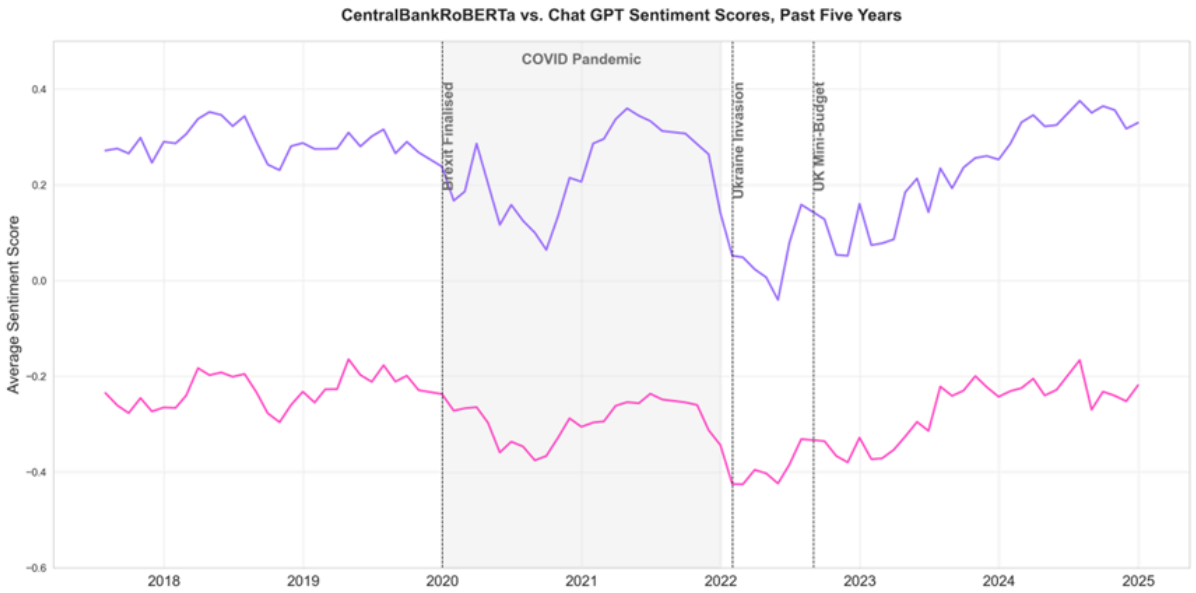
See Appendix 6 and *sentiment_model_analyis.ipynb* for detail on how models were processed.

A testing process established a benchmark and gauged model suitability:

- **Test sample:** we manually labelled 59 random speeches as positive/negative/neutral
- **Testing:** we tested sentiment models against manual-label classification.
- **Assessment:** we calculated accuracy, precision, recall, and F1 scores using scikit-learn modules
- **Results:** CentralBankRoBERTa (CBRoBERTa) and ChatGPT showed strongest performance

Model selection:

- **We further compared ChatGPT and CentralBankRoBERTa**, including examining patterns over time and correlations with economic indicators
- While ChatGPT retained a slight edge on correlations, the two models showed comparable performance



- **We chose CentralBankRoBERTa as our primary model** due to advantages including:
  - Second dimension: agent-specific sentiment
  - Domain specificity (trained on central-bank communications)
  - Free, stable, reproducible (vs. generative LLMs)

See a technical overview of CentralBankRoBERTa in Appendix 7.

Pros and Cons of Each Sentiment Model

CentralBankRoBERTa offers the best trade-off of cost, performance and specialisation

	Financial lexicon model	ML transformer-based finance models		Generative LLM
	LM Dictionary	**FinBERT**	**CentralBankRoBERTa**	**ChatGPT**
Pros	• Simple mechanics • Lightweight to run • Free	• Financial-trained ML model • Fast(er) to run (vs. CBRoBERTa) • Free	• Central-bank specific • 2nd dimension (agent decomposition) • Demonstrated predictive performance • Free	• Up-to-date, intelligent LLM • Slightly stronger correlations than CBRoBERTa with several economic indicators
Cons	• Not context aware (bag-of-words model) • Not central-bank specific	• Not central-bank specific • Lower predictive performance (vs. CBRoBERTa)	• Slow to run	• Costly • Less stable/ reproduceable than pre-trained models

5

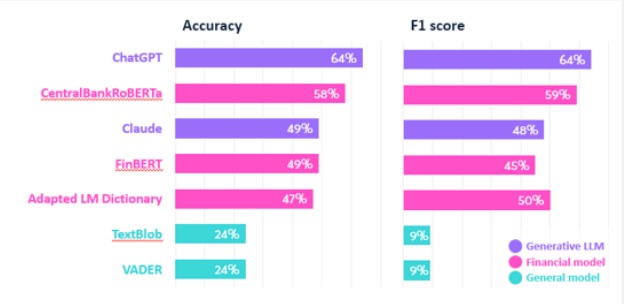
Visualisations demonstrated the testing process and CentralBankRoBERTa patterns. We created plots in Python and PowerPoint using the BoE colour palette for brand adherence.

Python version

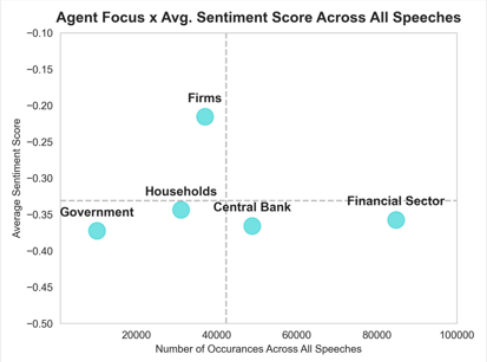
Sentiment Model Comparison:

	Accuracy	Precision	Recall	F1-Score
tb_label	23.73%	5.63%	23.73%	9.10%
vader_label	23.73%	6.27%	23.73%	9.92%
lm_label	47.46%	52.00%	47.46%	49.53%
finbert_sentiment	49.15%	46.84%	49.15%	45.48%
gpt_sentiment	64.41%	63.24%	64.41%	63.54%
claude_sentiment_clean	49.15%	49.03%	49.15%	47.96%
cbroberta_label	57.63%	61.12%	57.63%	58.98%

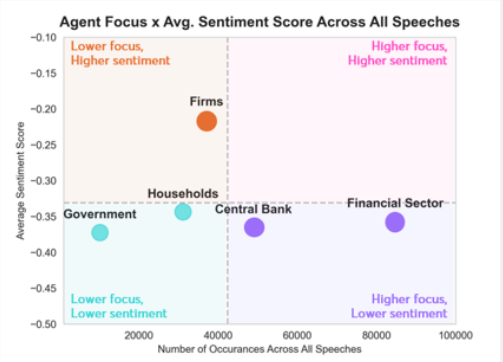
Presentation version – bar chart used for faster visual recognition of score patterns; BoE brand colours convey different model types



Python version:



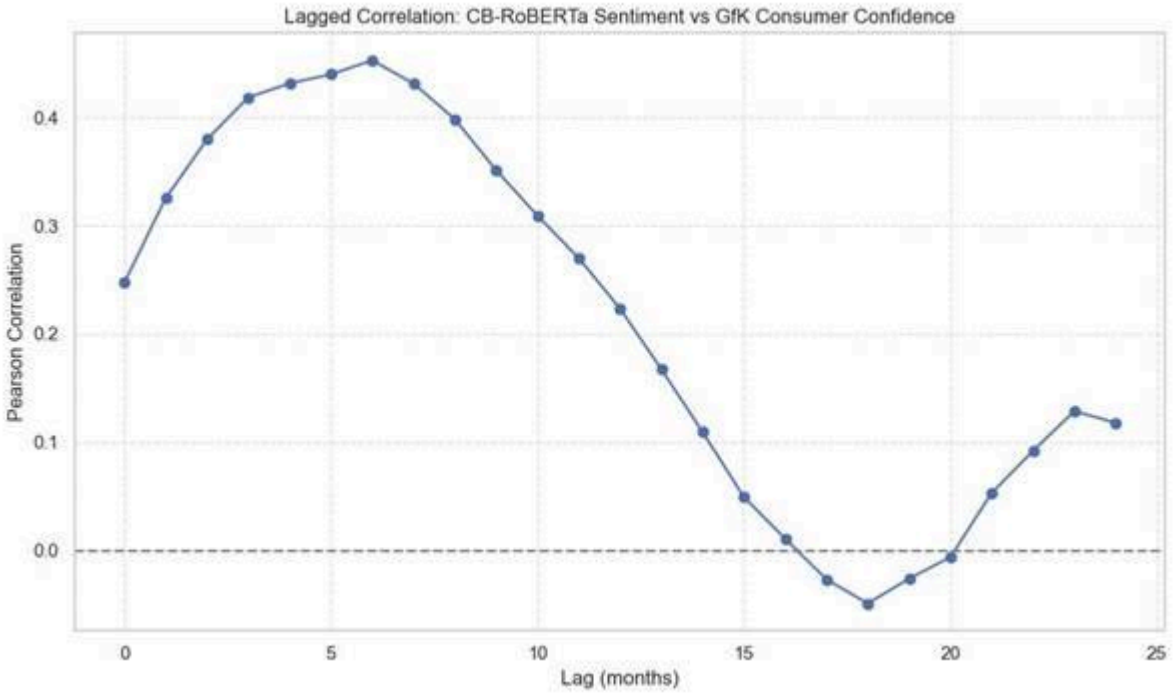
Presentation version – annotated with meaningful quadrants using BoE colours



Linear regression (GfK MLR model)

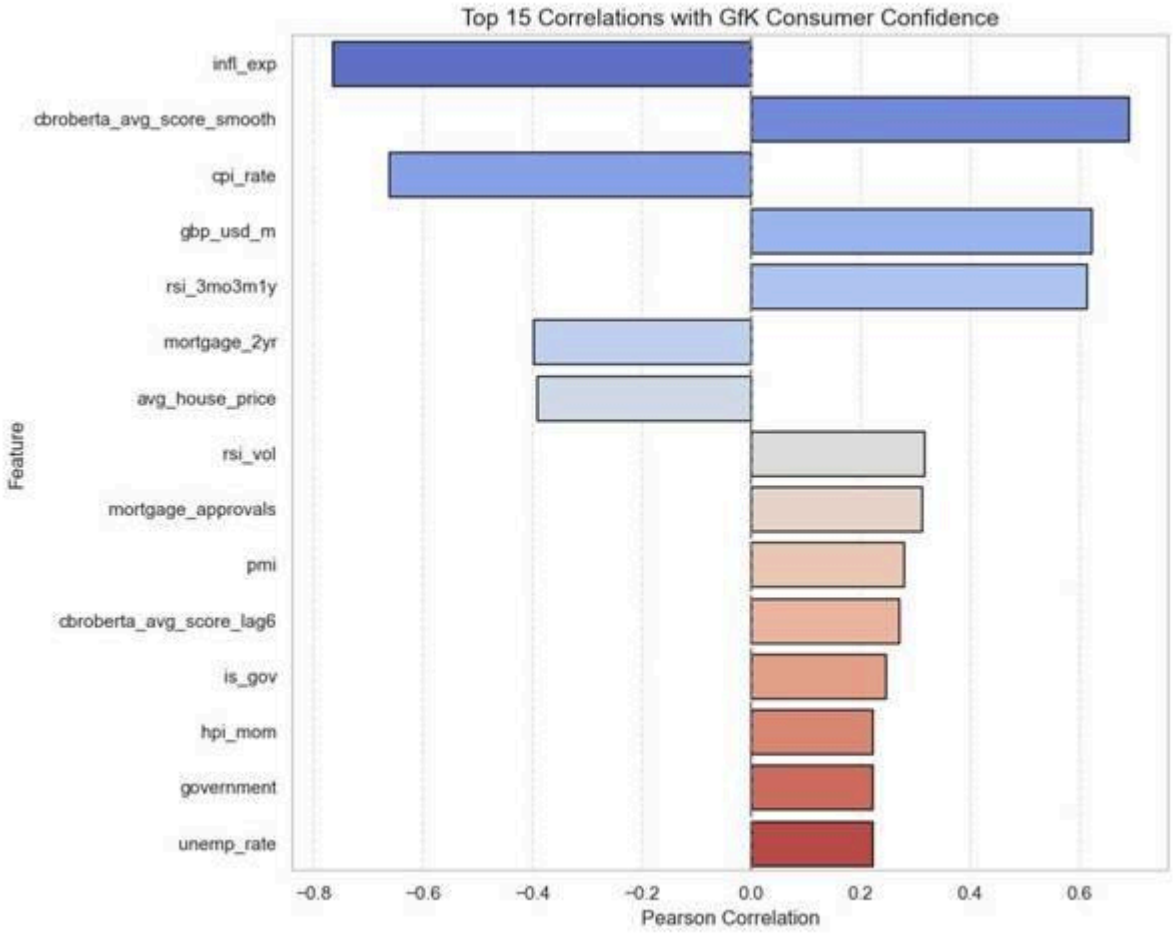
To establish if speech sentiment can influence economic indicators, we focused on GfK Consumer Confidence (see *GF_GfK modelling_Volat analysis_v3.ipynb*).

We plotted the correlation between gfk_cc (from our indicators research) and the different lagged values of cbroberta_avg_score, used here as our main sentiment metric.



This establishes that the highest correlation occurs when lagging sentiment by 6 months.

Next, we analysed the correlations between gfk_cc, cbroberta_avg_score (smoothed to reduce noise), and several economic indicators:

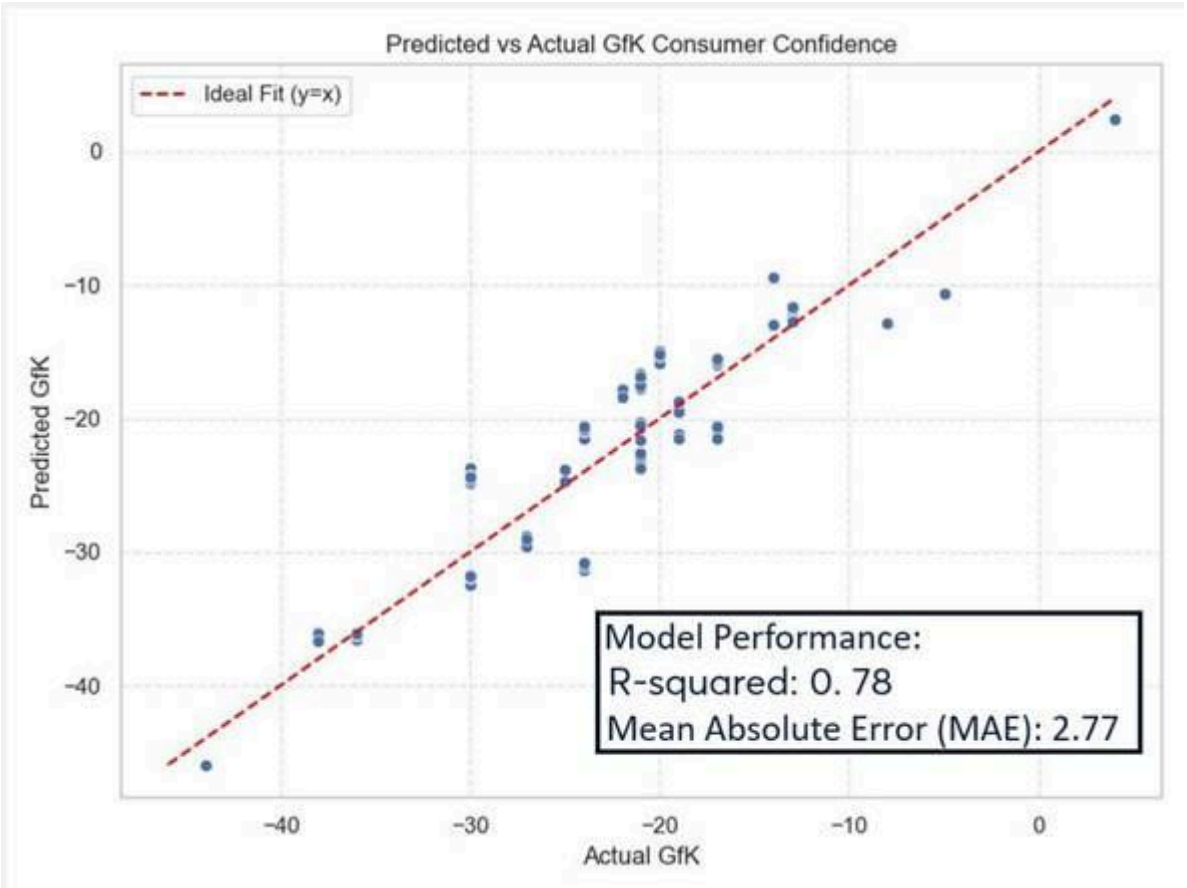


This enabled us to build a Multi Linear Regression model to predict gfk_cc using these features:

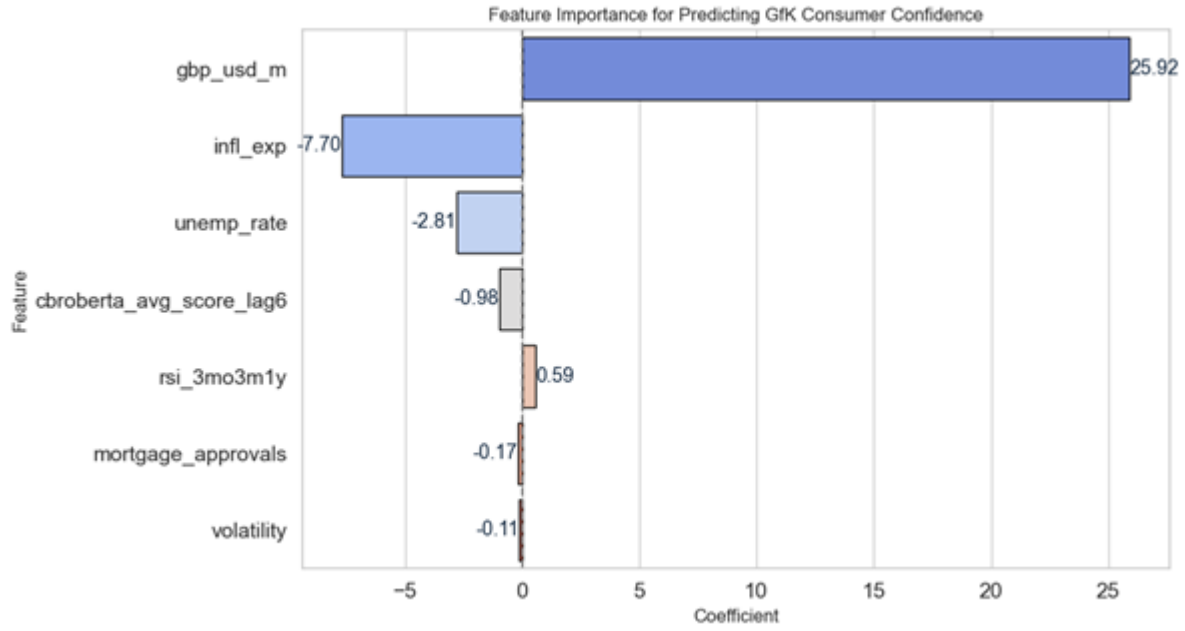
[18]:
```
# Define feature set
feature_candidates = [
    'infl_exp',
    'cpi_rate',
    'unemp_rate',
    'rsi_3mo3m1y',
    'gbp_usd_m',
    'mortgage_approvals',
    'cbroberta_avg_score_lag6',
    'volatility'
]
```

We ran a VIF test to avoid multicollinearity between features, such as CPI and inflation expectations. We decided to keep infl_exp as our predictor for inflation because of its higher correlation score.

The model explained 78% (R2) of the variance in GfK Consumer Confidence, with an average prediction error (MAE) of 2.77 points.



cbroberta_avg_score_lag6 **proved to add value to the model** as a booster for interpretability, with a coefficient of -0.98.



See Appendix 8 for a comparison of the model's performance in relation to different volatility scenarios.

Gradient boosting regression

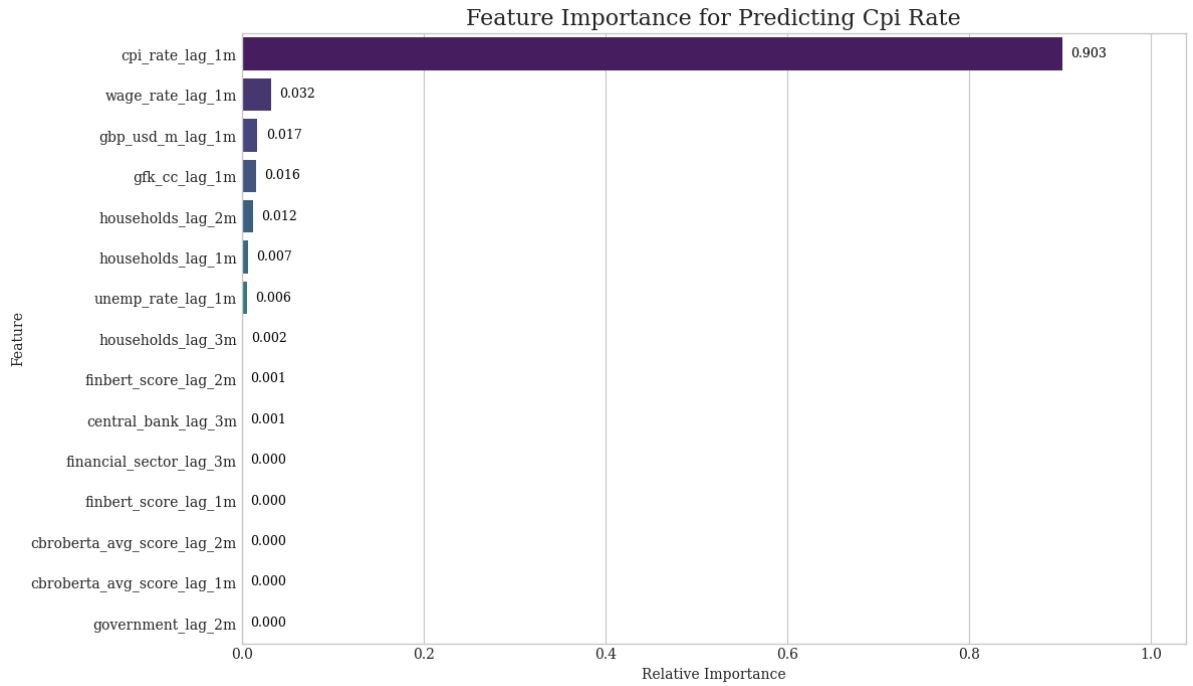
*(*See *Gradient Boosting - Monthly.ipynb* for details)

Our primary objective was to model the relationship between speech sentiment and CPI rates to predict future inflation.

We aggregated sentiment scores to monthly averages to mitigate noise from individual speeches. We created 1, 2, and 3-month lags for each sentiment indicator, which allowed us to check if sentiment from previous months influenced current economic conditions. We chose a 1-month lag for inflation because the most recent past value of a time series is usually its strongest predictor.

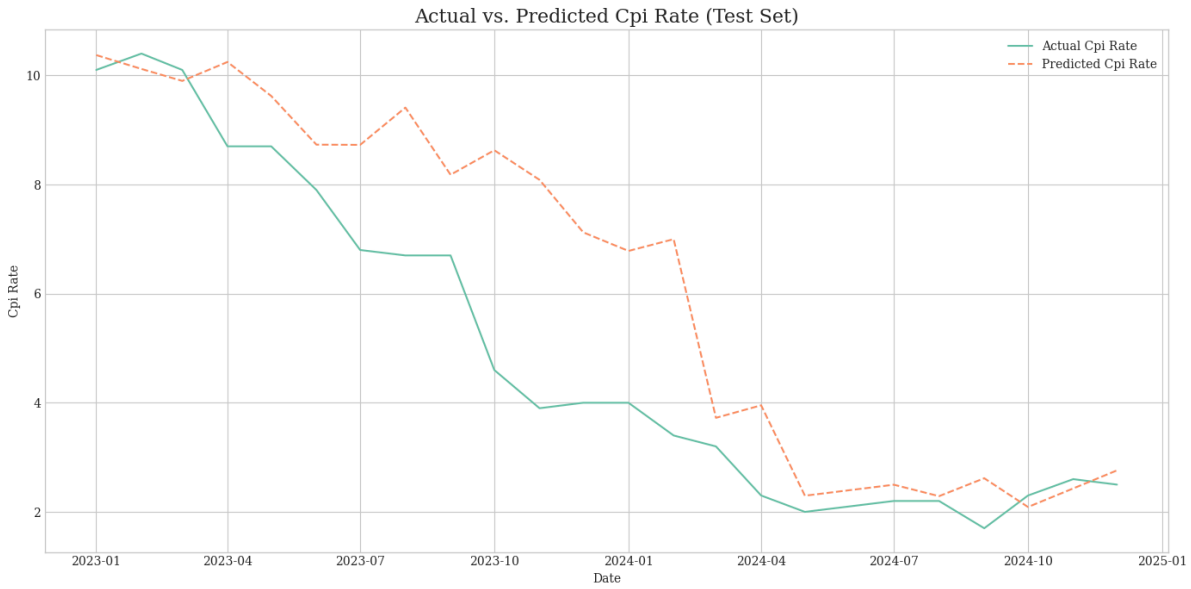
We selected Gradient Boosting Regressor from scikit-learn as the core analysis tool due to its ability to handle complex non-linear relationships. Standard Scaler was applied to features for consistency. We set 'shuffle=False' during the train/test split to preserve the chronological order of the data.

We visualised feature importance bar graphs to highlight which features are the most influential in making predictions. This allowed us to directly quantify which sentiment features contribute most to predicting inflation. The agent-specific lagged CBRoBERTa score ‘households’, with a 2-month delay (Importance: 0.012) and a 1-month delay (Importance: 0.007), turned out to be the most influential. These insights show that the Bank's tone on households significantly contributes to predicting future inflation. As anticipated, the previous month's value of CPI was its strongest predictor. This underscores the inherent persistence and auto-correlation in inflation time series.



Our conclusions on correlations are rooted directly in the quantitative evidence provided by the feature importance scores. A higher importance score for a sentiment feature indicates that the model found this aspect of sentiment to be highly effective in reducing prediction error for the target variable.

The model shows that past speech sentiment can help explain and predict the direction of future CPI rates. Our model predicts CPI rates well, explaining 57% of its variation. A mean squared error of 3.73, while low, shows further model refinement is needed to make better predictions. Options could include experimenting with different sentiment lags and hyperparameter tuning.



For further details, see Appendix 9.

Seasonal AutoRegressive Integrated Moving Average with Exogenous Variable (SARIMAX)

SARIMAX - Model Configuration

We employed SARIMAX to demonstrate how BoE speech sentiment may be utilised for predictive modelling. Exogenous inputs were CPI lags and CBRoBERTa sentiment scores for speeches given between 2015 and 2024. Up to six CPI lags were selected via *auto_arima,* with the selection procedure carefully designed to ensure that only non-overlapping lags were included as exogenous predictors.

The optimal SARIMAX configuration was determined to be SARIMAX(1, 2, 2)(2, 0, 0, 12).

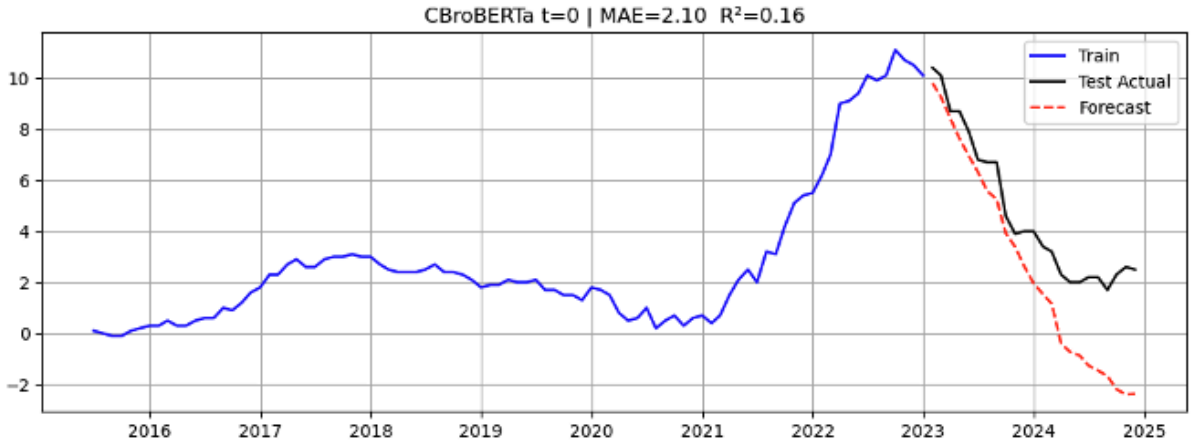
SARIMAX(1,2,2)(2,0,0,12)		
Component	**Meaning**	**Value**
p	Number of autoregressive (AR) terms	1
d	Number of differences to make data stationary	2
q	Number of moving average (MA) terms	2
P	Seasonal AR terms	2
D	Seasonal differencing	0
Q	Seasonal MA terms	0
s	Seasonal period: 12 for monthly data with yearly seasonality	12

An 80/20 train-test split was applied, with the test set covering 2023-2024.

SARIMAX – Key insight

- Using CBRoBERTa economic agent sentiment *(Agents = firms, households, central bank)* scores strongly improved forecasting accuracy compared to the model using CBRoBERTa average sentiment only (R^2: 0.65 v 0.16, MAE: 1.46 v 2.10). Although MAE of 1.46 may still be too high given the goal of predicting CPI rate.

- Contemporaneous *central_bank* CBRoBERTa sentiment capturing the BoE's outlook, tone and confidence is a powerful coincident indicator. Its positive coefficient was statistically significant (Coeff: 0.385, p-value: 0.0237).



SARIMAX - Technical Recommendations

- Explore non-linear models (Random Forest, XGBoost, etc.): to gauge whether non-linear interactions or high-dimensional feature sets can improve forecast accuracy, even if SARIMAX is ultimately chosen for ease of explanation.

- Cross-Validation on existing model Agents t=0: may yield better results and help determine which model setup (lags, seasonal terms, exogenous inputs) truly gives the smallest errors when predicting using unseen data. Reduces risk of overfitting.

- Continuously recalibrate lags and feature sets: macroeconomic data often experience structural breaks (e.g., recessions, policy shifts). To maintain robustness, periodically revisit choice of lag order and candidate predictors.

- Consider TVP-VAR: as (Primiceri, 2005) has shown, a Time-Varying Parameter Vector Autoregression (TVP-VAR) allows coefficient and effectively the lag-structure to drift over time according to a stochastic process. A TVP-VAR may adapt to structural breaks more gracefully than a fixed-parameter SARIMAX.

- Explore Generalized Autoregressive Conditional Heteroskedasticity (GARCH) Modelling: helps with forecasting not just point estimates but also uncertainty (e.g., conditional variance of an inflation rate, interest-rate volatility).

For further details, see Appendix 10.

Patterns, trends, and insights

Key findings

- **Communications matter:** Speech sentiment exerts influence, even without policy changes. Our findings support the Bank's use of forward guidance and show that integrating speech sentiment into forecasting models adds predictive value.
- **Sentiment has limited predictive power:** Sentiment scores (especially from CBRoBERTa) are important indicators that *supplement* traditional economic forecasting tools. Nevertheless, economic indicators have more weight on the overall performance of the models.
- **Lagging variables makes a difference:** Depending on the variables and the predictive model, the impact of speech sentiment features might be instantaneous or may take several months to develop.

Key recommendation

The Bank of England should incorporate agent-specific speech sentiment into a variety of predictive models, including MLR, Gradient Boosting, and SARIMAX. It should also experiment with using contemporaneous and lagged features in these predictive models. The predictive power of agent-specific speech sentiment can benefit the Bank's communication strategy by indicating which audiences it should address in its efforts to influence particular economic indicators and fulfil its mandate of promoting price stability. Effective communication directed towards specific audiences enables economic actors to navigate uncertainty with greater confidence, supporting more efficient and proactive economic planning.

Appendices

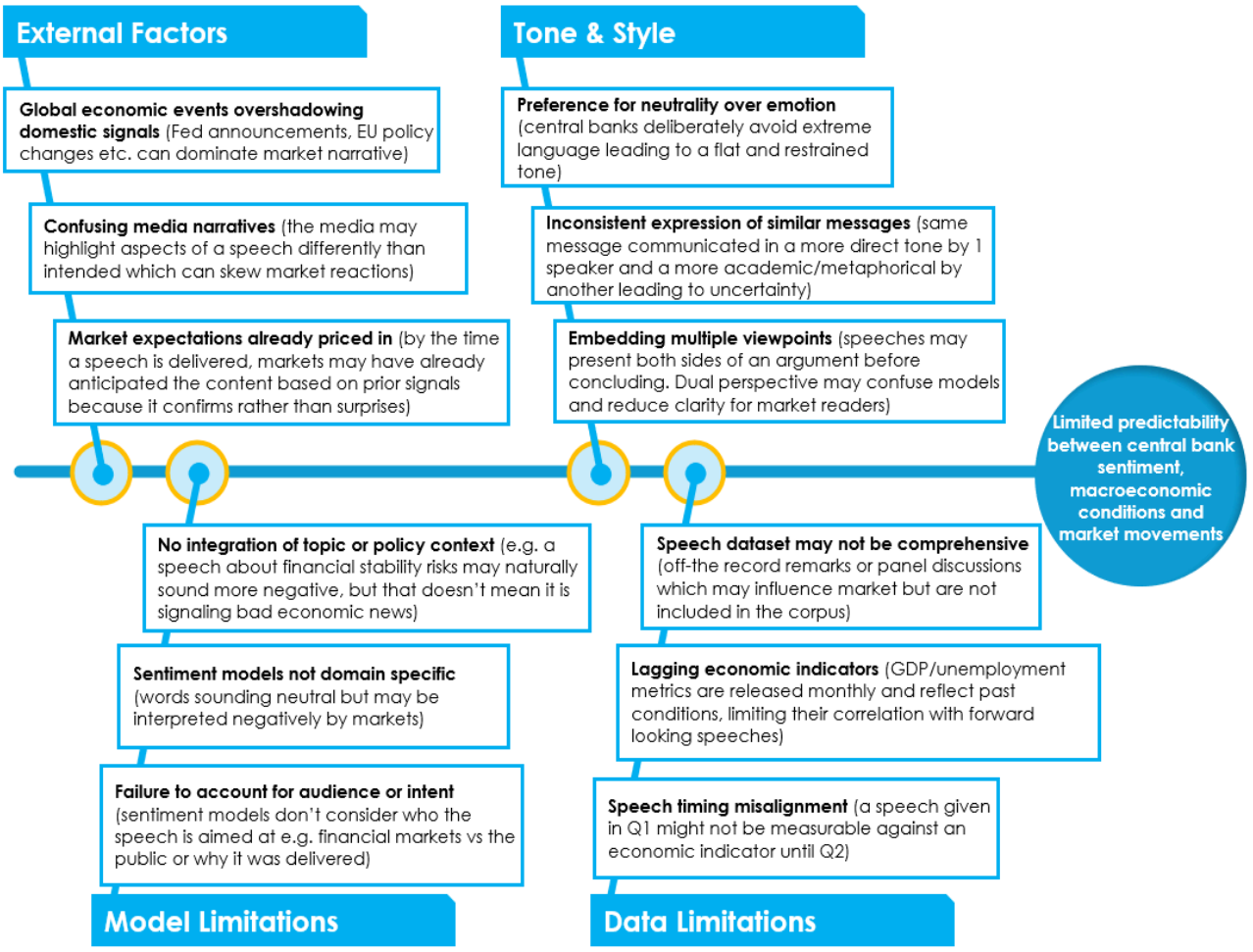
Appendix 1: Business questions and problem-solving framework

Business questions:

Main: Can speech sentiment help predict economic indicators?

Supporting:

- Which sentiment-analysis model is most effective?
- How can sentiment best be incorporated into predictive models?
- What are some next steps for BoE?



Appendix 2: Indicators

Core Indicators

- Inflation (CPI, RPI)
- Unemployment rates
- GDP growth
- Interest rates (e.g. BoE Bank Rate)

Secondary Indicators

Include only if they contribute to relevant insights:

- Business Confidence Index (BCI)
- Consumer Confidence Index (CCI)
- S&P Global UK Consumer Sentiment Index
- Gilt yields
- FTSE 100 performance (quick reactions - 1 day and 1 week post speech)
- FTSE 100 volatility (VFTSE)
- Exchange rate (e.g. USD/GBP)
- PMI

Leading Influence Indicators

We will assess the reaction of the following indicators to BoE sentiment:

- Yield curve shifts
- GBP exchange rates
- Equity market performance (e.g. FTSE indices)

Lagging Response Indicators

We could analyse the sentiment's reaction to the following data:

- Inflation (CPI, RPI)
- Unemployment rates
- GDP growth
- Interest rates
- Geopolitical or financial shocks

Note: At the suggestion of BoE we focused exclusively on leading influence models and did not pursue lagging response models.

Appendix 3: Web scraping additional speeches

To get the most recent speeches data, we performed web scraping techniques on the www.bankofengland.co.uk website. This will allow us to obtain the speech date, title, author, and links to each speech's content.

See *Boe latest speeches scraping_v2 -Copy1.ipynb* to follow this process

We used libraries such as:

- Requests: used to retrieve the HTML content
- BeautifulSoup: Extracts specific elements from the web pages (titles, dates, links, and full speech texts) by targeting HTML tags and classes.
- Pandas: Handling data structures.
- re (Regular Expressions): Clean and refine text by removing unwanted characters or whitespace patterns
- time: Introduce delays (time.sleep()) between HTTP requests to avoid overloading the BoE server.

The first step is to establish the connection to the url and send the request. We used common headers and verification tokens to ensure access.

```
[2]: # Endpoint and headers
url = "https://www.bankofengland.co.uk/_api/News/RefreshPagedNewsList"

headers = {
    "Accept": "*/*",
    "Accept-Language": "en-US,en;q=0.6",
    "Content-Type": "application/x-www-form-urlencoded; charset=UTF-8",
    "Origin": "https://www.bankofengland.co.uk",
    "Referer": "https://www.bankofengland.co.uk/news/speeches",
    "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/135.0.0.0 Safari/537.36",
    "X-Requested-With": "XMLHttpRequest"
}

# Cookie with session token (replace if expired)
cookies = {
    "shell#lang": "en",
    "__RequestVerificationToken": "F0TRyiMm0Nwv7WY9BlRedjFyTyzk5BExaj_WC8N-TXLOB75rrftgDCk55SpI9VN0uoHCkj0FqJk3ZD36jWZnPiilGoE1"
}
```

Next, we used prettify() allow us to see the entire structure of the HTML:

```
soup = BeautifulSoup(response.text, 'html.parser')
# Let's see how this Looks
print(soup.prettify())
```

- Each individual speech item was contains within a "div" element with a specific class, which groups the title, link, date, and sometimes the speaker's name together.
- Speech title and the hyperlink to the full speech page were found inside tags
- Publication dates are in a separate "div" or "time" element
- Speakers/authors are inconsistently available: Not all speeches include a named speaker in the listing; some require fetching from the detailed speech page, while others did not list the speaker at all.

Next, we used a while loop to handle pagination. On each iteration:

- The script fetches the current page.
- Parses the HTML content with BeautifulSoup.
- Locates each speech entry on the page and extracts key elements (title, date, speaker, URL, and summary).
- Moves to the next page using the "next page" link, if available.

```
import random

user_agents = [
    "Mozilla/5.0 (Windows NT 10.0; Win64; x64)...",
    "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)...",
    "Mozilla/5.0 (X11; Linux x86_64)...",
]


# Function to extract the full text of a speech from a given BoE speech page
def extract_speech_text(url):
    headers = {"User-Agent": random.choice(user_agents)}

    try:
        response = requests.get(url, headers=headers, timeout=10)
        if response.status_code != 200:
            return f"HTTP {response.status_code}"

        soup = BeautifulSoup(response.text, "html.parser")
        main_section = soup.select_one("div#output > section.page-section")
        if not main_section:
            return "Speech section not found"

        paragraphs = main_section.find_all(['p', 'h2', 'h3', 'h4', 'ul', 'ol'])
        return "\n\n".join(p.get_text(separator=" ", strip=True) for p in paragraphs)
    except Exception as e:
        return f"Error: {str(e)}"

# Apply to all rows with a small delay
df['speech_text'] = df['link'].apply(lambda url: extract_speech_text(url))

sleep(random.uniform(2.5, 20))  # Wait between 2.5 to 20 seconds  # optional: delay between requests if scraping more than a few

# Save to new CSV
#df.to_csv("boe_speeches_speech_text.csv", index=False)
#print("Saved", len(df), "speeches to boe_speeches_speech_text.csv")
```

The loop continues until no further pages are found, collecting data page by page.

Once all data is collected, the script consolidates the lists into a pandas DataFrame.

Expanding the data with actual text from each URL

The next section loops through the collected URLs and sends requests to each individual speech page. It then:

- Parses the HTML
- Extracts all <p> elements within the main content area (the body of the speech)
- Concatenates them into a single text block.

This is appended to a new column in the DataFrame called *text*.

Additional columns and data wrangling

We created a function to find the name of the speaker in the speech titles.

The function *extract_author_from_title(title)*:

- Looks at the title of a speech (e.g., “The UK Economy – Speech by Andrew Bailey”).
- Searches for common separators (like " – ", " - ", or " — ") that often divide the title and the speaker's name.
- After the separator, it looks for patterns like "by [Name]" using a regular expression.
- If it finds a match, it returns the speaker’s name, cleaned of any extra spaces.

```
[ ]: # Load the dataset
df = pd.read_csv("boe_speeches_speech_text.csv")

# Combined function to extract speaker's name from title
def extract_author_from_title(title):
    # Try both types of separators
    for separator in [" – ", " - ", " — "]:
        if separator in title:
            event_part = title.split(separator)[-1]
            match = re.search(r"\bby ([\w\s\.\-']+)", event_part, re.IGNORECASE)
            if match:
                return match.group(1).strip()
    return None

# Apply to rows where 'author' is missing or new extraction is preferred
df['author'] = df['title'].apply(extract_author_from_title)


# Preview results
df[['title', 'author']].head()
```

After that, we applied the correct data types to specific columns (like dates and string columns)

Next, we used a custom function to assign the ***is_gov*** value to True in the rows where 'Bailey' was the author.

```
[ ]: # List of last names of BoE Governors from 1998 to 2022
governors_last_names = ["Bailey"]

# Function to check if the author is a governor
def is_governor(author):
    if pd.isna(author):
        return 0
    return int(any(last_name in author for last_name in governors_last_names))

# Apply the function and insert the column after 'author'
df2.insert(loc=df2.columns.get_loc('author') + 1, column='is_gov', value=df2['author'].apply(is_governor))

df2
```

This later introduced an error with some speeches by David Bailey, that were addressed manually, checking with BoE web archive.

Some other manual procedures were performed to further complete the data:

- 20 names for 'author' were added when scraping was unsuccessful. The names were found on the 'speech by' part of the titles.
- 38 speeches showed 'Speech section not found'. This was probably due to inconsistencies on the HTML structures of every speech details page. The contents where manually copied and pasted to the CSV in Excel

Finally, the dataset was saved to 'recent speeches_v2.xlsx'

Appendix 4: Data cleaning and processing

See *BEACON_LSE_EP_Assignment3_EDA.ipynb*

1. Edward George is marked as a governor of the BoE (1 in 'is_gov') as per the corresponding term (1 July 1993 – 30 June 2003).
2. Terms of Mervyn King and Andrew Bailey were aligned with the official dates.
3. Since BoE gained operational independence over monetary policy on May 6, 1997 (House of Lords #) all BoE governors are assigned full names (Bank of England) to avoid confusion between David Bailey and Andrew Bailey, and to simplify further analysis.
4. 'united kingdom' is changed to 'UK'. At this point the final joined dataset is 'speeches_1900_2025_all.csv'
5. Erroneous and duplicated UK speeches are excluded for the purpose of sentiment analysis. The dates of these speeches are included in the total counts as per cross-check with the BoE-Speeches website.
6. The initial dataset erroneously attributed to **Andrew Bailey** (governor) some speeches that were in fact given by **David Bailey**. 4 instances of this issue were found when cross-checking with BoE speeches archive. The file *'speeches_corrected_all_v2.csv'* was corrected by hand to reflect David Bailey's correct speeches.

   Since 'all_speeches.csv' does not contain full names for 'author', we can not be sure that other authors with the same last names were affected by similar problems.

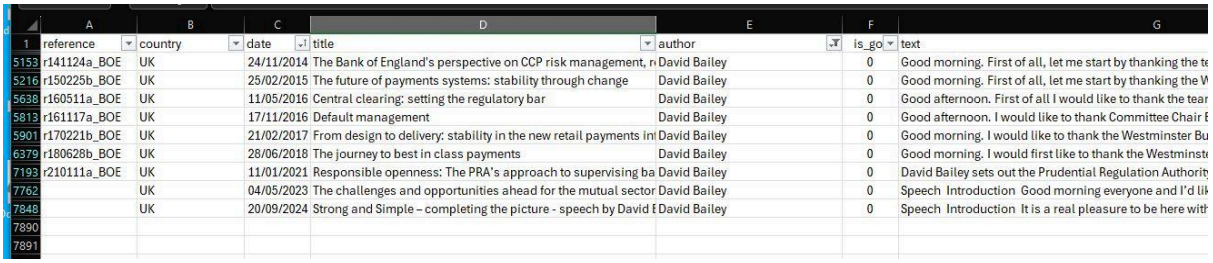
source: BoE

5791	r161117a_BOE	united kingdom	17/11/2016	Default management	bailey	1	Good afternoon. I would like to thank Committee Chair Bowen for inviting
5812	r170221b_BOE	united kingdom	21/02/2017	From design to delivery: stability in the new retail payments infrastructure	bailey	1	Good morning. I would like to thank the Westminster Business Forum fo
5898	r180628b_BOE	united kingdom	28/06/2018	The journey to best in class payments	bailey	1	Good morning. I would first like to thank the Westminster Business Forum
6030	r200713a_BOE	united kingdom	13/07/2020	LIBOR: entering the endgame	bailey	1	The passing of summer,' I thought. 'There can be no summer in this land
6035	r200828a_BOE	united kingdom	28/08/2020	The central bank balance sheet as a policy tool: past, present and future	bailey	1	It's a great pleasure to be participating in the virtual Jackson Hole confer
6037	r200903a_BOE	united kingdom	03/09/2020	Reinventing the wheel (with more automation)	bailey	1	The Covid-19 pandemic is having a severely disruptive economic impact
6052	r201109a_BOE	united kingdom	09/11/2020	The time to push ahead on tackling climate change	bailey	1	Our Governor Andrew Bailey explains what we are doing to ensure the fir
6063	r210111a_BOE	united kingdom	11/01/2021	Responsible openness: The PRA's approach to supervising banks	bailey	1	David Bailey sets out the Prudential Regulation Authority's approach to s
6067	r210205a_BOE	united kingdom	05/02/2021	Modern challenges for the modern central bank: perspectives from the Bank of England	bailey	1	Andrew Bailey looks at how central banks are adapting to new types of s
6068	r210210a_BOE	united kingdom	10/02/2021	The case for an open financial system	bailey	1	As we look forward - and for so many reasons we must look forward - it i
6075	r210308a_BOE	united kingdom	08/03/2021	Getting over Covid	bailey	1	It's a great pleasure to be participating in a Resolution Foundation event
6077	r210325a_BOE	united kingdom	25/03/2021	Banknote character	bailey	1	It's my great pleasure to unveil the new PS50 note, which will feature one
6081	r210421b_BOE	united kingdom	21/04/2021	Meeting varied people	bailey	1	It's important we engage with a more diverse range of people and institut
6088	r210512a_BOE	united kingdom	12/05/2021	Taking our second chance to make MMFs more resilient	bailey	1	Thank you for inviting me to speak today. I want to start by thanking ISDA
6097	r210601a_BOE	united kingdom	01/06/2021	Tackling climate for real: the role of central banks	bailey	1	We all tend to reach for the payment methods that we are accustomed to
6099	r210615b_BOE	united kingdom	15/06/2021	Innovation to serve the public interest	bailey	1	It's a pleasure to be participating in the City UK Annual Conference. Amo

speeches_corrected_all | all_speeches | Sheet1

Ready | Filter Mode | Accessibility: Investigate | Average: 21640 | Count: 7 | Sum: 43280

initial dataset ('all_speeches.csv')



	A	B	C	D	E	F	G
1	reference	country	date	title	author	is_go	text
5153	r141124a_BOE	UK	24/11/2014	The Bank of England's perspective on CCP risk management, r	David Bailey	0	Good morning. First of all, let me start by thanking the te
5216	r150225b_BOE	UK	25/02/2015	The future of payments systems: stability through change	David Bailey	0	Good morning. First of all, let me start by thanking the W
5638	r160511a_BOE	UK	11/05/2016	Central clearing: setting the regulatory bar	David Bailey	0	Good afternoon. First of all I would like to thank the tear
5813	r161117a_BOE	UK	17/11/2016	Default management	David Bailey	0	Good afternoon. I would like to thank Committee Chair E
5901	r170221b_BOE	UK	21/02/2017	From design to delivery: stability in the new retail payments int	David Bailey	0	Good morning. I would like to thank the Westminster Bu
6379	r180628b_BOE	UK	28/06/2018	The journey to best in class payments	David Bailey	0	Good morning. I would first like to thank the Westminste
7193	r210111a_BOE	UK	11/01/2021	Responsible openness: The PRA's approach to supervising ba	David Bailey	0	David Bailey sets out the Prudential Regulation Authorit
7762		UK	04/05/2023	The challenges and opportunities ahead for the mutual sector	David Bailey	0	Speech Introduction Good morning everyone and I'd lik
7848		UK	20/09/2024	Strong and Simple – completing the picture - speech by David E	David Bailey	0	Speech Introduction It is a real pleasure to be here with
7890							
7891							

corrections made

To add custom **unique IDs** to each speech, this function was applied:

```
[2]: def assign_custom_reference(df):
    # Ensure 'date' is datetime and 'author' is a string
    df['date'] = pd.to_datetime(df['date'], errors='coerce')
    df['author'] = df['author'].astype(str)

    # Extract first 3 characters of country
    country_part = df['country'].str[:3].fillna('UNK')

    # Format date to YYYY-MM-DD
    date_part = df['date'].dt.strftime('%Y-%m-%d')

    # Extract last name from author (assumes it's the last word)
    last_name_part = df['author'].str.strip().str.split().str[-1].fillna('unknown')

    # Concatenate to form custom reference
    df['custom_ref'] = country_part + '_' + date_part + '_' + last_name_part

    return df
```

It generates a string ID based on the country, date and author and creates a new column in the dataframe.

After applying the function to the uk_speeches_cleaned.csv data, we checked that no duplicates were present. The final dataset was exported as ***uk_speeches_cleaned_v2.csv***

Appendix 5: Economic indicator collection and cleaning

Economic indicator data collection

We selected economic indicators on the basis of being mentioned in the project brief, in meetings with BoE, in the Economic Indicators: Key statistics for the UK economy report, and in the scholarly literature we reviewed to prepare our research design. Whenever possible we collected data from official sources such as the Office for National Statistics (ONS).

Here is a list of the economic indicators we collected:

- Bank Rate
- Gross Domestic Product (GDP)
- Consumer Price Index (CPI)
- Unemployment Rate
- Average Weekly Earnings (AWE)
- Citi/YouGov Inflation Expectations Survey
- Purchasing Managers Index (PMI)
- Sterling Overnight Index Average (SONIA)
- FTSE Volatility Index (FTSE VIX)
- S&P 500 Volatility Index (VIX)
- GBP/USD Exchange Rate
- Retail Sales Index
- Mortgage Interest Rates
- Household Debt
- FTSE 100 Index (including historical volatility)
- FTSE 250 Index (including historical volatility)
- FTSE All Share Index (including historical volatility)
- Gilt Yields
- Overnight Index Swap (OIS)
- GfK Consumer Confidence
- UK House Price Index
- House Prices
- Mortgage Approvals
- Loans to Private Sector
- Business Confidence
- Consumer Spending

Economic indicator data cleaning

Please see the *Economic Indicator Cleaning and Merging* Jupyter Notebook to follow along step-by-step.

CSV files downloaded from ONS have metadata in first few rows so we skipped them when loading into Pandas DataFrames:

1	Title	CPI ANNUAL RATE 00: ALL ITEMS 2015=100			
2	CDID	D7G7			
3	Source datas	MM23			
4	PreUnit				
5	Unit	%			
6	Release date	16-04-2025			
7	Next release	21-May-25			
8	Important notes				
9	1989	5.2			
10	1990	7			

```
# Load .csv
cpi = pd.read_csv('cpi_rate.csv', skiprows=7)
cpi
```

	Important notes	Unnamed: 1
0	1989	5.2
1	1990	7.0

We renamed columns so each indicator had a unique name and the name of the date column was consistent for joining:

```
# Rename columns
cpi.rename(columns={'Important notes': 'date', 'Unnamed: 1': 'cpi_rate'}, inplace=True)
cpi
```

Some ONS files had annual, quarterly, and monthly data in the same column so we created a function (with AI assistance) to split the data into three separate DataFrames. It also changed the data type for each new DataFrame to its appropriate period frequency:

```python
def split_dataframe(df, date='date', prefix=None):
    """
    Split a dataframe with mixed date formats into three separate dataframes
    and save them as CSV files in the current directory.

    Parameters:
    - df: Dataframe containing the mixed data
    - date: Column name containing the date values
    - prefix: String to use as prefix for the saved csv files

    Returns:
    - df_annual: Dataframe with only annual values
    - df_quarterly: Dataframe with only quarterly values
    - df_monthly: Dataframe with only monthly values
    """
    # Create masks for each type of date format
    annual_mask = df[date].apply(lambda x: bool(re.match(r'^\d{4}$', str(x))))
    quarterly_mask = df[date].apply(lambda x: bool(re.match(r'^\d{4}\s+Q[1-4]$', str(x))))
    monthly_mask = df[date].apply(lambda x: bool(re.match(r'^\d{4}\s+[A-Z]{3}$', str(x))))

    # Split the dataframe based on the masks
    df_annual = df[annual_mask].copy()
    df_quarterly = df[quarterly_mask].copy()
    df_monthly = df[monthly_mask].copy()

    # Convert date columns to pandas Period format
    if len(df_annual) > 0:
        # Convert annual dates to Period with annual frequency
        df_annual['date'] = df_annual['date'].apply(lambda x: pd.Period(str(x), freq='A-DEC'))

    if len(df_quarterly) > 0:
        # Convert quarterly dates to Period with quarterly frequency
        def quarter_to_period(q_str):
            year, quarter = q_str.split()
            quarter_num = int(quarter[1])
            return pd.Period(f"{year}Q{quarter_num}", freq='Q-DEC')

        df_quarterly['date'] = df_quarterly['date'].apply(quarter_to_period)

    if len(df_monthly) > 0:
        # Convert monthly dates to Period with monthly frequency
        def month_to_period(m_str):
            year, month = m_str.split()
            dt = pd.to_datetime(m_str, format='%Y %b')
            return pd.Period(dt, freq='M')

        df_monthly['date'] = df_monthly['date'].apply(month_to_period)

    # Save the dataframes as CSV files if a prefix is provided
    if prefix:
        annual_file = f"{prefix}_annual_clean.csv"
        quarterly_file = f"{prefix}_quarterly_clean.csv"
        monthly_file = f"{prefix}_monthly_clean.csv"

    df_annual.to_csv(annual_file, index=False)
    df_quarterly.to_csv(quarterly_file, index=False)
    df_monthly.to_csv(monthly_file, index=False)

    return df_annual, df_quarterly, df_monthly

## Example usage:
# cpi_annual, cpi_quarterly, cpi_monthly = split_dataframe(cpi, 'date', prefix='cpi')
```

Some indicators like GDP are only released quarterly so we created a function (with AI assistance) to create a new DataFrame with the values in monthly rows so it could be joined with other monthly indicators:

```python
def period_quarterly_to_monthly(df_quarterly, date_col):
    """
    Convert a dataframe with values that are quarterly into a dataframe
    with rows for each month so it can be merged with other monthly dataframes.

    Parameters:
    - df: Dataframe containing the quarterly data
    - date_col: Column name containing the date values

    Returns:
    - monthly_data: Dataframe with values for each month
    """

    monthly_data = []

    for _, row in df_quarterly.iterrows():
        quarter_period = row[date_col]  # This is like "2024Q4"

        # Get the year and quarter number
        year = quarter_period.year
        quarter = quarter_period.quarter

        # Calculate the 3 months in this quarter
        start_month = (quarter - 1) * 3 + 1

        for month_offset in range(3):
            month_num = start_month + month_offset
            month_period = pd.Period(year=year, month=month_num, freq='M')

            new_row = row.copy()
            new_row['date'] = month_period
            monthly_data.append(new_row)

    return pd.DataFrame(monthly_data)

## Example usage:
## Convert your quarterly data to monthly
# df_monthly = period_quarterly_to_monthly(df_quarterly, 'date_column')
```

To ensure that bank rate values (which are decided 8 times per year) could be joined to the speech sentiment DataFrame we created a new DataFrame that forward-filled the values on a daily basis:

```python
# Create a complete date range to merge with speech dates
all_dates = pd.date_range(
    start=bank_rate['date'].min(),
    end=bank_rate['date'].max(),
    freq='D'
)

# Reindex bank rates to daily frequency and forward fill
bank_rates_daily = (bank_rate
                    .set_index('date')
                    .reindex(all_dates)
                    .ffill()
                    .reset_index()
                    .rename(columns={'index': 'date'}))
bank_rates_daily.head(10)
```

	date	bank_rate
0	1975-01-20	11.25
1	1975-01-21	11.25
2	1975-01-22	11.25
3	1975-01-23	11.25
4	1975-01-24	11.25
5	1975-01-25	11.25
6	1975-01-26	11.25
7	1975-01-27	11.00
8	1975-01-28	11.00
9	1975-01-29	11.00

After cleaning, we merged indicator DataFrames of the same period frequency using an outer join because the date range of each DataFrame varies:

```
# Join using reduce with merge
dfs = [cpi_monthly, ftse_vix_monthly, gbp_usd_monthly, gfk_cc, hpi_mom, hpi_yoy,
       house_prices, infl_exp, mortg_apprv, mortgage_rates, rsi_monthly, unemp_monthly,
       wage_monthly, pmi_monthly, ftse100_volat, ftse250_monthly, ftse250_volat,
       ftse_all_share_monthly, ftse_all_share_volat, gdp_monthly, household_debt_monthly]

# Join all DataFrames using reduce with an outer merge on the date column
monthly_indicators = reduce(lambda left, right: pd.merge(left, right, on='date', how='outer'), dfs)

# Sort by date for clarity
monthly_indicators = monthly_indicators.sort_values('date').reset_index(drop=True)

monthly_indicators.tail(5)
```

	date	cpi_rate	ftse_vix	gbp_usd_m	gfk_cc	hpi_mom	hpi_yoy	avg_house_price	infl_exp	mortgage_approvals
837	2025-01	3.0	NaN	1.2348	-22.0	0.6	2.9	298815.0	3.5	66.04
838	2025-02	2.8	NaN	1.2545	-20.0	-0.2	2.8	298274.0	3.9	65.09
839	2025-03	2.6	NaN	1.2911	-19.0	-0.5	2.8	296699.0	NaN	64.31
840	2025-04	NaN	NaN	1.3131	-23.0	NaN	NaN	NaN	NaN	NaN
841	2025-05	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

We then merged these DataFrames with the speech sentiment DataFrame to perform our predictive analyses. We used an outer join to retain the values of monthly economic indicators in months when no speeches were delivered:

```
# Perform an outer join
sentiments_monthly_indicators_all = sentiments.merge(monthly_indicators, on='month', how='outer')
sentiments_monthly_indicators_all
```

	date	custom_ref	reference	title	author	is_gov	text	finbert_pos	finbert_neg	finbert_neu	...	wage_rate	pmi	ftse100_monthly
0	1998-09-15	UK_1998-09-15_George	r980915a_BOE	Speech	Edward George	1.0	Thank you, Chairman. I'm actually very pleased...	0.081732	0.281070	0.637198	...	NaN	NaN	NaN

Appendix 6: Notes on sentiment-model processing

- Varied **text-cleaning** functions were applied to speech text based on different recommended approaches for each model (i.e. lighter cleaning for FinBERT, tokenisation for LM dictionary).
- **LM Dictionary:** Two types of scores were explored: a) simple calculations using positive/negative word counts, b) weighted calculations accounting for word counts in all sentiment categories (e.g. cautious, litigious …)
- **FinBERT:** Two methods were applied: a) overall speech level, b) individual sentence scores averaged across each speech. Findings: option b had a flattening effect on sentiment and was dropped.
- **Generative LLMs** were tested via each model's API. Positive/negative/neutral labels were generated for both Claude and ChatGPT; continuous scores were also generated with ChatGPT due to strong performance on manual tests.
- **CentralBankRoBERTa:** See detailed description in Appendix 7.
- See more detail and code in *sentiment_model_analysis.ipynb* and *sentiment_testing_results.ipynb*.

Appendix 7: CentralBankRoBERTa technical overview

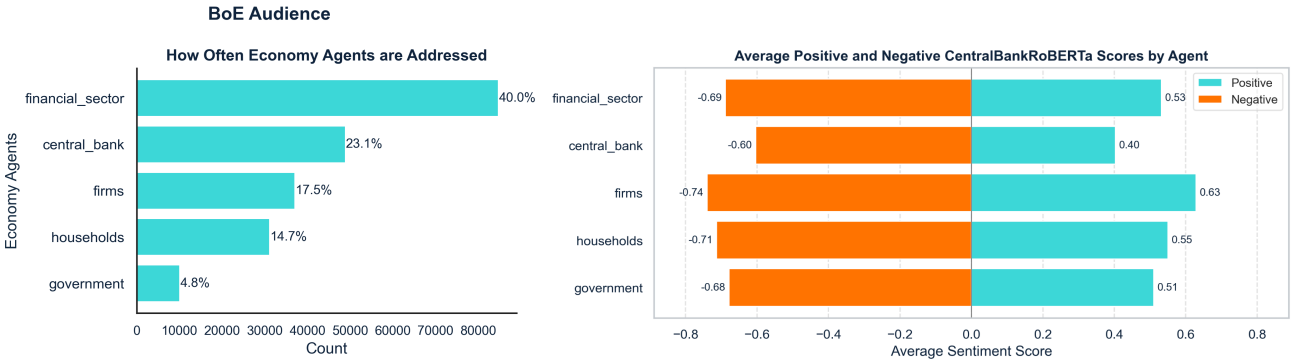
See *BEACON_LSE_EP_Assignment3_EDA.ipynb*

Model Acquisition and Sentiment Score Extraction:
To analyse the sentiment conveyed in BoE speeches, transformer models from the CentralBankRoBERTa suite were leveraged, specifically those fine-tuned for:

- agent classification – "Moritz-Pfeifer/CentralBankRoBERTa-agent-classifier",
- sentiment detection – "Moritz-Pfeifer/CentralBankRoBERTa-sentiment-classifier".

These models were obtained from a publicly available GitHub by means of pipelines listed on Hugging Face. The sentiment classifier was modified to support a regression head rather than a classification head by means of 'softmax'. This allows the extraction of a continuous sentiment score for each sentence or passage in the speech, enabling a more granular representation of tone beyond simple positive, neutral, or negative categories.

Pivoting Agent-Level Sentiment Data:
After inference, each sentence in a speech was tagged with both an economic agent (e.g., households, firms, financial sector, etc.) and a sentiment score:



To analyse sentiment at the level of entire speeches, the resulting data was pivoted into a wide-format table, aggregating sentiment scores for each agent per speech. This pivoted structure enabled easy comparison of how different economic agents were addressed or perceived in a given speech, and facilitated time series analysis and correlation with market indicators.

Optimising Inference for Scalability
Given the volume of textual data and the number of model inferences required per sentence, optimisation of processing time was critical. The inference pipeline was adapted to utilise available hardware — executing on GPU (1hr 3mins processing time) when available for significant speed-up, and falling back to CPU otherwise (7hrs 40 mins processing time). This hardware-agnostic implementation allows to scale the sentiment labelling process efficiently across large batches of speech data, ensuring consistency in results while reducing processing bottlenecks.

Weak Supervision with Snorkelling for Label Generation
We used the Snorkel framework to generate sentiment and policy stance labels for BoE speeches via weak supervision. Custom labelling functions were built using economic keyword patterns, heuristics, and domain-specific lexicons. These noisy labels were aggregated into probabilistic annotations using a generative model. The resulting data was used to evaluate CentralBankRoBERTa's accuracy on the full dataset (1365 speeches) in the absence of large-scale manual annotations.

Appendix 8: GfK MLR model and volatility scenarios

To test the performance of our MLR model under extraordinary conditions, we used FTSE-100 Historical Volatility to define scenarios of high, normal and low economic tensions.

Historical volatility is a statistical measure used to analyze the general dispersion of security or market index returns for a specified period of time. (source: https://www.tradingview.com/support/solutions/43000589145-historical-volatility/)

The data was exported from Tradingview charts. We created a pandas dataframe 'ftse_d' to structure the data. Then, using monthly aggregation (the average volatility for the month) we defined three different buckets, based on the threshold for high and low volatility. The thresholds were calculated as 0.33 and 0.66 quantiles of the sample.

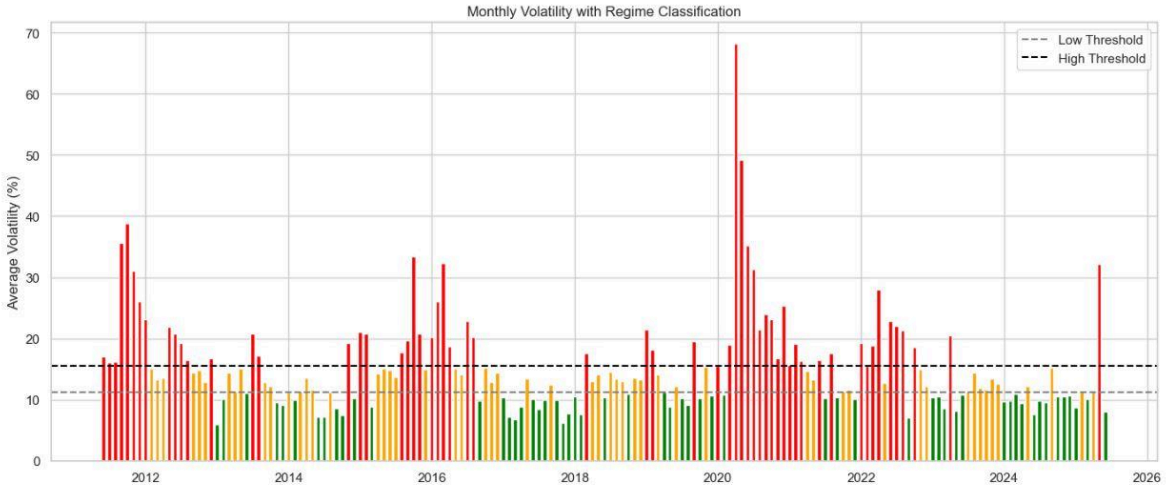
[9]:
```
# Set datetime index
ftse_d.set_index('date', inplace=True)

# Resample volatility to monthly frequency and compute the average
monthly_volatility = ftse_d['volatility'].resample('M').mean()

# Define thresholds for volatility regimes using quantiles
low_threshold = monthly_volatility.quantile(0.33)
high_threshold = monthly_volatility.quantile(0.66)

# Classify regimes
def classify_regime(vol):
    if pd.isna(vol):
        return 'unknown'
    elif vol < low_threshold:
        return 'calm'
    elif vol > high_threshold:
        return 'crisis'
    else:
        return 'neutral'

monthly_regime = monthly_volatility.apply(classify_regime)
monthly_regime.name = 'market_regime'
```



This volatility data was merged with the main dataframe (containing sentiment and indicators) to have specific scenarios to evaluate the model on.

[26]:
```
# Copy the clean dataset used for modeling and add 'market_regime'
model_df = maindf_merged[features + [target, 'market_regime']].dropna()

# Segment data by volatility regime
results_by_regime = {}

for regime in ['crisis', 'neutral', 'calm']:
    subset = model_df[model_df['market_regime'] == regime]

    if len(subset) < 10:
        continue  # skip if too few samples

    X_sub = subset[features]
    y_sub = subset[target]

    # Split (no shuffle to preserve timeline consistency)
    X_train, X_test, y_train, y_test = train_test_split(X_sub, y_sub, test_size=0.2, shuffle=False)

    model = LinearRegression()
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)

    mae = mean_absolute_error(y_test, y_pred)
    r2 = r2_score(y_test, y_pred)

    results_by_regime[regime] = {'MAE': mae, 'R2': r2, 'n_samples': len(subset)}

results_by_regime_df = pd.DataFrame(results_by_regime).T
results_by_regime_df.index.name = 'Volatility Regime'
results_by_regime_df.reset_index(inplace=True)


results_by_regime_df
```

[26]:

	Volatility Regime	MAE	R2	n_samples
0	crisis	7.348943	-1.013763	204.0
1	neutral	3.656209	0.447694	234.0
2	calm	4.634563	-0.116240	219.0

The results suggest that **neutral periods have more predictable structure.**

During **crisis** periods, consumer sentiment can shift abruptly due to shocks not captured by the features (e.g. news, extreme market moves, etc).

In **calm** periods, GfK might not move much, making it hard for the model to capture signal (minor shifts become hard to predict, and noise dominates).

Appendix 9: Gradient boosting regression

1. Data preparation and feature engineering

Our primary objective was to model the relationship between daily speech sentiment and monthly economic indicators. The initial df_final dataframe contained daily speech sentiment scores (from FinBERT and CBRoBERTa, including agent-specific sentiments) and monthly economic data (CPI rate, GBP/USD, GFK Consumer Confidence, Unemployment Rate, Wage Rate). Please refer to 'sentiments_topics_monthly_indicators.csv' for the raw data.

```
<class 'pandas.core.frame.DataFrame'>
Index: 664 entries, 0 to 663
Data columns (total 16 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   date                    664 non-null    datetime64[ns]
 1   month                   664 non-null    object
 2   custom_ref              664 non-null    object
 3   finbert_score           664 non-null    float64
 4   central_bank            664 non-null    float64
 5   financial_sector        664 non-null    float64
 6   firms                   664 non-null    float64
 7   government              664 non-null    float64
 8   households              664 non-null    float64
 9   cbroberta_avg_score     664 non-null    float64
 10  cbroberta_avg_sentiment 664 non-null    float64
 11  cpi_rate                664 non-null    float64
 12  gbp_usd_m               664 non-null    float64
 13  gfk_cc                  664 non-null    float64
 14  unemp_rate              664 non-null    float64
 15  wage_rate               664 non-null    float64
dtypes: datetime64[ns](1), float64(13), object(2)
memory usage: 88.2+ KB
```

The project team decided to focus on the period between 20/01/2015 and 16/12/2024 which was guided by the last 10 years being the key focus as mentioned by Bank of England and interesting observations in the correlation and time series analysis.

At a later stage into the project, we decided to focus on only the CPI Rates economic indicator as this analysis revealed interesting insights.

2. Approach and rationale

A critical decision involved aligning these disparate data frequencies. We opted to aggregate daily sentiment scores to monthly averages. This approach provides a robust representation of the Bank's overall communication tone within a given month, mitigating noise from individual daily speeches and ensuring a consistent monthly granularity for correlation with economic indicators. This decision was crucial for avoiding data leakage and ensuring that our sentiment features truly reflected a monthly aggregated signal.

Furthermore, to explore predictive power, lagged features were engineered. For each sentiment indicator, 1, 2, and 3-month lags were created. This allowed the Gradient Boosting Regressor to learn if sentiment from previous months influenced current economic conditions. Critically, the economic indicators themselves were also lagged by one month. The choice of 1 month lag for economic indicators was chosen because the most recent past value of a time series is usually its strongest predictor. All rows with Nan values, resulting from the lagging process, were subsequently removed to ensure a clean dataset for modelling.

```
# Align Daily Sentiment with Monthly Economic Indicators & Handle Monthly Data
# Aggregate daily sentiment scores to monthly averages
monthly_sentiment = df_final.groupby('month')[
    ['finbert_score', 'cbroberta_avg_score', 'central_bank', 'financial_sector', 'firms', 'government', 'households']
].mean().reset_index()
```

```
# Merge monthly sentiment with the monthly economic indicators.
monthly_econ_data_unique = df_final[['month', 'cpi_rate', 'gbp_usd_m', 'gfk_cc', 'unemp_rate', 'wage_rate']].drop_duplicates(subset=['month'])

# Combine monthly sentiment with monthly economic data
monthly_df = pd.merge(monthly_sentiment, monthly_econ_data_unique, on='month', how='inner')

# Sort by month for correct lagging
monthly_df = monthly_df.sort_values(by='month').set_index('month')

print("\nMonthly Aggregated DataFrame Head (for lagging):")
print(monthly_df.head())
print("\nMonthly Aggregated DataFrame Info:")
monthly_df.info()
```

```
# Feature Engineering: Create Lagged Features
# We'll create lags for both sentiment indicators and the economic indicators themselves.
sentiment_cols = ['finbert_score', 'cbroberta_avg_score', 'central_bank', 'financial_sector', 'firms', 'government', 'households']
economic_cols = ['cpi_rate', 'gbp_usd_m', 'gfk_cc', 'unemp_rate', 'wage_rate']

# Number of months to lag sentiment and economic indicators
# Let's consider 1, 2, and 3-month lags for sentiment
# For economic indicators, 1-month lag is usually sufficient as a strong predictor
lags_sentiment = [1, 2, 3] # Months
lags_econ = [1] # Months

# Create lagged sentiment features
for col in sentiment_cols:
    for lag in lags_sentiment:
        monthly_df[f'{col}_lag_{lag}m'] = monthly_df[col].shift(lag)

# Create lagged economic indicator features (these will be predictors for the *next* month's value)
for col in economic_cols:
    for lag in lags_econ: # Typically just 1-month lag for the target variable itself
        monthly_df[f'{col}_lag_{lag}m'] = monthly_df[col].shift(lag)
```

```
# Drop rows with NaN values resulting from lagging
df_model_ready = monthly_df.dropna()

print("\nDataFrame with Lagged Features Head:")
print(df_model_ready.head())
print("\nDataFrame with Lagged Features Info:")
df_model_ready.info()
```

3. Modelling approach and tooling: leveraging gradient boosting for insight

For this predictive analysis, Python was chosen as the primary environment, leveraging its robust data science ecosystem.

Tool Selection & Rationale:

The Gradient Boosting Regressor from scikit-learn was selected as the core analysis tool. Its rationale is multi-faceted:

Predictive Power: Gradient Boosting is known for its high predictive accuracy, capable of capturing complex, non-linear relationships within the data.

With the help of AI, we were able to build a gradient boosting regression model that would iterate through all the features of the data and output key regression metrics such as R-Squared, MSE, as well as feature importance and actual VS predicted plots. A potential next step could be to refine this model further by experimenting with different values of sentiment lags, hyperparameter tuning (tree depth, learning rate etc..) and further feature engineering.

Feature Importance: Crucially for the Bank of England, Gradient Boosting models provide clear feature importance scores. This allows us to directly quantify which sentiment features (and their lags) contribute most to predicting specific economic indicators, offering actionable insights beyond simple correlation.

Robustness to Scaling: As a tree-based ensemble model, Gradient Boosting is inherently less sensitive to the scaling of input features, simplifying the preprocessing pipeline. While StandardScaler was applied to features, it was primarily for consistency and potential future model flexibility, not a strict requirement for GBR performance.

Data Splitting: A standard 80/20 train-test split was implemented. A vital decision for time-series data was setting shuffle=False during the split. This preserves the chronological order of the data, preventing data leakage where the model might "see" future information during training, leading to overly optimistic performance estimates.

```
# Train and Evaluate Model for Each Economic Indicator
for target_econ_indicator in economic_targets:
    formatted_target = target_econ_indicator.replace('_', ' ').title()
    print(f"\n--- Modeling for Target: {formatted_target} ---")

    # Define the features (X) and the specific target (y)
    # Features will include all lagged sentiment and the lagged version of the current economic indicator
    current_features = [col for col in X_scaled.columns if col != f'{target_econ_indicator}_lag_1m']
    current_features.append(f'{target_econ_indicator}_lag_1m')

    X = X_scaled[all_lagged_features] # Use all lagged features that were scaled
    y = df_model_ready[target_econ_indicator]

    # Align X and y indices after dropping NaNs and before splitting
    common_index = X.index.intersection(y.index)
    X_aligned = X.loc[common_index]
    y_aligned = y.loc[common_index]

    # Split data into training and testing sets (80/20, shuffle=False for time series)
    X_train, X_test, y_train, y_test = train_test_split(
        X_aligned, y_aligned, test_size=0.2, random_state=42, shuffle=False
    )
```

```python
# Visualize Actual vs. Predicted (for each target)
plt.figure(figsize=(14, 7))
plt.plot(y_test.index, y_test, label=f'Actual {formatted_target}')
plt.plot(y_test.index, y_pred, label=f'Predicted {formatted_target}', linestyle='--')
plt.title(f'Actual vs. Predicted {formatted_target} (Test Set)', fontsize=16)
plt.xlabel('Date')
plt.ylabel(formatted_target)
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.savefig(f'actual_vs_predicted_{target_econ_indicator}.png')
plt.show()

# Visualize Top Feature Importances for each target
plt.figure(figsize=(12, 7))
# Using importance_df.head(15) directly for plotting to ensure consistent order
sns.barplot(x='Importance', y='Feature', data=importance_df.head(15), palette='viridis')

# Add numeric labels to the bars
# Iterate through the data
for i, (feature, importance) in enumerate(zip(importance_df['Feature'].head(15), importance_df['Importance'].head(15))):
    plt.text(importance + (importance_df['Importance'].max() * 0.01),
             i,
             f'{importance:.3f}',
             color='black', ha='left', va='center', fontsize=9)

plt.title(f'Feature Importance for Predicting {formatted_target}', fontsize=16)
plt.xlabel('Relative Importance')
plt.ylabel('Feature')
# Adjust x-axis limit to accommodate labels
plt.xlim(0, importance_df['Importance'].head(15).max() * 1.15)
plt.tight_layout()
plt.savefig(f'feature_importance_{target_econ_indicator}.png')
plt.show()
```

```python
# Initialize and Train Gradient Boosting Regressor
gbr = GradientBoostingRegressor(random_state=42)
gbr.fit(X_train, y_train)

# Store the trained model
models[target_econ_indicator] = gbr
joblib.dump(gbr, f'gbr_model_{target_econ_indicator}.pkl') # Save model

# Make predictions
y_pred = gbr.predict(X_test)

# Evaluate model performance
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(f'  Mean Squared Error (MSE): {mse:.4f}')
print(f'  R-squared (R2): {r2:.4f}')

# Feature Importance Analysis
feature_importance = gbr.feature_importances_
importance_df = pd.DataFrame({
    'Feature': X_train.columns,
    'Importance': feature_importance
}).sort_values(by='Importance', ascending=False)

print("\n  Top 10 Feature Importances:")
print(importance_df.head(10))
```

4. Visualisations

Visualizations were central to communicating complex relationships effectively.

Correlation Heatmaps: 2 distinct heatmaps were generated:

Current Sentiment vs. Current Economic Indicators: To show simultaneous linear relationships.

Lagged Sentiment vs. Current Economic Indicators: The most critical for predictive insights, illustrating if past sentiment correlates with future economic movements.

Rationale: Heatmaps provide an immediate, intuitive overview of linear correlations across many variables. Seaborn was used for its aesthetic quality and annot=True for displaying exact correlation coefficients. cmap='coolwarm' was chosen for clear distinction between positive (warm) and negative (cool) correlations.

Actual vs. Predicted Plots: For each economic indicator, a line plot comparing actual vs. predicted values on the test set was created.

Rationale: This visualizes the model's overall fit and ability to track trends over time.

Feature Importance Bar Plots: For each economic indicator, a horizontal bar plot displayed the top 15 most important features.

Rationale: This directly answers "which factors predict what?" The use of seaborn. Barplot provides a clear comparison. A key enhancement involved adding numeric labels (importance scores) directly to the end of each bar. This decision was made to provide precise quantitative evidence at a glance, making the plots highly informative and actionable without requiring constant cross-referencing. plt.text was used for this, with careful adjustment of x-axis limits (plt.xlim) to ensure labels were fully visible and not cut off.

5. Patterns, trends and insights

Our analysis reveals compelling patterns:

Lagged Sentiment as a Predictor: Consistently, specific lagged sentiment features emerged as statistically significant predictors for future economic indicators, even after accounting for the indicator's own past values. This is a powerful finding, suggesting the Bank's communication is not merely reactive but possesses a proactive influence.

For Inflation (CPI Rate), sentiment towards Households (2-month and 1-month lags) showed notable importance, indicating that the Bank's tone on household well-being can precede inflation shifts.

For Consumer Confidence (GFK CC), Overall CBRoBERTa sentiment (1-month lag) and Financial Sector sentiment (2-month lag) were key, suggesting market participants take time to process financial stability signals, impacting broader consumer sentiment.

For Wage Rate, again, Household sentiment (2-month and 1-month lags) played a significant role, highlighting the Bank's influence on labour market expectations.

Dominance of Past Economic Data: As anticipated, the previous month's value of an economic indicator was almost always its strongest predictor. This underscores the inherent persistence and auto-correlation in economic time series. However, sentiment's contribution, though smaller in magnitude, represents a crucial additional layer of predictive insight.

Topic-Specific Impact: The analysis showed that different facets of sentiment (e.g., 'financial sector' vs. 'households') have varying degrees and lags of impact on different economic indicators. This points to the nuanced way specific communication themes resonate across the economy.

Appendix 10: SARIMAX

Data

tsm_data

date	cpi_rate	unemp_rate	cbroberta_avg_score	central_bank	financial_sector	firms	government	households
2015-01	0.3	5.6	-0.300475877	-0.235685535	-0.341904784	-0.184261956	-0.372016383	-0.257171993
2015-02	0	5.5	-0.082323883	-0.203433018	-0.201116929	0.287657444	0.145076329	0.036422907
2015-03	0	5.5	-0.34361561	-0.343755976	-0.452808711	-0.159987166	-0.347925763	-0.346886271
2015-04	-0.1	5.6	-0.34361561	-0.343755976	-0.452808711	-0.159987166	-0.347925763	-0.346886271
2015-05	0.1	5.6	-0.396377079	-0.317329514	-0.461771167	-0.126308243	-0.505255659	-0.398512734
2015-06	0	5.5	-0.281783354	-0.244775605	-0.355566172	-0.147263898	-0.313056507	-0.263501904
2015-07	0.1	5.4	-0.353835149	-0.239627534	-0.3444139	-0.276625977	-0.484361325	-0.246656901
2015-08	0	5.3	-0.496853115	-0.49647883	-0.443991521	-0.586309108	-0.367993395	-0.256065056
2015-09	-0.1	5.2	-0.320271749	-0.280191471	-0.382010136	-0.092863162	-0.325416294	-0.370509386
2015-10	-0.1	5.1	-0.251705666	-0.360639843	-0.32921042	0.052723969	-0.202317273	-0.203548891

tsm_data_1

date	cpi_rate	unemp_rate	cbroberta_avg_score	central_bank	financial_sector	firms	government	households	gfk_cc	hpi_mom	infl_exp	wage_rate
2015-01	0.3	5.6	-0.300475877	-0.235685535	-0.341904784	-0.184261956	-0.372016383	-0.257171993	1	0.2	1.2	1.5
2015-02	0	5.5	-0.082323883	-0.203433018	-0.201116929	0.287657444	0.145076329	0.036422907	1	0.4	1	1.8
2015-03	0	5.5	-0.34361561	-0.343755976	-0.452808711	-0.159987166	-0.347925763	-0.346886271	4	0.5	1.4	2.2
2015-04	-0.1	5.6	-0.34361561	-0.343755976	-0.452808711	-0.159987166	-0.347925763	-0.346886271	4	0.8	1.1	2.6
2015-05	0.1	5.6	-0.396377079	-0.317329514	-0.461771167	-0.126308243	-0.505255659	-0.398512734	1	0.7	1	2.7
2015-06	0	5.5	-0.281783354	-0.244775605	-0.355566172	-0.147263898	-0.313056507	-0.263501904	7	0.6	1.4	2.7
2015-07	0.1	5.4	-0.353835149	-0.239627534	-0.3444139	-0.276625977	-0.484361325	-0.246656901	4	1	1.6	2.8
2015-08	0	5.3	-0.496853115	-0.49647883	-0.443991521	-0.586309108	-0.367993395	-0.256065056	7	1.2	1.4	2.7
2015-09	-0.1	5.2	-0.320271749	-0.280191471	-0.382010136	-0.092863162	-0.325416294	-0.370509386	3	0.3	1.5	2.4
2015-10	-0.1	5.1	-0.251705666	-0.360639843	-0.32921042	0.052723969	-0.202317273	-0.203548891	2	0.3	1.4	2

tsm_data.csv was used initially. When macro indicators were included, tsm_data_1.csv was used instead.

As the values for all variables fall within a comparable order of magnitude. No standardisation or min-max scaling was applied prior to modelling.

Technical overview of the code

Functions to determine statistically safe lags:

```
def get_safe_lags(target, max_lag, order, seasonal_order):
    p, d, q = order
    P, D, Q, s = seasonal_order
    exclude = set(range(1, max(p, q)+1))
    for i in range(1, max(P, Q)+1):
        lag = i * s
        if lag <= max_lag:
            exclude.add(lag)
    included = [f"{target}_lag{lag}" for lag in range(1, max_lag+1) if lag not in exclude]
    return included
```

The above function excludes any lag that overlaps with the nonseasonal AR/MA or seasonal P/Q terms implied by the model's (order) and (seasonal_order).

```
def fit_sarimax_comparison(
    df,
    target_col,
    exog_vars,
    max_lag=6,
    test_size=0.2,
    m=12
):
    df = df.copy()
    for lag in range(1, max_lag+1):
        df[f"{target_col}_lag{lag}"] = df[target_col].shift(lag)
    df_lags = df.dropna().copy()
    split_idx = int(len(df_lags) * (1 - test_size))
    train = df_lags.iloc[:split_idx]
    test = df_lags.iloc[split_idx:]
    safe_lags = get_safe_lags(target_col, max_lag, (1,2,2), (2,0,0,12))  # placeholder, will be updated below
    model_auto = auto_arima(
        train[target_col],
        exogenous=train[exog_vars],
        seasonal=True,
        m=m,
        trace=True,
        stepwise=True,
        error_action='ignore',
        suppress_warnings=True,
        max_p=3, max_q=3, max_P=2, max_Q=2,
        max_order=6,
        information_criterion='aic'
    )
    safe_lags = get_safe_lags(target_col, max_lag, model_auto.order, model_auto.seasonal_order)
    final_exog = exog_vars + safe_lags
    model = SARIMAX(
        train[target_col],
        exog=train[final_exog],
        order=model_auto.order,
        seasonal_order=model_auto.seasonal_order,
        enforce_stationarity=True,
        enforce_invertibility=True
    )
    results = model.fit(disp=False)
    forecast = results.predict(start=test.index[0], end=test.index[-1], exog=test[final_exog])
    mae = mean_absolute_error(test[target_col], forecast)
    r2 = r2_score(test[target_col], forecast)
    return results, train, test, forecast, mae, r2, model_auto.order, model_auto.seasonal_order, final_exog
```

The above function *fit_sarimax_comparison* does several things in sequence:

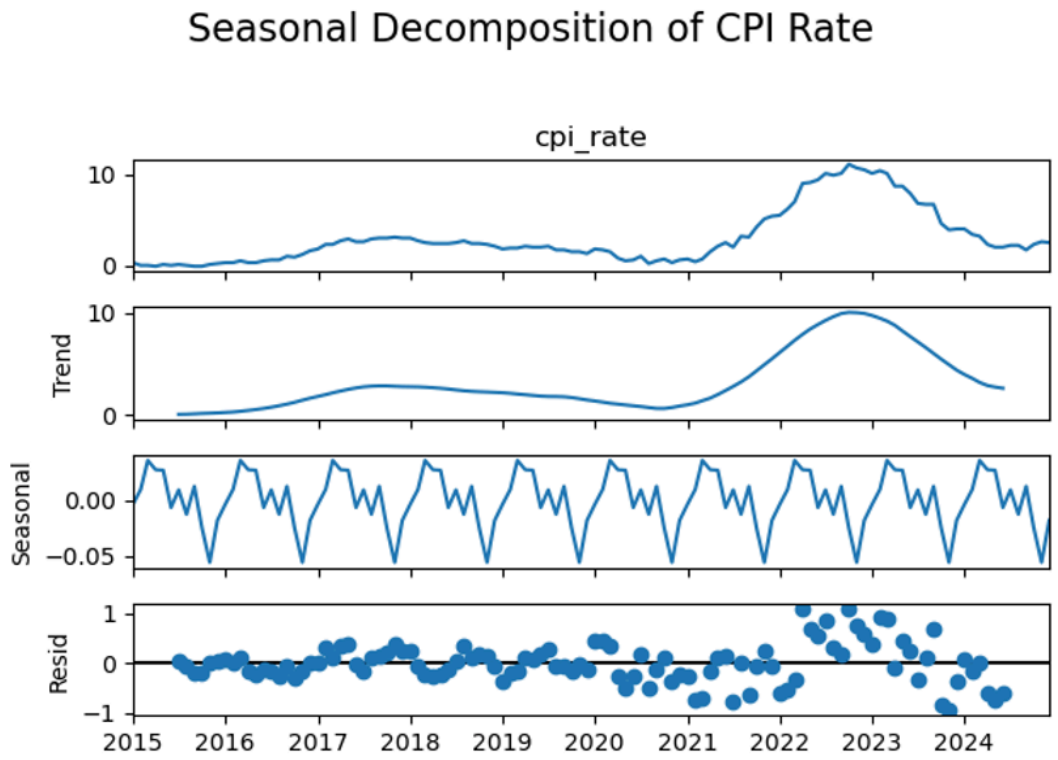
1. Create lagged columns for the target
2. Drop rows with missing values (due to shifting)
3. Split into train/test
4. Run auto_arima to select good (p,d,q) and seasonal (P,D,Q,s)
5. Use *get_safe_lags* function to decide which of the lagged columns to keep as exogenous
6. Fit a SARIMAX model with exogenous variables
7. Forecast on the test set and compute common metrics (MAE, R^2)

Using these functions, the optimal configuration for all models was SARIMAX(1,2,2)(2,0,0,12). When allowing for up to six lags of CPI to be generated, only *cpi_rate_lag3*, *cpi_rate_lag4*, *cpi_rate_lag5*, and *cpi_rate_lag6* were included.

AIC (Akaike Information Criterion), not BIC, was used as this:

- Calculates how well the model fits the data (the likelihood)
- Adds a penalty if the model uses too many parameters (to avoid overfitting)
- Lower AIC = better model (more accurate with fewer unnecessary parameters)

Why SARIMAX?



The seasonal decomposition makes two things very clear:

1. Seasonal component

In the “Seasonal” panel, a repeating up-and-down pattern at the same time each year is evident. CPI does not simply wander randomly as every 12 months there’s a systematic rise and fall. A plain ARIMA (no seasonal term) cannot capture that regular annual cycle. By contrast, a SARIMAX with a seasonal period of 12 explicitly estimates parameters for those 12-month patterns (the “P”, “D”, “Q”, and s=12 in SARIMAX).

2. Trend

The “Trend” panel shows a gradual up-swing in CPI from roughly 2015 to 2022, then a decline toward 2024. This means the raw CPI series is not stationary. An ARIMA model on the raw data would give misleading results. SARIMAX enables the ability to specify non-seasonal differencing (the “d” in (p,d,q)) to remove that overall trend before fitting any AR/MA or seasonal terms.

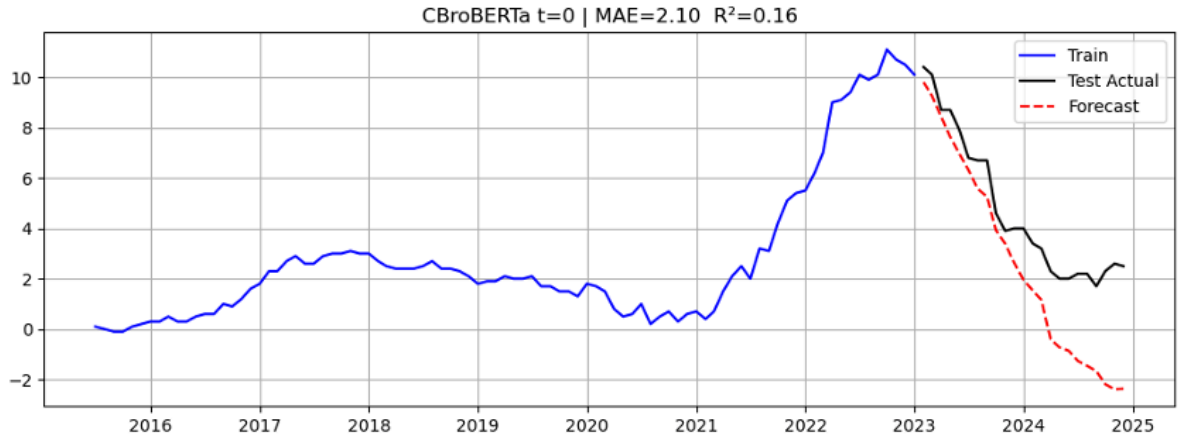
SARIMAX also allows the inclusion of exogenous regressors all in one framework.

SARIMAX - Detailed patterns, trends and findings

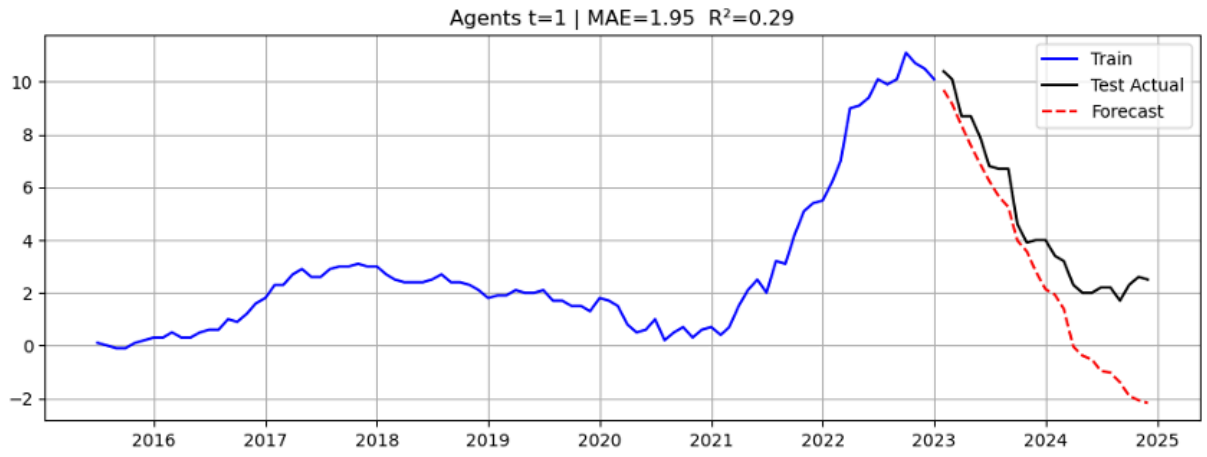
- Inflation is highly sticky: Past inflation (cpi_rate_lag6 in particular) was the most robust predictor of future inflation, underscoring its persistent, self-reinforcing nature.

- Using CBRoBERTa economic agent sentiment (Agents = firms, households, central bank) scores strongly improved forecasting accuracy compared to the model using CBRoBERTa average sentiment (R^2: 0.65 v 0.16, MAE: 1.46 v 2.10). Although MAE of 1.46 may still be too high given the goal of predicting CPI rate.

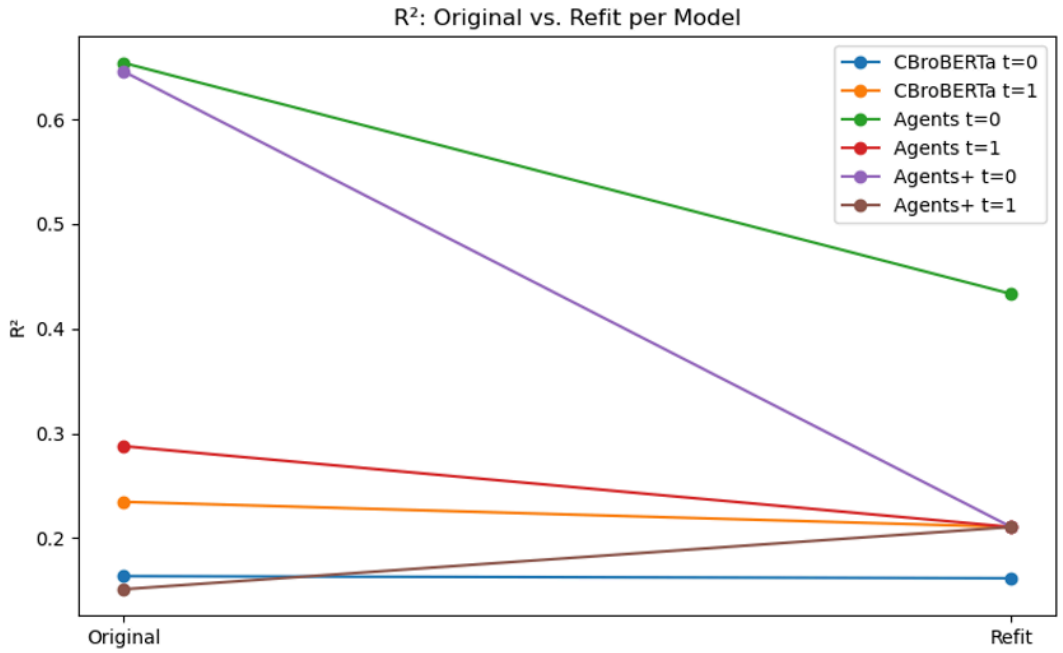
- Contemporaneous central_bank CBRoBERTa sentiment capturing the BoE's outlook, tone and confidence is a powerful coincident indicator. Its positive coefficient was statistically significant. (Coeff: 0.385, p-value: 0.0237)

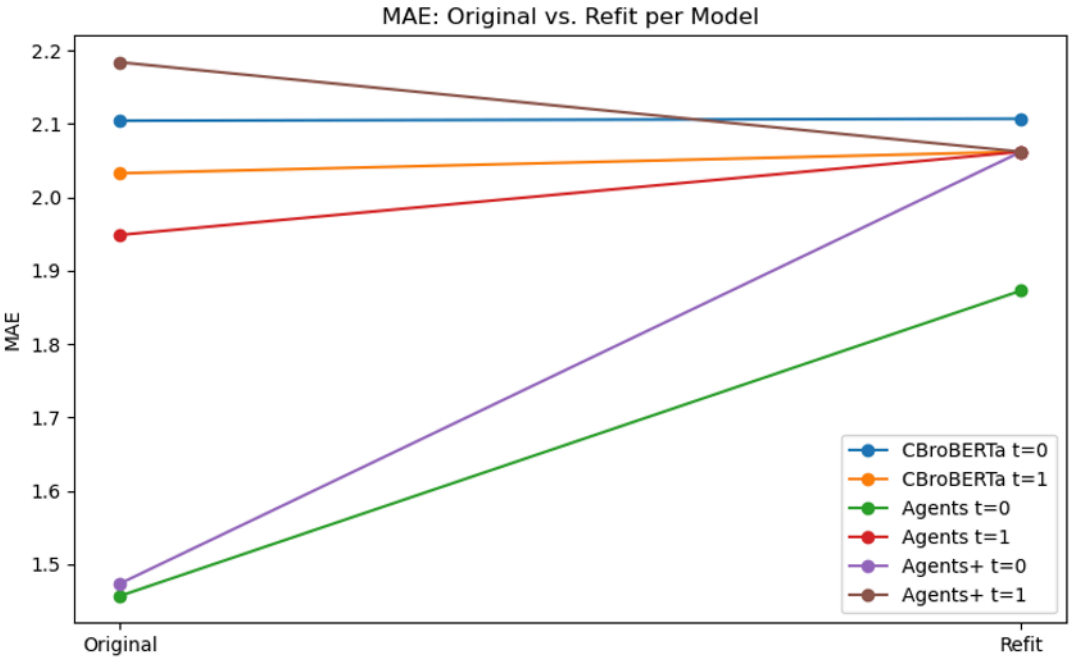


- Sentiment effects are immediate, not delayed. The model using contemporaneous speech sentiment outperformed lagged speech sentiment model, indicating that market reactions to BoE communication happen quickly. (R^2: 0.65 v 0.29, MAE: 1.46 v 1.95)



- Efforts to simplify models by removing non-significant predictors often reduced forecast accuracy, reflecting a trade-off between interpretability and performance. Adding CBRoBERTa sentiment in relation to government, financial sector (Agents+) degraded performance.





- Adding selected macro indicators lagged by a month to best-performing model Agents t=0, worsened the performance of the model. Possible reasons may be poor lag structure or indicators not being truly exogenous. Crucially, speech sentiment about the BoE itself remained statistically significant even with added macro indicators (Coeff: 0.5399, p-value: 0.014). Including even longer-lagged macro indicators may mask the signal from speech sentiment.

SARIMAX Results

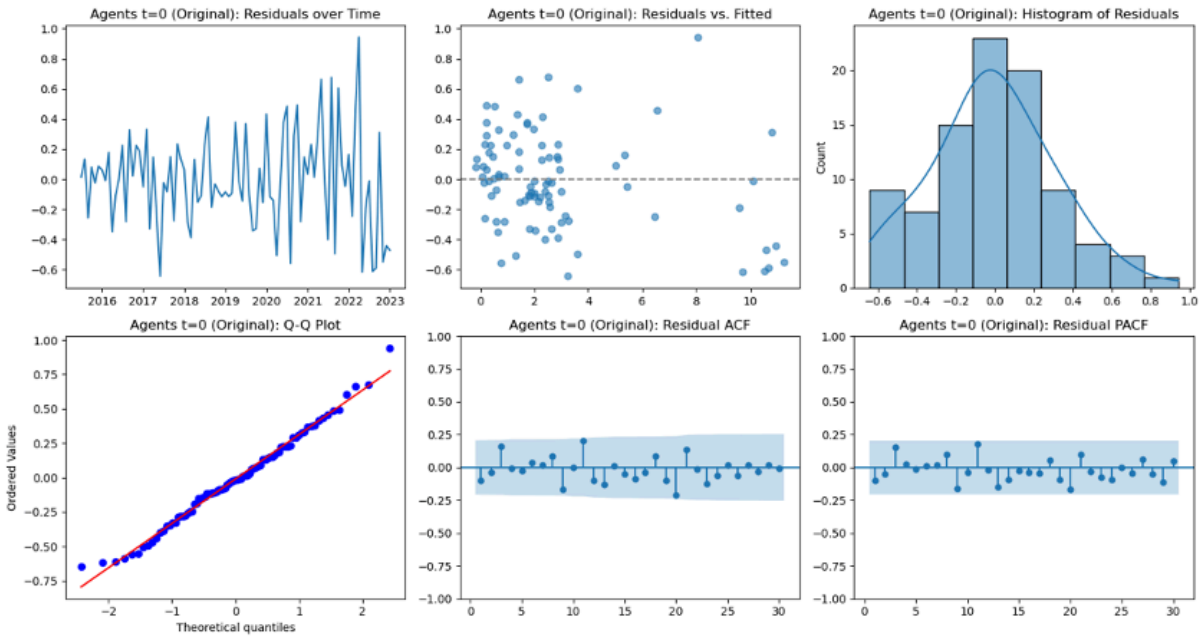
Dep. Variable:	cpi_rate	No. Observations:	91
Model:	SARIMAX(1, 2, 2)x(2, 0, [], 12)	Log Likelihood	-23.522
Date:	Fri, 30 May 2025	AIC	83.045
Time:	22:58:18	BIC	127.840
Sample:	07-01-2015 - 01-01-2023	HQIC	101.100
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
firms	-0.2703	0.199	-1.362	0.173	-0.659	0.119
central_bank	0.5399	0.220	2.451	0.014	0.108	0.972
households	0.0946	0.182	0.521	0.602	-0.261	0.451
gfk_cc_lag1	0.0004	0.008	0.045	0.964	-0.015	0.016
hpi_mom_lag1	0.0912	0.067	1.358	0.175	-0.040	0.223
infl_exp_lag1	0.2770	0.117	2.358	0.018	0.047	0.507
wage_rate_lag1	-0.0384	0.091	-0.423	0.672	-0.216	0.139
unemp_rate_lag1	-0.2184	0.380	-0.574	0.566	-0.964	0.527
cpi_rate_lag3	0.0067	0.145	0.046	0.963	-0.277	0.291
cpi_rate_lag4	-0.0110	0.153	-0.072	0.943	-0.311	0.289
cpi_rate_lag5	-0.0042	0.140	-0.030	0.976	-0.278	0.269
cpi_rate_lag6	0.4602	0.123	3.749	0.000	0.220	0.701
ar.L1	-0.0887	1.205	-0.074	0.941	-2.450	2.272
ma.L1	-0.8676	1.222	-0.710	0.478	-3.262	1.527
ma.L2	0.0839	1.130	0.074	0.941	-2.131	2.299
ar.S.L12	-0.4639	0.185	-2.505	0.012	-0.827	-0.101
ar.S.L24	-0.4630	0.160	-2.885	0.004	-0.778	-0.148
sigma2	0.0910	0.018	4.933	0.000	0.055	0.127

Ljung-Box (L1) (Q):	0.02	Jarque-Bera (JB):	0.18
Prob(Q):	0.89	Prob(JB):	0.91
Heteroskedasticity (H):	3.25	Skew:	0.10
Prob(H) (two-sided):	0.00	Kurtosis:	2.91

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).

Best model agents t=0 residual analysis and model diagnostics



Errors are well-behaved, so statistical inference and forecasts are reliable.

- Ljung-Box (Q): Residuals have no significant autocorrelation.
- Jarque-Bera (JB): Residuals not highly non-normal.
- Heteroskedasticity (H): Residuals may show changing variance (often seen with macro time series). Could explore GARCH instead.

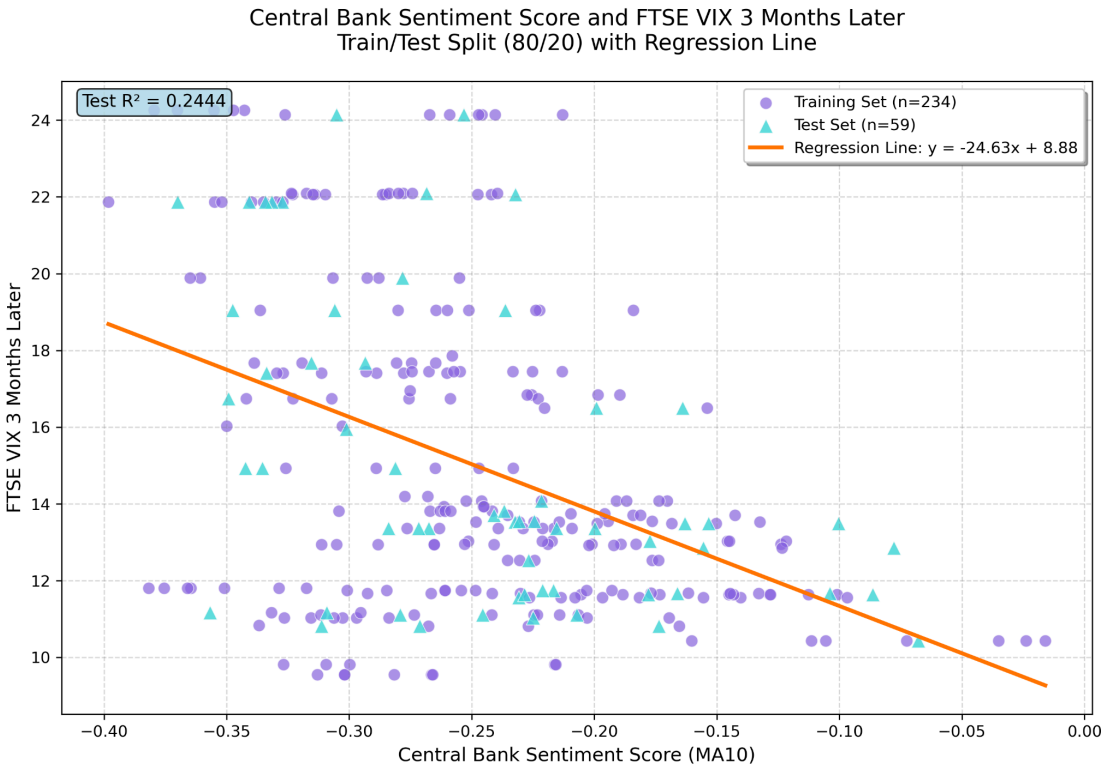
Multicollinearity was not deemed a problem for prediction accuracy given that focus was on out-of-sample predictive power (R^2, MAE on the test set).

Appendix 11: What can speech sentiment predict on its own?

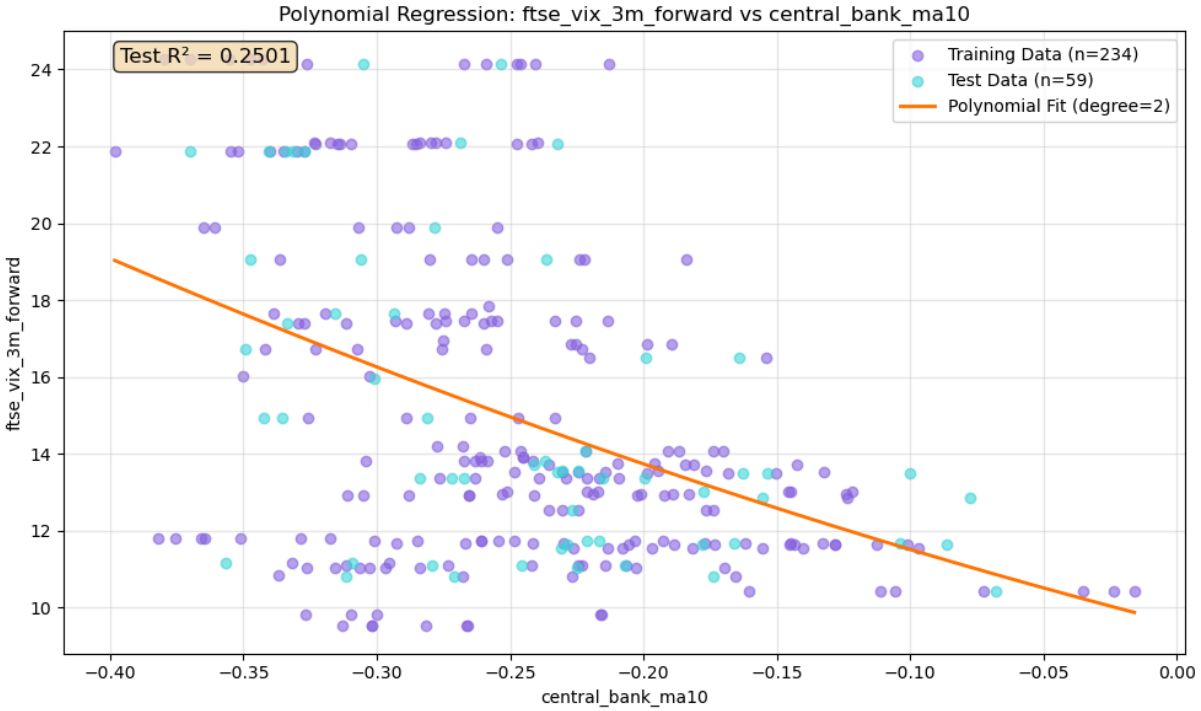
(See *Predictive Power of Speech Sentiment.ipynb* for step-by-step details.)

The strongest correlation (-0.4321) between a sentiment score and an economic indicator is the 10-speech moving average for central bank sentiment and the FTSE volatility index (VIX) three months later. It is a negative relationship which suggests that as central bank sentiment increases, volatility in the FTSE decreases on a three-month lag in keeping with the Bank of England's mandate to promote economic stability. Note that the correlation isn't very strong in absolute terms and that there is a smaller sample size because there is no data for the FTSE VIX after June 2019.

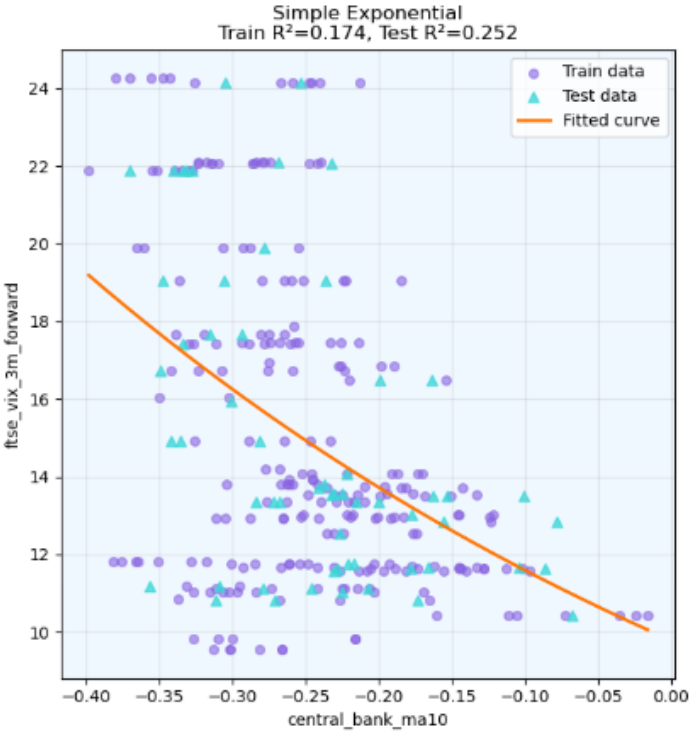
Here are the results of a simple linear regression:



The pattern of the scatter plot slightly resembles an exponential decay curve, but a polynomial regression does not make much of a difference to the R-squared value:



The same is true of a simple exponential nonlinear squares model:



Conclusion

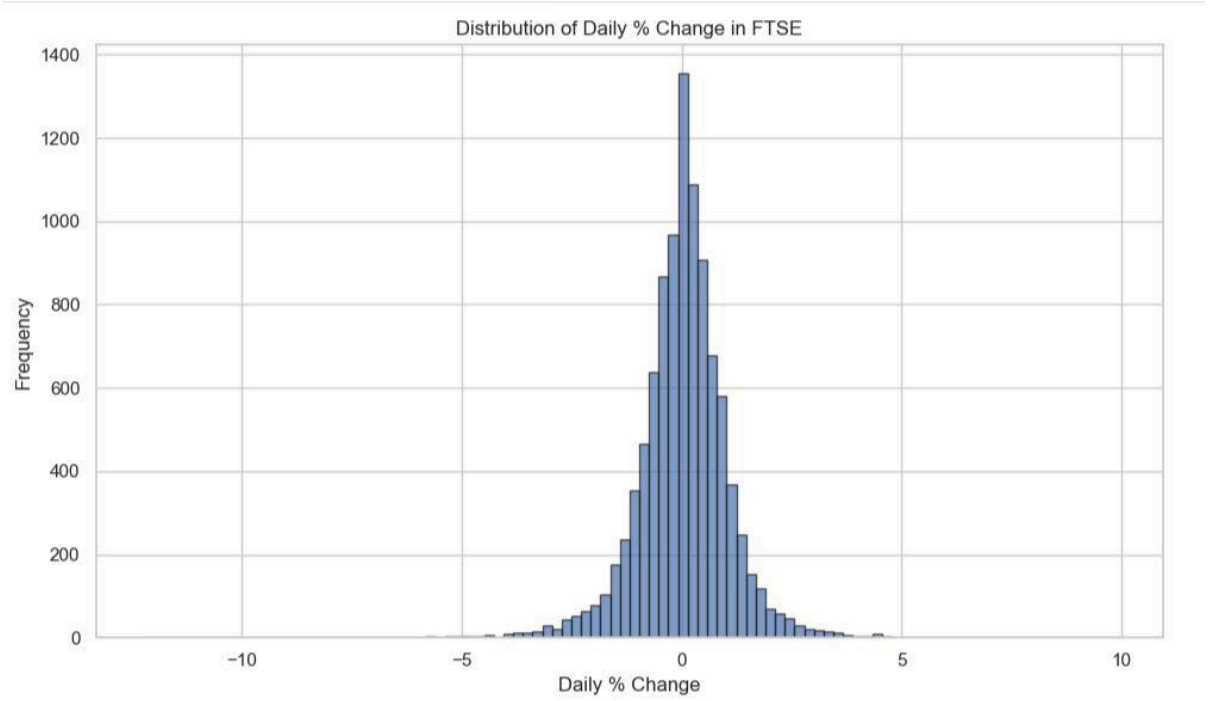
The 10-speech moving average of central bank sentiment appears to predict approximately 25% of the variation in FTSE VIX values 3 months after the month a speech is delivered. Due to the small sample size, however, this result is most likely specific to this particular set of test data and other randomized test sets will yield different (and lower) results. Unsurprisingly, speech sentiment has little predictive power in its own right.

Appendix 12: Finding patterns in FTSE-250

See *FTSE patterns.ipynb*

In this exercise, we explored the relationship between Bank of England speeches and significant movements in the FTSE-250 index, using sentiment analysis as a core tool.

Using FTSE-250 daily data from our cleaned indicators, we calculated daily percentage variation in the FTSE, and Identified days with absolute daily change > 2%



For each extreme FTSE movement day, we found BoE speeches given in the three days prior, and calculate the impact on the FTSE for each speaker

[8]:
```
# Find speeches within 3 days before each extreme FTSE movement
matches = []

for idx, row in extreme_days.iterrows():
    target_date = row['date']
    window_start = target_date - pd.Timedelta(days=3)

    recent_speeches = df[(df['date'] >= window_start) & (df['date'] < target_date)][
    ['date', 'country', 'title', 'author', 'is_gov', 'finbert_score']
    ].copy()

    recent_speeches['ftse_date'] = target_date
    recent_speeches['ftse_change'] = row['daily_pct_change']

    matches.append(recent_speeches)

# Combine and show results
results_df = pd.concat(matches, ignore_index=True)
```

[9]:
```
results_df.head()
```

[9]:

	date	country	title	author	is_gov	finbert_score	ftse_date	ftse_change
0	1998-09-15	UK	Speech	Edward George	1.0	-0.199338	1998-09-17	-3.000565
1	1998-11-01	UK	Economic policy, with and without forecasts	budd	0.0	-0.073561	1998-11-04	2.161933
2	1998-11-01	UK	Inflation targeting in practice: the UK experi...	vickers	0.0	0.000268	1998-11-04	2.161933
3	1998-12-01	UK	Speech	governor	0.0	-0.016364	1998-12-02	-5.766562
4	1999-01-12	UK	Speech	Edward George	1.0	0.442215	1999-01-13	-3.041297

For each of these speakers, we calculated the following metrics:

count: how many of their speeches occurred within 3 days before a large FTSE-250 movement.

avg_impact: the average % change in FTSE following their speeches.

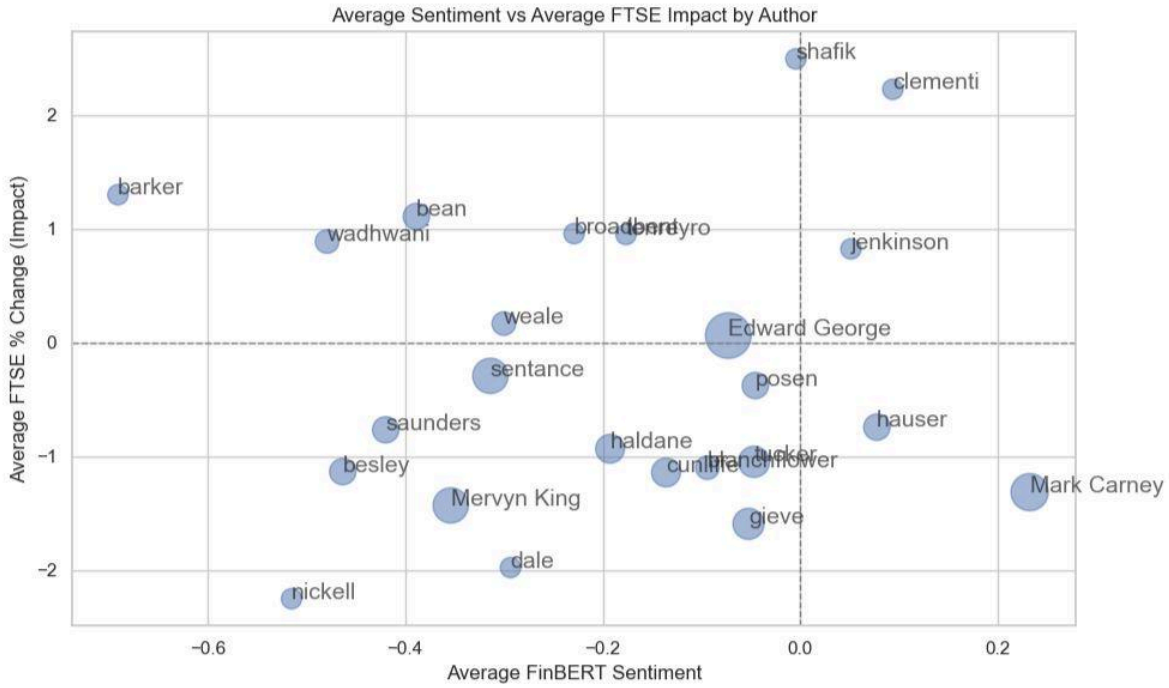
max_impact: the largest positive FTSE movement linked to their speech.

min_impact: the most negative FTSE movement.

std_impact: the variability (standard deviation) of the FTSE responses.

avg_sentiment: average FinBERT sentiment score across those speeches.

With this data, we can plot each speaker's impact on the market vs their sentiment scores (FinBert)



Interesting findings include:

- **Negative sentiment is often aligned with negative FTSE impact** (e.g. Nickell, King).
- Some speakers showed **positive FTSE responses despite neutral or negative sentiment,** suggesting nuanced market interpretation (e.g. Shafik, Broadbent).
- **Edward George** appears frequently in the dataset because many of his speeches occurred within three days prior to large FTSE-250 movements. However, his avg_impact is close to zero, meaning neutral or balanced messaging, as reported by his FinBERT sentiment score.
- Despite having a positive average sentiment score (≈ +0.23), **Carney's speeches** are associated with a strongly negative average FTSE-250 impact (≈ −1.31%). This could be due to uncertainty (e.g. Brexit) or maybe his positive sentiment in speeches might have led to markets projecting rate hikes.

Appendix 13: Global economic events affecting EU & UK (1997 - 2025)

1997 – Asian Financial Crisis (IMF)

- Triggered by the collapse of Thai baht; spread across Asia.
- **Impact**: Weakened global investor confidence, caused financial market volatility in Europe, slight economic slowdown.

1999 – Launch of the Euro (ECB)

- The euro was introduced for electronic transactions.
- **Impact**: Deepened economic integration across EU countries (UK opted out), set the stage for future monetary policy divergence.

2000 – 2002 – Dot-Com Bubble Burst (Wikipedia)

- Collapse of overvalued tech stocks.
- **Impact**: Market downturn in global equities, early 2000s recession affecting EU/UK exports and investment.

2001 – 9/11 Terror Attacks (The Centre for Economic Policy Research (CEPR))

- Shock to the global economy.
- **Impact**: Temporary fall in consumer/business confidence, decline in air travel and tourism, central bank rate cuts including BoE and ECB.

2007–2008 – Global Financial Crisis (ONS) (The Times)

- Triggered by U.S. subprime mortgage collapse.
- **Impact**: Severe recession in the UK and Eurozone. UK government nationalised banks; ECB and BoE slashed interest rates, housing boom peaked around 2007.

2009–2012 – Eurozone Sovereign Debt Crisis (Investopedia)

- Began with Greece's debt problems, spread to Ireland, Portugal, Spain, and Italy.
- **Impact**: EU bailouts, austerity programs, ECB created Outright Monetary Transactions (OMT); UK faced financial contagion risks.

2010 – 2015 Introduction of Austerity program (GOV.UK)

- Austerity in the UK began in 2010 as a government-led effort to reduce the budget deficit primarily through public spending cuts and welfare reform.
- **Impact**: slowed public sector investment, deepened regional inequality, and led to reductions in key services such as social care, policing, and local government.

2016 – Brexit Referendum (Coyle)

- UK voted to leave the EU (23 June 2016).
- **Impact**: GBP dropped sharply, uncertainty over trade and investment, BoE eased monetary policy.

2017 - 2020 - Brexit (Wright and Etherington)

- On 29 March 2017 the UK formally triggered Article 50 of the Treaty on European Union, starting the legal process to leave the EU, ending on 31 December 2020 (including transition period) with the UK's official withdrawal from the EU.
- **Impact**: led to reduced trade openness, slower investment growth, and labor market frictions, ultimately lowering the UK's potential GDP and intensifying inflationary pressures.

2020 – 2022 - COVID-19 Pandemic (ONS)

- Global economic shutdowns.
- **Impact**: EU and UK economies entered sharp recessions; stimulus packages, furlough schemes, ECB and BoE launched massive QE programs.

2021 – Global Supply Chain Crisis (Wilson) (Bolton) (UK Parliament) (Zettelmeyer)

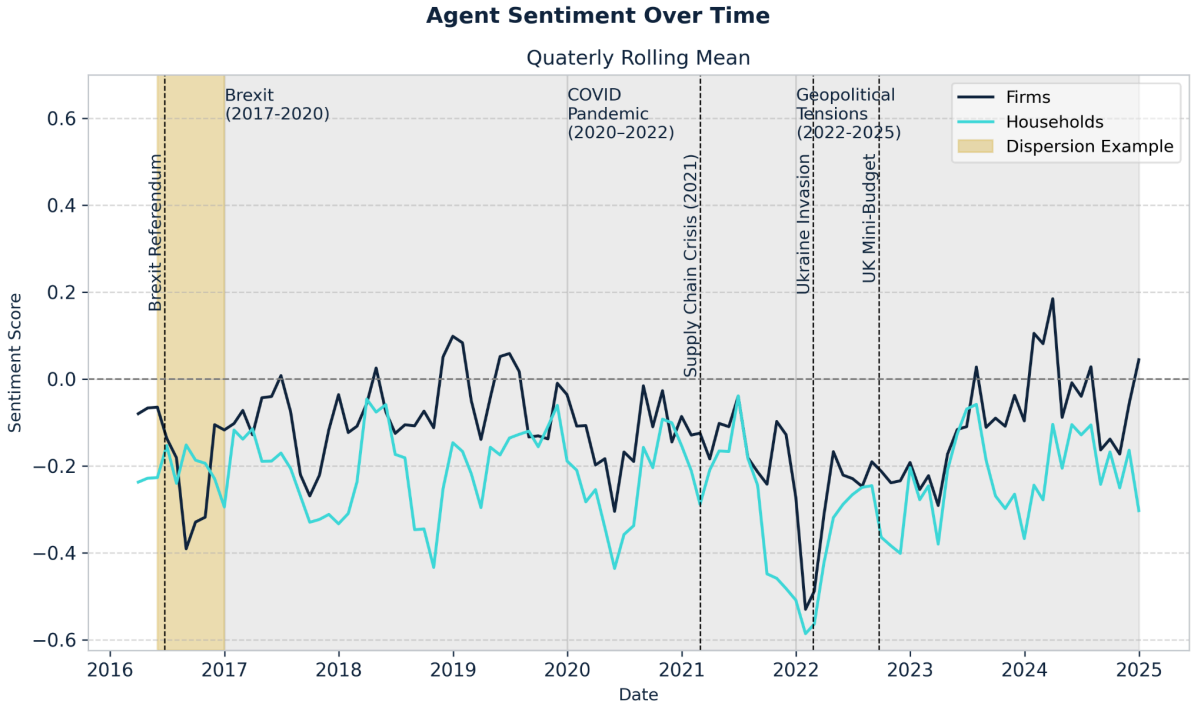
- COVID aftershocks, combined with container shortages and port congestion, A combination of post-pandemic demand surges and geopolitical tensions, notably Russia's invasion of Ukraine, led to unprecedented spikes in energy prices, exposing the UK's and EU's heavy reliance on imported fossil fuels.
- **Impact**: Inflation rose across the UK and EU; energy and input costs surged, straining recovery. The crisis triggered record-high inflation rates, strained household finances, and prompted substantial government interventions, leading to economic slowdowns and increased fiscal pressures across both regions.

2022 – Russia's Invasion of Ukraine (Kilfoyle). UK Mini-Budget (Thompson 2022)

- Full-scale invasion began 24 February 2022. Liz Truss's economic policy and the September 2022 "mini-budget" had significant repercussions for the UK economy and reverberated across the EU.
- **Impact**: Energy price shock in Europe (due to reliance on Russian gas), food inflation, refugee crisis, accelerated inflation and ECB/BoE policy tightening. Mini-budget: The announcement triggered a sharp decline in the British pound, a surge in government bond yields, and turmoil in pension markets.

2022 - 2025 - Geopolitical Tensions (Dolan) ("Europe saw stronger growth at start of year, but Trump's tariffs have darkened outlook") (CER)

- The UK's FTSE 100 index achieved a record 15-day winning streak, and the British pound reached a three-year high against the U.S. dollar, reflecting renewed investor confidence. In early 2025, the U.S. administration imposed sweeping tariffs, including a 20% levy on EU goods, exacerbating trade tensions. The UK is negotiating access to the EU's €150 billion Security Action For Europe (Safe) fund, with conditions including financial contributions and regulatory compliance. The IMF highlighted that an aging population is contributing to a slowdown in economic growth in advanced economies, including the UK and EU.
- **Impact**: This market rally indicates a positive shift in the UK's economic outlook, despite previous challenges related to Brexit and global trade tensions. Tariffs disrupted European exports, particularly affecting Germany's economy, which revised its 2025 growth forecast to zero. These negotiations aim to enhance UK-EU defense cooperation and could influence future trade relations and economic alignment. Labor shortages and increased healthcare costs are straining public finances and reducing economic productivity.



Note

The style of this document is based on the Bank of England logo and website guidelines (Bank of England, 2022).

References and bibliography

"About Gilts." n.d. dmo.gov.uk. On. Accessed May 4, 2025. https://www.dmo.gov.uk/responsibilities/gilt-market/about-gilts/.

Bank of England. 2022. "Our logo and website have a new look." https://www.bankofengland.co.uk/news/2022/march/new-visual-identity.

Bank of England. n.d. "The Bank of England publishes daily estimated yield curves for the UK." Bank of England. Accessed May 04, 2025. https://www.bankofengland.co.uk/statistics/yield-curves.

Bank of England. n.d. "Governors." Bank of England. Accessed April 30, 2025. https://www.bankofengland.co.uk/about/people/governors.

Bolton, Paul. 2025. "Gas and electricity prices during the 'energy crisis' and beyond." The House of Commons Library. https://commonslibrary.parliament.uk/research-briefings/cbp-9714/.

The Centre for Economic Policy Research (CEPR). n.d. "British voting intentions and the far reach of 9/11 terror in New York." CEPR. Accessed May 1, 2025. https://cepr.org/voxeu/columns/british-voting-intentions-and-far-reach-911-terror-new-york.

CER. n.d. "The geopolitical EU." Accessed May 03, 2025. https://www.cer.eu/hot-topics/geopolitical-eu.

Coyle, Christopher. 2021. “How has Brexit affected the value of sterling?” Economics Observatory. https://www.economicsobservatory.com/how-has-brexit-affected-the-value-of-sterling.

Dolan, Mike. 2025. “Sterling sneaks higher as UK assets rally.” Reuters. https://www.reuters.com/markets/europe/sterling-sneaks-higher-uk-assets-rally-mike-dolan-2025-04-30/.

Dolan, Simon, Tom Horn, Swanand Kant, and Rhys Phillips. 2022. “QE at the Bank of England: a perspective on its functioning and effectiveness.” Bank of England. https://www.bankofengland.co.uk/quarterly-bulletin/2022/2022-q1/qe-at-the-bank-of-england-a-perspective-on-its-functioning-and-effectiveness.

ECB. n.d. “The euro: the birth of a new currency.” European Central Bank. Accessed May 1, 2025. https://www.ecb.europa.eu/press/key/date/1999/html/sp990521.en.html.

“Europe saw stronger growth at start of year, but Trump's tariffs have darkened outlook.” 2025. The Independent. https://www.independent.co.uk/news/europe-donald-trump-frankfurt-germany-eurostat-b2742142.html.

GOV.UK. 2013. 2010 to 2015 government policy: deficit reduction. https://www.gov.uk/government/publications/2010-to-2015-government-policy-deficit-reduction#:~:text=Reducing%20the%20deficit%20%2D%20the%20gap,before%20the%20financial%20crisis%20began.

House of Lords. n.d. “Making and independent Bank of England work better.” *Economic Affairs Committee*. Accessed April, 2025. https://committees.parliament.uk/publications/42289/documents/210852/default/#:~:text=The%20Bank%20of%20England%20Act%201998%2C%20which,operational%20independence%2C%20was%20passed%2025%20years%20ago.&text=7%20However%2C%20just%20a%20few%20days%20after,of%.

IMF. 1998. https://www.imf.org/external/pubs/ft/fandd/1998/09/imfdirec.htm.

Investopedia. 2024. “European Sovereign Debt Crisis: Eurozone Crisis Causes, Impacts.” Investopedia. https://www.investopedia.com/terms/e/european-sovereign-debt-crisis.asp.

Kilfoyle, Michelle. 2023. “Ukraine: what's the global economic impact of Russia's invasion?” Economics Observatory. https://www.economicsobservatory.com/ukraine-whats-the-global-economic-impact-of-russias-invasion.

Mathur, Ragini, Shashwat Chauhan, Sruthi Shankar, and Mrigank Dhaniwala. 2025. “FTSE 100 flat as investors await speeches from BoE policymakers.” Reuters. https://www.reuters.com/world/uk/uks-ftse-100-starts-week-brighter-note-healthcare-defence-stocks-rise-2025-02-24/.

ONS. 2018. “The 2008 recession 10 years on.” Office for National Statistics. https://www.ons.gov.uk/economy/grossdomesticproductgdp/articles/the2008recession10yearson/2018-04-30.

ONS. n.d. “GDP and events in history: how the COVID-19 pandemic shocked the UK economy.” Office for National Statistics. Accessed May 1, 2025. https://www.ons.gov.uk/economy/grossdomesticproductgdp/articles/gdpandeventsinhistoryhowthecovid19pandemicshockedtheukeconomy/2022-05-24.

Pastorella, M. 2023. Central Banks’ communication strategy and its impact on the financial markets: A comparison of the Bank of England, the US Federal Reserve and the European Central Bank. https://esthinktank.com/2023/08/18/central-banks-communication-strategy-and-its-impact-on-the-financial-markets-a-comparison-of-the-bank-of-england-the-us-federal-reserve-and-the-european-central-bank/

Pfeifer, Moritz, and Vincent P. Marohl. 2023. “CentralBankRoBERTa: A fine-tuned large language model for central bank communications.” *The Journal of Finance and Data Science* 9: 100-114. https://doi.org/10.1016/j.jfds.2023.100114.

Primiceri, Giorgio E. 2005. “Time Varying Structural Vector Autoregressions and Monetary Policy.” *Review of Economic Studies*, 72 (3): 821–852. https://doi.org/10.1111/j.1467-937X.2005.00353.x

Reeves, Rachel, and Michael Sawicki. 2005. “Do financial markets react to Bank of England communication?” *External MPC Unit*, no. 15. https://www.bankofengland.co.uk/-/media/boe/files/external-mpc-discussion-paper/2005/do-financial-markets-react-to-boe-communication.pdf.

Scott, Gordon. n.d. “Yield Curve: What It Is and How to Use It.” Investopedia. Accessed May 4, 2025. https://www.investopedia.com/terms/y/yieldcurve.asp.

Thompson, Isobel. 2022. “The Liz Truss Era Begins With a Bungled Tax Plan That May Cost UK Conservatives.” Vanity Fair. https://www.vanityfair.com/news/2022/10/liz-truss-era-bungled-tax-plan-cost-uk-conservatives-labour.

The Times. n.d. “How the property market has changed since 2000.” Accessed May 25, 2025. https://www.thetimes.com/life-style/property-home/article/how-housing-market-changed-since-2000-lbcsw5mz2#:~:text=Over%20the%20past%2025%20years,time%20of%20Britpop%20and%20Britney.

UK Parliament. 2025. “Energy Bills Support.” https://publications.parliament.uk/pa/cm5901/cmselect/cmpubacc/511/report.html

Wikipedia. n.d. “Dot-com bubble.” Wikipedia. Accessed May 1, 2025. https://en.wikipedia.org/wiki/Dot-com_bubble.

Wilson, Joanna. n.d. “The impact of the global supply chain crisis.” Imperial College London. Accessed May 1, 2025. https://www.imperial.ac.uk/stories/global-supply-chain-crisis/.

Wright, Georgina, and Haydon Etherington. n.d. “Brexit transition period.” Institute for Government. Accessed May 3, 2025. https://www.instituteforgovernment.org.uk/article/explainer/brexit-transition-period.

Zettelmeyer, Jeromin. n.d. “Beating the European Energy Crisis.” International Monetary Fund (IMF). Accessed May 3, 2025.

https://www.imf.org/en/Publications/fandd/issues/2022/12/beating-the-european-energy-crisis-Zettelmeyer.