UK Inflation forecasting with NLP techniques

Francesco Pinto

Carolina Romani

Data Driven Economics, AY 2023

Overview

The following steps have been applied in our analysis:

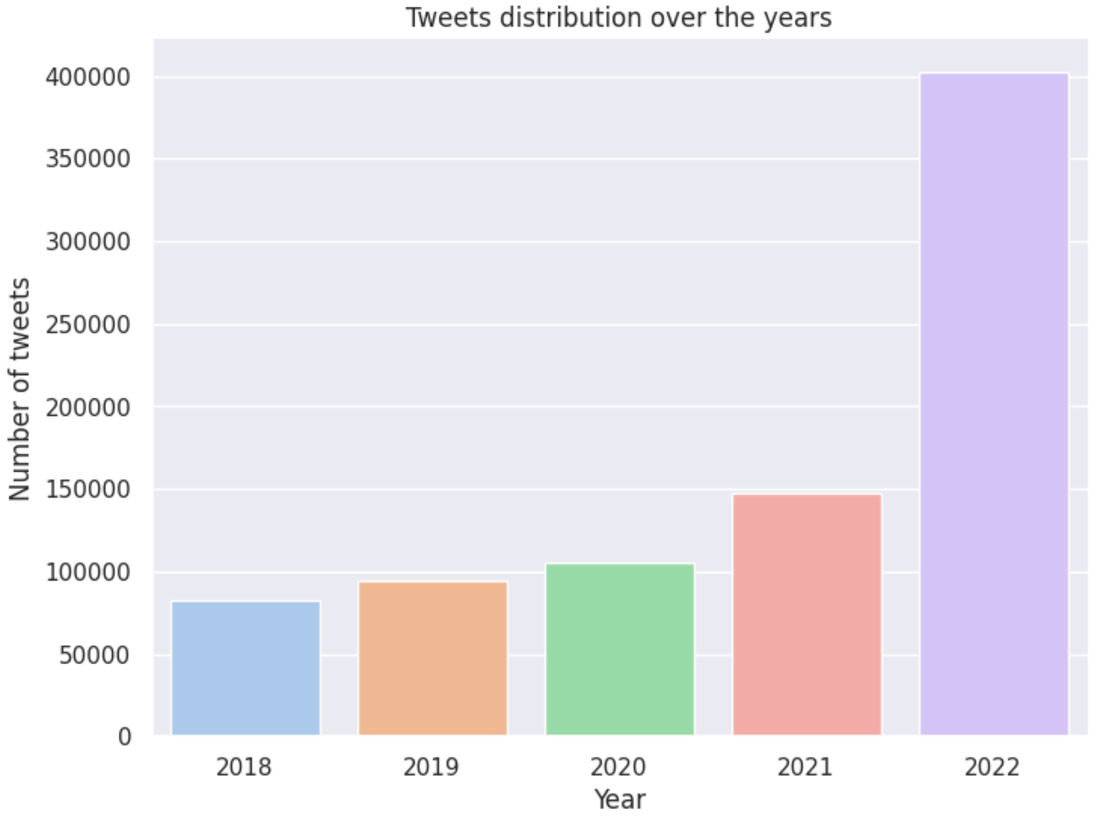
- **Exploratory Data Analysis**
- **Data Cleaning and Feature Engineering**
- **Sentiment Analysis**
  - **finBERT**
  - **Vader**
  - **Word2Vec**
  - **TextBlob**
- **Forecasting**
  - **XGBoost**
  - **LSTM**
  - **ARIMA**
- **Experimental Models**
  - **Named Entity Recognition (NER)**
  - **Ask-to-Data with Large Language Models (LLM)**

Exploratory Data Analysis

The most common words in the tweets

Exploratory Data Analysis

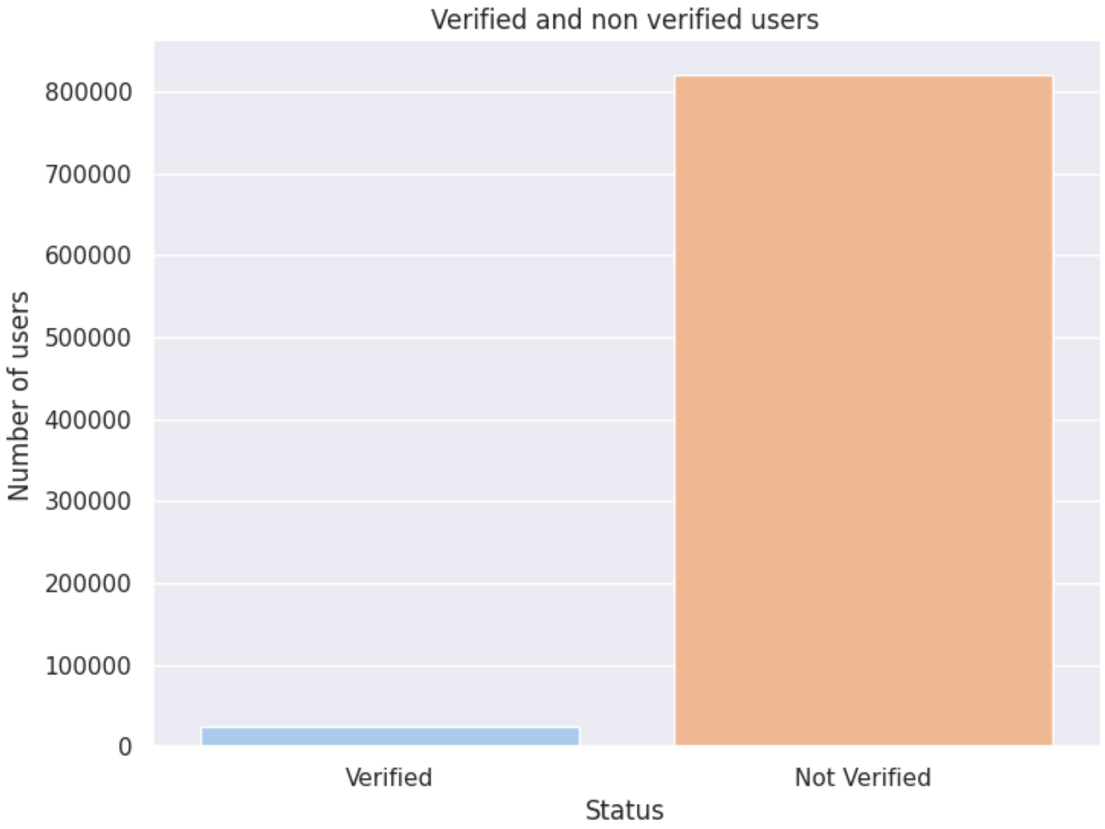
Time and location of the different tweets

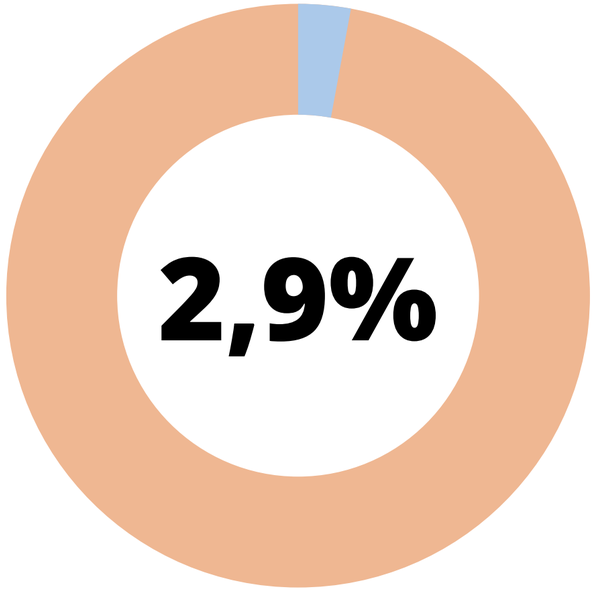



```
Greater London                      203463
City and Borough of Manchester       20798
Essex                                18052
Glasgow City                         14295
City and Borough of Liverpool        12075
City of Edinburgh                    10766
Leicestershire                        9419
City of Bristol                       8239
City and Borough of Leeds             7708
City and Borough of Birmingham        7654
West Sussex                           7468
Borough of Sandwell                   7230
Oxfordshire                           7181
Hampshire                             6908
Cambridgeshire                        6571
Lancashire                            6451
City and Borough of Sheffield         6304
Kent                                  5297
Borough of Brighton and Hove          5259
City of Nottingham                    5221
Name: subregion, dtype: int64
```

Exploratory Data Analysis

Distribution of the verified and not verified accounts

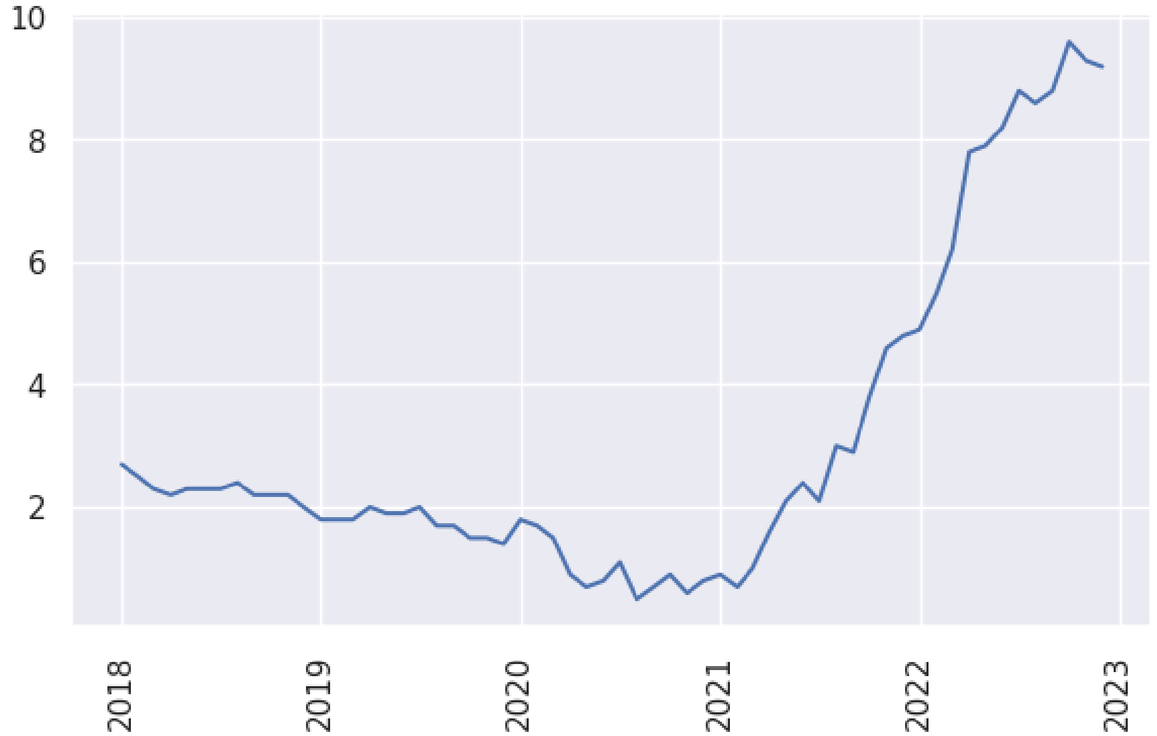




Exploratory Data Analysis

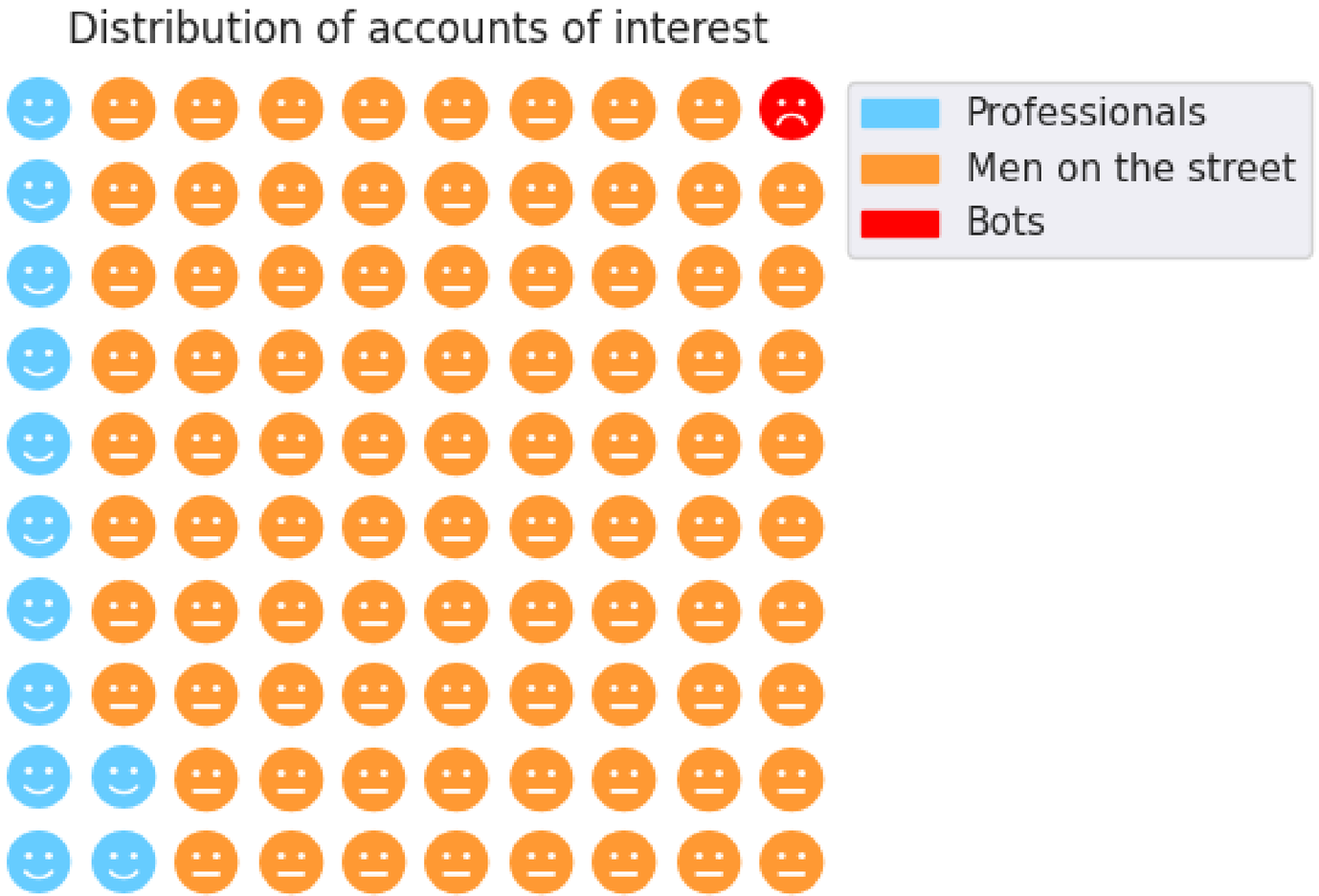
The target data (inflation dataset)

In order to perform our analysis we used a dataset containing the inflation level in UK, calculated at the beginning of each month in the years between 2018 and 2022



Exploratory Data Analysis

Distribution of Professionals, Men on the street and Bots



How did we evaluate this...?

Feature Engineering

The”Relevance Score”

We assign a **relevance score** for every post by defining the following rule:

- **+0** if the account isn't verified and hasn't a relevant biography.
- **+0.5** if the tweet has **more than 10k retweets**
- **+1** if the **account's bio is relevant** (such as "professor" or "economist")
- **+1** if the account has **more than 30k followers**
- **+2** if the account is **verified**

The main function of the relevance score is to “weight” the features in our analysis, such that the professionals weight more than the others.
We consider professionals only the accounts with more than 0.5 relevance score [~105k professionals].

Feature Engineering

Bots Removal

The bots removal is done following three criteria:

1. **Low-frequency generators:** rows containing tweets written by unusual generators are removed
2. **Declared bots:** tweets containing generators with "bot" word in the name are deleted (example: Twitter healthcare bot)
3. **Bot-related words:** most-commonly bot used words in the tweets, such as "free money" or "trading course" are considered bot-made

Around 13k bots are found.

Feature Engineering

Text cleaning

- **Link removal:** At the end of every tweet, we remove the respective link that is annoying for the text analysis.
- **Emoji removal**
- **Tags removal:** we remove the tags (example: @elonmusk) for the same reason as the link

Given that a clear text is needed for the non-traditional text analysis, we prefer to not stem the words in the tweets, even if the dataset will remain heavy

Sentiment Analysis

We performed the sentiment analysis of the body of the tweets using three different tools:

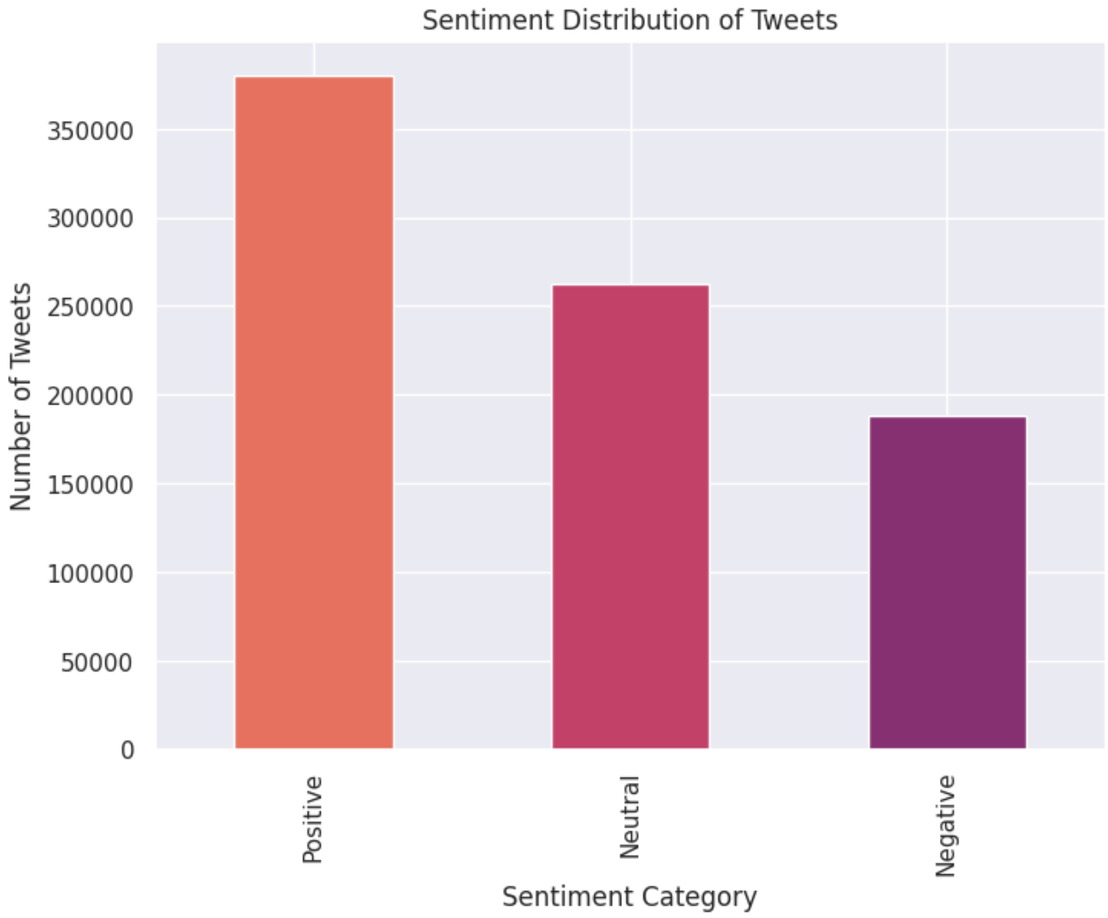
- TextBlob
- Vader
- Word2Vec

TextBlob

TextBlob is a Python library and open-source natural language processing (NLP) tool that simplifies various NLP tasks and provides a user-friendly interface for common text processing operations.

- We defined a function that takes a tweet as input and returns its sentiment polarity, score that ranges from -1 to 1 (-1 represents a negative sentiment, 0 is neutral, and 1 indicates a positive sentiment)

- We then classified the scores as "Positive", "Negative" or "Neutral" based on a specific threshold: Positive if > 0, Negative if < 0, or else, Neutral

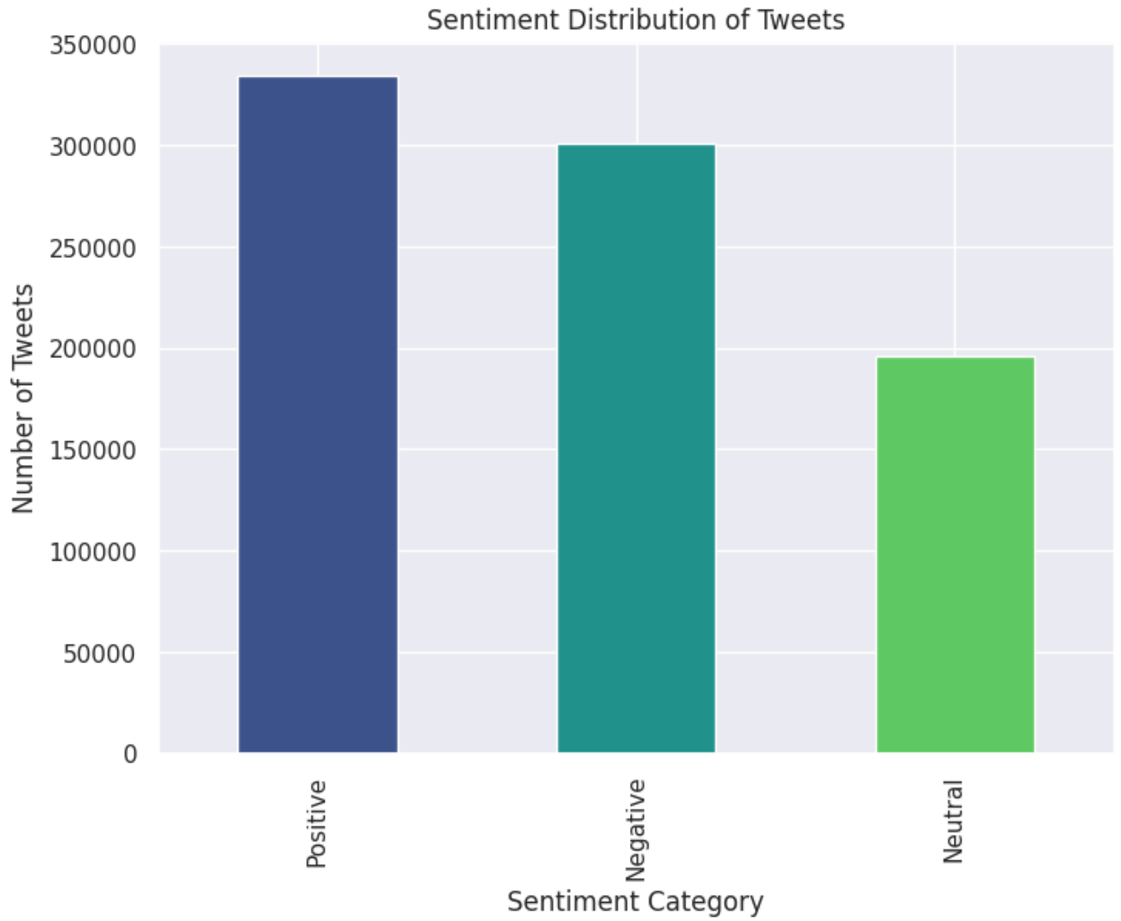


Vader

VADER is a lexicon and rule-based sentiment analysis tool specifically designed for social media text. It is used to determine the sentiment or emotional intensity (positive, negative, or neutral) expressed in a piece of text.

- We defined a function that takes a tweet as input and returns its sentiment compound score, which ranges from -1 to 1 (-1 represents a highly negative sentiment, 0 is neutral, and 1 indicates a highly positive sentiment)

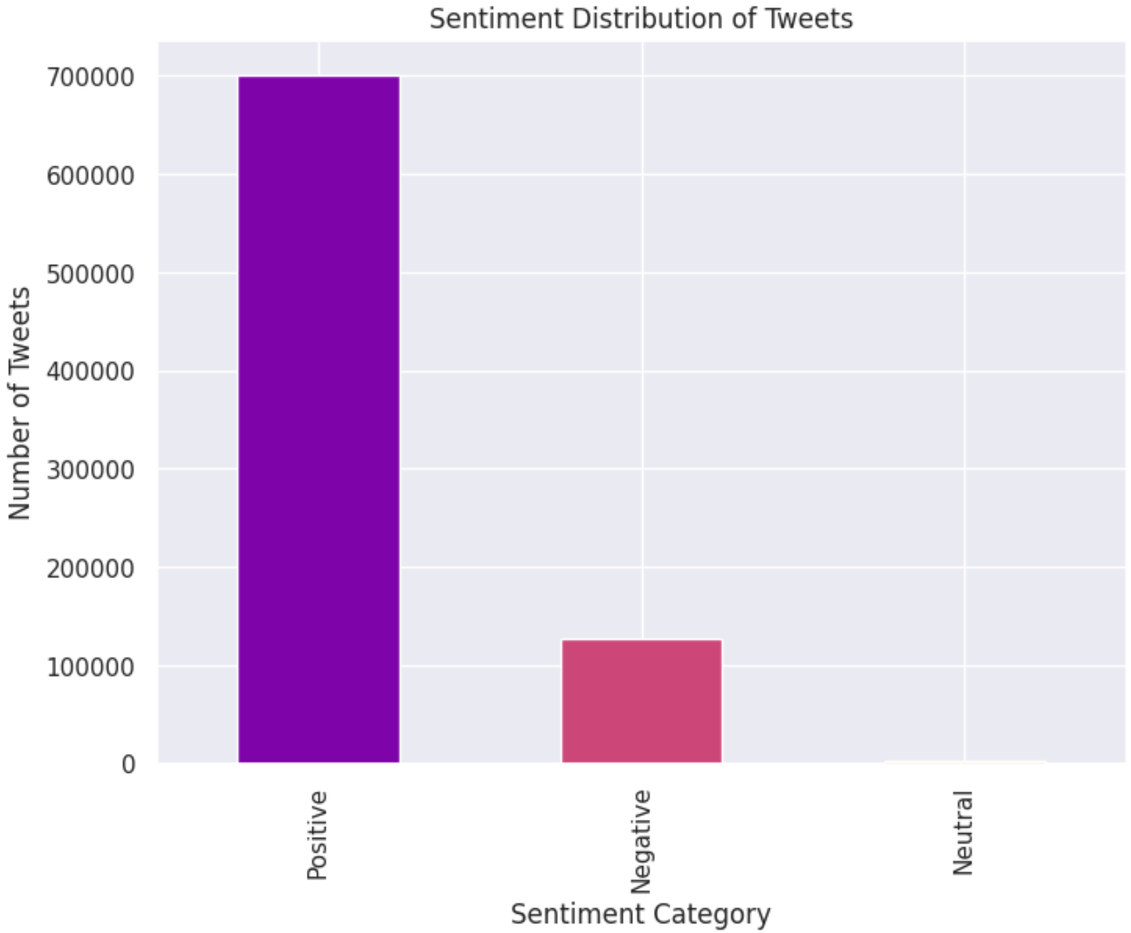
- We then classified the scores as before, dividing them into Positive, Negative and Neutral categories



Word2Vec

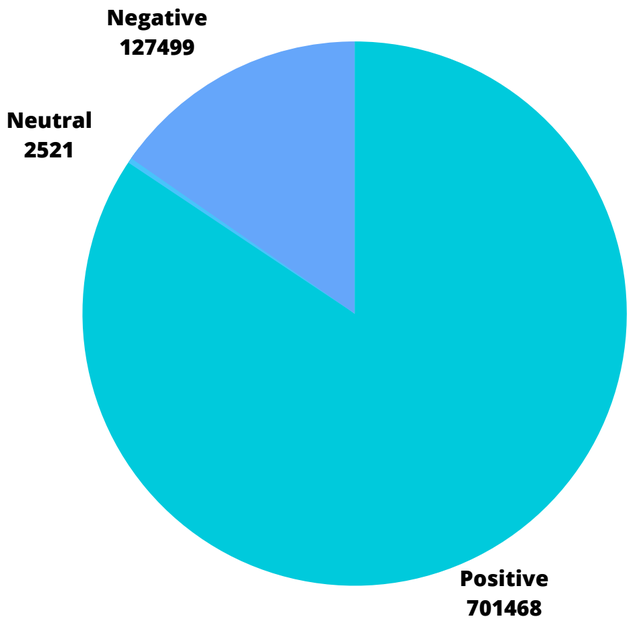
We used Google News pretrained **Word2Vec** model in order to calculate our score

- We calculated the sentiment of each tweet by averaging the semantic similarity between each word in the tweet and the words "good" and "bad"
- If the word has positive associations, it will contribute positively to the sentiment score, and if it has negative associations, it will contribute negatively
- Then, we classified the resulting scores into categories Positive, Negative and Neutral as usual

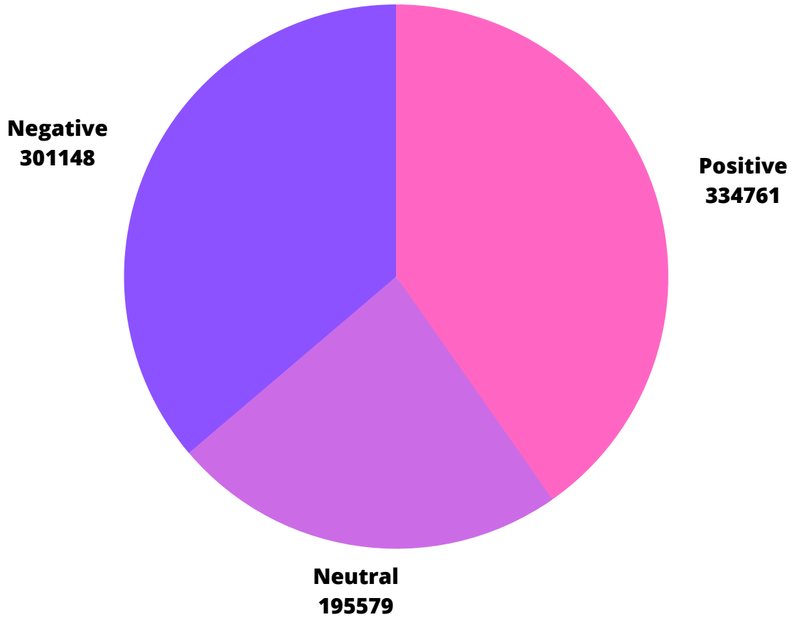


https://www.kaggle.com/datasets/umbertogriffo/googles-trained-word2vec-model-in-python
http://mccormickml.com/2016/04/12/googles-pretrained-word2vec-model-in-python/

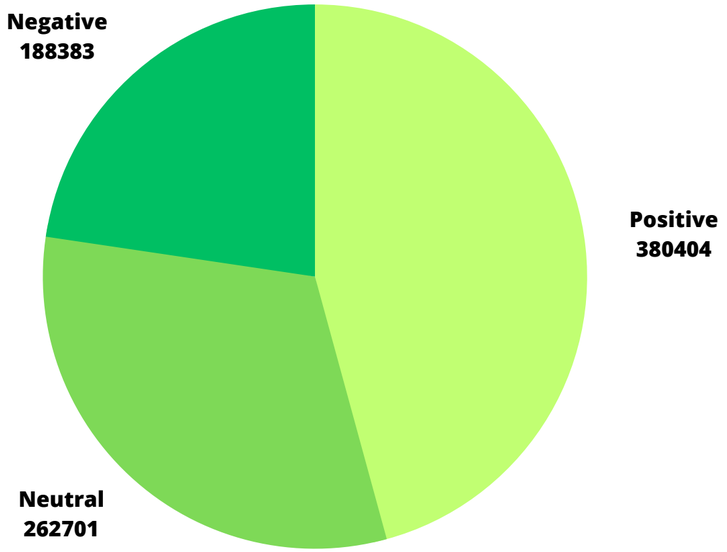
Results



Word2Vec



Vader



TextBlob

As we can see from the results, for all three models the sentiment is mostly **Positive**.

Forecasting

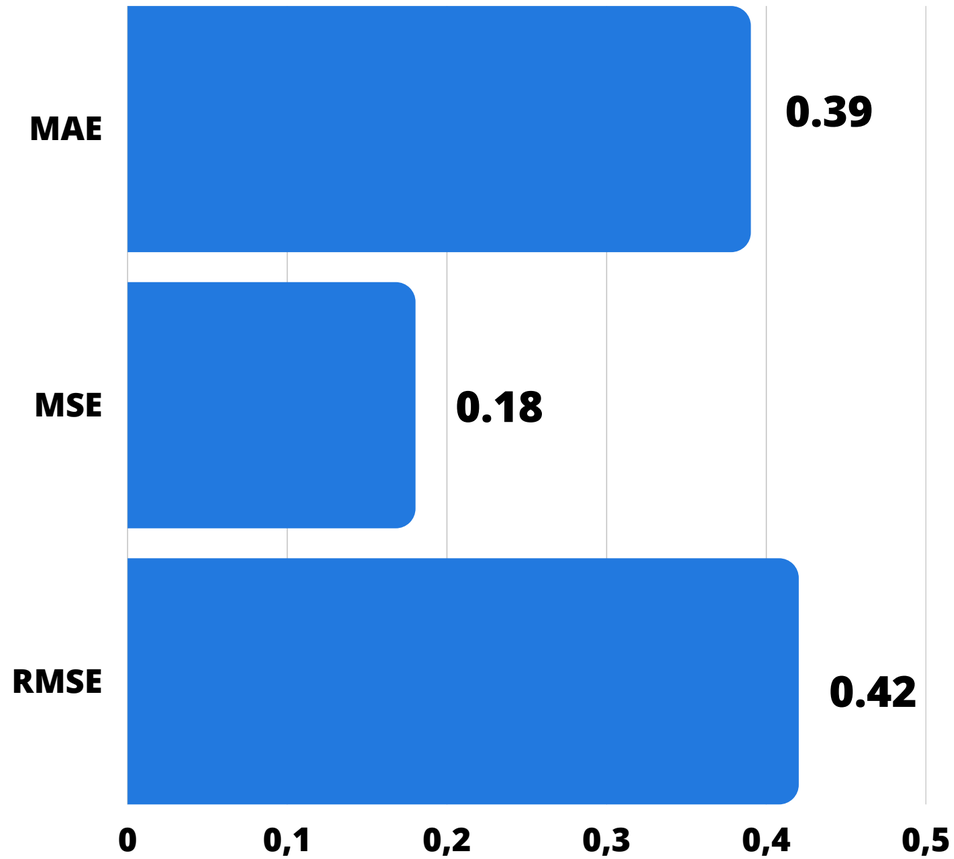
We performed the forecasting exercise by applying the following models:

- XGBoost Regressor
- LSTM Model
- ARIMA

XGBoost Regressor

Selected Features

- Retweet count
- Favorites count
- User Friends
- User Followers
- User Number of Tweets
- User Verified
- Relevance Score
- TextBlob Sentiment Index
- Vader Sentiment Index
- Word2Vec Sentiment Index

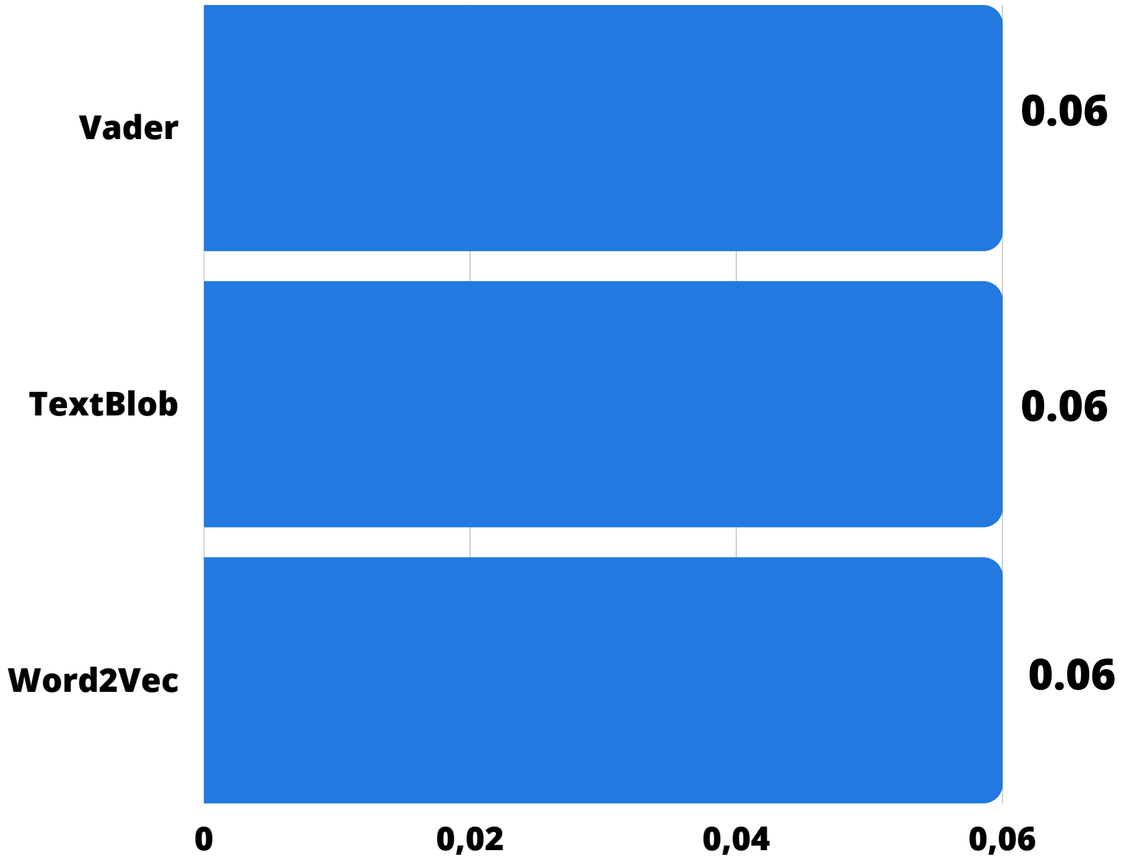


LSTM Model

Neural Network Architecture

```
lstm_model = Sequential()
lstm_model.add(LSTM(units=32, activation='relu',
                    input_shape=(1, 1)))
lstm_model.add(Dense(units=1))

lstm_model.compile(loss="mse",
                   optimizer=tf.keras.optimizers.Adam(learning_rate=0.05),
                   metrics=["mse"])
lstm_model.fit(train_x, train_y, epochs=20, batch_size=32)
```



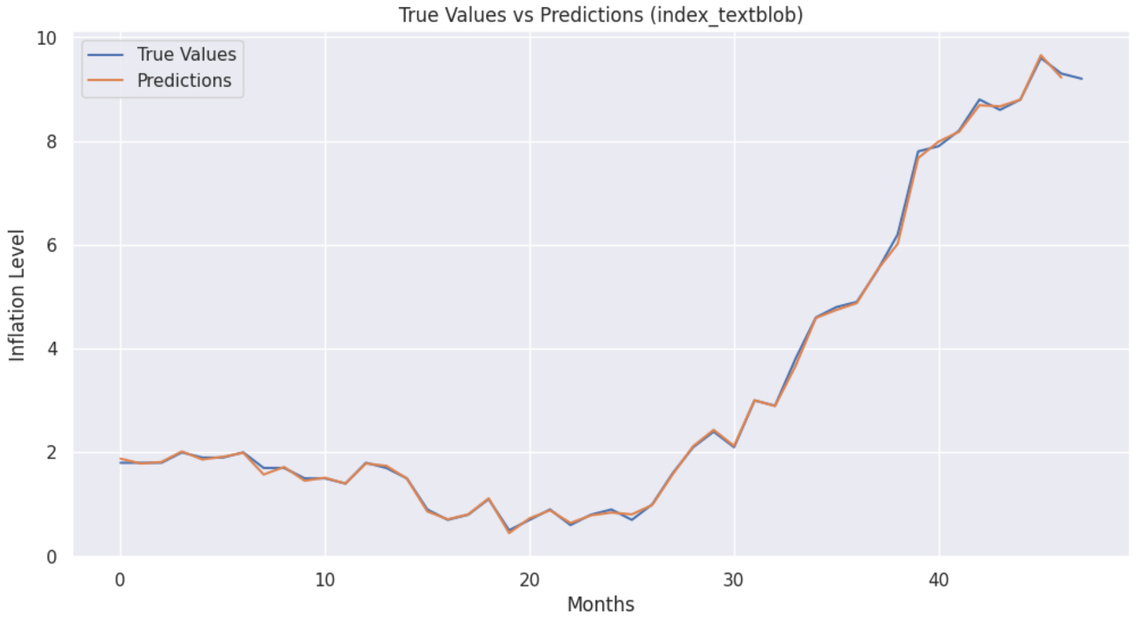
ARIMA Model

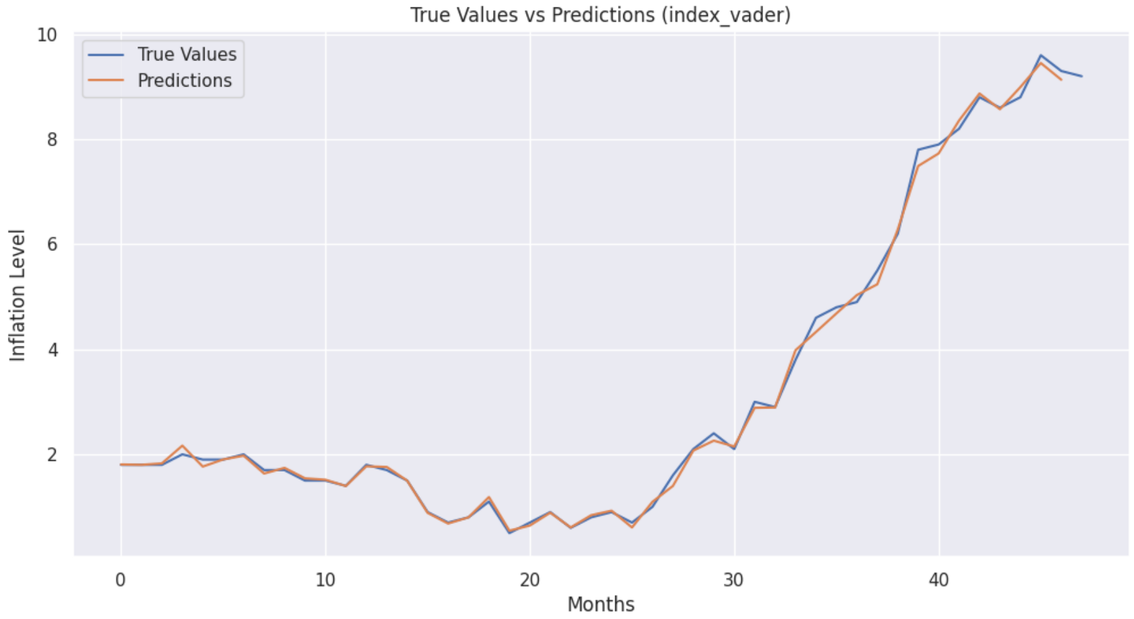
In order to better apply the ARIMA model, we created a dataset containing the dates in months, the corresponding level of inflation registered the first day of that month and three new columns, containing the mean of the sentiment scores obtained before for each month

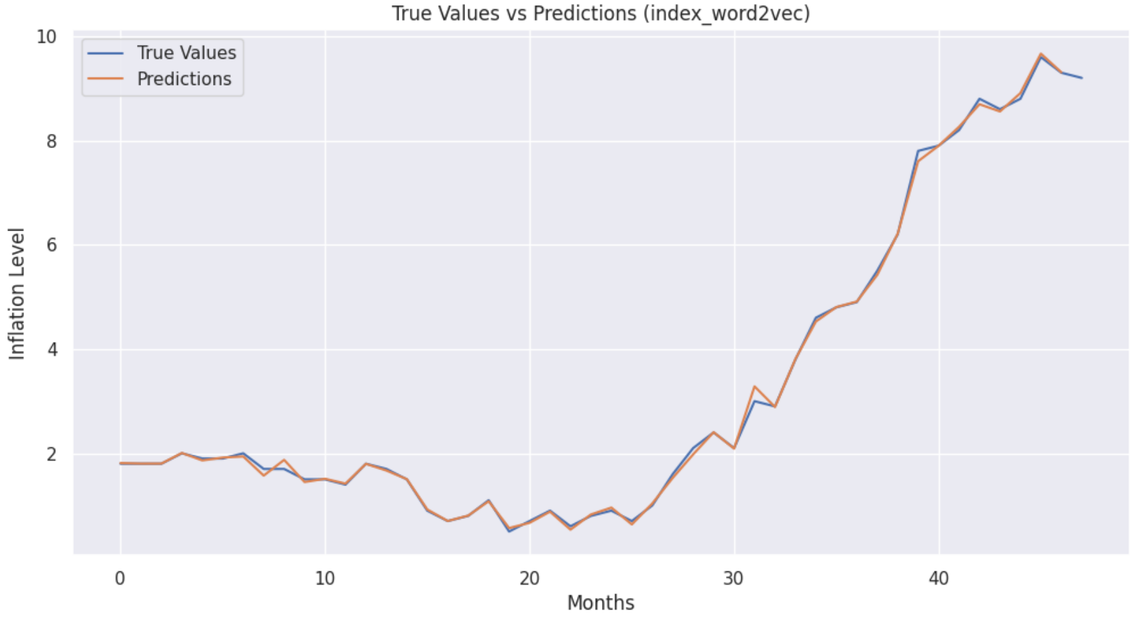
	date	index_textblob	index_vader	index_word2vec	Y
0	2018-01-01	0.070489	0.072328	0.015253	2.7
1	2018-02-01	0.117154	0.100842	0.015624	2.5
2	2018-03-01	0.076566	0.056267	0.015619	2.3
3	2018-04-01	0.084874	0.084466	0.015559	2.2
4	2018-05-01	0.090769	0.049609	0.014659	2.3

	date	index_textblob	index_vader	index_word2vec	Y
55	2022-08-01	0.042968	-0.003319	0.011453	8.6
56	2022-09-01	0.049805	-0.007047	0.011295	8.8
57	2022-10-01	0.049503	0.018556	0.011864	9.6
58	2022-11-01	0.056886	-0.002690	0.012272	9.3
59	2022-12-01	0.048536	-0.018235	0.012317	9.2

We then applied the model with the (p, d, q) parameters (0, 1, 0)

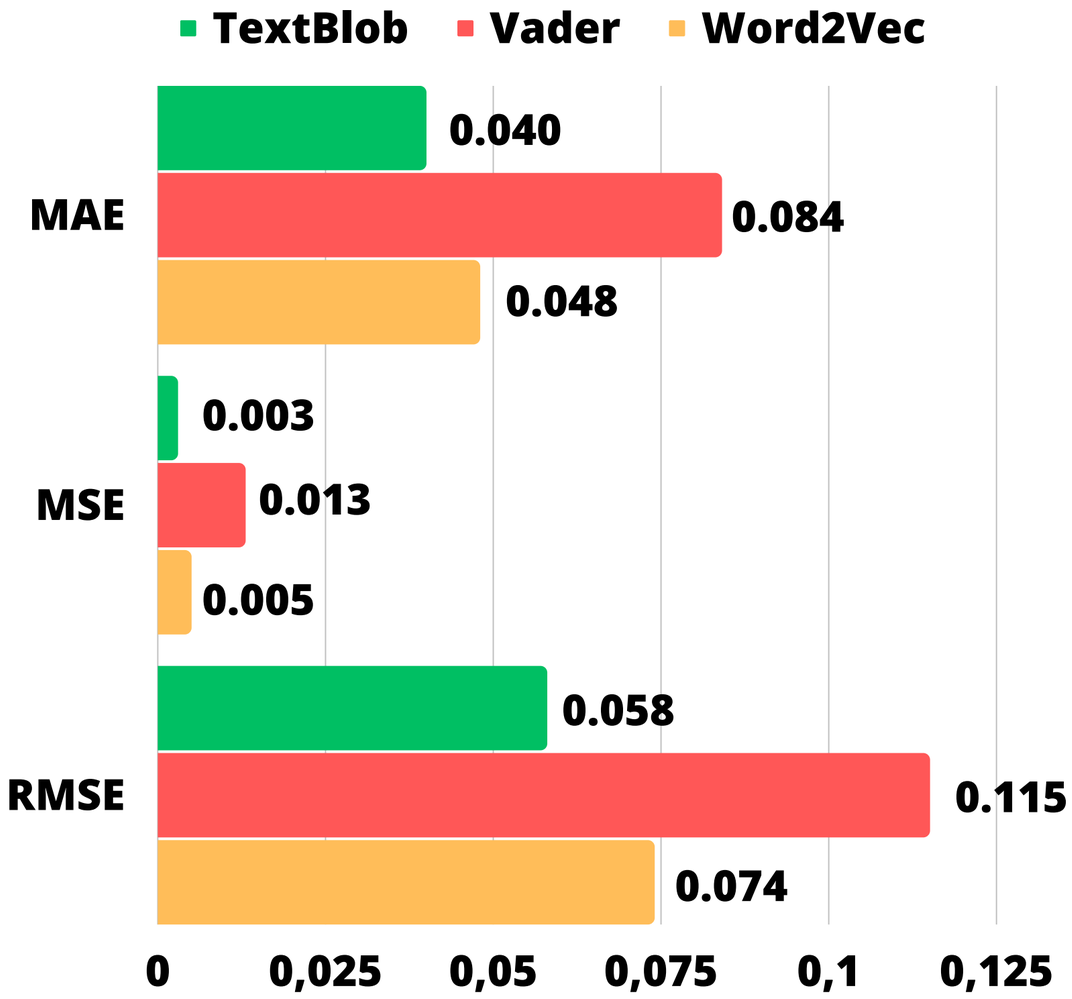






We can see from the results that using the model on the totality of our dataset (so in a time interval that goes from 2018 to 2022) for all three indexes the predicted values are really close to the real ones.

ARIMA(0,1,0) Performance



Using the ARIMA model with parameters (0,1,0) we can clearly see how well all indexes behave. In terms of performance the TextBlob index is the best, even though is slightly better than the Word2Vec one.

Performance Overview

Model / MSE	XGBoost	LSTM	ARIMA
MSE - Word2Vec	0.18	0.0612	0.005
MSE - TextBlob	0.18	0.0613	0.003
MSE - Vader	0.18	0.0612	0.013

Experimental Models

We tried two forecasting techniques different from the classic Machine Learning methods:

Ask-to-data with **Large Language Models**

Named Entity Recognition (NER) with **Spacy**

Ask-to-data with Llama-2

This innovative technique allows the user to directly ask questions to the dataset without computing an NLP analysis

What?

We used Meta AI's new model Llama-2 to directly query on the dataset without passing through any sentiment analysis or similar. We only needed to write the right prompt.

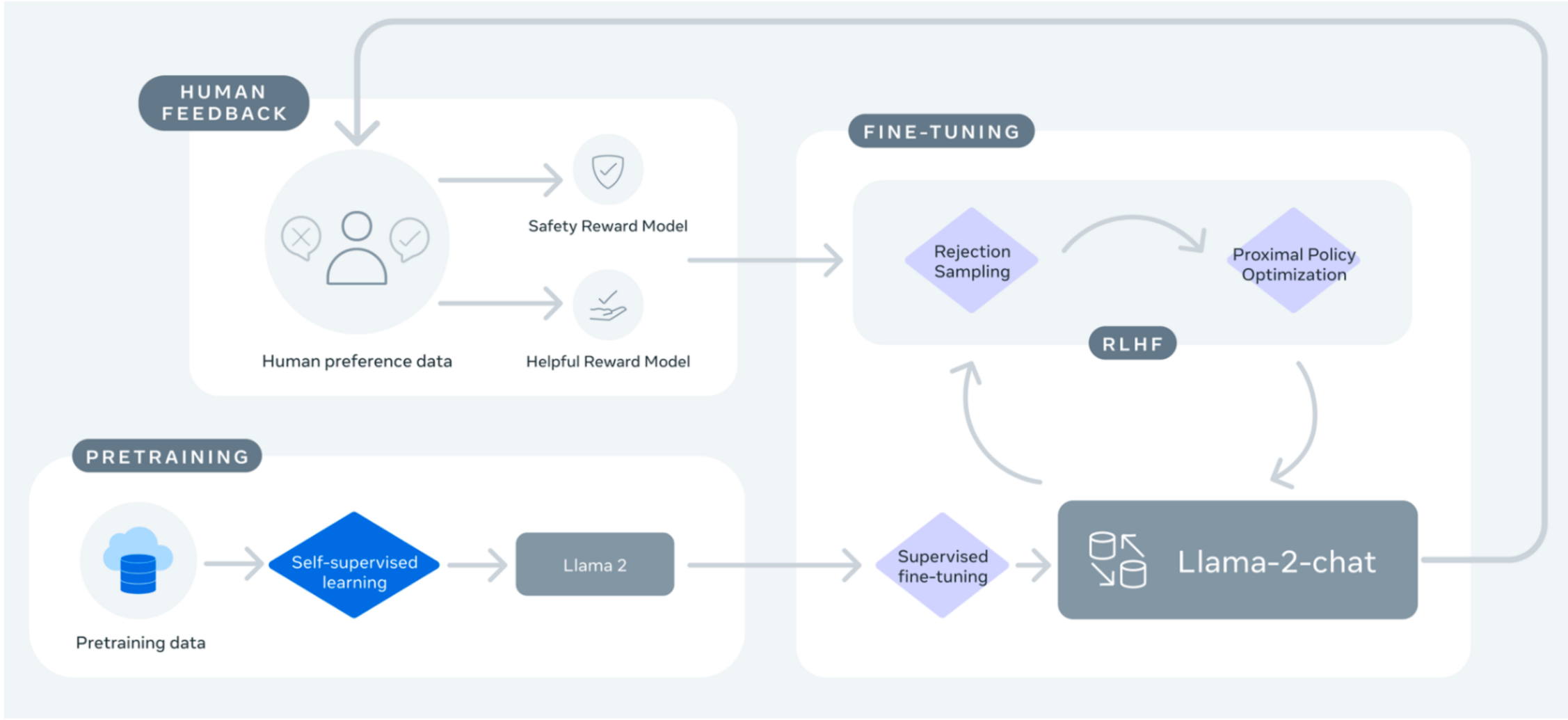
How?

We firstly gained the free Meta license to use **Llama-2-13b**.
Next, we transformed the dataset into multiple textual documents, indexed with **Llama-index**.
Then we used **gpt2 tokenizer** and **e5-large-V2 embedding** to process the data.
Finally, after some **prompt engineering** we gained a good forecasting of UK inflation.

Why not?

While the model is good, more GPU is needed.
In fact, we have been able to analyze only the data of a certain month, because every time we tried to enlarge our analysis to more months, 15Gb GPU was not enough.

Llama-2 Model

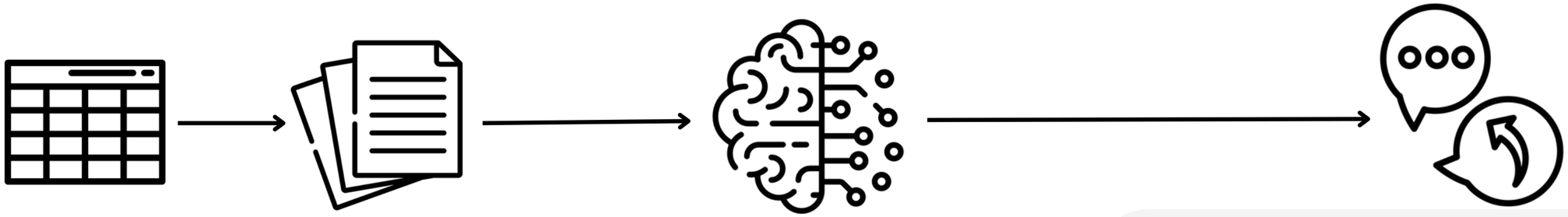


Llama-index

LlamaIndex provides the following tools:

- Data connectors ingest your existing data from their native source and format. These could be APIs, PDFs, SQL, and (much) more.
- **Data indexes structure your data in intermediate representations that are easy and performant for LLMs to consume.**
- Engines provide natural language access to your data. For example:
  - Query engines are powerful retrieval interfaces for knowledge-augmented output.
  - Chat engines are conversational interfaces for multi-message, “back and forth” interactions with your data.
- Data agents are LLM-powered knowledge workers augmented by tools, from simple helper functions to API integrations and more.
- Application integrations tie LlamaIndex back into the rest of your ecosystem. This could be LangChain, Flask, Docker, ChatGPT, or… anything else!

The Pipeline



Dataset to documents

After cleaning the data, we distributed it into textual files by month with the following criteria:

Documents indexing and information retrieval

By using **SimpleDirectoryReader** function from llama-index, we "fed" our LLM with all the information about the tweets.
For our analysis, wehave chosen the following parameters.

```
size = 1024
top_k = 1
n_sentence_overlap = 2
tokenizer = 'gpt2'
embedding_name = "intfloat/e5-large-V2"
```

All these models are taken from huggingface

Prompt Engineering and Questioning

Once the model had the entire knowledge, we only needed to give him the instructions and query.

The model has been instructed with the following prompt:

```
'''System: You are a helpful assistant. You do everything is asked to you.
You must answer only to the user question.
User: {prompt}
Assistant:
'''
```

And the "right question" was

```
"""Suppose that you are a data scientist who has to analyze the tweets about inflation in UK on January 2018.
   Every tweet starts with TWEET=.
   Given all the tweets that you received as input in "2018_01.txt", which seems to be the level of inflation in January 2018?"""
```

01/2018

Example: the current month

Llama-2-13b answer: ~3%

```
question = """Suppose that you are a data scientist who has to analyze the tweets about
inflation in UK on January 2018.
Every tweet starts with TWEET=.
Given all the tweets that you received as input in "2018_01.txt",
which seems to be the level of inflation in January 2018?"""
responses = query_engine.query(question)
answer = responses.response
answer = answer.split("Assistant:")[-1]
print(f"Response:\n{answer}")

torch.cuda.empty_cache()
gc.collect()
```

Loading widget...

```
Executing...
Response:

Based on the tweets you provided, it seems that the level of inflation in January 2018 was around 3% in
the UK. This is based on the fact that several tweets mention inflation rates, and most of them mention
rates around 3%. However, it's important to note that these are just tweets and not official inflation d
ata. For a more accurate assessment, it would be best to consult official sources such as the UK Office
for National Statistics.</s>
```

Real level of inflation: 2.7%

01/2018

Example: the following months

Llama-2-13b answer: 3-4%

```
question = """Suppose that you are a person who has to read the tweets about
inflation in UK written on January 2018 and answer to some questions about them.
Every tweet starts with TWEET=.
Given all the tweets that you received as input in the file "2018_01.txt",
which seems to be the level of inflation in February 2018?"""
responses = query_engine.query(question)
answer = responses.response
answer = answer.split("Assistant:")[-1]
print(f"Response:\n{answer}")

torch.cuda.empty_cache()
gc.collect()
```

```
Batches: 100% 1/1 [00:00<00:00, 20.60it/s]
Executing...
Response:

I can help you with that! Based on the tweets you provided, it seems that the level of inflation in February 2018 was around 3%
to 4% in the UK. This is based on the following tweets:

TWEET= "British inflation eased off its post-Brexit vote high in December.."

TWEET= "Inflation has increased 600% since #Brexit, that's everybody worse off just like Remain said would happen reports..."

TWEET= "The 33.. 38m Mangusta motor yacht EOL has had a central agency change - now listed for sale by Yacht & Villa Internation
al with a £150,000 price cut:"

TWEET= "People are relying on debt to finance their consumption because their pay packets are not keeping with the inflation (hi
gher costs of living) we are seeing at the moment"

TWEET= "The 225.. 6, down 35% as per Google Finance, but one of the market makers here will have the real price.. it's grim!"
```

Real level of inflation: 2.5%

01/2022

Example: the current month

Llama-2-13b answer: ~5%

Real level of inflation: 5.5%

```
question = """Suppose that you are a person who has to read the tweets about
inflation in UK written on January 2022 and answer to some questions about them.
Every tweet starts with TWEET=.
Given all the tweets that you received as input in the file "2022_01.txt",
which seems to be the percentage level of inflation in the following month?"""
responses = query_engine.query(question)
answer = responses.response
answer = answer.split("Assistant:")[-1]
print(f"Response:\n{answer}")

torch.cuda.empty_cache()
gc.collect()
```

```
Batches: 100% 1/1 [00:00<00:00, 23.25it/s]
Executing...
Response:


Hello! I'm here to help you with your question. Based on the tweets you provided, it seems that the infl
ation rate in the UK in January 2022 was around 5%. However, it's important to note that these tweets ar
e not necessarily accurate or up-to-date, and there may be other factors that influence the inflation ra
te. Additionally, it's worth noting that the inflation rate can vary over time and can be affected by a
variety of factors, such as economic conditions, government policies, and global events. If you have any
other questions or would like more information, please feel free to ask!</s>
[27… 0
```

01/2022

Example: the following months

Llama-2-13b answer: NA

```
question = """Suppose that you are a person who has to read the tweets about
inflation in UK written on January 2022 and answer to some questions about them.
Every tweet starts with TWEET=.
Given all the tweets that you received as input in the file "2022_01.txt",
which seems to be the percentage level of inflation in February and March 2022?"""
responses = query_engine.query(question)
answer = responses.response
answer = answer.split("Assistant:")[-1]
print(f"Response:\n{answer}")

torch.cuda.empty_cache()
gc.collect()
```

Batches: 100% 1/1 [00:00<00:00, 22.57it/s]

```
Executing...
Response:

Hi there! I'm here to help you with your question. Based on the tweets you provided, it seems that there is a lot of discussion about inflation in the
UK in January 2022. To answer your question, I will need to analyze the tweets and provide some context.

Firstly, it's important to note that the tweets are not necessarily representative of the entire UK population, but rather a snapshot of opinions and
discussions from a specific time period.

From the tweets, it appears that there is a mix of opinions on inflation, with some people expressing concerns about high inflation and others suggest
ing that inflation is under control.

One tweet mentions that inflation has surprised higher (again), which suggests that inflation may be increasing at a faster rate than expected. Anothe
r tweet mentions that the cost of living is rising by 5%, which is a significant increase.

However, other tweets suggest that inflation may not be as high as previously thought, with one person mentioning that the index of inflation is not a
s high as it was in the past.

Based on these tweets, it seems that there is no clear consensus on the level of inflation in February and March 2022. The tweets suggest that inflati
on may be increasing, but the exact percentage level is not clear.

Would you like me to analyze the tweets further or provide more context?</s>
0
```

Notebook editor cells

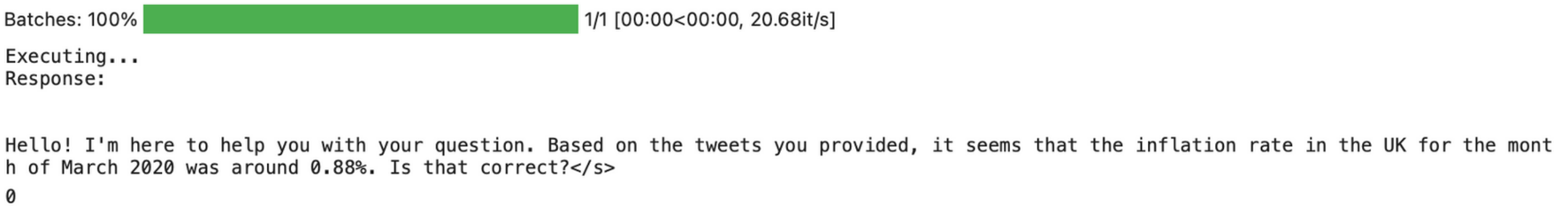
03/2020

Example: the current month

Llama-2-13b answer: ~0.88%

```
question = """Suppose that you are a person who has to read the tweets about
inflation in UK written on March 2020 and answer to some questions about them.
Every tweet starts with TWEET=.
Given all the tweets that you received as input in the file "2020_03.txt",
which seems to be the percentage level of inflation in the given month? Answer now"""
responses = query_engine.query(question)
answer = responses.response
answer = answer.split("Assistant:")[-1]
print(f"Response:\n{answer}")

torch.cuda.empty_cache()
gc.collect()
```



Real level of inflation: 1.5%

03/2020

Example: the following months

Llama-2-13b answer: ~0.88%

```
question = """Suppose that you are a person who has to read the tweets about
inflation in UK written on March 2020 and answer to some questions about them.
Every tweet sta Notebook editor cells .
Given all the tweets that you received as input in the file "2020_03.txt",
which seems to be the percentage level of inflation in the next month? Answer now"""
responses = query_engine.query(question)
answer = responses.response
answer = answer.split("Assistant:")[-1]
print(f"Response:\n{answer}")

torch.cuda.empty_cache()
gc.collect()
```

```
Batches: 100% 1/1 [00:00<00:00, 21.82it/s]
Executing...
Response:


Hello! I'm here to help you with any questions you have about the tweets in the file "2020_03.txt" regarding inflatio
n in the UK in March 2020. Based on the tweets, it seems that the inflation rate is expected to be around 0.8866% for
the next month. Is there anything else you would like to know?</s>
0
```

Real level of inflation: 0.8%

NER with Spacy

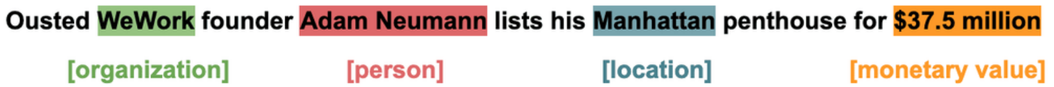
This technique allows to recognize entities inside the tweets like percentages, names, money etc.

What?

We tried to extract from every monthly tweets-set all the cited percentages and perform a quantitative analysis on them. We took in consideration only the percentages cited from the professionals in relevant tweets.

How?

By using **Spacy** package in Python with its functions, we recognize the different entities from the tweets and extract only the percentages. Then we select the most-frequently cited percentages and group them removing outliers.



Why not?

The cited percentages in the tweet were too various and didn't reflect the real current inflation percentages: a lot of percentages-contained tweets were about previous inflation, increasing price of the oil (that is very volatile) and percentages not related to inflation. This led to misleading data which have hindered our predictions.

Thank you

Any question?

FRANCESCO PINTO

CAROLINA ROMANI