Ca' Foscari University of Venice

Master's Degree

in

Erasmus Mundus

Joint Degree QEM

Final Thesis

Inflation perception indicator measurement based on US newspaper articles

Supervisor
Ch. Prof. Marco Corazza

Graduand
Arina Rakhaeva
Matriculation Number 896257

Academic Year
2023 / 2024

Contents

Acknowledgments

This thesis would not be possible without the help and support of Prof. Marco Corazza. His guidance and deep understanding of the topics were essential to getting this thesis to its current state. What is more, his communication skills helped to navigate any difficulties encountered during the preparation of the thesis and any questions that arose.

Moreover, I would like to thank the QEM program administration in both Venice and Paris as they were kind enough to offer support and advice and answer any questions.

Also, huge thanks to my colleagues and classmates for proofreading this thesis, and a separate thank you to my amazing parents, supporting me through every step of the thesis writing journey.

Abstract

Consumers' perception of the inflation rate which is usually measured by surveys, can help improve the accuracy and prediction of already existing indicators. However, conducting regular surveys presents several challenges including constraints on the time and resources needed. This thesis proposes a new inflation perception indicator based on the predictions from the machine learning models that take newspaper articles as input and after some training perform articles classification. It is shown that transformer models tackle this task most efficiently and that this new indicator can be computed continuously with the newspaper data allowing for timely and more accurate inflation rate assessment.

Keywords: Inflation, perceived inflation, inflation rate measurement, machine learning in economics, Bag of Words, Word2Vec, BERT, transformer models in economics.

1. Introduction

Accurate forecasts of macroeconomic variables hold significant implications for enhancing both monetary and public policies, given that numerous economic indicators rely on the nominal values provided by researchers. For example, statistical agencies and researchers would provide policymakers with nominal values for GDP, inflation, or unemployment rate as they represent the raw data that are directly observed and reported. Of particular importance is the prediction of inflation rates, especially in light of recent global inflationary surges, notably within the EU area, which peaked at 11.5 percent in October 2022 according to the European Central Bank (ECB). Accurate inflation rate forecasts are crucial for various stakeholders, including investors seeking to make well-informed decisions, businesses and governments engaged in strategic planning, and consumers adjusting their spending patterns. Consumer expectations regarding future inflation rates can directly influence their spending behaviors, with anticipated price increases prompting higher expenditures and expectations of low inflation leading to deferred purchases. Consequently, policymakers must factor in consumer sentiment when formulating policy adjustments and initiatives. It is important as consumer sentiment is an aggregate expression of behaviors of consumers in a particular market and can be an indicator of future spending (Gartner).

This work focuses on improving inflation rate perception indicators. Consumer inflation perceptions relate to perceived price changes in the past, and consumer inflation expectations refer to expected price changes in the future (ECB). Bank of France states that currently, in Europe, several indicators are used to predict expected inflation: i) expectations from financial markets, measured using inflation-linked bonds or inflation derivatives, ii) expectations of forecasters based on surveys (e.g. the Consensus Forecast or the Survey of Professional Forecasters), iii) expectations of companies (e.g. in France, the new survey of business leaders conducted by the Banque de France) and iv) expectations of households from surveys, for example, the European Commission's consumer survey, or the European Central Bank's Consumer Expectations Survey (CES). However, those indicators can be improved with the estimation of short-term inflation expectations and perceived inflation measurements, providing a more robust perception indicator. Most perception indicators are based on survey data and can not be updated regularly without significant investments. The introduction of a media-based indicator reflecting perceived inflation can help to solve this problem. It can also introduce leads for future periods' expectations.

Existing research indicates that media outlets wield significant influence over consumer expectations (Soroka, 2006). However, the majority of studies in this domain have predominantly focused on social media platforms, particularly Twitter, for text analysis. Yet, these platforms are known for their high volatility and noise levels, indicating a potential for enhanced predictive accuracy through the inclusion of newspaper headlines and articles in analysis frameworks (Angelico et al., 2022).

Although traditional econometric methods and models for predictions of economic phenomena have been widely used, based on the recent research it seems reasonable to expect that combining economic data with machine learning approaches can lead to more accurate results in this area (Athey, 2019). There are multiple solutions available at the moment. However, continuous progress in the Natural Language Processing (NLP) area allows for the improvement of existing algorithms, economic indices, and indicators.

The first part of this thesis offers an overview of the current literature on expected perceived inflation prediction. It outlines consumers' expectations, and behavioral economics theory, and explores the influence of media on shaping these expectations. Following this, a more technical background is provided, covering research on sentiment analysis and several machine learning (ML) models used for text mining and analysis, where text mining (text data mining) is a way of turning unstructured textual data into a structured format to extract actionable insights as stated by IBM.

This work uses New York Times archived articles for the predictions of the perceived inflation. Taking articles’ lead paragraphs and abstracts from the past thirty years allows us to control for major economic shocks and periods of recession. In the next part, the data source and the structure of available information are discussed. After acquiring the necessary articles and data on inflation in the US, articles were labeled as inflationary and non-inflationary for the subsequent model training.

After that, different NLP-based approaches for the implementation of an inflation perception indicator are discussed. Starting with the most common but biased Bag of Words (BoW) approach, Word2Vec embeddings, and state-of-the-art transformer BERT model, with a new layer adapted specifically for the task at hand.

The BoW approach uses a vocabulary created from all the unique words in the dataset to convert each text into a vector. Each element in this vector represents how many times a certain word from the vocabulary appears in the text. Another approach to converting words into vectors is by using pre-classified dictionaries with the terms in a specific area, assigning a score or description to each word in the text.

In this thesis also word embeddings via the Word2Vec model were used. This model also known as the "continuous bag of words" predicts the target word from the surrounding context words. This method is preferred to the ones mentioned before as it captures semantic relationships between words.

Finally, a transformer model was used for the inflation perception predictions. They were introduced in the paper "Attention is all you need" (Vaswani et. al, 2017). This is currently state-of-the-art architecture in NLP-based modeling as these models are better at handling long-range dependencies, like semantic relationships between words and sentences in the text. They are also able to parallelize training more efficiently than previous architectures.

After that, the results of each model's predictions were compared using several tests including accuracy, precision, and F1 score. Accuracy is measured by the proportion of correct predictions to the total number of predictions, precision shows the ratio of the true positives to all positives, and the F1 score is the harmonic mean of precision and recall metrics. Those tests help in spotting the errors and finding the best model configuration.

Comparing the results of several implementations, it is shown that while BOW and Word2Vec models have their advantages, in the inflation perception indicator prediction transformer models are preferred.

2. Literature review

Considerable volume of research has been produced when it comes to inflation. The rises and spikes in prices are an important indicator of economic instability and can affect both monetary and public policies as well as social welfare. However, rational expectations theory does not account for the fact that individuals have limited access to the information available on prices and understanding of the topic may significantly vary.

Perception and expectations of inflation were studied in depth not only by economists but also by researchers in psychology, marketing, business, and other fields. For example, it was pointed out that the research about inflation was still very much fragmented and concentrated on specific topics in an issue on the psychology of inflation (Warneryd, 1986).

In a monumental study by Behrend (1977) with the review of more than ten years of research concentrated on bridging the gap between psychology and economics, it was argued that while the prices and their movements are the dominant frame of reference for the individuals, it is not correct to assume that they apply the same rationality and economic instruments for estimating the prices' movements as experienced economists do. That is especially true in times of rising inflation when the attention is focused on gathering knowledge about specific commodities.

A more recent study (Leiser, Drori 2005) based on the results of the questionnaire given to people from different socio-economic groups found that while the overall social representation was nearly the same across the groups, the understanding of the inflation topic differed severely. The limited understanding of the topic makes it hard for individuals and households to account for inflation in a way perceived by economists in the past. People might not be able to fully understand and correctly interpret how inflation affects their purchasing power, savings, and overall financial well-being. Instead, an individual's decisions might be based on their cognitive constraints and biases as well as information available to them at any given moment. This brings us to the principles of bounded rationality (Kahneman, 2003).

An important factor influencing inflation perception is the amplification of the price movements by the media. Several researchers including Warneryd (1986) argued that individuals not only follow commodities prices directly but also through other channels like the press and media, as well as word of mouth. Soroka (2006) explored the asymmetry in information provided by the media by exploring the time-series analyses of UK media. The results show strong information asymmetry in providing positive and negative news with an incline towards the latter. He also found that media exposure significantly influenced inflation expectations across the demographic.

In another article (Van Raaij, 1988) the influence of mass media was researched from both economic and psychological points of view. The findings show that while there exists a scarcity in the information available to individuals, these pieces of information are often overinterpreted to the point where the expectations and behavior are biased towards the more stable trends. Consider a scenario in which people receive limited information about inflation through the news. Despite the complexity of economic factors affecting inflation, most people are likely to focus on a few positive sentences or headlines and interpret them as an indicator of long-term stability. As a result, they may develop expectations that inflation will remain low and adjust their spending and investments accordingly, potentially missing the possibility of future changes in inflation rates. In more recent studies on perceived inflation, a relationship was found between the media coverage and inflation expectations before and after the launch of the euro currency in the EU. In Italy Giovane and Sabbatini (2006) found that, while the prices indeed increased especially for the frequently purchased commodities, the consumers' perception can be at large explained by the strong upward price trend and the media amplification and increased coverage of the changeover.

In terms of the financial market indicators predictions, there has been substantial research over the past years, concentrated on the forecasts based on stock prices, financial indices, and overall stock data. Saxena (2021) introduced an analysis platform, to help investors in the real-time analysis of stock prices. Generalized linear regression, logistic regression, support vector machine, and random forest showed the best results in terms of precision (proportion of true positive to all positive results as stated by Google) and time necessary to train the model in comparison to other NLP-based algorithms.

There have also been published several papers analyzing textual data from popular news sources. Hagenau, Liebmann, and Neumann (2013) introduced a new approach to analyzing text-based data which allows to select semantically relevant features and prevents model overfitting, this approach has been shown to improve the price stock prediction based on the financial news. The big disadvantage of the method above as well as other statistical methods is in the disregard for the context in which the word is presented. To overcome this drawback NLP methods are largely applied. In terms of transforming words into vectors, there are several options. For example, word embeddings convert words into numerical vectors, they basically help machines to understand the meaning of the word and its relationships to other words in the text so that the words that often appear in the same context would be represented by the vectors located close to each other. The Word2vec model that uses word embedding was introduced in the paper by Mikolov et al. (2013) and consequentially discussed in an analysis by Levy and Goldberg (2014). This model uses neural networks to create word embeddings. The embeddings are determined based on the assumption that words in similar contexts share close meanings. Another example would be the Global Vectors for Word representation - GloVe model (Pennington, Socher, Manning, 2014), it is a global log-bilinear regression model that combines global matrix factorization with local context window methods to capture the semantic patterns using vectors. Simply put, it creates a large table with frequencies of each word co-occurring in the text. Then this table is simplified to create embeddings for each word.

Recently, even more sophisticated models were utilized using deep learning techniques. Deep learning is an area of machine learning concentrated on multi-layered neural network models that are designed to simulate the work of the human brain (IBM). These models are trained on large datasets and can recognize patterns better than the single-layer NN models. An example is a large language model BERT (bidirectional encoder representations from transformers), which utilizes the word embedding models while considering both the left and right context of the word so that the context is also taken into account. However, using an LLM comes with a big drawback of having to acquire large datasets that are not available freely and easily. The standard requirement nowadays states that the N-times increase of the model complexity (adding more layers, and parameters, using a larger architecture) comes not only with the N-times increase in the required data but also, as a consequence, with an N^2 increase in the required computing powers. For research purposes acquiring such resources can be infeasible. In the private sector though there already has been progress in this direction with the BloombergGPT Large Language Model (LLM) - a 50 billion parameter language model trained on the financial data and showing great prediction results (Wu et al 2023).

Analyzing public sentiments, attitudes, and emotions can also be valuable to understanding the market dynamics and stock price movements. In the text mining field, there are various techniques for opinion mining that can be done manually by the experts in a specific domain, but it also can be done using a learning-based approach with the introduction of naive Bayes, support vector machine (SVM) and artificial neural networks (ANN) models (Medhat, Hassan & Korashy, 2014; Boiy & Moens, 2009). Naive Bayes is a classification algorithm based on the Bayes' theorem, it is particularly well-suited for the classification tasks. SVM is also a classifier that searches for the hyperplane that best separates given data into different classes. ANN are more versatile as they process the input data and learn patterns through training, it is comprised of neurons, where each connection in between them has a weight that adjusts during the training stage.

Another example is a lexicon-based type of sentiment analysis. Rao et. al (2022) analyze news headlines from seven major financial news publications using the VADER model for sentiment analysis and logistic regression for the rating of a chosen stock. This model uses a lexicon-based approach utilizing an already pre-defined dictionary of words with a sentiment score. It can also detect the intensity of sentiment depending on the punctuation.

In terms of data sources, most of the researchers choose freely available data, hence, Twitter tweets, discussion boards, posts, and blogs are popular choices for the analysis (Bollen, Mao & Zeng, 2011; Angelico et al 2022). However, this data-sourcing approach has its drawbacks given that such data is highly volatile and susceptible to noise and bias. This type of data also can be corrupted by external interventions, there have been numerous incidents with bots, data leaks, and hacking.

What is more, there is a possibility of leveraging recurrent neural networks (RNN) for financial and economic analysis. They are focused on processing the long-term dependencies and sequences of data. These attributes make RNN suitable for the stock price data and distinguish them from other types of deep neural networks. Since the main purpose of RNNs is to work with large amounts of time-series data, they can be used to make predictions based on the information available at the moment. Liu, Qiu, and Huang (2016) used a new approach to learning for the RNN based on several tasks which improved the performance.

Long short-term memory (LSTM) models have a gated hidden layer that allows them to work better with long-term dependencies. Kulikovskikh and Voronkov (2020) used LSTM to analyze the news headlines from relevant publications and predict trends of market efficiency.

On the other hand, there is significantly less research for the prediction of the economic indices using NLP or any other machine learning techniques for that matter. Univariate benchmarks, such as random walk, autoregressive, and unobserved components stochastic volatility models are widely used for the task of forecasting inflation and macroeconomic variables. However, (Marcelo C. Medeiros, Gabriel F. R. Vasconcelos, Álvaro Veiga & Eduardo Zilberman, 2021) show that the ML methods are systematically performing better with the random forest model showing the best results.

This thesis is mostly based on two papers researching the use of newspaper articles in improving macroeconomic variables predictions with a focus on inflation perception prediction indicators.

The first paper (Kalmara et al, 2022) looks into various ways of data extraction from three UK newspapers' articles for use in economic forecasts of GDP, inflation, and unemployment. The main difference between this work and the thesis is that in this thesis the initial dataset focuses on US articles and data, introducing new machine learning methods for the predictions into the comparison. The second paper (de Bandt et al, 2023) uses press data from French newspapers to construct a new indicator of inflation perceptions in France. Adding to the impact of this Banque de France working paper, the thesis at hand introduces RNN and BERT models used for the improved indicator's performance.

3. Methodology

This thesis largely follows the methodology of the Bank of France working paper "New Indicator of Inflation Perceptions in France" (de Bandt et al, 2023) developing a new supervised method of measuring inflation perception. In the paper, several supervised and unsupervised methods for the predictions are used as well as some possible extensions of the indicator calculations. In the end indicator was calculated as a "balance of opinions" between articles indicating an increase in inflation and those indicating a decrease or stability of prices.

3.1. BOF working paper methodology

The authors of the working paper use the Factavia database as a newspaper article source. Firstly, they filter all the price-related articles on the Factavia platform directly using a "price"-query focusing on the keywords related to the lexical field of prices and inflation. The next step was to clean and pre-process the input data in the same way as it is usually done for any NLP tasks (i.e. removing the stop words, turning texts to lowercase). For the creation of vectors from the data Word2Vec embeddings were used with a set number of coordinates for each vector.

After that, a random sample of 2000 articles was manually labeled as "about price" or not by several economists. This is necessary for the supervised model training as the labeling procedure provides the correct examples for the models' training. Next, a Support Vector Machine (SVM) model was trained using the embeddings provided by the Word2Vec model. This model helps to categorize the input data by creating the best possible hyperplane to separate each group of labels.

The SVM labeling of the data separated it into several categories: (0) low or falling prices, (1) stable prices, (2) rising prices, and (3) no direction. Due to the low amount of articles on stable prices, categories (0) and (1) were subsequently merged. Finally, with the price-related articles divided into the appropriate categories, the inflation perception indicator was calculated following the formula below:

$$Press\ Inflation\ Perception\ Indicator = \frac{\#\,Up - \#\,Down}{\#\,Up + \#\,Down}$$

This is the most restrictive version of the indicator formula as the denominator excludes the articles without any direction of price movements. However, it was chosen because it represents the closest patterns to the traditional inflation indicators like the ones computed in the surveys in the form of a “balance of opinions” with a number of surveyed persons represented by the number of articles about the price movements.

It is harder to extract the time dimension of inflation from the articles as it requires further labeling finetuning. Also, in terms of proportions, only a few articles described future price movements while most of them focused on the past or current inflation rates. As a consequence, it was decided to restrict the scope of the working paper to the measurement of the perceived inflation levels.

3.2. Thesis methodology

This thesis focuses on the comparison of different vectorization methods and possible improvements of the last part of the working paper related to the ML algorithm used for the categorization of the articles.

So, after the preliminary cleaning of the texts and restricting the database to articles relating to the price movements the inflation perception indicator was computed. However, there is a restriction on the resources available for the implementation of this thesis, so the labeling of the texts was performed automatically instead of the manual procedure used in the working paper. This can be improved in future implementations.

Vectorization of the texts was performed via several methods: Bag of Words, using specified dictionaries, and Word2Vec model. The categorization of the new texts was performed via the Naive Bayes classifier and Linear regression. However, those methods didn’t show significant improvements to the model’s predictions, so a state-of-the-art transformer model was introduced. With some fine-tuning of parameters and after training the model on the available corpus of articles it shows promising performance compared to other methods.

The next chapter goes into further details of the implementation, describing each step taken during the analysis.

4. Tools

4.1. Models

Bag of Words

To process textual information automatically we need to convert it into a form that the machines would understand. So, to do so, we need to convert words into numbers. One of the most straightforward ways would be to create a vector with a length equal to the number of words in the language and put one in the position of the word we want to encode and zero in all other positions. Such procedure is called One-Hot-Encoding (OHE). This method is simple to implement but it does not allow to capture semantic relationships between words.

The next step would be to concentrate on specific texts as most of the natural languages are represented as texts. Here we can follow the same logic and count the number of appearances of unique words in the text as follows:

```
Text: "bag of words counts words"

BoW vector: (1,1,2,1)
```

bag	of	words	counts
1	1	2	1

This method is also suboptimal as it does not capture any information about the relative positions of words in the text. However, it allows us to compare different texts and assign specific numerical values to the words in a text.

Sentiment dictionaries

Another way of assigning numerical values to words is through pre-determined dictionaries in a specified area (i.e. economics, finance, psychology). This method allows to measure sentiment associated with a specific word. For example, in the AFINN dictionary the word “bargain” has a value of “2” and the word “debt” has a value of “-2”. One of the ways to calculate the score of a specific text would be to combine the BoW approach with the words’ weights assigned through the dictionary.

$$Text\ score\ =\ \sum_{0}^{\#\ of\ words} (frequency\ of\ a\ word\ \ *\ weight_{word}\ )$$

Word2Vec

So far, the methods mentioned above have not allowed us to capture the context of a text. The Word2Vec model is based on the idea of predicting the probability of a certain word's appearance based on its context (surrounding words).

In the training of this model we take sequentially (2k+1) words, the word in the center is the word, that should be predicted. The surrounding words are context of length k on each side. Each word in our model is associated with a unique vector, which we change during the training process of our model.

This approach is called CBOW - continuous bag of words, continuous because we feed our model sequential sets of words from the text, but the order of words in the context is still not considered.

After the model's training, we receive distributed representation, so no number individually means anything, they need to be considered together with the other words from the text, which means that linear algorithms will perform best for the predictions (i.e. predicting the next word in the text should be easier with a linear algorithm once all the previous words are considered).

Word2Vec unlike the BoW captures semantic relationships between words as words with similar meanings are located close to each other in the vector space (for example, “man” is close to “woman”). It also generates dense vectors of fixed dimensions, usually 100 or 200 dimensions. This allows for reduced computational complexity and lower use of memory. It also allows the Word2Vec approach to be more efficient when handling large datasets because of the dense vector representations and NN-based training algorithms.

RNN vs Transformers

The use of neural networks-based algorithms can significantly improve word embeddings and following predictions as in this case the whole texts are considered including semantic relationships between words.

In short, in a neural network, each neuron gets an input, processes it, and passes it to the output. Neurons from one layer are connected to the neurons in the next, and each of these connections has a weight that adjusts during the network’s learning process.

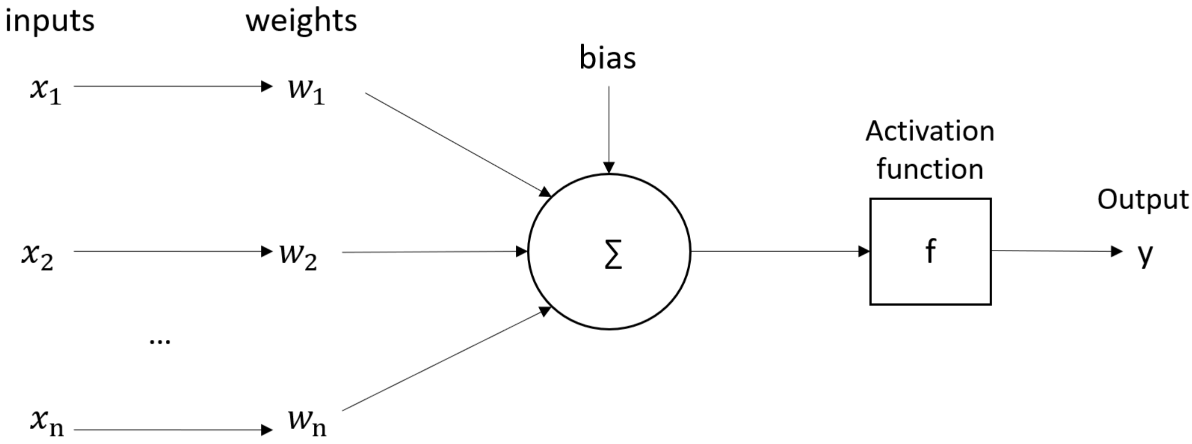


Fig. 1 - NN architecture

The network learns by forward and backpropagation processes. In the forward propagation, inputs are passed through the input layer to the hidden one, in which the weights and activation functions are applied, after that the processed data goes to the output layer producing the model's predictions. Backpropagation is the process in which the network calculates how much each weight contributed to the error. These processes are repeated for several iterations over the training dataset, updating the weights and subsequently improving the network's predictions.

In the first discussions about this thesis implementation RNN model was considered as a possible improvement from the SVM implemented in the BOF working paper (de Bandt et al, 2023). However, after a review of the current literature on the topic, it was decided to proceed with the transformer model's architecture. Below are some of the reasons for this decision.

RNN is considered an improvement of the neural network paradigm because, besides the hidden and input/output layers, it also has a time (memory) component. In principle, RNN work is based on the idea of recursion where the output from one step is also used as an input for the next step. All the information gathered in the previous steps is passed to the next steps for analysis and prediction creation. This is a very important feature that allowed RNN to perform better than other models when it comes to NLP-related tasks as it considers the context of words compared to other words in a sentence and other sentences in the text. It is comparable to reading a book: with the information from each new page we are already considering information from all the previous pages and can better understand the story. In addition to the connections between layers, the element receives a connection with itself and returns to itself, passing information from the current moment in time to the next one.

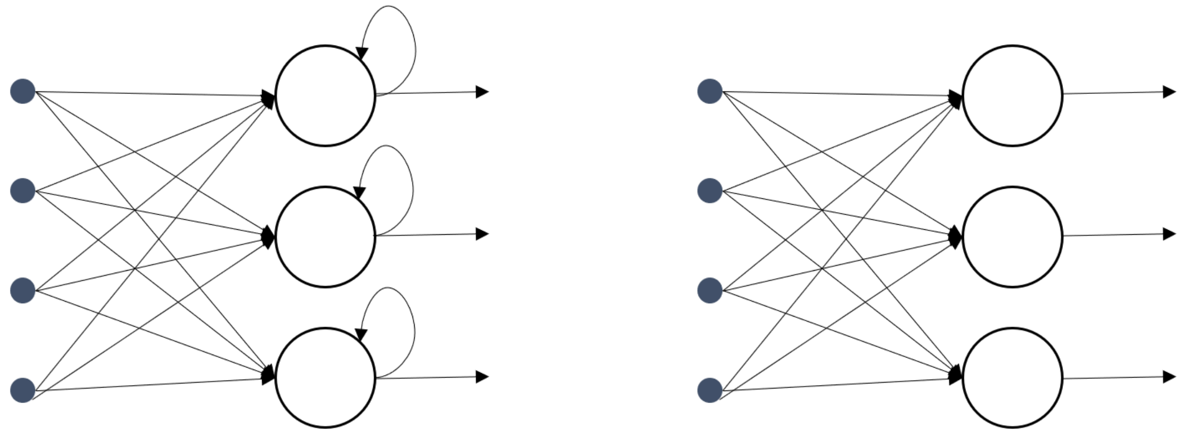


Fig. 2 - RNN and NN architecture schemas

Transformers use attention mechanisms in order to increase the model's training speed, they also perform more efficiently in the conditions of parallelization, cutting the required computing power.

On the high level, the model consists of multiple encoder and decoder layers. Inside the encoder, the inputs are first passed through the self-attention mechanism (it allows to capture the context during encoding of a specific word) and into the feed-forward neural network. The decoder has both of those layers, plus the encoder-decoder attention layer that allows the decoder to focus on the relevant parts of a sentence.

During the processing of a specific word by the model, the attention mechanism allows the model to consider other positions in the input sequence and find a way of better encoding the said word. For example, it might be challenging for a model to understand whether in the sentence below "it" refers to the word "way" or the word "model". This is where the attention mechanism comes in and allows the model to understand other relevant words in the sentence that are connected to the word that is being encoded.

`"The best way to learn transformer model is to use it in the thesis"`

Transformer models also consider the order of words in the input sentence by adding a positional vector for each input embedding. This allows to create appropriate distances between the embedding vectors.

So, in the transformer models, the encoder first receives an input sentence. The output of the first encoder is converted into the attention vectors, they are used by all the decoders in their encoder-decoder attention layer to focus on the most important parts of the sentence for the specific word that is being processed. Those steps are repeated until the decoder finishes the generation of outputs. In both encoder and decoder inputs the positional vectors are added to the embeddings. In decoder layers though the attention mechanism can only focus on the previous positions of the output.

Decoders return a vector of numbers and in order to convert it into words there are linear layear and softmax. The linear layer is a simple feed-forward NN that transforms the vector outputted by the decoder layers into a logit vector. Every cell in this vector would correspond to the weight of one of the unique words in the training corpus. The softmax layer converts those weights into probabilities so that the cell with the highest probability can be chosen. So, the output for a specific time period would be a word from that cell.

After training the model the idea is to get the probabilities that would allow us to label texts according to their categories. Given that the model generates one output element at a time, we can assume that the model selects the word with the highest probability from the probability distribution and discards all others.

The general transformer model architecture is represented in the figure below.

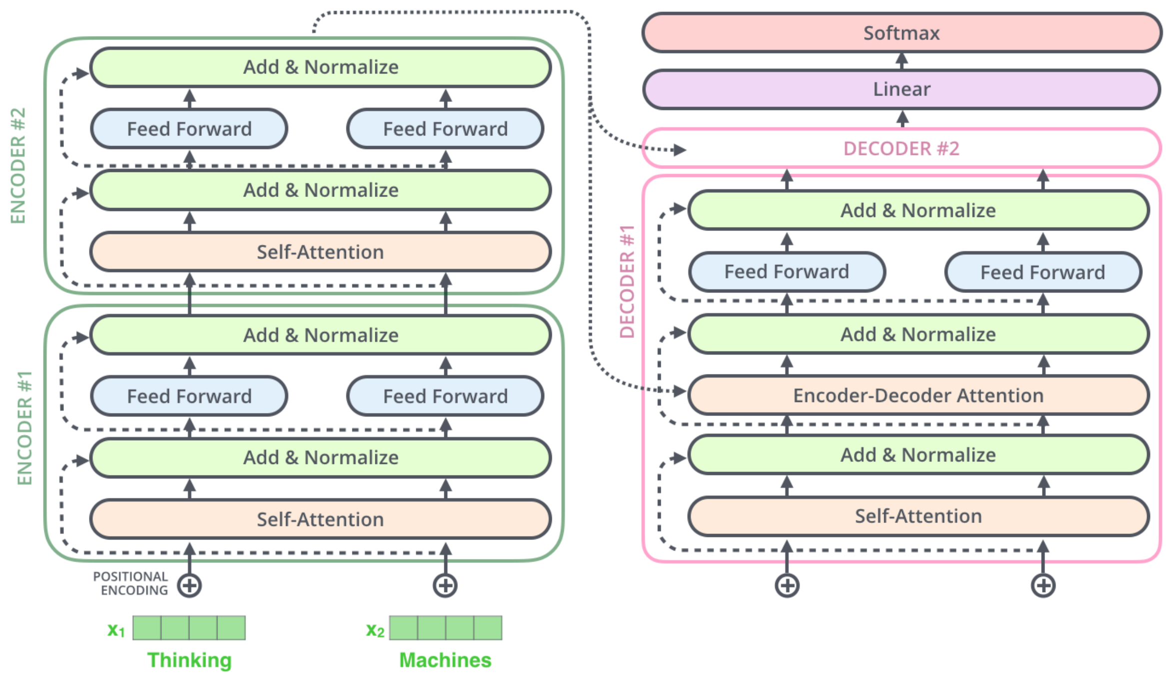


Fig. 3 - Transformer model architecture

4.2. Libraries and programming environments

Implementation of the methodology described in the previous chapters was only possible with the use of several programming environments and a number of libraries. The codes for implementation were written in Python and R programming languages. R is specifically designed for dealing with data and Python is a bit more flexible language with a variety of ML libraries available. Since multiple models were involved, it was decided to separate scripts for better project organization into data import, preliminary cleaning, data cleaning and reformatting, Bag of Words implementation, sentiment dictionaries, Word2Vec, and BERT implementation.

In terms of environments, for the Python scripts Visual Studio Code (VS Code) was used as the main text editor. It provides a lightweight environment for multiple programming languages. However, since the local recourses were restricted by the GPU and memory of the personal laptop, it was decided to also incorporate the Google Collaborators environment which utilizes Jupiter notebooks and allows to use extra GPU and memory for free or with the service subscription. For R scripts local RStudio environment was used.

In terms of Python libraries, firstly, for the loading of the relevant years' articles from the New York Times archives, some libraries for handling data files were used. JSON and CSV libraries in Python help with the creation, processing, opening, and writing of the files of the corresponding formats. JSON file structure allowed one to navigate through the plethora of information in the NYT files to find the texts of the articles and dismiss unnecessary information.

Some basic data transformations and formatting operations were done via Pandas and NumPy libraries, which are very powerful tools for working with data. NumPy is primarily used for calculations as it is implemented in C, making this library much faster than the native Python computations. Pandas library was built on top of NumPy and is primarily used for data manipulations.

Scikit-Learn library is one of the most commonly used ML libraries in Python, it has integrated preprocessing tools as well as various ML algorithms (e.g., SVM, Random Forest, K-Means). It is built on top of NumPy, SciPy, and matplotlib and provides tools for data analysis and data mining. In this thesis, it was primarily used for the preprocessing of text and automatic implementation of the BOW algorithm.

Natural Language Toolkit (NLTK) is a Python library specifically created for NLP. It is used for classification, tokenization, stemming, lemmatization, tagging, parsing, and semantic reasoning. In the thesis, it is used for the texts cleaning from the English stop words. For the World2Vec implementation Gensim library was used. It is especially good at handling large texts, making it one of the primary choices for NLP tasks such as word embedding creation and topic modeling using techniques like LDA (Latent Dirichlet Allocation). For various plots of the data statistics and models' evaluation results, matlplotlip was used, as it is highly customizable and is already integrated with NumPy and Pandas libraries.

One of the main reasons to use R was the textdata library that provides the most commonly used sentiment dictionaries in finance, economics, and psychology fields. R environment also allows for fast and effective experimentation with data formats and analysis.

For the implementation and fine-tuning of a transformer model Hugging Face community model was used. Hence, it was necessary to convert the data into the datasets format specific to the said community models, which supports large datasets and provides tools for dataset manipulation and processing. PyTorch library was also used for data formatting as it is one of the requirements for the BERT model inputs. PyTorch is an open-source library which had extensive support for neural network operations.

Finally, the state-of-the-art pre-trained model for NLP was accessed through the Hugging Face Transformers library. It includes implementations of the most popular NLP models and provides tools necessary for fine-tuning and deployment of these models.

5. Data (sources & cleaning)

Finding an appropriate dataset with open access is still one of the biggest challenges when it comes to newspaper article-related research. Most of the sources are proprietary and restrict downloads or scraping. Both aforementioned papers, which are most similar to the work at hand, use the Factavia database which has access to a plethora of papers and their articles.

The data for this research was acquired through an open New York Times archives API. It was possible to request a personal API key-password pair to download articles' data from 1990 to 2024. The data available is in the JSON format. JSON (JavaScript Object Notation) is a lightweight and widely used format that allows the storage of data in an understandable way for both developers and machines (JSON community). It is built on two main data structures: ordered sequences of values (a "list" in Python) and key-value paired information (also known as "dictionary" in Python).

Although full articles were not provided, leading paragraphs, snippets, and abstracts were available. According to some of the referenced academic papers, even the use of newspaper headlines is enough to improve predictions significantly (Hossain, Arafat, Md Karimuzzaman, Md Moyazzem Hossain, and Rahman, 2021). However, according to the available research, the results of this work could be improved in terms of accuracy with the availability of the full texts of the articles.

Another limitation comes from the fact that the dataset is restricted to only one newspaper introducing some bias when it comes to the tone (a quality in writing or speech that expresses a person's feelings and emotions towards a particular subject) and semantics (meaning of words in a language) used in the reporting. Since different newspaper reporters can have polar opinions on the same event, this can nudge the prediction closer to one of the extremes. To account for the writing style differences, all the words in the articles are lemmatized.

The period was constrained to the last 34 years as this is a long enough period to see trends as well as cyclic changes in the economic phenomena. It also includes several shocks to the economy such as the crisis of the real estate markets in 2008, COVID-19, and others.

There were two primary datasets included in the analysis. The "raw" version of the data, is ejected directly from the downloaded JSON files. Another version was preselected using methodology from the working paper to include only the articles containing relevant content to the inflation topic/economics lexicon (de Bandt et al, 2023).

Table with the lexicon fields:

Lexical fields	Keywords
Inflation with economic terms	CPI, deflation, inflation, disinflation, inflationary ; recession, stagflation, consumption basket
Energy	Gas, oil, petrol, fuel, electricity
Prices and costs	Price, cost, income, revenue, wage, expenditure, payment, rent, purchasing power, tariff, sale
Other	Tobacco

Then, both datasets were cleaned from non-alphanumeric characters and English language stop words. The openly available historical US inflation rates by month from 1990 to 2024 were associated with the corresponding months in the data. The new index was then constructed by counting the number of articles with an upward inflation trend minus the ones with a downward trend divided by the sum of the articles from both groups for each month.

The overall dataset was also divided into "training", "testing" and "validation" sets, where the testing set represented 20% of the available data, and the validation set represented 10%. The training set was shuffled to avoid biased behavior. However, this shuffle limits the representation of temporal relations in the data.

Some statistics on the data are presented in the Table below:

Metric	Total corpus	Restricted corpus
Total number of articles	5134546	483901
Inflation going up	12056	8649
Inflation going down	14739	10453
Average number of articles per month	6232	588
Average number of articles with inflation going up	29	21
Average number of articles with inflation going down	36	25

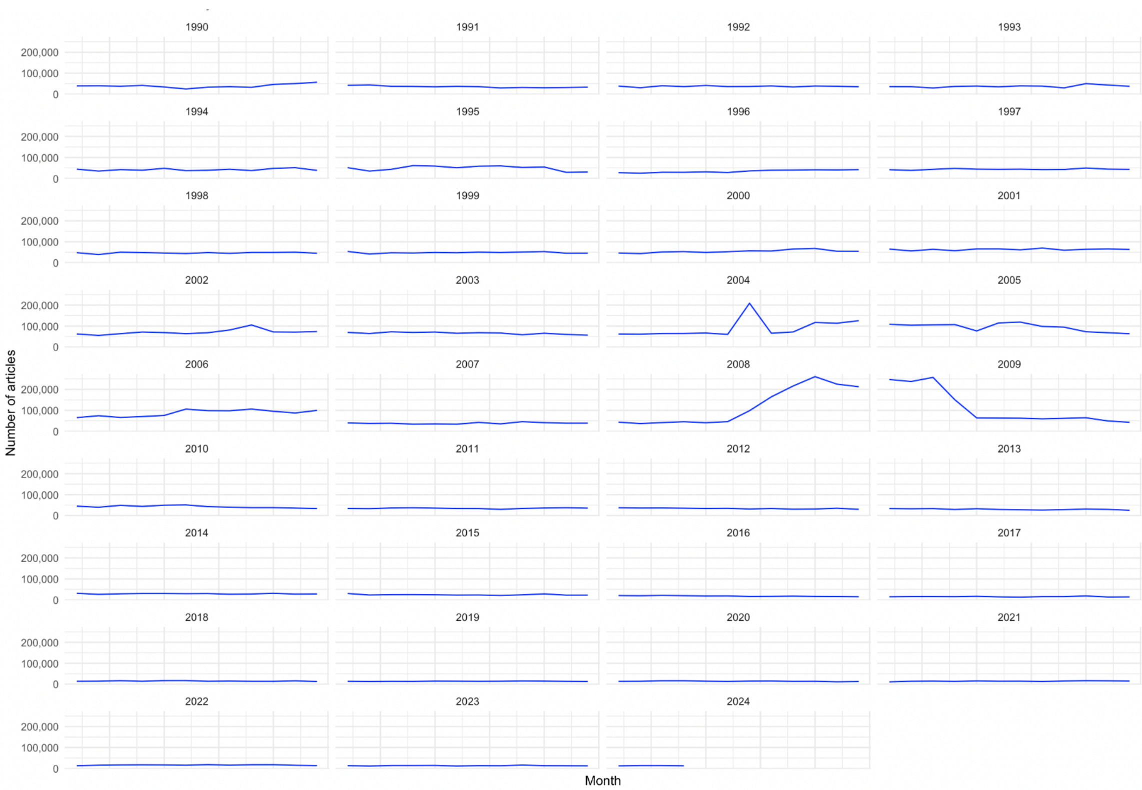


Fig. 4 - Number of articles connected to inflation and price lexical fields by month

6. Analysis

6.1. Pre-processing

Before any data manipulations could be done the dataset had to be pre-cleaned. First of all, all the words were converted to lowercase. After that, all the texts were split into individual words - tokens. Also, the texts were cleaned from the punctuation and special characters as well as the stop words (e.g. « is », « the », « and »), these words do not contribute to the analysis, however, processing them would still require time and computational power. In the final step of pre-processing words were converted to their base form - lemmatized. For example, « ran » and « running » after lemmatization would both appear as « run ». Lemmatization allows returning a word to its base form, which is called “lemma”. It is a necessary step to standardize words in the dataset and treat each variation of the same word in the same way.

The data was shuffled and divided into training, testing, and validation sets for the model's training and estimation. The training set consists of 70% of available data, it is usually the biggest part of the dataset and is needed for the model to understand and gather all patterns in the real-world data, which then can be used for future predictions. The testing set is about 20% of the dataset, it is used to test the classification and predictions made by the model. Finally, the validation set provides 10% of the data that is used to validate the results of the testing.

So, after all the preliminary steps the dataset looked like a data frame with two columns: pre-cleaned sentences and inflation movements described as 1 - for the upward movement of the perceived inflation indicator, 2 - for the downward movement, and 0 - for the texts where there was no inflation movements mentioned.

To conduct models’ training “Google Colaboratory” was used. It is a programming environment with extra storage and CPU power available for rent and free of charge usage. Also, for the compilation of models and pre-trained examples, the documentation on keras, pytocrh, and scktlearn libraries was used as well as recourses provided by the Hugging Face community.

6.2. Bag of Words

The first approach applied in this thesis is commonly known as the “bag of words", a popular NLP and text analysis approach. The primary step is to convert words into tokens. During the preliminary cleaning, all available paragraphs and snippets are converted into individual words (tokens). After tokenization, a vocabulary is formed subsetting all the unique words in the text.

Afterward, it is counted how many times each word appears in a given text to create a vector for each of the texts with the frequency of a word from the vocabulary in that text. In this thesis, the « count vectorizer » function from the sci-kit-learn library was used to perform the task of vectorizing (extracting features).

Then using a training set a simple supervised Naive Bayes classifier was trained. This classifier represents a rather straightforward approach for predictions using Bayes' theorem. «Naive » refers to the assumption that there is a conditional independence between every pair of features given the values of the class variable (category).

This approach is widely used, because of its straightforwardness. However, it has several notable limitations. The most important one is the fact that with the "bag of words" approach punctuation and word order are not considered. So, this approach can be improved as it is well-known that semantic relationships and context are important for the understanding of the overall idea of a given text.

6.3. Dictionaries

Another method of converting words into vectors is via financial and economic dictionaries where words in a specific area are already pre-classified as positive or negative. For the robustness of the analysis, a combination of several dictionaries was used: Nielsen (2011), Hu and Liu (2004), Hu et al. (2017), and Correa et al. (2017). In each text the number of positive and negative words was counted based on the dictionary, afterwards, the feature was created to represent those counts. In this case for the prediction of the index NB was used in the same way as with the BOW approach.

6.4. Word2Vec

Another approach of word vectorization was performed via word embeddings with the Word2Vec model. Word embeddings put words in a continuous vector space where semantically similar words are located close to each other. For each text, the average of the word vectors was computed for all the words in the text. It gives a fixed-size feature vector to represent each text. This feature allows for capturing the relations between words in a sentence allowing for a better texts' interpretation and analysis. This method allows putting weights on the words with consideration for the context around them. For the predictions, Linear Regression was used as it allows for negative vector values, unlike the NB classifier.

6.5. BERT

Finally, the most promising and well-performing class of NLP processing models at this moment was implemented - the transformers. In particular, BERT, developed by researchers at Google, allows us to not only consider the context of one sentence but several sentences and the article as a whole. There are a few pre-trained models available, that require some finetuning (creating an extra layer with the task's specifications) to achieve the best performance in the task. This allows researchers to achieve the best possible result without notable compromises on the training time and necessary computing power.

BERT (bidirectional encoder representation from transformers) is a bidirectional transformer pre-trained with a combination of masked language modeling objective and next sentence prediction on a large corpus including Wikipedia (2500M words) and Toronto Book Corpus (800M words). The said pretraining cuts on the required computational power and memory necessary to produce high-level results. Fine-tuning the existing pre-trained model adds another linear layer on top of the base model configuration, which produces a tensor indicating unnormalized scores for several labels for every example in the batch.

For this work supervised machine-learning technique was used. So, for the restricted data sample articles were divided into three categories: indicates upward inflation, indicates downward inflation rate movement, and no indication of the inflation movements. Due to restrictions on resources, the labeling was done automatically based on words most likely corresponding to one of the categories. In future implementations, the labeling should be done either by a group of economists or by a pre-trained algorithm in the unsupervised machine learning domain.

Multi-label prediction in the chosen finetunned BERT model helps to differentiate between the articles where the inflation rate is moving up or down, and the ones where the movement is absent. This classification allows us to then calculate the inflation perception indicator suggested earlier.

To estimate the model and test its accuracy several steps were taken. After the initial data cleaning that was done for all the models in this work and naive labeling of observations as "connected to inflation rate going up", "connected to inflation rate going down" and "not connected to inflation", the dataset was formatted in a specific way accepted by the model.

For the research question at hand, the “bert-based-uncased” pre-trained model for multi-label classification was chosen (from the Hugging Face community). This is a transformer-based model designed for natural language understanding tasks. There are two types of this model available: one is “base” and another one is “large”, truly large model that gives the best permoance. The “uncased” part means that the model does not distinguish between uppercase and lowercase letters, treating them in the same way. This can present an issue if it is important to distinguish between proper and common nouns. For example, the names of the cities or people and all other nouns. However, in this work, it is not the case, since during the pre-processing stage all the available data is converted to lowercase.

The next step was to convert the data into the dataset format, creating a dictionary of datasets. All the labels were collected into a list of labels that are needed for the model’s inputs. After that, the data was tokenized, padded on the right side (as the model requires fixed-length inputs), and converted to the “torch” format from the PyTorch library. This library was developed by Facebook’s AI Research lab and is used for building and training neural networks.

Example of the encoded sentence:

```
[CLS] " percy barnevik chairman abb europe big electrical engineering group unexpectedly step
today say take share responsibility sag profit stock price company instrumental shape " [SEP]
[PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD]
[PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD]
[PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD]
[PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD]
[PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD]
[PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD]
[PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD]
```

After that the parameters of the model were defined. The evaluation strategy specifies when to evaluate the model during training, the save strategy determines when to save the model checkpoint during training. Learning rate is a hyperparameter that controls step size at each iteration while moving towards a minimum of the loss function. It tells the researchers how fast the model learns. Weight decay prevents overfitting by discouraging too complex models, and maintaining small weights of the loss function. The parameters chosen for this work are common for the fine-tuning of the transformer models.

Parameters of the model are as follows:

```
args = TrainingArguments(
   f"bert-finetuned-sem_eval-english",
   evaluation_strategy = "epoch",
   save_strategy = "epoch",
   learning_rate=2e-5,
   per_device_train_batch_size=batch_size,
   per_device_eval_batch_size=batch_size,
   num_train_epochs=4,
   weight_decay=0.01,
   load_best_model_at_end=True,
   metric_for_best_model=metric_name)
```

In multi-label classification, the model outputs raw scores or logits for each of the labels. They can be positive or negative and do not sum to one. To convert those logit outputs into probabilities the sigmoid function is applied. It maps any number into the (0,1) interval, which afterward can be interpreted as the probability of belonging to a group with a certain label. After that, a threshold is applied to decide the final prediction for each label indicating whether the label is present or absent.

$$sigmoid(x) \;=\; \frac{1}{1+e^{-x}}$$

Estimation of this model with the available computing power takes about 1.5 hours for the sample dataset of 1,000 observations. Since the whole dataset contains more than 400,000 observations, it is not feasible to estimate the model on the whole dataset.

6.6. Evaluation

To evaluate the models several tests are used. The accuracy test represents the ratio of correct predictions to the total number of predictions. It is most useful when the classification categories are balanced. So, in this case, if all three classes ("connected to inflation rate going up", "connected to inflation rate going down" and "not connected to inflation") were to comprise approximately 33% of the dataset, the accuracy metric would be the most useful. However, this is not the case as the articles "not connected to inflation" present the majority.

$$Accuracy \;=\; \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions}$$

Another test that is used is the F1 score which represents the harmonic mean of precision and recall metrics. Precision shows the ratio of correctly predicted positive observations to the total number of predicted positives, while recall represents the ratio of correctly predicted positive observations to all observations in the actual class. F1 is especially useful if the class distribution is unbalanced.

$$F1\ Score\ =\ 2\ *\ \frac{Precision * Recall}{Precision + Recall}$$

$$Precision\ =\ \frac{True\ positives}{True\ positives + False\ positives}$$

$$Recall\ =\ \frac{True\ positives}{True\ positives + False\ negatives}$$

In addition to the accuracy tests used previously for predictions after using different vectorization techniques, for the BERT model, it was decided to introduce additional tests. ROC AUC (Receiver Operating Characteristic Area Under the Curve) is a curve that plots the true positive rate (TPR) against the false positive rate (FPR) and tells how well the model can distinguish between different classes. It is most useful in the models that output probabilities and is robust to the class imbalance.

$$AUC\ =\ \int_0^1 TPR\ (\ FPR^{-1}(x))dx$$

The resulting errors are measured differently for training and validation datasets and are presented in the plot in the results section. Training loss monitors how well the model is fitting the training data and validation loss is used as an early stopping mechanism to prevent overfitting. Overfitting happens if validation loss increases while training loss continues to decrease. Underfitting happens if both training and validation losses are high and do not decrease significantly.

7. Conclusion and challenges

Inflation rate measurement is an essential part of monetary policy decisions (ECB). An index for inflation perception measurement based on newspaper articles can help with improving the accuracy of inflation predictions and make it less costly in terms of time and costs incurred by conducting the surveys for the assessment of the more traditionally used indicators. In this thesis a new inflation perception indicator was proposed, which is based on newspaper articles, hence it can be updated regularly providing the most up-to-date results.

Another possible dimension of usage for the indicator is to concentrate the assessment on newspaper articles related to future inflation rate expectations. However, at this stage, it was challenging to separate articles according to their time-dimensions. Moreover, since this thesis relies on naive labeling for the supervised ML tasks, manual labeling of the randomized sample of articles including the label for time dimension would potentially help to improve the results as well as allow for the predictions of inflation expectations of the households.

The results of the tests for the first three models are provided in the table below. It is noticeable that although the results are quite close, the pre-assigned words' classification from the financial and economic dictionaries produces less accurate predictions than the methods derived based specifically on the data at hand.A

Model	Accuracy	Precision	Recall	F1 score
Bag of words	0.9798927475434228	0.9821357507246606	0.9798927475434228	0.9807510159280907
Dictionaries	0.9619346772610327	0.925318323317287	0.9619346772610327	0.9432712862887788
Word2Vec	0.9803370496275096	0.9784197981392366	0.9803370496275096	0.9783935880305006

Comparing the previous results to the tests for the BERT model we can see that all of them have higher values than previous models starting with the 2nd training epoch.

Epoch	Training Loss	Validation	F1	Roc Auc	Accuracy

		Loss			
1	0.065700	0.008450	0.975207	0.975806	0.998000
2	0.012600	0.003014	0.991870	0.991935	0.999333
3	0.007000	0.002879	0.991870	0.991935	0.999333
4	0.005200	0.002356	0.991870	0.991935	0.999333

We can also see that the training and validation loss are both decreasing with time as shown on the graph below, so there is no concern about underfitting and overfitting of the results. However, the accuracy of the model is so high due to the naive labeling of the data and the majority of it belonging to the "not connected to inflation movements" set. In order to bring it to more realistic results it would be advised to make the labeling of the data more precise and adjust the dataset to have equal proportions of data in each category.

An improved version of this model can be used for the inflation perception index predictions. The setup needed would include access to newly published articles so that they can be classified into one of the three categories. Afterward, the index can be computed following the methodology of the BOF working paper:

$$IPI = \frac{\#Upwards\ inflation\ movements - \#Downwards\ inflation\ movements}{\#Upwards\ inflation\ movements + \#Downwards\ inflation\ movements}$$

This index in combination with the survey-based indices can be used to more accurately predict perceived inflation rate movements and would be useful for policy adjustments.

In this thesis, several approaches to developing a news-based inflation perception index were discussed. There is not much research available in regard to the NLP implementations for economic indicators predictions. However, introducing NLP-based models can improve said predictions.

Finding the appropriate data source that would cover Finding the appropriate data source that would cover a substantial time period was one of the main challenges. After the data collection for the past 30 years via a free New York Times archives' API, initial pre-cleaning, and data shuffling, several models were implemented for texts' classification and prediction of the classes of the new texts.

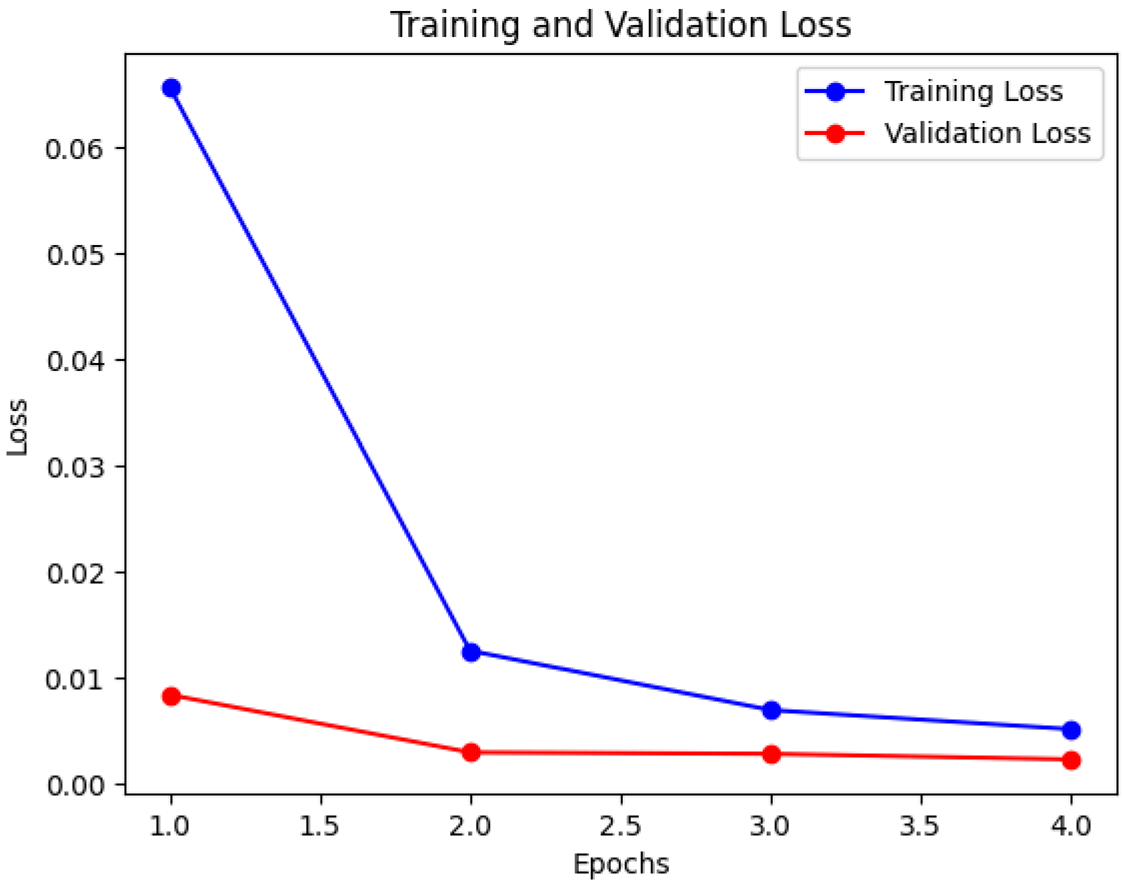


Fig. 5 - Training and validation loss plot

For the evaluation of the models, several tests were used including accuracy, precision, recall, and F1 score which is the most accurate in the case of this database as the label proportions in the dataset are not balanced.

The Bag of Words approach for the vectorization of input texts in combination with the Naive Bayes classifier for the predictions of the categories of the new texts shows good results but has some limitations. For example, the BOW approach doesn't take into account semantic dependencies in between words in the sentences.

The introduction of the dictionaries on financial and economics topics with already pre-classified words in theory should have improved the prediction accuracy as the labeling in these dictionaries was done manually by specialists and researchers in the area. However, since the words were not specific to the dataset at hand, this approach gave the worst performance.

Word2Vec embeddings do consider the context around the word, unlike previous methods of vectorization. It does require a bit more time to train the model but in the end, it gave results comparable to the BOW method.

According to the results of this thesis transformer models, in particular, BERT are the most accurate and, hence the preferred way of classifying news data related to inflation topics. Nevertheless, there are several challenges associated with this model's implementation. Firstly, choosing the right model for the task can be tricky as there are many possibilities of fine-tuning already pre-trained models and constructing new model variations based on the transformers' architecture. What is more, such models require substantial computational power and time for training on such a large dataset. As well as, specific parameterization that can be calculated only with the trial of different model configurations. In the frame of this thesis training and predictions were conducted on a sample of the dataset as it was not feasible to conduct the training on the whole dataset with the computational power available. Otherwise, the model shows promising results and can be used for the construction of a news-based perceived inflation indicator and its prediction.

In the next phases of the research in this area, it would be interesting to implement a transformer model that takes into consideration temporal dependencies in between texts. For example, the time-series transformer model.

8. Extension

As an extension of this thesis probabilistic time series forecasting with transformer models can be implemented. This type of model allows considering multiple time series at the same time building the predictions on a more "global" model than more traditional methods that are applied to one time series in a dataset. Those methods also produce a point-wise prediction, meaning that there will be a specific value for each predicted time period while in reality, it might be more useful to also understand the uncertainty of predictions. This can be done via probabilistic forecasting, where the output of the model is a probabilistic distribution, which can be sampled.

Transformer models are well-suited for prediction tasks. Although it usually can not be trained on the whole dataset due to time and memory constraints, the model can be trained on sampled data. The encoder-decoder architecture allows the decoder to use for training only the values that preceded the input sample, thus the data can be sampled randomly eliminating possible selection bias.

This type of model can also incorporate any missing values without imputing them by not including the padding in the computation of the attention matrix. This matrix is one of the crucial components of the transformer architecture as it calculates the weights that are placed on the input values to assess their importance for the predictions.

What is more, transformer models use an "autoregressive generation" mechanism, which means that every predicted step helps to predict the new one. This is also one of the advantages of the encoder-decoder architecture. For example, in the task of generating new text, one would sample the next word and then pass it back into the decoder.

In the context of the inflation perception indicator based on the news articles, implementing the probabilistic forecast transformer model would allow the use of data from several newspapers in the same global model constructing a more robust indicator with consideration of temporal dependencies in the data.

References

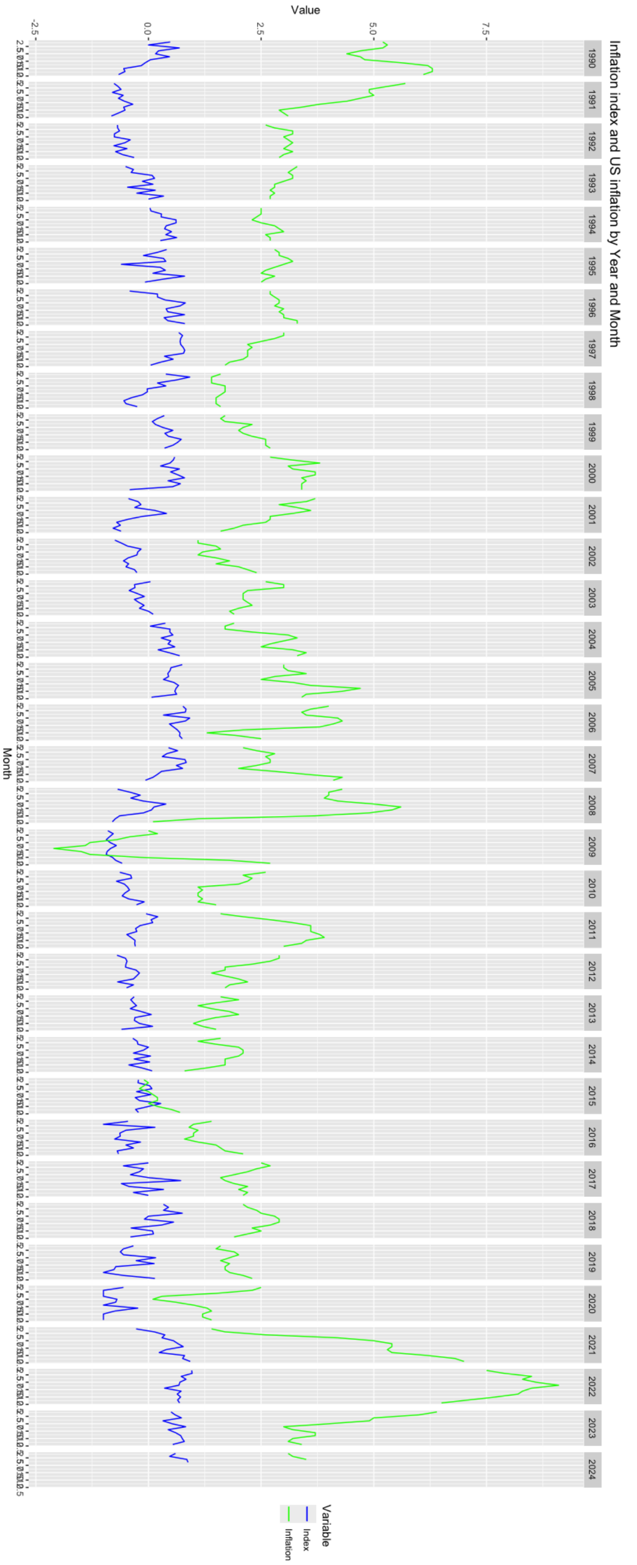
1. “Consumer Expectations Survey.” n.d. European Central Bank. Accessed June 13, 2024. https://www.ecb.europa.eu/stats/ecb_surveys/consumer_exp_survey/html/index.en.html.
2. “Definition of Consumer Sentiment.” n.d. Gartner. Accessed June 13, 2024. https://www.gartner.com/en/marketing/glossary/consumer-sentiment.
3. Lane, Philip R. 2024. “The 2021-2022 inflation surges and monetary policy in the euro area.” European Central Bank. https://www.ecb.europa.eu/press/blog/date/2024/html/ecb.blog240311~968c707650.en.html.
4. “What are business leaders' inflation expectations?” 2022. Banque de France. https://www.banque-france.fr/en/publications-and-statistics/publications/what-are-business-leaders-inflation-expectations.
5. Meyler, Aidan, and Lovisa Reiche. n.d. “Making sense of consumers' inflation perceptions and expectations – the role of (un)certainty.” European Central Bank. Accessed June 13, 2024. https://www.ecb.europa.eu/press/economic-bulletin/articles/2021/html/ecb.ebart202102_02~32e2ff1af1.en.html.
6. “Monetary policy decisions.” 2024. European Central Bank. https://www.ecb.europa.eu/press/pr/date/2024/html/ecb.mp240606~2148ecdb3c.en.html.
7. “New indicators of perceived inflation in France based on media data.” 2022. Banque de France. https://www.banque-france.fr/en/publications-and-statistics/publications/new-indicators-perceived-inflation-france-based-media-data.
8. Soroka, S. N. (2006). Good News and Bad News: Asymmetric Responses to Economic Information. *The Journal of Politics, 68*(2), 372–385. https://doi.org/10.1111/j.1468-2508.2006.00413.x
9. Angelico, Cristina, Juri Marcucci, Marcello Miccoli, and Filippo Quarta. "Can we measure inflation expectations using Twitter?." Journal of Econometrics 228, no. 2 (2022): 259-277.
10. Athey, Susan. "21. The Impact of Machine Learning on Economics" In *The Economics of Artificial Intelligence: An Agenda* edited by Ajay Agrawal, Joshua Gans and Avi Goldfarb, 507-552. Chicago: University of Chicago Press, 2019. https://doi.org/10.7208/9780226613475-023
11. “What Is Text Mining?” n.d. IBM. Accessed June 13, 2024. https://www.ibm.com/topics/text-mining.
12. Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need." *Advances in neural information processing systems* 30 (2017).
13. Wärneryd, Karl-Erik. "Introduction The psychology of inflation." *Journal of economic psychology* 7, no. 3 (1986): 259-268.
14. Behrend, Hilde. “Research into Public Attitudes and the Attitudes of the Public to Inflation.” *Managerial and Decision Economics* 2, no. 1 (1981): 1–8. http://www.jstor.org/stable/2560532.
15. Leiser, David, and Shelly Drori. "Naïve understanding of inflation." *The Journal of Socio-Economics* 34, no. 2 (2005): 179-198.

16. Kahneman, Daniel. "Maps of bounded rationality: Psychology for behavioral economics." *American economic review* 93, no. 5 (2003): 1449-1475.
17. Pieters, Rik GM, and W. Fred Van Raaij. "The role of affect in economic behavior." In *Handbook of economic psychology*, pp. 108-142. Dordrecht: Springer Netherlands, 1988.
18. Del Giovane, Paolo, Silvia Fabiani, and Roberto Sabbatini. *What's behind "inflation perceptions"? A survey-based analysis of Italian consumers*. Springer Berlin Heidelberg, 2008.
19. Saxena, Ashima. "Does Aging Impacts on Financial Behavior and Investment Decisions." (2021).
20. "Classification: Precision and Recall | Machine Learning." 2022. Google for Developers. https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall.
21. Hagenau, Michael, Michael Liebmann, and Dirk Neumann. "Automated news reading: Stock price prediction based on financial news using context-capturing features." Decision support systems 55, no. 3 (2013): 685-697.
22. Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean. "Efficient estimation of word representations in vector space." arXiv preprint arXiv:1301.3781 (2013).
23. Goldberg, Yoav, and Omer Levy. "word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method." *arXiv preprint arXiv:1402.3722* (2014).
24. Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. "Glove: Global vectors for word representation." In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532-1543. 2014.
25. "What is Deep Learning?" n.d. IBM. Accessed June 13, 2024. https://www.ibm.com/topics/deep-learning.
26. Wu, Lingfei, Yu Chen, Kai Shen, Xiaojie Guo, Hanning Gao, Shucheng Li, Jian Pei, and Bo Long. "Graph neural networks for natural language processing: A survey." *Foundations and Trends® in Machine Learning* 16, no. 2 (2023): 119-328.
27. Medhat, Walaa, Ahmed Hassan, and Hoda Korashy. "Sentiment analysis algorithms and applications: A survey." *Ain Shams engineering journal* 5, no. 4 (2014): 1093-1113.
28. Boiy, Erik, and Marie-Francine Moens. "A machine learning approach to sentiment analysis in multilingual Web texts." *Information retrieval* 12 (2009): 526-558.
29. D.R, Harish. (2017). Automatic Product Review Sentiment Analysis Using Vader and Feature Visulaizaton. International Journal of Computer Science Engineering and Information Technology Research. 7. 53-66. 10.24247/ijcseitraug20178.
30. Bollen, Johan, Huina Mao, and Xiaojun Zeng. "Twitter mood predicts the stock market." *Journal of computational science* 2, no. 1 (2011): 1-8.
31. Liu, Pengfei, Xipeng Qiu, and Xuanjing Huang. "Recurrent neural network for text classification with multi-task learning." *arXiv preprint arXiv:1605.05101* (2016).
32. Kulikovskikh, Gregory Alexandrovich, and Ilia Mikhailovich Voronkov. "Quotes forecasting method based on news analysis as part of an internet cloud service." In *2020 International Scientific and Technical Conference Modern Computer Network Technologies (MoNeTeC)*, pp. 1-5. IEEE, 2020.
33. Medeiros, Marcelo C., Gabriel FR Vasconcelos, Álvaro Veiga, and Eduardo Zilberman. "Forecasting inflation in a data-rich environment: the benefits of machine learning methods." *Journal of Business & Economic Statistics* 39, no. 1 (2021): 98-119.

34. Kalamara, Eleni, Arthur Turrell, Chris Redl, George Kapetanios, and Sujit Kapadia. "Making text count: economic forecasting using newspaper text." *Journal of Applied Econometrics* 37, no. 5 (2022): 896-919.
35. “Using the Press to Construct a New Indicator of Inflation Perceptions in France.” 2023. Banque de France | Publications. https://publications.banque-france.fr/en/using-press-construct-new-indicator-inflation-perceptions-france.
36. n.d. JSON.org. Accessed June 13, 2024. https://www.json.org/json-en.html.
37. Hossain, Arafat, Md Karimuzzaman, Md Moyazzem Hossain, and Azizur Rahman. "Text mining and sentiment analysis of newspaper headlines." *Information* 12, no. 10 (2021): 414.
38. “TONE | English meaning - Cambridge Dictionary.” n.d. Cambridge Dictionary. Accessed June 13, 2024. https://dictionary.cambridge.org/dictionary/english/tone.
39. “SEMANTICS | English meaning - Cambridge Dictionary.” n.d. Cambridge Dictionary. Accessed June 13, 2024. https://dictionary.cambridge.org/dictionary/english/semantics.
40. Nielsen, Finn Årup. "A new ANEW: Evaluation of a word list for sentiment analysis in microblogs." *arXiv preprint arXiv:1103.2903* (2011).
41. Xu, Hongzhi, Kai Zhao, Likun Qiu, and Changjian Hu. "Expanding Chinese sentiment dictionaries from large scale unlabeled corpus." In *Proceedings of the 24th Pacific Asia Conference on Language, Information and Computation*, pp. 301-310. 2010.
42. Ricardo Correa, Keshav Garud, Juan M Londono, Nathan Mislang, Sentiment in Central Banks’ Financial Stability Reports, *Review of Finance*, Volume 25, Issue 1, February 2021, Pages 85–120, https://doi.org/10.1093/rof/rfaa014
43. “The Model Hub.” n.d. Hugging Face. Accessed June 13, 2024. https://huggingface.co/docs/hub/en/models-the-hub.
44. “The Illustrated Transformer – Jay Alammar – Visualizing machine learning one concept at a time.” 2018. Jay Alammar. https://jalammar.github.io/illustrated-transformer/.

Appendix

A1. Inflation perception index and US inflation rate by month



A2. Code snippets from the main analysis steps

Data import:

```
# Iterate through all the months and years
for year in years:
   for month in months:

        # Try/except in case of loading error
        try:
            json_name = year + month + 'output.json'
            json_data = open(json_name)

           # Convert to json loads for easier navigation in the doc
            data = json.load(json_data)

            lead_paragraph_with_ids = []

            for idx, doc in enumerate(data["response"]["docs"]):
                try:
                    lead_paragraph = doc.get("lead_paragraph", "")
                    cleaned_lead_paragraph = items_in_dictionary(dictionary_inflation,
lead_paragraph)

                   # Write lead paragraphs with unique ids to track in case of errors
                    lead_paragraph_with_ids.append({"id": idx, "lead_paragraph":
cleaned_lead_paragraph})
                except:
                    continue

            # Write outputs to a dataframe
            lead_paragraph_df = pd.DataFrame(lead_paragraph_with_ids, index = None)
```

Data cleaning:

```
# Clean df
lp_df = lead_paragraph_df.drop_duplicates (subset=['lead_paragraph'], keep=False)
lp_df = lp_df.drop(lp_df.columns[0], axis=1)
lp_df = lp_df.dropna(subset=['lead_paragraph'], how='all')

# Removing non-alphabetic characters
lp_df['lead_paragraph'] = lp_df.lead_paragraph.str.replace("'", "")
lp_df['lead_paragraph'] = lp_df.lead_paragraph.str.replace("[^A-Za-z']+", " ",
regex=True).str.lower()

# Lemmatazing, removing stopwords
nlp = spacy.load("en_core_web_sm")
nlp = en_core_web_sm.load()

# Lead paragraphs db
lead_paragraph = [nlp(row) for row in lp_df.lead_paragraph]
lead_paragraph_txt = [cleaning(doc) for doc in lead_paragraph]
lp_df_clean = pd.DataFrame({'clean_lp': lead_paragraph_txt})
lp_df_clean["year"] = year
lp_df_clean["month"] = month
lp_df_clean = lp_df_clean.dropna().drop_duplicates()
```

BOW:

```
# Preprocessing and Bag of Words vectorisation
vectorizer = CountVectorizer(stop_words='english')
X = vectorizer.fit_transform(df['"sentences"'])
y = df['infl_movements']

# Splitting the dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Model training
```

```
model = MultinomialNB()
model.fit(X_train, y_train)

# Predicting sentiment of the new text
new_text = ["Inflation rate is going up."]
new_text_transformed = vectorizer.transform(new_text)
prediction = model.predict(new_text_transformed)
print('Predicted sentiment:', prediction[0])

Predicted sentiment: 1 [inflation-up]
```

Dictionaries:

```
# Sentiment dictionary
positive_df = pd.read_csv("/content/positive.csv", quoting = csv.QUOTE_NONE,
encoding='utf-8')
negative_df = pd.read_csv("/content/negative.csv", quoting = csv.QUOTE_NONE,
encoding='latin-1')

pos = []
neg = []

pos = positive_df['"positive"'].to_list()
neg = negative_df['"negative"'].to_list()

financial_sentiment_dict = { "positive": pos, "negative": neg}

# Feature extraction
def sentiment_features(tokens, sentiment_dict):
    pos_count = sum(1 for word in tokens if word in sentiment_dict['positive'])
    neg_count = sum(1 for word in tokens if word in sentiment_dict['negative'])
    return [pos_count, neg_count]

X = np.array([sentiment_features(tokens, financial_sentiment_dict) for tokens in
df['tokens']])
y = df['sentiment']

# Splitting the dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Model training
model = MultinomialNB()
model.fit(X_train, y_train)
```

Word2Vec:

```
# Train Word2Vec model
word2vec_model = Word2Vec(sentences=df['tokens'], vector_size=100, window=5, min_count=1,
workers=4)

# Feature extraction
def get_average_word2vec(tokens, model, vector_size):
    vectors = [model.wv[word] for word in tokens if word in model.wv]
    if vectors:
        return np.mean(vectors, axis=0)
    else:
        return np.zeros(vector_size)
X = np.array([get_average_word2vec(tokens, word2vec_model, 100) for tokens in
df['tokens']])
print(X)
y = df['infl_movements']

# Splitting the dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Model training
model = LogisticRegression(max_iter=1000)
```

```
model.fit(X_train, y_train)

# Predicting sentiment of new text
new_text = "Inflation increases."
new_tokens = preprocess(new_text)
new_text_vector = get_average_word2vec(new_tokens, word2vec_model, 100).reshape(1, -1)
prediction = model.predict(new_text_vector)
print('Predicted sentiment:', prediction[0])

Predicted sentiment: 1 [inflation-up]
```

Data preparation for the BERT model:

```
# Download data
df = pd.read_csv("/content/words_by_month.csv", quoting = csv.QUOTE_NONE, encoding='utf-8')
df = df.iloc[:, [1, 4, 5]]

# Shuffle
df = df.sample(frac=1, random_state=42).reset_index(drop=True)

# Get the first 10k rows for training
df = df[:10000]

# Split the data into train, validation, and test sets
train_data, temp_data = train_test_split(df, train_size=0.7, random_state=42)
validation_data, test_data = train_test_split(temp_data, train_size=0.5, random_state=42)
print(train_data)

# Convert to dataset format
train_dataset = Dataset.from_pandas(train_data)
validation_dataset = Dataset.from_pandas(validation_data)
test_dataset = Dataset.from_pandas(test_data)

# Create a dictionary of datasets
dataset = DatasetDict({'train': train_dataset, 'validation': validation_dataset,
'test':test_dataset})
```

BERT model setup:

```
model = AutoModelForSequenceClassification.from_pretrained(
       "bert-base-uncased",
       problem_type="multi_label_classification",
       num_labels=len(labels),
       id2label=id2label,
       label2id=label2id)

args = TrainingArguments(
    f"bert-finetuned-sem_eval-english",
    evaluation_strategy = "epoch",
    save_strategy = "epoch",
    learning_rate=2e-5,
    per_device_train_batch_size=batch_size,
    per_device_eval_batch_size=batch_size,
    num_train_epochs=4,
    #num_train_epochs=2,
    weight_decay=0.01,
    load_best_model_at_end=True,
    metric_for_best_model=metric_name,
)
trainer = Trainer(
    model,
    args,
    train_dataset=encoded_dataset["train"],
    eval_dataset=encoded_dataset["validation"],
    tokenizer=tokenizer,
    compute_metrics=compute_metrics
)
trainer.train()
```