S&P 500 Daily Index Time Series Forecasting Given Global News Headlines Using LSTM, BERT, and GloVe Embeddings

Arav Santhanam

Abstract

The ability to project capital markets performance to meet rapidly increasing accuracy demands has material implications for investors and capital raising environments. The evolution of natural language processing (NLP) and deep learning techniques has provided a previously unutilized approach to accurately forecast stock market performance, progress that has largely been allowed by the research and development of cutting-edge tools such as Google Bidirectional Encoder Representations from Transformers (BERT), Global Vectors for Word Representation (GloVe), and Word2vec. NLP models have been proven to be useful in projecting stock prices, inflation and economic factors, and fundraising potential, serving as vital tools for economists and others closely tracking these markets. The aim of this analysis is to produce an accurate NLP model for stock market price prediction utilizing pretrained BERT, LSTM and CNN-based models trained on global news headlines and correspondingly labeled daily percentage returns for the S&P 500 index. Artificial intelligence (AI) neural networks are reliable methods to accurately forecast stock market performance based on global news headlines in conjunction with opinion mining techniques, while accuracies over random sampling, e.g. 50% for a binary model and approx. 17% for a hex factor model, can have substantial impacts on economic forecasting. This project utilized three models and achieved a maximum accuracy of 54% for binary classification with BERT and 45% for multiclass with GloVe.

1 Introduction

While other papers have pursued similar ideas, such as, “Predicting Stock Market Movements Using News Headlines” by Kurohara et al. 2018[1], this paper seeks to introduce a novel neural network for stock market prediction using global headlines. Specifically, neural networks are a series of algorithms that attempt to come to a conclusion based on underlying patterns in data through a process modeled on the human brain[2]. These networks are composed of an input layer, hidden layers, and an output layer and each layer is constructed from parallel computing units, known as “nodes” that resemble artificial neurons and have designated “weights”, indicating importance, and a threshold that determines when the node “fires”, or sends data to the next layer, based on its output. These neural networks are trained on data to “learn” how to map the input features to outputs and this subset of machine learning has been applied in various fields from facial recognition in phones to object detection in self-driving cars. The project data includes article headlines from the Reddit WorldNews Channel from the Kaggle Dataset “Daily News for Stock Market Prediction” from August 8, 2008 to July 1, 2016[3] and historical S&P 500 daily open and closing index values and percentage changes during the same timeframe from the Thomson Reuters Refinitiv Workspace.

[1] Kurohara, J., Chang, J., & Hoskins, C. (2018). Predicting Stock Market Movements Using Global News Headlines. *CS230*.
[2] IBM Cloud Education. (2020, August 17). *What are Neural Networks?* IBM. https://www.ibm.com/cloud/learn/neural-networks
[3] Sun, J. (2016). *Daily News for Stock Market Prediction, Version 1*. https://www.kaggle.com/aaron7sun/stocknews.

2 Sentiment and Context in NLP

The fundamental principles of natural language processing revolve around implementing efficient and accurate methods to capture the context and sentiment of words and phrases in a machine-understandable format. At the human level, biological sentiment is characterized by neural activity (more specifically, the firing of specific neurons) in specific regions of the brain[4]. The vague concept of "sentiment" refers to the emotion, attitude or perspective that accompanies a given circumstance[5]. While more intricate definitions are varied among cognitive psychologists and others, they are described on a broader basis as subjective mental impressions that can vary from mild to intense within a spectrum.

In the context of NLP, there are two major types of sentiment analysis and classification: subjectivity/objectivity and feature/aspect-based identification (see Footnote 5). Subjectivity and objectivity identification in NLP encompasses classifying a phrase or text into subjective or objective categories (or opinionated and factual). However, a substantial challenge in this type of classification is the contextual dependency of a word or phrase's definition. A word's context and the phrases or words that define its "context" are highly variable depending on the specific circumstance the word appears in. Most importantly, context illustrates the relative importance of parts of phrases and highlights the relevance and meaning of specific words. A basic method to conduct general context analysis is through n-gram analysis[6]. These are sequences of one or more words that represent entities, aspects, themes, or objects in text, and are important to performing theme extraction on a phrase. Additionally, aspect-based sentiment analysis (ABSA) is a more advanced approach compared to basic techniques such as lexicon-based analysis, in which a sentiment score is calculated based on the number of positive and negative words, with sentiment defined by a valence dictionary[7]. ABSA represents a more granular approach to sentiment classification. Rather than define a phrase's sentiment as an "average" or large-scale representation of the entire text's sentiment, feature-based classification breaks text into individual subsections to identify unique aspects and create a more nuanced map of the sentiment of different ideas within the phrase[8]. Additionally, aspect similarity co-occurrence in sentiment analysis allows algorithms to intelligently group specific features or words together based on how frequently they are present in the similar contexts. For this reason, among others, many elaborate natural language projects deploy aspect-based sentiment analysis (ABSA) for emotion mining.

To conduct effective sentiment analysis, it is essential to utilize functional tools and systems for measurement. More rigorous sentiment analysis often consists of NLP and machine learning (ML) techniques, including pretrained models such as BERT, word embeddings such as GloVe and Word2Vec, and convolutional neural networks (CNNs), which are beginning to expand outside of their traditional applications in computer vision and image recognition and are

[4] Research Institute of Molecular Pathology. (2012, September 17). The biology of emotions. *ScienceDaily*. Retrieved August 25, 2022 from www.sciencedaily.com/releases/2012/09/120917111056.htm.
[5] Brown, R. (2021, September 2). What are the different types of sentiment analysis ? *Nerd For Tech*. https://medium.com/nerd-for-tech/what-are-the-different-types-of-sentiment-analysis-808f36ef89ee.
[6] *Context analysis in NLP: Why it's valuable and how it's done*. (2019, February 19). Lexalytics. https://www.lexalytics.com/blog/context-analysis-nlp/.
[7] *Lexicon-Based sentiment analysis: A tutorial*. (n.d.). KNIME. Retrieved August 17, 2022, from https://www.knime.com/blog/lexicon-based-sentiment-analysis
[8] Repustate Team. (2022, January 4). Aspect based sentiment analysis. *Repustate*. https://www.repustate.com/blog/aspect-based-sentiment-analysis/

gaining popularity in the NLP domain. These more complex analysis methods often work by allocating weighted emotion scores to specific aspects of a sentence or phrase, as opposed to the aforementioned lexicon-based systems which typically do not employ machine learning techniques.

3 Natural Language and Sentiment Analysis in Economics

When analyzing natural language and sentiment classification applications in economics, especially when discussing the use of news headlines to predict stock market performance, it is vital to address the following questions related to machine learning algorithms and their advantages:

1. What are logical relationships or connections between headlines and fluctuations in the stock market?
2. How can a machine learning model potentially capture these correlations and what are its advantages in doing so?

This paper attempts to provide a foundation for answering these questions by emphasizing the underlying economic principle of supply and demand and the subtle yet consequential influence of the media on markets.

The most basic rule governing stock markets is that fluctuations in the ratio of buyers to sellers, or supply and demand, impacts stock prices. For example, if there are many buyers and great demand for a stock with few sellers, the stock price will increase, sidelining “excess” consumers for whom there is no supply due to the shortage of available sellers. Conversely, if there are few buyers and less demand for a stock with many sellers and abundant supply, the stock price will decrease. In other words, if the general market trend is that consumers want to sell a certain stock, the price of that stock will decrease, and if there is widespread motivation to buy a certain stock, the value will increase.

This seemingly simple yet crucial element of stock markets can, in part, explain the relationship between breaking news and the resulting stock movements. Both positive and negative news can have real-time impacts on the stock market, directly impacting consumer practices. For example, positive news, such as financial reports showing growth and stability, product launches, new corporate assets, and other economic measures such as GDP, average consumer spending, inflation, and unemployment statistics can drive shareholders to buy stocks, culminating in increased stock prices. On the other hand, negative news, such as financial reports that expose corporate incompetency, large-scale sociopolitical and economic uncertainty, and unforeseen events such as weather events and geopolitical conflicts can ultimately translate into sharp dips in stock prices.

After breaking news is published, investors reflect on the incoming information and brainstorm ideas as to the potential impact on stocks. Following these decisions, the stock market will experience gradual shifts to account for the change in consumer thought and behavior. In addition, the incoming news and the shift in stock prices can be both a positive and negative sign for various stakeholders. For example, following catastrophic floods or natural disaster events, the stock price of insurance firms may decrease while that of home renovation

and repair companies may increase as a result of the increased demand for their services and anticipation of elevated sales in the future.

After illustrating the association between news headlines and stock price variations, it is valuable to review the efficacy of machine learning algorithms to map these associations, relating inputs such as news headlines to outputs ranging from general identification of whether a market or specific index will rise or fall to a more detailed result with predictions for returns within percentage ranges. This regards the second proposed question in this paper concerning the ability of neural networks to capture and process these correlations in a machine-understandable format.

Natural language processing has the unique capacity, even among other subsets of machine learning, to understand and process human language and places this specific technology in a prime position to analyze headlines and their relationships to predefined numeric labels regarding percentage market returns. Herz et al. (2014) describes their patented approach to utilizing named entity recognition (NER), in which a neural network can fill structured templates detailing a company's actions based on recognized company names and other entities by parsing text and matching patterns on context words[9]. These templates can then be organized in clusters based on statistical correlations with variations in stock market prices. Moreover, traditional machine learning models for stock market prediction that are employed in an active trading environment are often trained on historical stock data and other numerical factors. However, an advantage of utilizing NLP in economics is that models can also be trained on qualitative data, such as financial reports, text from other corporate documents, or news headlines as explored in this paper. This allows models to capitalize on a multifaceted and diverse range of features that can be used to vastly improve model accuracy and understanding, a deciding factor to maximize profit in deployment in real-world trading or investing scenarios.

4 Historical Overview of Stock Market Prediction and Project Context

The stock market is an aggregate of exchanges where financial assets and/or securities owned by public companies are bought and sold. Its primary function is to serve as a marketplace for the trading of financial assets and securities between individuals, investors, and corporations. These markets can also provide opportunities for firms to fundraise and distribute financial risk among various stakeholders. Stock market prediction involves an attempt to foretell variations in the stock market, on an individual level regarding a company's stock, an industry or market sector level, a larger body or aggregate of companies such as the S&P 500 index, or the market as a whole. Often, expert forecasters and economists take advantage of two major analytical methods to draw conclusions about stock values and become more informed as to the variations in singular stocks in relation to the broader market: fundamental analysis and technical analysis.

The aim of fundamental analysis is to discover the approximate inherent value of financial assets, such as stocks, bonds, and currency[10]. In the case of stocks, the purpose is to

[9] Herz, F., Ungar, L., Eisner, J., & Labys, W. (2014). *Stock market prediction using natural language processing*. https://patentimages.storage.googleapis.com/df/93/5d/4cc361daa8ee8c/US20030135445A1.pdf

[10] Corporate Finance Institute. (2019, March 26). Fundamental analysis. *Corporate Finance Institute*. https://corporatefinanceinstitute.com/resources/knowledge/trading-investing/fundamental-analysis/

determine the market price in addition to the supposed “intrinsic” value. If the market is undervaluing the stock, investors take advantage of the opportunity to buy at low prices in hopes of selling later for profit. As explained by Graham & Dodd (2008):

> … security analysis does not seek to determine exactly what is the intrinsic value of a given security. It needs only to establish either that the value is adequate—e.g., to protect a bond or to justify a stock purchase—or else that the value is considerably higher or considerably lower than the market price. (p. 66)[11]

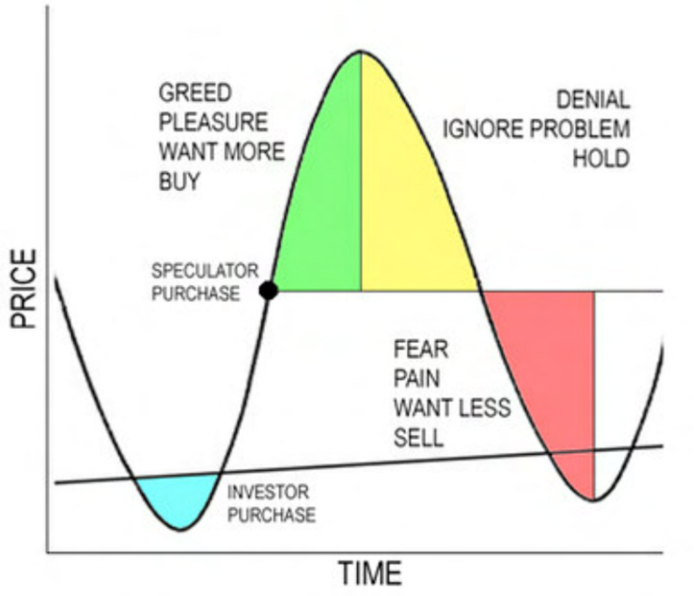
In essence, fundamental analysis seeks to approximate a stock or company’s true value and compare that with the general sentiment or market price to inform investor actions, such as buying or selling. This method of analysis is further broken down into two perspectives: a top-down and bottom-up approach. A top-down approach begins reviewing higher-level economic indicators, such as GDP, current state and stability of the economy, and the latest market news, and scrutinizing how these factors impact the business and the industry. Inversely, the bottom-up approach is often more quantitatively focused and begins with specific company metrics, including revenue and management statistics.

Next, technical analysis is the examination of securities and their historical prices with the aim of forecasting[12]. A fundamental principle of this methodology is that markets and prices move self-similarly, that future price trends will recur based on prior patterns, and that these trends are a reflection of market psychology. Also known as behavioral finance, technical analysis is at its core a study of human sentiment and thought as it pertains to their collective financial decisions in the stock market. In this regard, a price chart can be labeled with a spectrum of emotions, including caution, optimism, greed, pessimism, fear, and momentum, or the herd mentality, serving as a representation of the future expectations of consumers at certain points in time. The following exhibit is a generic price chart demonstrating consumer sentiment at various moments based on prior trends and expected future returns. Investors closely observing technical factors buy when the stock price dips below the fundamental value (line that intercepts price axis) for likelihood of maximum returns (blue region). This spectrum of emotions represents the collective consumer psyche that prompts short-term behavioral impulses, often without sight of a long-term investment.

[11] Graham, B., & Dodd, D. (2008). *Security Analysis: Sixth Edition, foreword by Warren Buffett*. Mcgraw-hill.
[12] *What is Technical Analysis and How Does it Work?* (n.d.). Nadex. Retrieved August 18, 2022, from https://www.nadex.com/learning/introduction-to-technical-analysis/

Figure 1

Price Chart Labeled With Consumer Sentiment Reflecting Expectations for Returns



Source: "5 charts to tell if stock buyers are too bullish," by J. Burton, 2013, *MarketWatch*

Due to the increasing availability of textual data that serves as an abundant supplement to traditional quantitative data and that is inherently high-dimensional (Gentzkow, Kelly & Taddy, 2019), machine learning strategies have evolved to address the wide array of data analysis and summarization requirements in applied econometrics. Many forms of analysis in economics include the identification and distillation of patterns in data, and one of the most common tools is linear regression, although the rise in popularity of machine learning algorithms have made more accurate nonlinear methods such as classification, regression trees, random forests, penalized regression such as LASSO, neural networks, and support vector machines (SVMs)[13]

5 Dataset and Initial Manipulation

The headline data for this project was taken from the Kaggle dataset *Daily News for Stock Market Prediction*[14], with the top 25 headlines ranked by Reddit users on the WorldNews channel, each day from August 8, 2008 to July 1, 2016 (excluding non-trading days). The Combined_News_DJIA.csv file consists of 27 columns (two columns for date and label and 25 for the top 25 daily headlines) and 1989 rows, each corresponding to a specific day. The label column as in the original dataset referred to the Dow Jones Industrial Average and indicated 1 if the daily closing index value rose over the open or was constant, and indicated 0 if the index value fell. However, this index data was later changed to reflect the S&P 500 data.

Figure 2

[13] Varian, H. (2014). Big Data: New Tricks for Econometrics. *Journal of Economic Perspectives*, *28*(Spring), 3–28.
[14] Sun, J. (2016). *Daily News for Stock Market Prediction, Version 1*. https://www.kaggle.com/aaron7sun/stocknews

Partial Sample Row from Combined_News_DJIA.csv File From Kaggle

Date	Label	Top1	Top2	Top3	Top4
2008-08-08	0	b"Georgia 'downs two Russian warplanes' as countries move to brink of war"	b'BREAKING: Musharraf to be impeached.'	b'Russia Today: Columns of troops roll into South Ossetia; footage from fighting (YouTube)'	b'Russian tanks are moving towards the capital of South Ossetia, which has reportedly been completel...

Source: "Daily News for Stock Market Prediction," by J. Sun, 2016, *Kaggle*.
Note. Partial sample from the Combined_News_DJIA.csv file provided by the Kaggle dataset. The date column indicates the specific day, the label column indicates the numerical value attributed to the change in index value from open to close, and the next 25 rows contain byte strings of the top 25 news headlines for that day, ranked by Reddit users in the WorldNews channel.

For this project, the index value data for the S&P 500 was downloaded from the Thomson Reuters Refinitiv Workspace during the same time frame (August 8, 2008 to July 1, 2016). The labels were then modified for both binary and multiclass classification tasks. The labels were changed to represent 0 for a lower closing index value compared to the open value for the S&P 500 (a fall in price during the day) and 1 for a higher or constant closing value (the index value either rose or stayed the same).

The following modifications were made to the original Combined_News_DJIA.csv file for the binary classification (a similar process was used for multiclass classification, except the numerical labels were changed to reference ranges of percentage returns):

a) Deleted the date and label columns
b) Created a separate spreadsheet with the Refinitiv Workspace S&P 500 data, including the opening and closing index values
c) Added the new S&P 500 price data into the original spreadsheet and use conditional statements in Excel to populate a new 'Label' column
d) Deleted the S&P 500 index values
e) Added column headers (Label and Headline 1… Headline 25) to ensure easy translation from CSV file to pandas DataFrame
f) Converted the byte strings to standard strings for easier manipulation in Python by removing 'b' and excess ' and " characters (conversion can also be done in Python, although here the process was executed in Excel)

Figure 3
Sample of the First Row from Final Binary Classification CSV File

Label	Headline 1	Headline 2	Headline 3	Headline 4	Headline 5	Headline 6	Headline 7	Headline 8	Headline 9	Headline 10	Headline 11	Headline 12	Headline 13	Headline 14	Headline 15
1	Georgia 'downs two Russian warplanes' as countries move to brink of war"	BREAKING: Musharraf to be impeached.'	Russia Today: Columns of troops roll into South Ossetia; footage from fighting (YouTube)'	Russian tanks are moving towards the capital of South Ossetia, which has reportedly been completely destroyed by Georgian artillery fire'	Afghan children raped with 'impunity,' U.N. official says - this is sick, a three year old was raped and they do nothing"	150 Russian tanks have entered South Ossetia whilst Georgia shoots down two Russian jets.'	Breaking: Georgia invades South Ossetia, Russia warned it would intervene on SO's side"	The 'enemy combatent' trials are nothing but a sham: Salim Haman has been sentenced to 5 1/2 years, but will be kept longer anyway just because they feel like it."	Georgian troops retreat from S. Osettain capital, presumably leaving several hundred people killed. [VIDEO]'	Did the U.S. Prep Georgia for War with Russia?'	Rice Gives Green Light for Israel to Attack Iran: Says U.S. has no veto over Israeli military ops'	Announcing:Class Action Lawsuit on Behalf of American Public Against the FBI'	So--- Russia and Georgia are at war and the NYT's top story is opening ceremonies of the Olympics? What a fucking disgrace and yet further proof of the decline of journalism ."	China tells Bush to stay out of other countries' affairs"	Did World War III start today?'

Note. A sample of the first row of the final binary classification CSV file, with the label row and the next 25 rows indicating the top 25 news headlines.

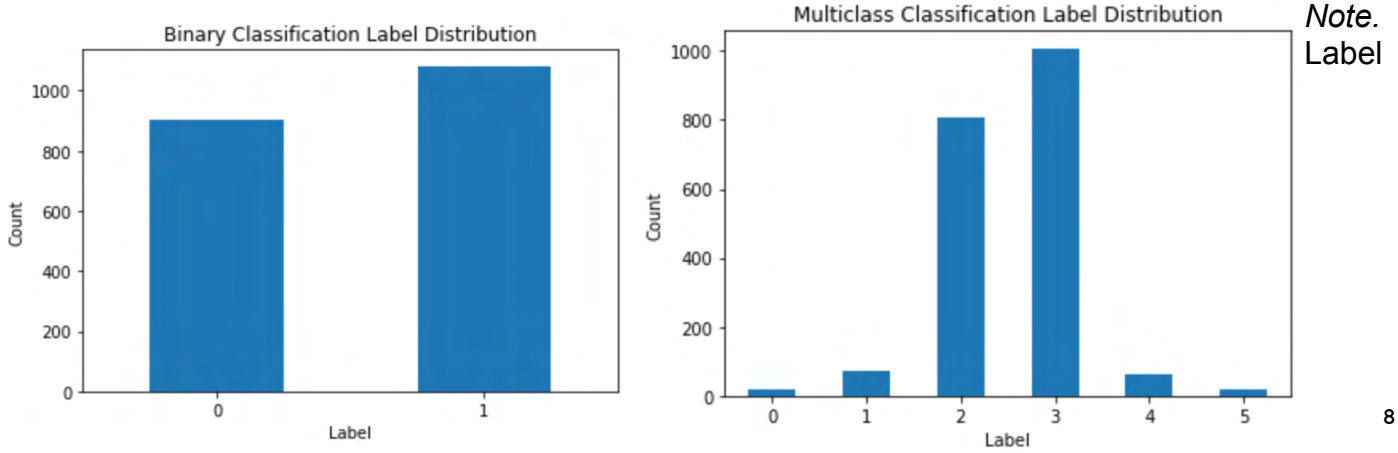
The multiclass labels were defined according to the following daily return percentage ranges:

a) 5 ⇒ Return of 4% or above
b) 4 ⇒ 2 to 4% return
c) 3 ⇒ 0 to 2% return
d) 2 ⇒ -2 to 0% return
e) 1 ⇒ -4 to -2% return
f) 0 ⇒ Below -4% return

Below is the label distribution for the binary and multiclass classification tasks. While label frequency for the binary classification data is approximately equal, the multiclass dataset is unevenly distributed, necessitating custom class weights computed by sklearn.utils.compute_class_weight. This lack of uniformity is likely caused by the great variability in stock market daily returns during the span of 8 years, especially when further broken down into more narrow categories.

Figure 4

Label Distribution of Binary and Multiclass Classification Tasks



Note. Label

frequencies for 2 class and 6 class classification. Clearly, the unevenly distributed data will vastly influence model performance, as the model will “memorize” the labels rather than learn the connections between the headlines and their corresponding labels. Therefore, custom class weights need to be computed, or some form of data resampling must occur on the dataset. Here, we will explore the weighting approach.

Figure 5

Sample of First 9 Rows from Final Multiclass Classification CSV File

Label	Top1	Top2	Top3	Top4	Top5	Top6	Top7	Top8	Top9	Top10	Top11	Top12	Top13	Top14	Top15
4	Georgia 'dc	BREAKING:	Russia Tod	Russian tar	Afghan chil	150 Russia	Breaking: C	The 'enemy	Georgian tr	Did the U.S	Rice Gives (	Announcin(	So---Russia	China tells	Did World '
3	Why wont .	Bush puts f	Jewish Geo	Georgian a	Olympic op	What were	Russia ang	An America	Welcome T	Georgia's n	Russia pres	Abhinav Bi	U.S. ship h	Drivers in a	The French
2	Remember	Russia 'end	"If we had	Al-Qa'eda i	Ceasefire ir	Why Micro	Stratfor: Th	I'm Trying t	The US mili	U.S. Beats \	Gorbachev:	CNN use fo	Beginning a	55 pyramid	The 11 Top
2	U.S. refuse	When the p	Israel clear	Britain\'s p	Body of 14	China has n	Bush annou	Russian for	The comma	92% of CNI	USA to sen	US warns a	In an intrig	The CNN Ef	Why Russia
3	All the exp	War in Sou	Swedish wr	Russia exag	Missile Tha	Rushdie Co	Poland and	Will the Ru	Russia exag	Musharraf	Moscow M	Why Russia	Nigeria has	The US and	Russia appa
3	Mom of mi	Russia: U.S	The govern	The Italian	Gorbachev:	China fakes	The UN's cr	Russian ger	Russia can	Russia-Geo	Business W	Under Sovi	Ministers h	Russia: Gec	Russians 'S
2	In an Afgha	Little girl, y	Pakistan's I	Tornado th	Britain's tei	Iran 'fires s	Rights of N	Tour of Tsk	The Great F	Over 190,0	Russia mov	a President	"Democrati	New Cold V	Georgian Si
2	Man arrest	The US mis	Schrder lan	Officials: 1(	These ten l	Russia seiz	Muslims ar	Taliban For	Assaults, ki	South Osse	Finally, an (	New York l	US left isol	Driven: Sha	NATO free
3	Two elderly	The Power	We had 55	"I live here	Russia senc	The Americ	Abkhazia o	Russia war	India Sets A	Elderly Chir	Plane skids	Taliban mo	150 Feared	Was Weste	Spanish Fre

Note. Sample of first 9 rows from the final multiclass classification CSV file (labels are now ranging from 0 to 5 depending on the S&P 500 daily return).

6 Data Preprocessing and Pre-Model Preparation

The methods for data preprocessing were slightly different with regard to the various models. For both LSTM models (binary and multiclass classification), a load_data function reads the CSV file using the csv module and iterates through each row to extract the labels and headlines per day into separate lists. Next, the headlines (X_train_token and X_val_token) are cleaned by removing punctuation, stopwords, and converting the string to lowercase before tokenizing each headline. Before creating a vocabulary and vectorizing the headlines, the maximum length of each headline is calculated by updating a counter and iterating over X_train_token and X_val_token (list with number of elements equal to some number of days, each with a list of 25 headlines, with each headline represented as a list of individual words, or tokens). Next, the vocabulary is defined by creating a dictionary and assigning each unique word in the combined train and validation headlines with a specific value, or number (represents a simple scalar or 1D vector). The buildVocabulary function that X_train_token and X_val_token are passed to returns the size of the vocabulary, a dictionary mapping words to indices (vectors), and another dictionary word_count that maps each unique word to the number of occurrences in the training and validation samples. Subsequently, a vectorizer function vectorizes each token in X_train_token and X_val_token by replacing zeros in a NumPy array of length 46 (maximum number of tokens for all headlines) with the token’s vector (corresponding index from the dictionary mapping unique words to indices). Finally, all labels are one-hot encoded by the Keras to_categorical() method.

For all models, custom class weights are computed to ensure classes that have less samples receive higher weights and classes with more samples are designated lower weights. This is one solution to address the problem of an unbalanced dataset.

For both the pre-trained BERT and GloVe embedding implementations, the bulk of preprocessing included reformatting the pandas DataFrame object (read from the CSV file) of dimensions 1989 x 26 into 49725 x 2 by establishing each headline as its own row. This means that each row, previously containing 25 headlines, becomes 25 rows with the same label and one headline each, thus, the dimensions are expanded.

To continue data preparation for the BERT model in TensorFlow, all train and validation headlines and labels are converted to TF tensors, and train and validation datasets are created by initializing a tf.data.Dataset.from_tensor_slices instance from the tensor headlines and labels and finally, creating a PrefetchDataset with buffer size tf.data.AUTOTUNE, meaning elements of the inputted datasets (the tf.data.Dataset.from_tensor_slices instance) are automatically retrieved before they are requested to be added to the new dataset, improving runtime. Next, the encoder[15] and preprocessing[16] handles are accessed from the Tensorflow Hub (the preprocessing handle is selected based on the chosen BERT model). More on the models themselves will be discussed in the Models and Results section.

The GloVe embeddings were extracted as a zip file and contain individual files with 50, 100, 200, and 300 dimensional versions of the GloVe vectors[17]. Additional preparation for the GloVe model included initializing a TextVectorization Keras layer with max_tokens set to 20000 and the output_sequence_length set to 300. The text dataset was then created as a TensorFlow dataset with tf.data.Dataset.from_tensor_slices with headlines as input, separating consecutive data elements into batches of 64. The vectorizer is then adapted on the text dataset with vectorizer.adapt(), rather than being fed a predefined vocabulary. The vocabulary is then extracted from the vectorizer and a dictionary mapping words to indices is created. Next, an embedding dictionary is generated mapping words to their NumPy vectorized representation. The predefined embedding dictionary is then used to produce an embedding matrix, where the vector at position *i* corresponds to the vector at index *i* in the dictionary. This is inputted to a Keras Embedding layer with the number of tokens equal to the number of words in the vocabulary + 2 (accounting for PAD and OOV tokens), the embedding dimension set to 300 (length of the GloVe vectors), and the embedding_initializer is set as keras.initializers.Constant(embedding_matrix). The trainable parameter of the layer is set to False, so the GloVe vectors are not updated during training.

7 Models and Outputs

This project utilizes three distinct architectures: pre-trained BERT, RNN (LSTM-based) and CNN-based (with GloVe embeddings). Both binary and multiclass classification models will be trained and tested using each of the above base architectures, for a total of 6 models. This section will detail the three main architectures, the slight variations in their binary and multiclass applications, and intricate visualizations describing the layers, inputs, outputs, and data transformations of each model. Regarding visualizations each type of layer is allocated a

[15] Turc, I., Chang, M.-W., Lee, K., & Toutanova, K. (2019, August 23). *Well-Read students learn better: On the importance of pre-training compact models.* ArXiv.Org. https://arxiv.org/abs/1908.08962.
[16] *TensorFlow hub*. (n.d.). Retrieved August 25, 2022, from https://tfhub.dev/tensorflow/bert_en_uncased_preprocess/3.
[17] (N.d.). Stanford. https://downloads.cs.stanford.edu/nlp/data/glove.6B.zip.

specific color, and large arrows between layers indicate the direction of forward propagation through the network. Differences between the implementation for the binary and multiclass tasks are also noted.

7a BERT Implementation

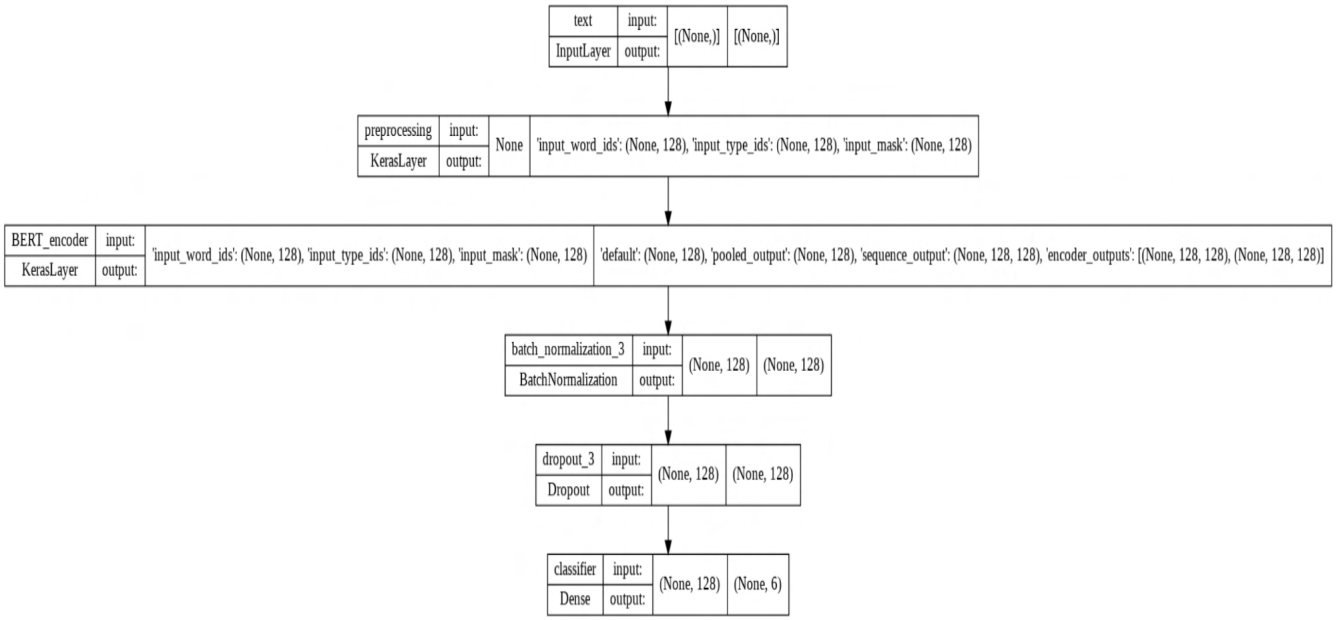
BERT is a novel approach to language modeling utilizing a transformer's bidirectional characteristic[18]. In essence, the BERT encoder reads an entire subset of text at once, allowing the model to learn the necessary surrounding context (hence bidirectional). While transformers in general consist of self-attention and fully connected layers, and an encoder and decoder, BERT is a language model, meaning it only requires the encoder to input and process text, and doesn't involve a decoder, which outputs predictions. In the model outlined below, the classifier is the final Dense layer with softmax activation. Additionally, each encoder contains multi-head self-attention components and a fully connected network[19]. Multi-head self-attention is a module for parallel processing, where each head learns unique weighted averages for input features (see Footnote 19).

While BERT is a powerful model that has revolutionized the NLP domain, there are limitations. BERT is primarily trained on English text from the BookCorpus (a vast series of unpublished novels, containing around 11,000 books and the English Wikipedia), although there are other corpora and models available in different languages. Additionally, BERT faces some of the same key issues as other language models in that it is impossible to eradicate all bias, and the pre-trained models can potentially contain offensive language and hate speech, including racial and other social biases[20]. Next, the vast computing power and capability of BERT calls for immense and efficient resource allocation. The full-scale BERT models like BERT-base and BERT-large often consist of hundreds of millions of parameters and hours-long training times. This project utilizes a miniature BERT model with two transformer blocks, a hidden size of 128 and 2 attention heads.

The BERT implementation in this project involves a Keras Input layer for the headline input, a preprocessing layer defined by the BERT preprocessing handle, an encoder layer defined similarly with the BERT encoder handle (both defined with hub.KerasLayer), the pooled outputs of the encoder (contextual embeddings from BERT output layer), a batch normalization layer, a dropout layer, and the final Dense classifier with the given number of output classes (6 for multiclass and 2 for binary). The following is the architecture of the BERT classifier (the only difference between the binary and multiclass models is the number of units in the final dense layer classifier).

[18] Horev, R. (2018, November 17). BERT Explained: State of the art language model for NLP. *Towards Data Science*. https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270.
[19] Özateş, M. N. (2021, February 20). Transformer architecture: How transformer models work? *CARBON CONSULTING*. https://medium.com/carbon-consulting/transformer-architecture-how-transformer-models-work-46fc70b4ea59.
[20] Ahn, J., & Oh, A. (2021, September 13). *Mitigating language-dependent ethnic bias in BERT*. ArXiv.Org. https://arxiv.org/abs/2109.05704.

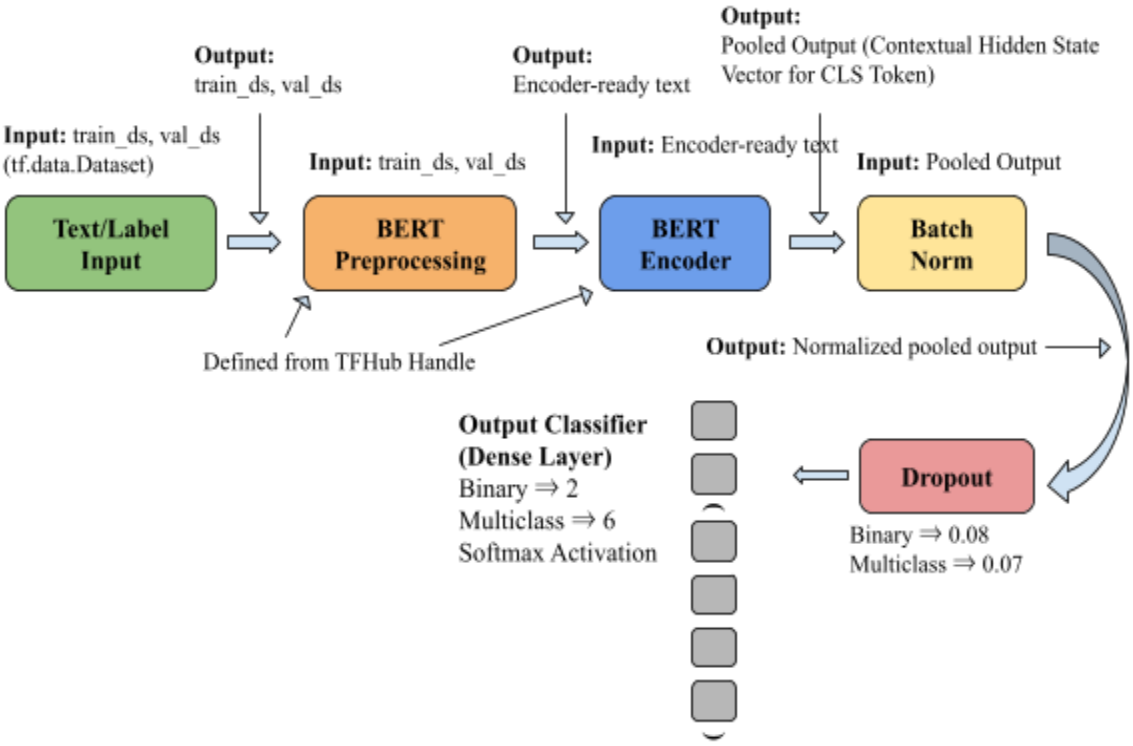
Figure 6
BERT Model Architecture



Note. Pre-trained BERT architecture illustrating the inputs and outputs of each layer and their shapes.

As depicted in the figure above, the shape of the input layer is None as the shape of the textual data input (headlines arrays) is unknown in certain dimensions, such as batch size (during training, an error was also raised if the shape was defined in the model). This means the number of samples are free to be chosen during training. The length of 128 on the second axis refers to the maximum sequence length for the tokenized and vectorized headlines. The shape of the pooled_outputs is (batch_size, max sequence len) and the output itself is a contextual hidden state vector of the CLS token (provides enough context for the sequence).

Figure 7
BERT Visualization with Layer Inputs and Outputs



Note. Visualization of the BERT model, with different labels colored differently. The general model structure is headline and label input, BERT preprocessing, the BERT encoder, batch normalization, dropout, and the final classifier. The model is relatively simple aside from the BERT components as the purpose of the model is to gain a more complete understanding of BERT functionality in price prediction.

More specifically, the train and validation datasets (Prefetch Datasets) are fed to the Input Layer, which sends the outputs to the BERT preprocessing layer chosen based on the encoder handle from the TensorFlow Hub. The preprocessing model prepares and returns the headline text to be encoded. The encoder returns the pooled_output (the hidden state vector for the CLS token at the beginning of the headlines that provides necessary context), that is normalized through batch normalization. The following dropout layer randomly inactivates 8% and 7% of neurons for the binary and multiclass tasks respectively, and the final classifier has two output units for the binary classification and six for the multiclass classification, with softmax activation.

More on specific hyperparameters below:

a) 1 epoch for binary classification as opposed to 2 epochs for multiclass classification
b) Batch size is 10
c) Training steps per epoch defined as cardinality or length of train_ds
d) Total training steps is equal to the number of training steps per epoch multiplied by the number of epochs

e) Number of warmup steps for the learning rate annealing approach is set to 10% of the number of total training steps
f) Initial learning rate is 3e-5
g) Sparse Categorical Cross Entropy loss due to integer labels
h) Sparse Categorical Accuracy metric
i) AdamW optimizer (same as original BERT training) with learning rate warmup with the parameters set above
j) L2 regularizer on output classifier set to 1e-4 for binary classification as opposed to 3e-4 for multiclass classification

The BERT models achieved an accuracy of 54% for the binary classification and 39% for the hex factor classification. The model performed only slightly better than random guessing (50%) for the binary classification task, although it performed around twice as well as random guessing (17%) for the multiclass (6 class) classification task.

7b RNN (LSTM) Implementation

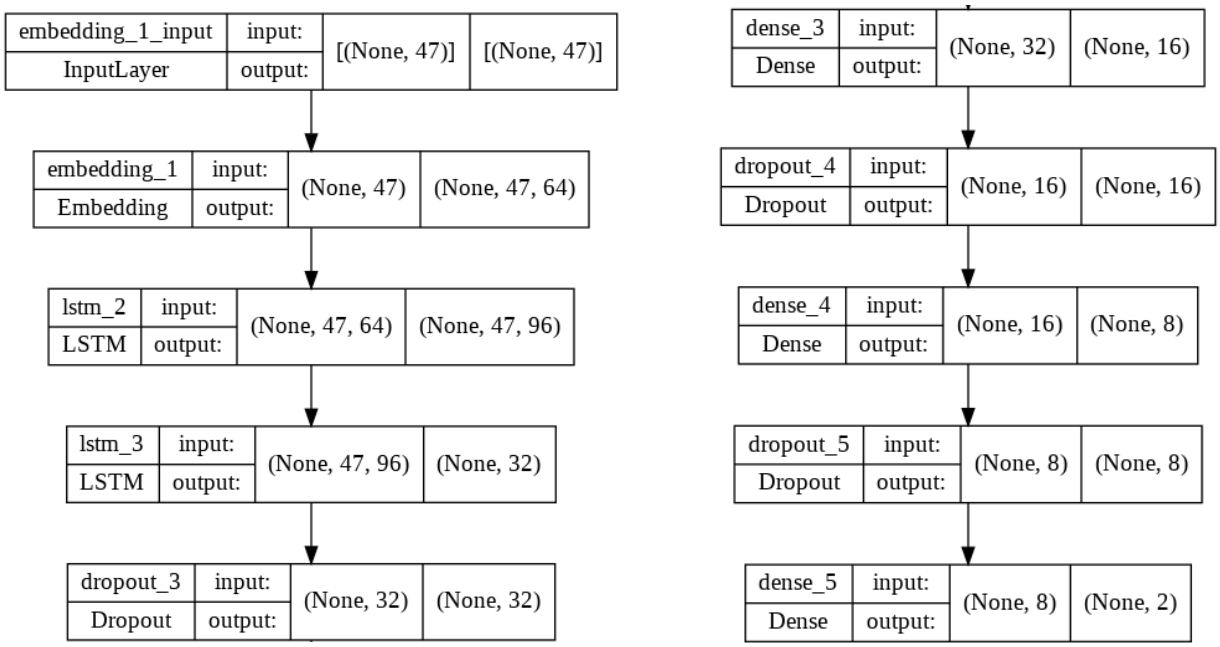
Before the advent of BERT and other state-of-the-art language models, Long Short-Term Memory (LSTM) models were much more widely utilized in text classification and NLP. These models have an intrinsic advantage in being able to memorize and learn feature patterns, preserve, and propagate them throughout a neural network[21]. This trait also means that LSTMs can be fed whole sentences and phrases in addition to sole words, improving model understanding and accuracy. LSTMs are extensions of traditional recurrent neural networks (RNNs) and are especially useful in analyzing patterns in temporal data, as is the case with stock prediction in this project[22].

The LSTM implementation in this project involves an embedding layer which receives inputted headlines, two LSTM layers with 96 and 32 units respectively, a dropout layer, a dense layer with 16 units and ReLu activation, another dropout layer, a dense layer with 8 units and ReLu activation, a final dropout layer, and the dense classifier with softmax activation and number of units equal to the number of output classes (2 for binary and 6 for multiclass classification tasks).

Figure 8
LSTM Model Architecture

[21] *LSTM for text classification*. (2021, June 14). Analytics Vidhya. https://www.analyticsvidhya.com/blog/2021/06/lstm-for-text-classification/.

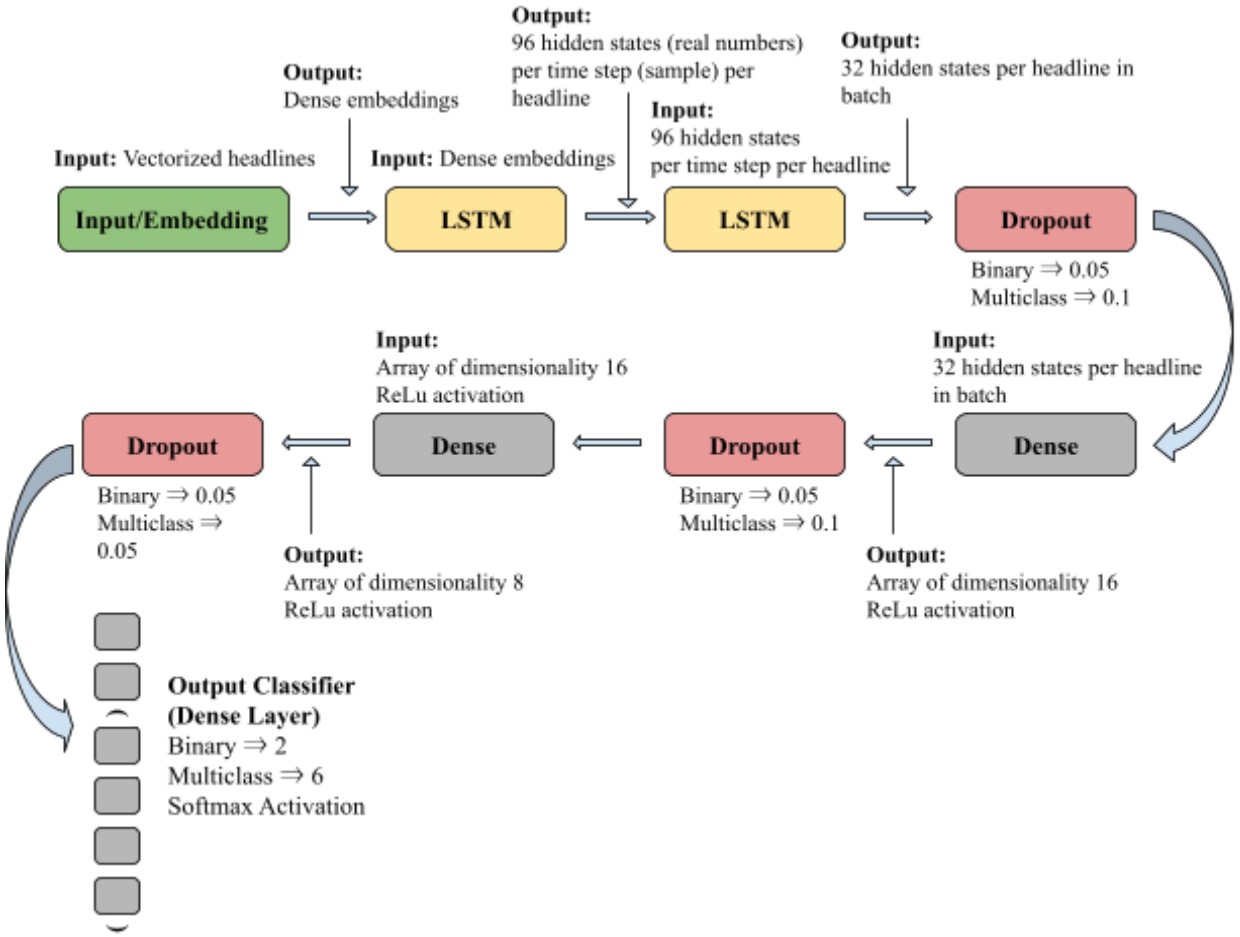
[22] *Sentiment analysis with LSTM*. (2022, January 17). Analytics Vidhya. https://www.analyticsvidhya.com/blog/2022/01/sentiment-analysis-with-lstm/.



Note. The LSTM architecture illustrates the inputs and outputs of each layer and their shapes. The model reads from the left column to the right (the dropout_3 layer outputs to the dense_3).

To elaborate, the input receives vectorized headlines of length 47, and the Embedding layer outputs dense embeddings of dimension 64 (output length) for each headline. These embeddings are passed to the following LSTM layers, with 96 and 32 units respectively, before a series of dropout and dense layers ending at the final dense classifier with number of units equal to the number of output classes.

Figure 9
LSTM Model Visualization with Layer Inputs and Outputs



Note. Visualization of the LSTM model, with different layers colored differently. The structure of this model is input, embedding, LSTM layers, three instances of dropout followed by a dense layer before the final classifier.

With the LSTM model, the vectorized train and validation headlines are fed to the input of the embedding layer, which outputs dense embeddings for the headlines to the first LSTM. This LSTM outputs 96 hidden states (corresponds to the dimensionality of the output vector space defined by the number of units) for each time step per headline/sample in the batch. The next LSTM outputs 32 hidden states per batch (rather than per time step as return_sequences=False). The rate for the following dropout layer is set 0.05 for the binary classification and 0.1 for the multiclass classification. The next dense layer takes the input from the LSTM that has not been inactivated by the dropout and outputs an array of dimensionality 16 with ReLu activation. This continues with another dropout and dense layer, with the dense layer outputting an array with length 8 instead. The final dropout layer has a rate of 0.05 and the final classifier with softmax activation outputs predictions.

More on the specific hyperparameters below:

a) One epoch for binary classification as opposed to two epochs for multiclass classification
b) Batch size is 4
c) Adam optimizer with learning rate 0.0001

d) Categorical Cross Entropy loss due to one-hot encoded labels
e) Accuracy metric

The LSTM models achieved an out-of-sample accuracy of 46% for the binary classification and 41% for the multiclass classification. The model performed slightly worse than random guessing (50%) for the binary classification task, although the multiclass (6 class) accuracy was more than double that of random guessing (17%).

7c GloVe Embeddings

In addition to language models, pre-trained word embeddings are also showing great promise in natural language processing. Word embeddings are a method by which to represent words so that words similar in meaning have similar representations, or vectors[23]. Often, word embeddings are in a machine-understandable format to translate words into numbers.

This project utilizes pre-trained GloVe embeddings. GloVe is an unsupervised algorithm that produces contextual word embeddings, developed at Stanford by counting the number of times a particular word is present in the same or similar context as another word (co-occurrence matrix)[24]. The embeddings that are produced demonstrate broad patterns between words and ideas, also known as linear substructures when the words are depicted in a vector space.

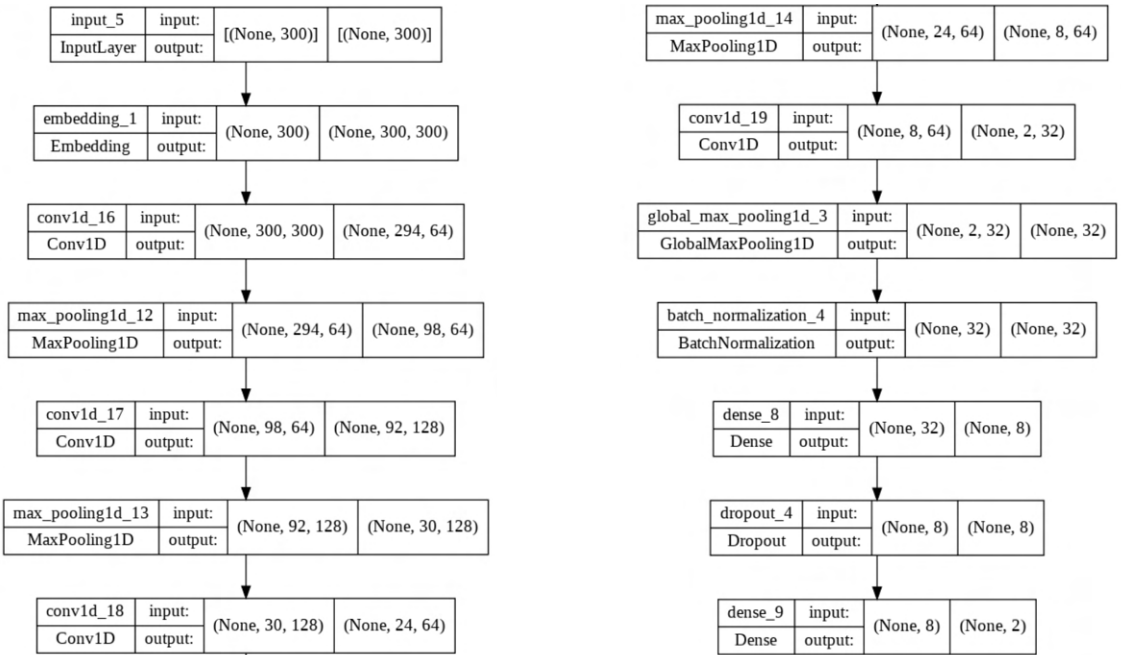
Furthermore, many within the NLP sector have begun to explore the use of Convolutional Neural Networks (CNNs) in sentiment analysis and text classification tasks. While CNNs are generally thought of as inherent to computer vision and image processing, the core concepts can be translated to NLP tasks. In addition, text inputs, such as news headlines, have similar structures to images, and therefore well-trained convolutional layers can be well-suited to analyze minute patterns and features within text. For example, images can be represented as matrices of pixel values, similar to how text can be represented numerically as arrays, or matrices, of word vectors or embeddings. In the same way that a filter convolves over an image and creates a feature map, filters, or weight matrices, can slide horizontally across a sentence represented by a series or array of word vectors and compute weighted sums to create feature maps, input to activation functions, and output values[25].

The GloVe implementation in this project includes an input layer with shape equal to (embedding_dim,), where the embedding dimension is defined as the length of the GloVe vectors (300). Next, the predefined GloVe embedding layer (defined by the embedding matrix) returns the 300 dimensional embeddings for each headline, followed by a 1D convolutional layer, and a max pooling layer. These last two layers are repeated twice more, and a fourth convolutional layer is followed by a global max pooling layer. Next, data is normalized via a batch normalization layer, passed to a Dense layer with 8 units and ReLu activation, a dropout layer, and the final output classifier with softmax activation and the number of units equal to the number of output classes.

---

[23] Brownlee, J. (2017, October 10). *What are word embeddings for text?* Machine Learning Mastery. https://machinelearningmastery.com/what-are-word-embeddings/
[24] Chawla, J. S. (2020, July 6). What is GloVe? - Analytics Vidhya - Medium. *Analytics Vidhya*. https://medium.com/analytics-vidhya/word-vectorization-using-glove-76919685ee0b
[25] Binhuraib, T. (2020, October 16). NLP with CNNs. *Towards Data Science*. https://towardsdatascience.com/nlp-with-cnns-a6aa743bdc1e

Figure 10
GloVe Model Architecture

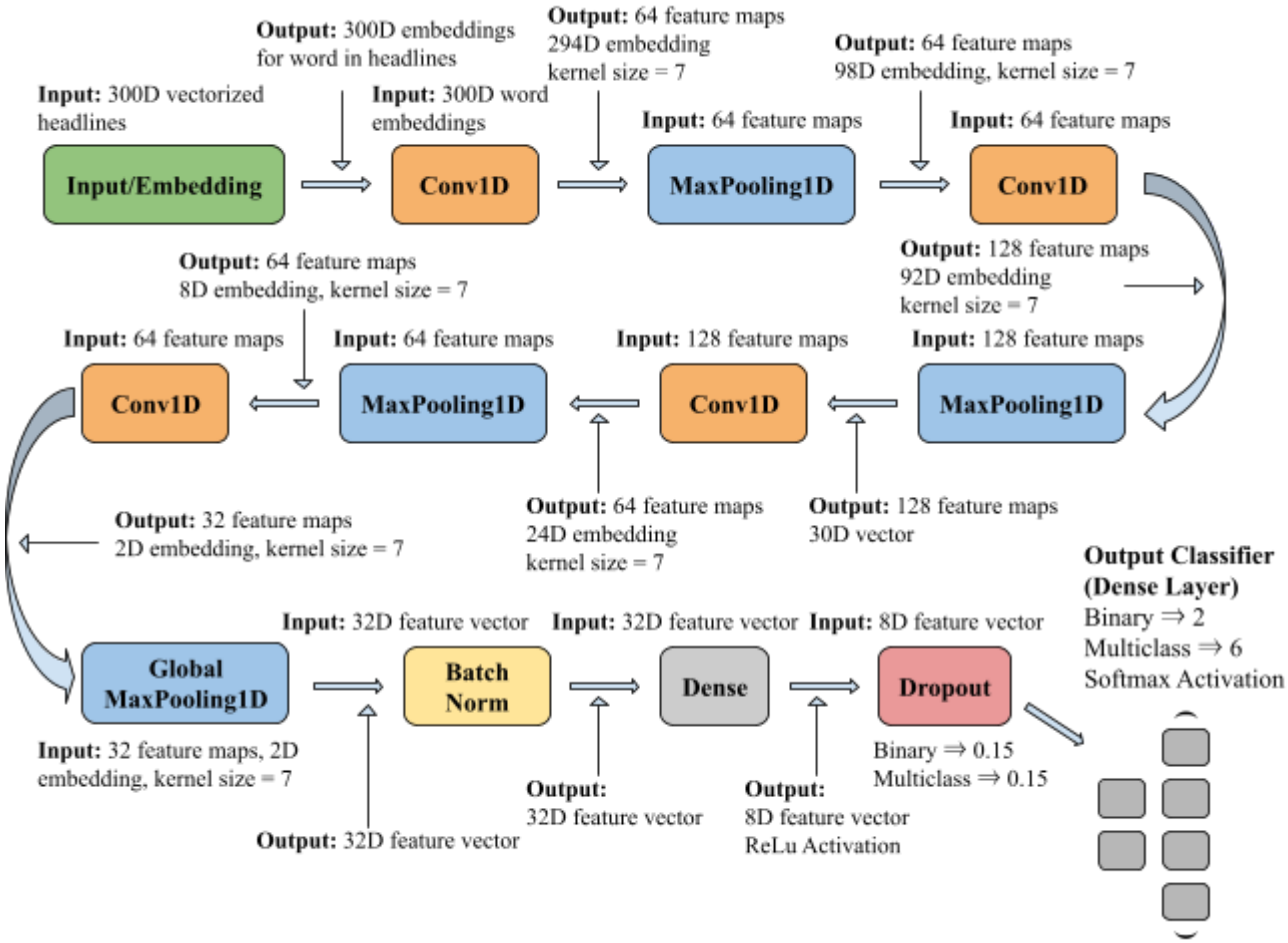


Note. The LSTM architecture illustrates the inputs and outputs of each layer and their shapes. The model
reads from the left column to the right (the conv1d_18 layer outputs to the max_pooling1d_14 layer).

Regarding the input layer, the shape (None, 300) indicates the model expects batches of 300 dimensional vectors (vectorized headlines). The embedding layer that follows outputs 300 dimensional vectors for each word within each headline. The values in the output shape of the convolutional layers indicate the number of steps or convolutions across the input and the number of filters respectively. The final classifier is a dense layer with the number of units equal to the number of output classes.

Figure 11

GloVe Model Visualization with Layer Inputs and Outputs



Note. Visualization of the GloVe model, with different layers colored differently. The basic structure is input/embeddings followed by three instances of a convolutional layer with max pooling, a fourth convolutional layer, global max pooling, batch normalization, a dense layer, dropout, and the output classifier.

In the GloVe model, the train and validation headlines are inputted to the Embedding layer in the form of 300-dimensional vectors. These vectors are generated according to a Keras TextVectorization layer initialized during the pre-model preparation stage, with max_tokens set to 20000 and the output sequence length to 300 (same length as the GloVe vectors). These vectorized headlines are passed to the input of the embedding layer and then to the layer itself (defined by the embedding matrix also created in the pre-model preparation phase), where the word vectors are replaced by their corresponding GloVe embeddings. The outputs of the embedding layer are 300-dimensional vectors for each word in each headline. These outputs are inputted to the first convolutional layer that slides over the headline with 64 filters of size 7x7, creating 64 unique feature maps and an output shape of (294, 64). The subsequent max pooling layer further shrinks the embedding vector to 98D, and these two layers are repeated in this manner 3 times, with the second convolutional layer outputting 128 feature maps instead of

64. In this time, the dimensionality of the embedding vector is constantly becoming smaller. Next, a fourth convolutional layer receives an 8D vector and 64 feature maps and outputs 32 feature maps and a 2D embedding, that is in turn transformed into a simple 32D vector by a global max pooling layer. After a batch normalization layer, a dense layer, and a dropout layer with rate 0.15, the final output classifier outputs probabilities with softmax activation.

More on specific hyperparameters below:

a) Seven epochs for binary classification as opposed to two epochs for multiclass classification
b) Batch size is 20
c) Training steps per epoch is defined as cardinality or length of train_df (this DataFrame contains the train headlines and labels)
d) Total training steps is equal to the number of training steps per epoch multiplied by the number of epochs
e) Number of warmup steps for the learning rate annealing approach is set to 10% of the number of total training steps
f) Initial learning rate is 3e-5
k) Sparse Categorical Cross Entropy loss due to integer labels
l) Sparse Categorical Accuracy metric
m) AdamW optimizer with learning rate warmup with the parameters set above
n) GloVe embedding layer trainable is set to False to ensure the embeddings are not updated during training

The GloVe models achieved a validation accuracy of 51% for the binary classification and 45% for the multiclass classification. The model performed slightly better than random guessing (50%) for the binary classification task, and the multiclass (6 class) accuracy was more than double that of random guessing (17%).

8 Final Results

All 6 models tested experienced various degrees of accuracy in predicting stock price fluctuations based on news headlines.

Table 1

Comparing Model Accuracies Among 3 Base Architectures

	Binary	Multiclass	Parameters
BERT	54%	39%	~ 4,300,000
LSTM	46%	41%	~ 2,400,000
GloVe	51%	45%	~ 6,300,000

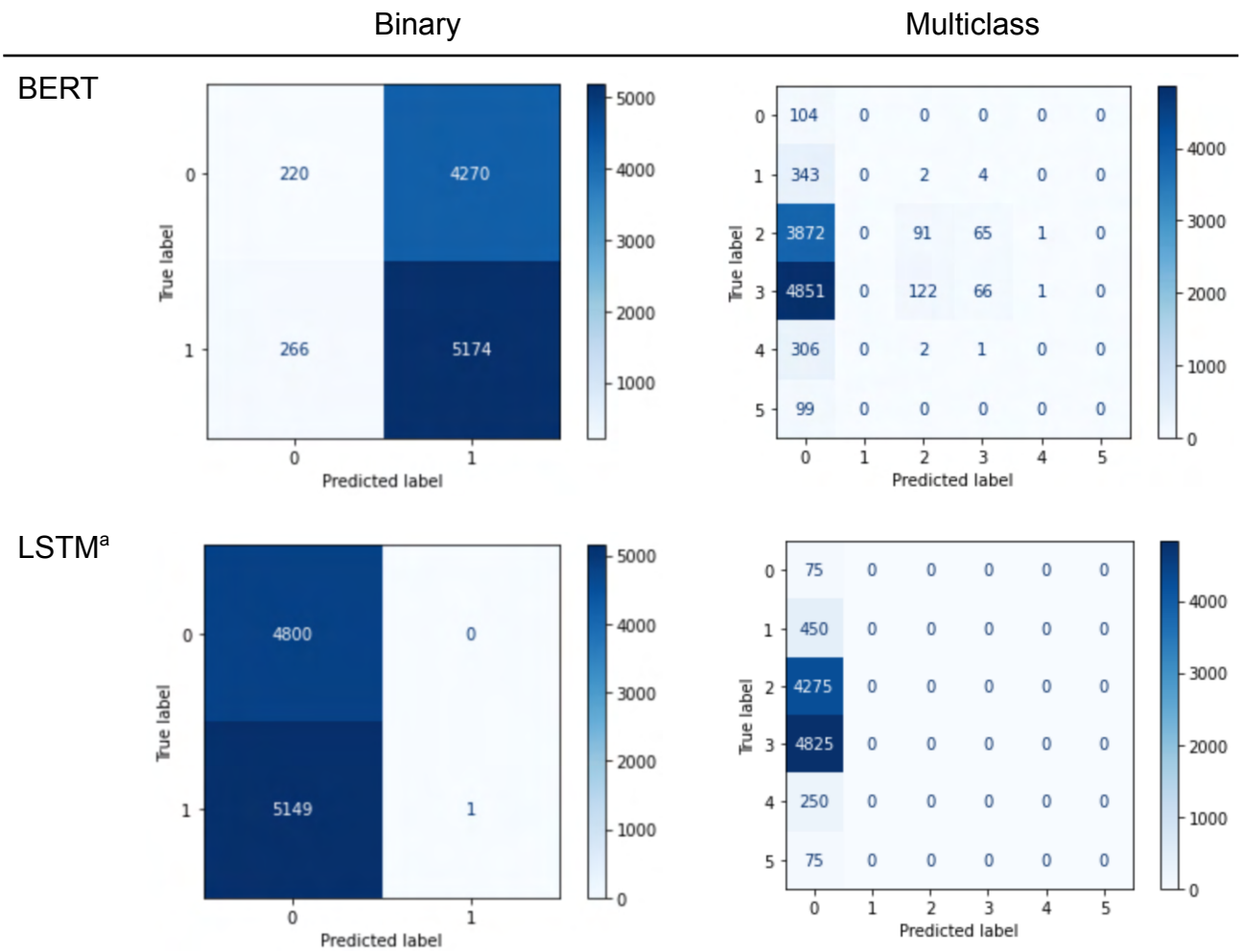
Note. This table demonstrates the highest accuracies achieved by each base architecture in both the binary and multiclass tasks, with the approximate number of parameters of the model.

As described in the table above, the BERT model achieved the highest accuracy on the binary classification task (54%), while the GloVe model achieved the highest accuracy on the multiclass (6 class) classification task (45%).

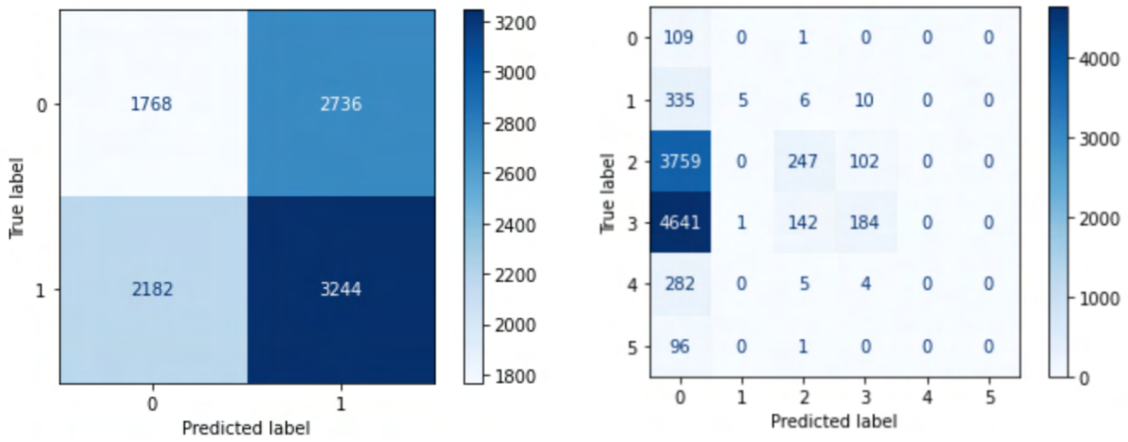
Another useful technique to visualize out-of-sample model accuracy is a confusion matrix. Below are the matrices for all three base architectures in both binary and multiclass classification tasks.

Figure 12

Confusion Matrices for BERT, LSTM, and GloVe Models



GloVe



[a]The LSTM multiclass matrix only has predictions for a single class. This may be due to the fact that the LSTM model could not map input features to output classes specifically enough for the other classes 1-5, and 'defaulted' to class 0.

Generally, it seems like all the models are inclined to predict class 0 for the multi classification task. This may be because class 0 indicates returns of below -4%, which potentially reflects very evidently in the sentiment or content of the headlines that lead to that category of market returns, as opposed to more subtle changes in index value, such as within a few percent.

Conversely, the models never predicted class 5 for the multiclass prediction task, as many factors may influence whether or not an index value increases by 4% or greater.

For the binary tasks, it is also interesting to observe the patterns in the matrices among the various architectures. While the BERT model tended to predict class 1, the LSTM model tended to predict class 0 and the GloVe model seemed to strike a balance with more evenly distributed predictions. This may be due to the unique methods through which the models learn, i.e. BERT's encoder or preprocessing mechanisms and the pre-trained GloVe embeddings.

9 Discussion and Conclusion

While neither of the models performed exceptionally, these models provide a key foundation to further exploring sentiment analysis in economics, specifically in time series forecasting. Two overwhelming questions remain:

a) What other factors would be essential to creating a more accurate model?
b) What are the impacts of technical/fundamental factors on market and model performance

The scope of these models is solely limited to news headlines. Therefore, due to the vast array of factors that impact economic performance, it is exceedingly difficult to develop an accurate, market-ready model exclusively based on news headlines. In fact, it is extremely difficult to create an accurate model based wholly on any single or handful of variables; accurate economic forecasting requires a comprehensive approach, although this can pose a challenge for individuals or researchers without great computational capabilities and access to sensitive or private data. Other factors that impact stock price on a broad or company-specific level can

include laws and policies, tax rates, individual company metrics, and beyond, and the nuance in economic analysis inherently means that a very simple solution to forecasting a broad market will be impossible.

For individuals and market researchers alike, it may also be worthwhile to explore similar strategies as outlined in this paper for specific companies, analyzing both technical and fundamental factors such as revenue and accounting statistics and company reviews and public sentiment. Additionally, it could be useful to explore the effect of tracking multiple different markets or exchanges and observing the impact of a set of headlines on the performance of these various markets.

Architecturally, it will be fascinating to continue investigating various model types to discover novel approaches to building more accurate forecasting models, such as combining various architectures or creating even deeper networks.

References

[1] Ahn, J., & Oh, A. (2021, September 13). *Mitigating language-dependent ethnic bias in BERT*. ArXiv.Org. https://arxiv.org/abs/2109.05704

[2] *Applications of machine learning - Javatpoint*. (n.d.). Www.Javatpoint.Com. Retrieved August 17, 2022, from https://www.javatpoint.com/applications-of-machine-learning

[3] Binhuraib, T. (2020, October 16). NLP with CNNs. *Towards Data Science*. https://towardsdatascience.com/nlp-with-cnns-a6aa743bdc1e

[4] Brown, R. (2021, September 2). What are the different types of sentiment analysis ? *Nerd For Tech*. https://medium.com/nerd-for-tech/what-are-the-different-types-of-sentiment-analysis-808f36ef89ee

[5] Brownlee, J. (2017, October 10). *What are word embeddings for text?* Machine Learning Mastery. https://machinelearningmastery.com/what-are-word-embeddings/

[6] Burton, J. (2013, June 17). 5 charts to tell if stock buyers are too bullish. *MarketWatch*. https://www.marketwatch.com/story/5-charts-to-tell-if-stock-buyers-are-too-bullish-2013-06-17

[7] Chawla, J. S. (2020, July 6). What is GloVe? - Analytics Vidhya - Medium. *Analytics Vidhya*. https://medium.com/analytics-vidhya/word-vectorization-using-glove-76919685ee0b

[8] *Context analysis in NLP: Why it's valuable and how it's done*. (2019, February 19). Lexalytics. https://www.lexalytics.com/blog/context-analysis-nlp/

[9] Corporate Finance Institute. (2019, March 26). Fundamental analysis. *Corporate Finance*

Institute. https://corporatefinanceinstitute.com/resources/knowledge/trading-investing/fundamental-analysis/

[10] Gentzkow, M., Kelly, B., & Taddy, M. (2019). Text as data. *Journal of Economic Literature*, *57*(3), 535–574. https://doi.org/10.1257/jel.20181020

[11] Graham, B., & Dodd, D. (2008). *Security Analysis: Sixth Edition, foreword by Warren Buffett*. Mcgraw-hill.

[12] Herz, F., Ungar, L., Eisner, J., & Labys, W. (2014). *Stock market prediction using natural language processing*. https://patentimages.storage.googleapis.com/df/93/5d/4cc361daa8ee8c/US20030135445A1.pdf

[13] Horev, R. (2018, November 17). BERT Explained: State of the art language model for NLP. *Towards Data Science*. https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270

[14] IBM Cloud Education. (2020, August 17). *What are Neural Networks?* IBM. https://www.ibm.com/cloud/learn/neural-networks

[15] Kurohara, J., Chang, J., & Hoskins, C. (2018). Predicting Stock Market Movements Using Global News Headlines. *CS230*.

[16] *Lexicon-Based sentiment analysis: A tutorial*. (n.d.). KNIME. Retrieved August 17, 2022, from https://www.knime.com/blog/lexicon-based-sentiment-analysis

[17] *LSTM for text classification*. (2021, June 14). Analytics Vidhya. https://www.analyticsvidhya.com/blog/2021/06/lstm-for-text-classification/

[18] Özateş, M. N. (2021, February 20). Transformer architecture: How transformer models work? *CARBON CONSULTING*. https://medium.com/carbon-consulting/transformer-architecture-how-transformer-models-work-46fc70b4ea59

[19] Repustate Team. (2022, January 4). Aspect based sentiment analysis. *Repustate*. https://www.repustate.com/blog/aspect-based-sentiment-analysis/

[20] *Sentiment analysis with LSTM*. (2022, January 17). Analytics Vidhya. https://www.analyticsvidhya.com/blog/2022/01/sentiment-analysis-with-lstm/

[21] Sun, J. (2016). *Daily News for Stock Market Prediction, Version 1*. https://www.kaggle.com/aaron7sun/stocknews

[22] *TensorFlow hub*. (n.d.-a). Retrieved August 25, 2022, from https://tfhub.dev/tensorflow/small_bert/bert_en_uncased_L-2_H-128_A-2/2

[23] *TensorFlow hub*. (n.d.-b). Retrieved August 25, 2022, from https://tfhub.dev/tensorflow/bert_en_uncased_preprocess/3

[24] Turc, I., Chang, M.-W., Lee, K., & Toutanova, K. (2019, August 23). *Well-Read students learn better: On the importance of pre-training compact models*. ArXiv.Org. https://arxiv.org/abs/1908.08962

[25] Varian, H. (2014). Big Data: New Tricks for Econometrics. *Journal of Economic Perspectives*, *28*(Spring), 3–28.

[26] *What is Technical Analysis and How Does it Work?* (n.d.). Nadex. Retrieved August 18, 2022, from https://www.nadex.com/learning/introduction-to-technical-analysis/

[27](N.d.). Stanford. https://downloads.cs.stanford.edu/nlp/data/glove.6B.zip