FINANCIAL ECONOMETRICS (FIN-407)

STOCK VOLATILITY FORECASTING USING TWITTER DATA: AN EMPIRICAL STUDY ON TESLA'S STOCK AND ELON MUSK'S TWEETS

Group 7

Maxim J. Oppitz

Tuomas J. A. Pääkkönen

Ali ElGuindy

Selim Khalfallah

June 4, 2023

Abstract

The increasing prevalence and influence of social media platforms like Twitter present a rich source of data for financial market analysis. This study harnesses this dormant potential by investigating the predictive power of Twitter data, specifically Elon Musk's tweets, on the volatility of Tesla's stock. We employ natural language processing to extract relevant features from the text and numerically represent sentiment. The study then examines the impact of these features on stock volatility using a blend of well-established econometric and machine learning techniques. Considering the unique position of Elon Musk as CEO and his notable influence on the stock market through Twitter, the research aims at providing novel insights into the effect of corporate communication on financial markets. Our results, while not entirely conclusive, indicate that the sentiment variable might hold some predictive power, especially within the framework of more complex models. This suggests a potential avenue for future exploration, underscoring the nuanced role of sentiment analysis in the realm of financial forecasting. The research culminates in a comparative analysis of all approaches, providing a comprehensive view on the effectiveness and limitations of the different methods in the context of stock volatility forecasting using Twitter data. The outcomes of this research contribute to the evolving field of volatility modelling by integrating unconventional data sources and diverse analytical methodologies.

Contents

1 Introduction

1.1 Motivation

In the evolving landscape of financial markets, there exists a potential for a relationship between corporate communication and the volatility of stocks. These communications, once confined to formal announcements and press releases, have extended into the realm of social media, amplifying the reach and immediacy of corporate disclosures. As an innovative channel of information dissemination, social media introduces an additional layer of complexity to the intricate dynamics of stock markets, thus warranting an in-depth investigation of its influence.

Traditional financial theories, such as the Efficient Market Hypothesis, argue that all public information is instantly incorporated into stock prices, rendering it difficult to predict stock movement based on publicly available data. However, these theories often do not account for the nuances of human sentiment and perception, particularly as they relate to the interpretation of corporate communication on social media. Herein lies the unique value of our research: it ventures into the relatively uncharted waters of examining the potential impact of sentiment, expressed through social media by key corporate personalities, on stock volatility.

Our empirical case study focuses on Tesla, Inc., an American electric vehicle and clean energy company, and the tweets of its CEO, Elon Musk, who is also a public figure and visionary entrepreneur. This case is particularly relevant due to several distinguishing factors. First, Tesla, under Musk's leadership, has been at the forefront of significant advancements in the electric vehicle industry, making it a company of interest to investors, market analysts, and the public. Its stock is highly traded, leading to potentially high volatility. Second, Musk's active presence on Twitter and his propensity to share substantial company-related information make for a unique case of study. He has, on several occasions, used the platform to announce strategic corporate decisions, product updates, and future visions, often leading to immediate market reactions. This situation presents a real-world instance of the potential for corporate communication, particularly through social media, to influence stock volatility. However, Musk's use of Twitter has been far from traditional, often resulting in controversy and significant market fluctuations. This unusual behavior deviates from conventional corporate communication norms and may introduce new forms of volatility in Tesla's stock, providing us with a rich context to analyze the effects of social media-based corporate communication on stock volatility.

Amidst the evolving dynamics of this field, the academic discourse is on the brink of exciting discoveries in the field of volatility forecasting using text sentiment. Therein, this research is driven by a desire to enrich our understanding of the fascinating interplay between social media, corporate communication, and stock market volatility.

Through a blend of econometric and machine learning techniques, we embark on this exploratory journey to uncover the relationship between Twitter data, specifically sentiment expressed in Elon Musk's tweets, and Tesla's stock volatility. Our research is not solely aimed at proposing a foolproof model for predicting stock prices but also seeks to provide a comprehensive understanding of the underlying dynamics that govern this complex system.

1.2 Review of Literature

The use of social media data in predicting stock market trends has been a subject of interest in recent years. Awan et al. (2021) explored the use of machine learning models to study the stocks of top companies using data from social media and historical stock prices. Their study demonstrated that models such as linear regression, random forest, and generalized linear regression could provide an accuracy of 80% to 98% in predicting share price movements. This aligns with our approach, where we also employ machine learning algorithms, specifically Long Short-Term Memory (LSTM), to forecast stock volatility. Similarly, Mehta et al. (2021) proposed a stock price prediction tool that considers public sentiment, opinions, news, and historical stock prices to forecast future stock prices. Their experiments were performed using machine-learning and deep-learning methods including Support Vector Machine, MNB classifier, linear regression, Naïve Bayes and LSTM. Their successful use of LSTM further validates our choice of this particular machine learning algorithm. Maguluri and Rengaswamy (2020) also focused on predicting stock market trends using real-time stock technical data and stock social media data. Their work underscores the potential of social media data in providing real-time insights for stock market predictions, a strategy that we also employ in our study by using Twitter data.

In a slightly different context, Valencia et al. (2019) applied similar techniques to predict the price movement of cryptocurrencies. They found that Twitter data alone could be used to predict certain cryptocurrencies and that neural networks outperformed other models. This suggests that our focus on Twitter data for predicting stock volatility is well-founded. Skuza and Romanowski (2015) designed a system that used sentiment analysis of Twitter data within a big data distributed environment for stock prediction. Their work demonstrated the efficiency of the chosen approach in predicting stock prices, reinforcing our decision to use Twitter data in our study. In the context of volatility forecasting, Xing et al. (2019) explored sentiment-aware volatility forecasting. Their work highlights the importance of considering public sentiment in forecasting stock volatility, a factor that we incorporate in our study. Similarly, Zhang et al. (2023) focused on volatility forecasting with machine learning and intraday commonality. Their work underscores the potential of machine learning techniques in predicting stock volatility, aligning with our use of LSTM for this purpose.

In conclusion, the literature suggests that social media data, particularly from Twitter, combined with machine learning techniques, can be a powerful tool in predicting stock market trends and volatility. Our study builds on this existing research by using a mixture of econometric methods and machine learning algorithms to forecast stock volatility using Twitter data. However, the accuracy of these predictions can vary, and further research is needed to refine these models and improve their predictive power.

2 Data

2.1 Twitter Data Acquisition & Preprocessing

To acquire Elon Musk's tweets for analysis, we accessed the dataset available on the website: here. The dataset features a comprehensive collection of tweets from Elon Musk, who is the

incumbent CEO of Tesla. His tweets are seen as a rich resource for insights and sentiment concerning Tesla and its shares. The dataset we procured consists of two columns - date and tweet. The tweets dated from December 2, 2017, at 19:33:00 up until May 10, 2023, at 05:44:00, providing a vast spectrum of Elon Musk's thoughts and communications. However, to construct a dataset suitable for volatility analysis, we concentrated on a particular time frame. We handpicked tweets that were posted starting from January 1, 2020, at 05:32:00 and went on until May 10, 2023, at 05:44:00. The rationale behind omitting tweets before January 1, 2020, was to ensure that the corresponding stock market data was readily available to facilitate precise volatility analysis.

2.2 Tesla Stock Data Acquisition & Preprocessing

We started by using YahooFinance API to get a daily stock return. Nonetheless, the task of predicting volatility necessitates the use of dependable and abundant data. Consequently, our next step was to collect intra-day stock price data to ensure that we could calculate daily volatility with a sufficient number of samples for a solid analysis. As suggested by the method outlined in Andersen & Bollerslev (1998), the most suitable data to use seemed to be 5-minute price data. However, the acquisition of this data presented quite a challenge. While a number of prominent platforms, such as Yahoo Finance, provided daily price data extending back several years, intra-day price data proved harder to come by. Where it was available, it was usually restricted to data for just one year. Nevertheless, we were able to obtain the required data spanning from January 2020 to the start of May 2023, thanks to a data service provider online called *Twelve Data*.

2.3 Feature Engineering

Upon extraction of 5-minute closing prices every three months into CSV files - a restriction dictated by the free API we were using - we combined these files to create a single Pandas DataFrame, thereby covering the entire time period of interest. Subsequently, we used a straightforward group-by method to compute daily volatility across this timeframe, excluding weekends. The final stage involved merging the volatility data frame with sentiment analysis data for each day.

In terms of sentiment analysis (more comprehensive description of the model utilized is provided later), we utilized both VADER and TextBlob to ascertain the polarity of each tweet, with the aim being to compute the average daily polarity. As we used all of Elon Musk's tweets from the required period to maximize our amount data points, with the argument that Musk's public image may also impact Tesla's performance even with tweets not directly related to the company, we decided to give more weight to the polarity of tweets containing 'Tesla' (or 'TSLA'), with the additional weight being an additional hyperparameter.

The absolute mean polarity score for each day is calculated using the formula:

$$\mu_{pol\_abs} = |\sum_{i=1}^{n} polarity(tweet_i) * (1 + add\_weight\_tesla * 1_{\{tweet_i \text{ contains 'Tesla' or 'TSLA'}\}})| \quad (1)$$

With $polarity(x)$ representing VADER's compound score for tweet x, add_weight_tesla representing the hyperparameter of additional weight added to the seemingly more relevant

tweets and $1_{\{x\}}$ being an indicator function conditional on x happening.

The reasoning here is that if tweets are published before the New York Stock Exchange opens (which is 09:30 local time or 13:30 GMT), they will be included in the same day's polarity average. Tweets published during or after market hours are attributed to the subsequent day. This principle also applies to weekends: tweets posted before market hours on Friday are assigned to Friday, whereas those published after market hours on Friday, or any time over the weekend, are assigned to Monday. Thus, we end up with several corresponding time series, one showing daily volatility and the rest showing the daily mean polarity of tweets for its underlying NLP model. This final dataset contains all the base features needed for model implementation.

2.4 Descriptive Statistics

Please note that while we have presented a selection of plots in this section, we recommend referring to the appendix for additional plots.

2.4.1 Tweets

To begin with, a comprehensive analysis was carried out to enhance comprehension of the data by visualizing the frequency of tweets posted by Elon Musk. Initially, the focus was on evaluating the yearly distribution of tweets (refer to Figure 1):

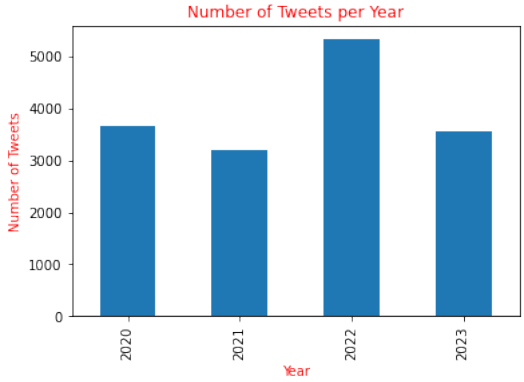

Figure 1: Number of tweets per year

Additionally, a graphical representation of the average daily tweets over the course of each year was generated to determine the specific periods when Elon Musk tends to tweet the most throughout the week (refer to Figure 2):

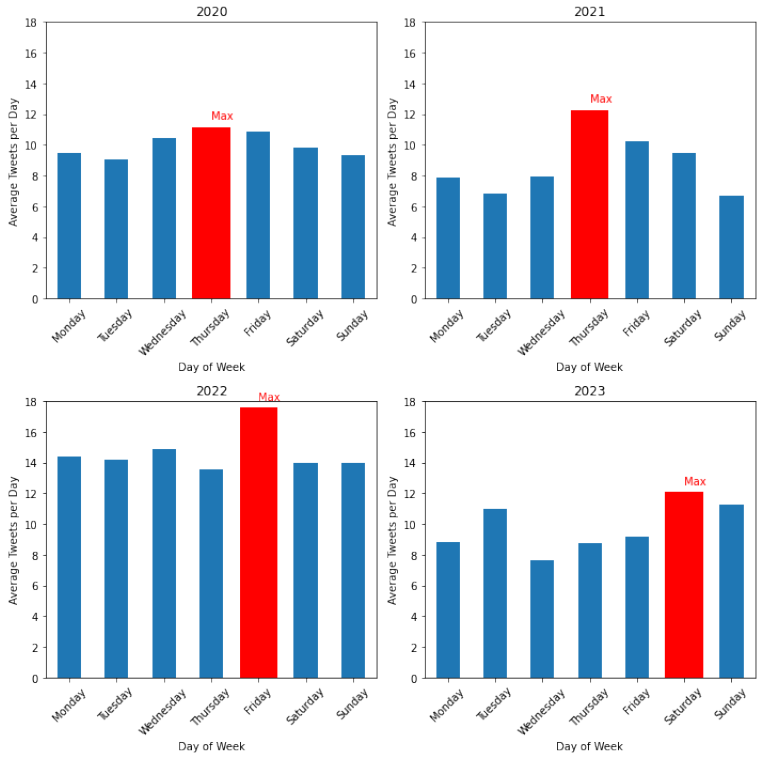

Figure 2: Average tweets per day

Next, we redirect our focus to the tweets that include the term 'Tesla.' We examine the yearly frequency of 'Tesla' or 'TSLA' mentions in Musk's tweets. A comparison between two plots, one for SpaceX and the other for Tesla, reveals a notable disparity. The attention given to Tesla in Elon Musk's tweets surpasses that of SpaceX by a significant margin. This highlights Tesla as the predominant subject of his tweets, showcasing a greater level of focus and emphasis on the company (refer to Figure 3):

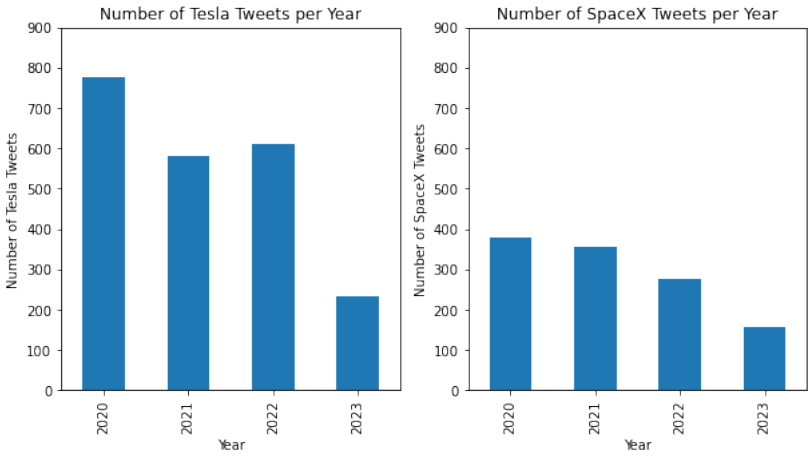

Figure 3: Comparison of number of Tesla tweets vs number of SpaceX tweets on yearly basis

In conclusion, we comprehensively analyzed Elon Musk's entire tweet dataset and identified the top 10 most commonly used words. Through the utilization of a bar plot, we gained valuable

insights into the main topics and subjects that garnered the highest level of attention in his tweets. Notably, the word 'tesla' emerged as the most frequently recurring term (refer to Figure 4):

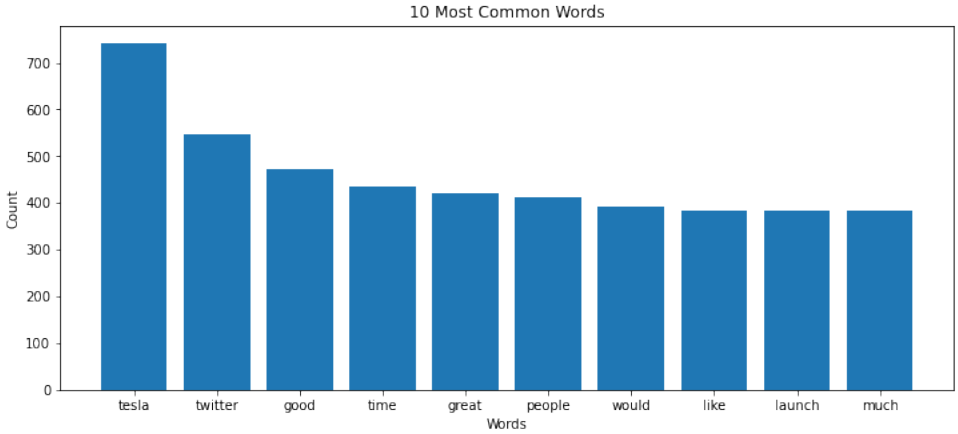

Figure 4: Ten most common words with their counts

2.4.2 Daily stock return of Tesla

By utilizing Yahoo Finance, statistical analysis was performed. Initially, we plotted the daily returns (Adjusted Close) of the SP 500, Tesla stock, and Nasdaq to evaluate their respective levels of volatility. This analysis provided insights into the relative volatility among the three stocks, with Tesla stock exhibiting significantly higher levels of fluctuation (refer to Figure 5):

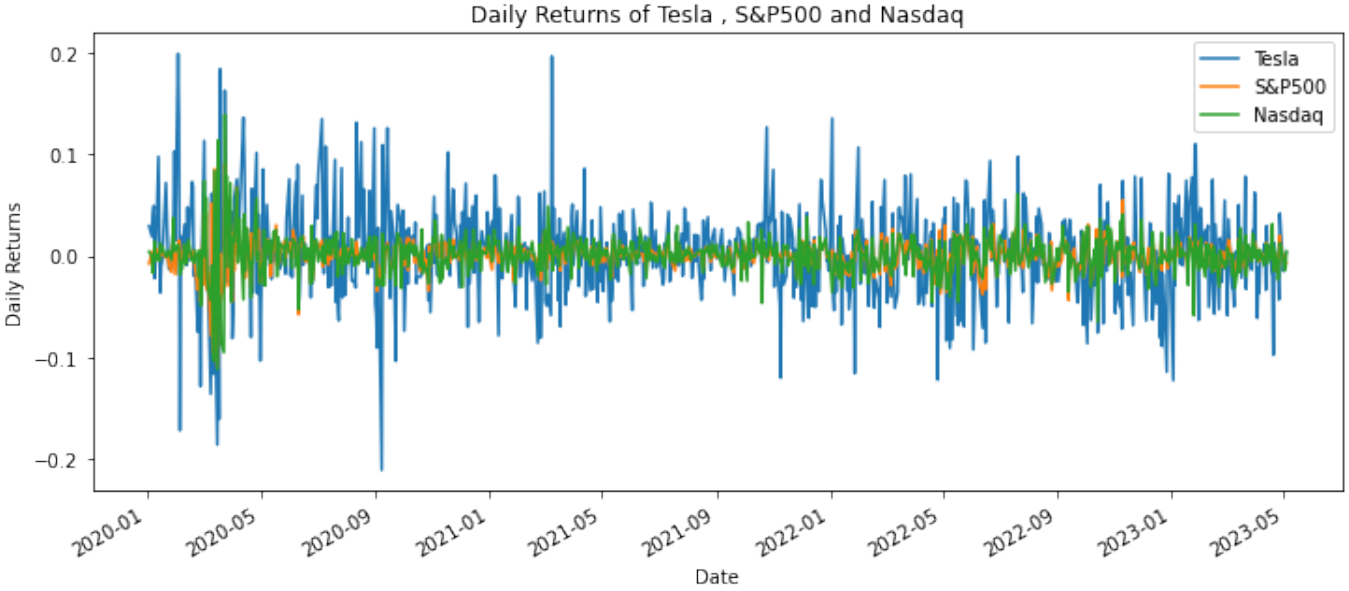

Figure 5: Daily stock return of SP 500 ,Tesla and Nasdaq

Furthermore, we evaluated the daily stock returns by visualizing their probability distribution. This analysis revealed fatter tails compared to the normal distribution, indicating a higher

likelihood of extreme events. Additionally, we observed a relatively narrow interquartile range, suggesting a reduced spread of values within the dataset (refer to Figure 6):

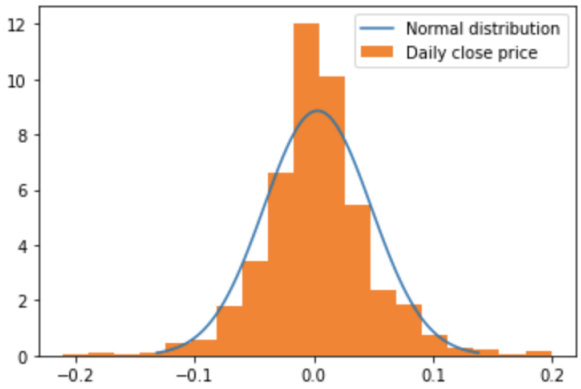

Figure 6: Probability distribution of daily stock return of Tesla

In addition, the kurtosis test examines the null hypothesis that the kurtosis of the population, from which the sample was taken, follows a normal distribution. The data underwent a kurtosis test, resulting in a test statistic of 7.49 and an exceptionally small p-value of 7.07×10^{-14}. The minuscule p-value indicates a significant deviation of the data's kurtosis from that of a normal distribution, highlighting a distinct level of "tailedness" in comparison to the normal distribution.

2.4.3 Volatility dataset of Tesla

Through the compilation of the volatility dataset, which was created using intraday stock prices at a 5-minute frequency, we proceeded to perform descriptive statistics. The summary of these statistics is presented in the table provided below:

Statistic	**Value**
Count	837
Mean	0.070398
Standard Deviation	0.055558
Minimum	0.013640
25% Quantile	0.043881
Median	0.059596
75% Quantile	0.081032
Maximum	1.196186

Table 1: Descriptive Statistics of Volatility dataset (annualized)

3 Methods

3.1 Sentiment Analysis

The objective of the project is to forecast the daily volatility of Tesla stock by analyzing Elon Musk's tweets. To measure the impact of these tweets, polarity was deemed as an appropriate metric, determined through sentiment analysis. Considering Musk's dual role as the CEO of the company and a prominent public figure with substantial influence, his positive or negative tweets could trigger fluctuations in stock prices.

As individuals in the market may swiftly react to his social media messages, polarity emerged as a suitable choice for several reasons. Firstly, it provides a more balanced measure compared to a simplistic positive, negative, or neutral classification. Additionally, it eliminates the need for setting a threshold to assign sentiments, reducing the complexity of parameter tuning. Moreover, utilizing polarity offers flexibility as both the polarity value and its absolute value could be incorporated into the models. In this project, since both positive and negative tweets by Musk hold the potential to impact volatility, exploring both the actual and absolute values of the tweets proved relevant.

3.1.1 Model Selection

To accomplish this task, we opted to experiment with two distinct models. Initially, we employed TextBlob, a relatively basic model that necessitates more extensive data preparation. Additionally, we utilized a more advanced model called Valence Aware Dictionary for sEntiment Reasoning (VADER). The objective was to compare a straightforward model based on the bag-of-words approach with a more intricate one that considers sentence structure and additional language features. This approach aimed to avoid unnecessary complexity and determine the most suitable model for the task at hand.

3.1.2 TextBlob - Naive Bayes classifier

TextBlob (for more details: TextBlob documentation) is a user-friendly Python library that offers a simple and intuitive API for executing common natural language processing (NLP) tasks. These tasks include part-of-speech tagging, sentiment analysis, and language translation. It is built on top of the NLTK (Natural Language Toolkit) and provides a convenient interface for processing textual data.

Prior to utilizing TextBlob, several cleaning operations must be conducted to preprocess the text. These operations involve removing usernames, converting emojis to their textual representations, tokenizing the tweet into individual words, eliminating stopwords, punctuation, empty strings, links, and words with only one character. Additionally, lemmatization is performed using the nltk library to transform the words in the tweet into their base forms. The polarity score, which ranges from -1 to 1 (with -1 representing the most negative sentiment and 1 indicating the most positive sentiment), is computed for both the overall tweets and those specifically mentioning 'Tesla' or 'TSLA'. The mean polarity scores for these categories are calculated and summarized below:

Mean polarity	Value
All tweets	0.11
tweets containing Tesla	0.14

Table 2: Mean polarity of all tweets and tweets containing Tesla using textBlob

3.1.3 VADER- Lexicon and rule-based sentiment analysis

Within the VADER library, the "SentimentIntensityAnalyzer" class proved invaluable, particularly its "polarity_scores()" function. This function provided us with three sentiment scores (positive, negative, and neutral), as well as a global compound score. We relied on the compound score to determine the sentiment of each tweet, following the guidelines established in the original VADER paper (Hutto, C.J., Gilbert, E., 2014). As our aim was to capture Musk's sentiment in a nuanced manner rather than simply classifying it as positive, negative, or neutral, the compound score aligned well with our objectives.

However, it is important to acknowledge that while VADER may handle full sentences with punctuation better than TextBlob's lemmatized and cleaned tweets, it may still struggle with detecting irony or misspelled words. Despite these potential limitations, we computed and summarized both the mean polarity for all tweets and specifically for those containing the term 'Tesla' (or 'TSLA'), as presented below:

Mean polarity	Value
All tweets	0.15
tweets containing Tesla	0.25

Table 3: Mean polarity of all tweets and tweets containing Tesla using VADER

3.1.4 Model Evaluation

To assess the similarity between Elon Musk's tweets about Tesla and a dataset of tweets related to Apple (Crowdflower, 2016), we employed the latter dataset. The Apple tweets appeared comparable to Musk's in terms of content and vocabulary. To ensure data reliability, we filtered out tweets with a confidence score below 1, which was the highest confidence level available. The resulting evaluation dataset comprised 1886 tweets, with 159 classified as positive, 1024 as neutral, and 703 as negative. Although the distribution of sentiments was uneven, particularly for positive tweets, our model choice was validated by the obtained results. For the evaluation, we opted for a classification task, despite utilizing the polarity score in our models. Finding reliable datasets specifically for polarity, particularly ones similar to our data, proved to be considerably challenging due to the difficulty of accurately judging sentiment polarity.

VADER achieved accurate predictions for 1176 tweets (62.35%), while TextBlob scored 1113 (59.01%) correctly classified tweets. Although the difference may not appear significant, it can be attributed to several factors. Analyzing the confusion matrices below, it becomes apparent that VADER performs notably better in classifying highly polarized labels, surpassing TextBlob in both positive and negative label classification.

Another possible contributing factor could be the thresholds used for comparison. To ensure a fair and equal comparison between the models, we employed the thresholds of -0.05 and 0.05, as originally stated in the paper, for both TextBlob and VADER. Since our models rely on polarity rather than specific labels, as previously mentioned, we did not delve further into determining optimal thresholds for each model.

The key takeaway from this observation is that, given our focus on polarity for volatility prediction, it appeared logical to select VADER as our polarity classifier. VADER demonstrated better performance in handling more extreme polarities and overall classification tasks.

	Actual		
Predicted	**Positive**	**Neutral**	**Negative**
Positive	137	18	4
Neutral	285	602	137
Negative	137	129	437

Confusion matrix for VADER

	Actual		
Predicted	**Positive**	**Neutral**	**Negative**
Positive	117	39	3
Neutral	262	682	80
Negative	134	255	314

Confusion matrix for TextBlob

3.2 Time Series Modelling

We will now explore econometric modelling methods for volatility forecasting. These fundamental methods will serve as our benchmark model that we will contrast with more advanced machine learning methods (cf. section 3.3). This part of our work is mainly motivated by two factors.

The first one being parsimony, time series models are often times more interpret-able and understandable compared to deep learning algorithms. The second one is performance, especially out of sample predictive power. These two factors will be discussed and compared within models in the Results section (cf. section 4). Next we will expose what properties do our time series require to fit the chosen econometric model, along with the reasons of the choice of the model and its intended relevance for our work.

3.2.1 Statistical Requirements

The econometric model we have selected to serve as our benchmark throughout this paper is the Heterogeneous Autoregressive (HAR) model (Corsi, 2009). The rationale behind this choice and the theoretical framework associated with the model will be detailed in the subsequent section (refer to section 3.2.2). If one aims to employ a HAR model for realized volatility (RV) forecasting, specifically a HAR-RV model, it is critical to ensure certain statistical prerequisites are met.

Initially, we necessitate our realized volatility data to exhibit stationarity, hence we employ an Augmented Dickey-Fuller test (ADF) (Dickey, Fuller, 1979) to test for this.

Next, we verify the presence of autocorrelation in the time series using autocorrelation function (ACF) and partial autocorrelation function (PACF) plots, as well as statistical tests like the Ljung-Box test (Ljung, Box, 1978) to determine the significance of autocorrelation. This is paramount as the model integrates its own lags, thus they must exhibit correlation for us to apply the model.

Our HAR-RV model will also encompass the lagged sentiment variable, hence, another concern at this stage was to ascertain if previous values of the sentiment variable ($sentiment_{tn}$) can be utilized to predict the value of realized volatility (RV_t). If it is indeed the case that ($sentiment_t$) can predict (RV_t), then it must be that ($sentiment_t$) Granger causes (RV_t) (realized volatility). Given this property, we can administer a Granger Causality test. This test necessitates that both variables RV_t and $sentiment_t$ be stationary (Enders, 2014). The regression to evaluate this assumes the following form:

$$RV_t = \sum_{i=1}^{n} \alpha_j RV_{t-j} + \sum_{i=1}^{n} \beta_j sentiment_{t-j} + \varepsilon_j \tag{2}$$

Assuming that ε_t represents uncorrelated white noise, the null hypothesis for the Granger causality test posits that β_j equals zero, as proposed by Granger (1969). While the detailed theoretical aspects and implementation specifics of these tests will not be explored here, as they are not directly pertinent to the subject matter of our work, Section 4.1 will present the obtained results along with a brief discussion.

3.2.2 Benchmark Model: Heterogeneous Autoregressive (HAR)

As we embark on modeling Tesla's return volatility using Twitter sentiment data, we opt to utilize the current state-of-the-art heterogeneous autoregressive (HAR) framework as proposed by Corsi (2009). Based on current literature, HAR modeling has seen broad adoption in financial econometrics for the purpose of volatility forecasting (Corsi, Mittnik, Pigorsch, Pigorsch, 2008; Degiannakis Filis, 2017; Zhang, Wei, Zhang, Jin, 2019).

Furthermore, it appears to surpass other volatility clustering techniques such as ARCH (Autoregressive Conditional Heteroskedasticity) or GARCH (Generalized Autoregressive Conditional Heteroskedasticity) models, as suggested by Sévi (2014) and Prokopczuk, Symeonidis, and Wese Simen (2016). The ARCH model postulates that the conditional variance of a variable is dependent on its lagged squared residuals.

However, financial data often display volatility that not only demonstrates persistence but also short-term dynamics influenced by higher-frequency information. The HAR model accounts for varying frequencies of data to encapsulate these dynamics. It integrates lagged squared returns at multiple frequencies, typically daily, weekly, and monthly, as predictors of the conditional variance. By assimilating information from diverse frequencies, the HAR model seeks to enhance the accuracy of volatility forecasting. The objective here is to forecast realized volatility (RV_t) using weekly and monthly lags ($RV_{t-5,t}$ and $RV_{t-22,t}$ respectively) in addition to the lagged sentiment $(sentiment_{t-1})$. We can define this more specifically as:

$$RV_{t-5,t} = \frac{1}{5}(RV_{t-1} + RV_{t-2} + RV_{t-3} + RV_{t-4} + RV_{t-5}) \tag{3}$$

and $RV_{t-22,t}$ is defined in the same rolling window fashion. Further we can introduce the first HAR model specification;

$$RV_t = \beta_0 + \beta_d^{RV} RV_{t-1} + \beta_w^{RV} RV_{t-5:t} + \beta_m^{RV} RV_{t-22:t} + \beta_{sentiment} Sentiment_{t-1} + \varepsilon_t \quad (4)$$

where ε_t is the stochastic noise component of autoregressive models.

A natural extension of this model is to take the absolute value of sentiment $|sentiment_{t-1}|$ instead of the original variable. In our analysis, directional effects (positivity or negativity of sentiment) do not really matter, as it is the strength of the sentiment that should in theory push the prices either up or down depending on the respective polarity, and therefore generate a higher volatility on the given trading day. Thus, we have the following alternative model specification:

$$RV_t = \beta_0 + \beta_d^{RV} RV_{t-1} + \beta_w^{RV} RV_{t-5:t} + \beta_m^{RV} RV_{t-22:t} + \beta_{|sentiment|} |Sentiment_{t-1}| + \varepsilon_t \quad (5)$$

The estimation technique chosen to fit the above specifications is Ordinary Least Squares (OLS), which easily allows us to incorporate lags as well as lagged exogenous variables which we require.

3.3 Machine Learning Methods

3.3.1 LSTM

Long Short-Term Memory (LSTM) is a type of Recurrent Neural Network architecture, used in order to overcome the vanishing gradient problem. The network is capable of retaining or forgetting information from previous time steps, making it able to effectively model and predict sequential data, which suits the problem of volatility time series prediction (Kim, & Won, 2018). LSTM models are capable of learning complex relationships between past and current data as well as being able to handle non-linear relationships and incorporate additional features, like polarity in our case, making it well-suited in theory for this task. Our inputs were the time series of volatility and absolute mean polarity for each day (calculated with VADER's compound score, with the additional weights, as mentioned above).

In order to figure out the best model specifics, one has to start from a very basic model and added layers and neurons step by step, in order to avoid the issue of the model getting too complex.

To best capture the impact of different LSTM architectures on volatility prediction, we proceeded with one simple and one more complicated architecture, which will be described later in the paper. Simple LSTMs consist of a single layer of LSTM units with relatively few hidden nodes. They are easy to train and interpret, but may fail to capture complex dependencies and nonlinearities in our data, which we may need in the case of volatility prediction. Complex LSTMs have multiple layers of LSTM units, with more hidden nodes. They are more powerful in modeling nonlinearities, but are more difficult to train and interpret.

In order to try to mitigate both the lack of training data as well as the impact of nonlinearities, we thus decided to proceed with both models, aiming to find the best-performing ones for each category.

In order to optimize the performance of both LSTM models, several common hyperparameters were taken into account. However, due to the large amount, it was impossible to test all of them, as the combinations grow exponentially. We thus made some decisions based on theory and left some other hyperparameters to be tested manually.

First of all, the "add_weight_tesla" parameter was introduced to determine the additional weight given to tweets containing the term "Tesla". This allowed for a greater impact of these tweets on the daily sentiment analysis. In our implementations, various weights were considered. After extensive testing, giving the Tesla-containing tweets a weight ten times larger than to the regular tweets seemed to yield the best results in general. Giving less weights to these tweets (and thus more weight to the potentially unrelated tweets) yielded less significant results, with way lower R^2_{OOS} scores and giving them more did not improve the results. The sequence length parameter was also carefully tuned to define the number of previous input tuples considered by the LSTM for each output, enabling the model to capture relevant context and memory. Another important hyperparameter was the batch size, which determined the number of input sequences processed in parallel during each training iteration or forward pass. A larger batch size facilitated better pattern recognition by the LSTM. The number of epochs, specifying the iterations over the entire dataset during training, was optimized to balance model convergence and computational efficiency.

For models using bootstrapping, we had to consider how many resamplings of the dataset were considered to enhance the model's robustness. To prevent overfitting, a dropout rate was added, temporarily ignoring randomly selected units or connections during training. Ours was set to 0.1 following careful testing. However, as our models did not seem to be prone to overfitting with our relatively low amount of epochs, its use was rarely required. The training and validation split parameter determined the allocation of samples in the training set for actual training and validation at each epoch, aiding in model assessment and preventing overfitting. The training and validation sets (used for the R^2_{OOS}) were split according to a 80%/20% split, used in many papers within the industry, notably in (Kim, & Won, 2018). The choice of loss function played a vital role in measuring the model's performance by comparing predictions to true values on the validation set. We tried multiple loss functions with no large differences in the results, so we decided to proceed with the MSE, for its efficiency, suitability for regression tasks and due to theoretical backing (J. Jakubik, A. Nazemi, A. Geyer-Schulz, & F.J. Fabozzi, 2021). The optimizer, responsible for adjusting model weights and biases, was carefully chosen to optimize the loss function during training. For our model, Adam, able to provide adaptive learning rates, momentum optimization, bias correction, and adaptive regularization, seemed like a relevant choice, as it was also used in one of the papers used as a reference (Kim, & Won, 2018).

Finally, an appropriate activation function was employed to introduce non-linear transformations within the LSTM cells, enabling the model to capture complex relationships in the data. In our case, we proceeded with the ReLU in order to introduce non-linearity, as it is efficient, as well as often the most appropriate choice for regression tasks with positive outputs (other activation functions, like hyperbolic tan, sigmoid and even a simple linear function were tried, but ReLU yielded the best results). By fine-tuning the remaining hyperparameters, the LSTM models could be optimized to achieve accurate and reliable predictions for the given task.

3.3.2 Requirements

The usage of LSTM requires a few assumptions about the data to hold.

First of all, as the model will use previous values of volatility to predict the next ones, we assume that volatility data has temporal dependencies. This seems like a legitimate assumption, as stock price volatility generally exhibits time-series properties, due to volatility being affected by many factors, such as macroeconomic conditions or market sentiment, which are time-dependent.

Another assumption that is not strictly required, due to the learning capacities of LSTM networks, but that could make training the model easier, is the stationarity of the dependent variable, volatility. As a LSTM model is able to learn nonlinearities, as well as long-term patterns, non-stationarity becomes less of a concern in theory. However, in order for the network to learn these long-term patterns, a lot of data is required, which could be a problem in our case. As we are predicting volatility, which, being the square root of the variance (positive), is considered to be positive, we chose to work with the absolute value of polarity, to capture in a similar way the effect both positive and negative sentiment of Musk's tweets. LSTM requires inputs of shape $(amount\_samples, window\_length, nb\_features)$ for the training and testing sets and $(amount\_samples)$ as the training and testing labels, thus we had to transform our DataFrame containing the daily volatility and the mean polarity in order to be able to train the network. The training and testing sets contain an amount of observations of both volatility and polarity for each day within the $window\_length$ before the current label (volatility).

Due to the lack of data in our training and testing sets, we used block stationary bootstrapping (Politis, & Romano, 1994) to generate multiple versions of our samples in a random way. The method involves dividing the time series data into blocks of fixed size and then randomly resampling these blocks with replacement to create new synthetic time series. Block bootstrapping is the appropriate method to use with time series, due to the temporal dependence of the data. Another possible method would have been circular bootstrapping, but as our time series is stationary, we proceeded with stationary bootstrap, requiring this property. We then trained a model for each boostrapped instance of the dataset and averaged out the total in order to avoid some outlier values in predictions.

One last thing to take into account with LSTM networks is that since they can exhibit non-deterministic behavior due to the presence of random initialization and non-linear activation functions, they can lead to variations in output predictions for the same input sequence during different model runs or training iterations. This non-determinism can impact the reproducibility and stability of results, so seeds had to be set before running anything related to the network. The required seeds to set were the environment variable *PYTHONHASHSEED*, Python's built-in pseudo-random generator's seed and the seeds for *numpy.random* as well as *Tensorflow.random*. In the current implementation, all values are set to 42 arbitrarily.

3.4 Model Evaluation

In order to assess the efficiency of our models against each other, we decided to use out-of-sample R-squared (with $sklearn.metrics.r2\_score$), measuring the model's ability to generalize

to new data that was not used during training, providing a better estimate of the model's predictive performance in real prediction tasks. It helps to ensure that the model is not overfitting to the training data.

$$R^2_{OOS} = 1 - \frac{RSS}{TSS} = 1 - \frac{\sum_{t=1}^{T}(y_t - y_{t,pred})^2}{\sum_{t=1}^{T}(y_t - \mu_y)^2} \tag{6}$$

Where RSS is the residual sum of squares, TSS the total sum of squares, y_t are the actual values of the dependent variable (volatility) in the out-of-sample testing set, $y_{t,pred}$ are the predicted values, obtained from the inputs of the testing set and μ_y is the sample mean of the testing set dependent variables.

3.5 Limitations

A first limitation for the usage of neural networks with our data was the size of the datasets. Due to there being a limited number of tweets by Elon Musk in our time interval, as well as the difficulty to find stock's price data about Tesla with 5-minute intervals, we managed to gather data over three years and five months. As the week-ends and holidays reduce the price data availability for the period, it limited even further the amount of days we could work with. Once everything was ready, the networks had 807 samples to work with and this is something that could be improved in the future, using different twitter datasets or price data over a longer period. Even while using bootstrapping, more samples would help the models perform better.

4 Results & Discussion

In the forthcoming section, we bring forward the empirical findings of our study. Our presentation initiates with the outcomes of a series of statistical tests that we conducted, which were prerequisites to fitting the models of interest. Post that, we proceed to explore the benchmark model, examining its in-sample (IS) and out-of-sample (OOS) efficacy. The most substantial part of this section, however, is dedicated to unraveling the insights derived from our employment of machine learning models.

We embark on an exhaustive conversation comparing the conventional econometric approach, which served as our reference point, with the machine learning techniques we utilized. Further, we delve into an internal comparison within the non-linear, more complex machine learning models. The focus of this discourse lies in understanding how adjustments in hyperparameters can influence the performance of these models in out-of-sample predictions. This step-by-step journey through our results is designed to provide a complete and nuanced understanding of our findings.

4.1 Statistical Requirements - Results

This section begins by evaluating the statistical prerequisites of our time series. As we have previously noted, certain conditions must be met in advance. We initiated our investigation by checking for stationarity, a property we discussed in detail in section 3.2.1. To confirm this, we employed the Augmented Dickey-Fuller (ADF) test across all our time series, and the results are presented in the Appendix (Tables 1-6). As evidenced by the tables, all our time series reject the null hypothesis, thereby negating non-stationarity or unit roots.

Furthermore, due to some models' utilization of Realized Volatility (RV) lags, it was necessary for us to analyze the autocorrelation and partial autocorrelation functions. This analysis was conducted using Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) plots, which are illustrated in Figures 13 and 14. These plots clearly indicate the presence of significant autocorrelation within the RV time series.

Finally, we performed the Granger Causality test to investigate a potential linear causal relationship between various sentiment polarity variables and Realized Volatility. The outcomes, again presented in the Appendix (Tables 7-10), were rather underwhelming, suggesting the absence of any notable linear causal relationship between these variables. This will be further substantiated in the results of the forthcoming model.

4.2 Benchmark Model - Results

We applied four distinct HAR model specifications for the sake of parsimony and robustness. The variations between the models were due to the use of either the VADER or TextBlob polarity time series and their corresponding absolute value analogs. These results are exhibited in Tables 11-14 of the Appendix.

Initially, it is observable that, regardless of model specification, sentiment (or the absolute value thereof) has coefficients that are not statistically significant. This is consistent with the Granger Causality findings we discussed earlier, which suggested an absence of a linear relationship between sentiment and Realized Volatility (RV). In addition, it is clear that only short-term and mid-term RV impacts, particularly daily and weekly lags, seem to have a linear relationship with RV. This is further supported by the Autocorrelation Function (ACF) plot, where lags up to 15 appear to be significant, thus capturing daily and weekly effects.

However, the overall fit and effectiveness of this benchmark model leave much to be desired. The in-sample R-Squared is approximately 7% across all specifications, indicating that linear relationships may not be the primary drivers of inter-variable connections. Although the out-of-sample performance—obtained using a rolling window method—is not exceedingly poor relative to the in-sample predictive power, it was evident that our investigation needed to extend further, aiming to capture more complex, non-linear patterns for potential improvement in results.

4.3 LSTM Models - Results

Important remark: In order to reproduce the same results, the LSTM notebooks need to be run exactly once after restarting the kernel, in the same exact order, with the same seeds.

4.3.1 Simple LSTM

When building the simple LSTM, one clear takeaway could directly be found after the first tries. As recurrent neural networks with only one or two layers are not able to find complex non-linear relationships within the data, this approach was not going to work even with lots of fine-tuning.

However, we still tried to find a configuration working well enough. Large batch sizes seemed to help the predictions slightly, so we went with a size of 128.

20 epochs yielded the best results out of the amounts we tried, with less, the predictive power was even lower and with more, the predictive power decreased too.

In practice, with the best configuration we could find, we managed to obtain a R^2_{OOS} of -0.62 (Table 16 of the Appendix), which is way lower than the HAR model and has less predictive power than a sample average predictor. We can thus notice that more complexity and non-linearity is required with our input data in order to reach some actual predictive power.

4.3.2 Complex LSTM

For the complex LSTM, after careful testing, a few insights emerged about the hyperparameters.

Large batch sizes helped tremendously in achieving better values for the R^2_{OOS}, the best choices were between batch sizes of 128 and 256. The latter one did slightly better in most of the configurations, included in the final one.

Bootstrapping helped in general to stabilize the values and avoid the influence of outlier predictions on average, however, as Tesla's stock is highly volatile, in some configurations, it smoothed predictions out too much.

The number of epochs was carefully considered. Configurations with epochs set between 10 and 20 seemed to have the best results, afterwards, the extra training did not seem to help to improve the validation score, but in most of the cases, 10 epochs was not quite enough to finish improving the predictions, so we proceeded with 20.

After careful experimenting, the optimal sequence length seemed to be in the range from 9 to 15. In the final model, the most optimal sequence length was 15, as the performance increased slightly up to that, afterwards, it started decreasing again as the prediction smoothed out.

The most important part for the LSTM was the configuration of the network, with decisions required to be made about the amount of layers, whether they are bidirectional or not as well as about the amount of cells in each layer.

Bidirectional layers help the model to capture both past and future context when making predictions on sequential data, as they allow information to flow both in the forward and the backward direction within the inputs (Siami-Namini, Tavakoli, & Siami Namin, 2019). In the case of this paper, adding one bidirectional layer to the model yielded the best results.

Adding more layers increases both the depth and the complexity of the network. This allows it to capture better more abstract and nonlinear patterns, as well as understand complex dependencies between samples. In this case, adding a few layers, between 5 and 10, yielded the best results. The more layers, the more the network seemed to capture well the complex patterns in the validation set volatility, up to a certain point, where performance started to decrease again.

Adding more cells increases the representational capacity of the network, allowing it to capture more intricate patterns in the input sequences, which may also improve its capability to learn complex dependencies and nonlinear patterns. However, in our case, adding more cells actually made the predictive power slightly worse, as it smoothed out the volatility predictions closer to the mean and thus did not seem like a good approach for a highly volatile stock like Tesla. Also, due to the increased computational complexity, their addition should be considered with care. Nevertheless, for other stocks with less volatility, this approach should still be considered. The best results in this case were obtained with 1 and 6 cells per layer.

After a lot of testing, we managed to obtain the best results with a model without using bootstrapping. Despite the imitations of the sample size, we managed to obtain a R^2_{OOS} of 0.085 (Table 16 of the Appendix), not only beating the HAR-based approach in the out-of-sample prediction, but also yielding a better out-of-sample result than HAR managed to obtain with its

in-sample one and beating the simple LSTM by far.

With boostrapping, even our best network did not perform as well, yielding a R^2_{OOS} of 0.064 (Table 17 of the Appendix). This is probably due to the smoothing of the prediction by the averaging between multiple LSTMs, or by the breaking of some potentially important dependencies within the time series due to the block bootstrapping.

However, in order to showcase the potential power of boostrapping for other, less volatile stocks (or with less dependencies in the long run), we still decided to showcase our results, as with bootstrapping, we could achieve a predictive power which was pretty close to the regular approach. Our R^2_{OOS} was only 0.021 away from the above approach and still beat both the out-of-sample R^2 of the HAR model.

The details of the models can be found below, explaining in detail the hyperparameters, the model specifications and other metrics.

4.4 Summary of the Results

In order to have a closer look on the performances of each model, we can summarize our results as follows:

Model	**Description**
Benchmark model (HAR-based approach)	Parsimonious, simple and straightforward Short-term dynamics seem prevalent in RV (Realized Volatility) as daily and weekly lags are significant No significant linear relationship between sentiment and RV In-sample R^2 = 0.07 OOS R^2 = 0.05
Simple LSTM	Insufficient for non-linearity Insufficient for complex dependencies Low computational cost OOS R^2 = -0.62
Complex LSTM without bootstrapping	More depth and complexity added to the network Better representational capacity Better for non-linearity and complex dependencies High computational cost OOS R^2 = 0.0859 Beats the benchmark model (both in- and out-of-sample)
Complex LSTM with bootstrapping	Smoothing of the predictions by averaging between multiple LSTM Breaking of potentially important dependencies Slightly worse performance OOS R^2 = 0.064 Beats the benchmark model out-of-sample

Table 4: Summary of Models

5 Potential Improvements

As discussed in the part about limitations, the lack of data of good quality, available easily, for a longer period was one of the main issues. A first obvious fix would be to gather price data over a longer time, if possible.

Another potential issue we had was the fact, that we considered the impact of several tweets over the volatility of an entire day, due to the lack of tweets by Musk for most of the days. In the future, with tweets from various sources, one could assess the impact of tweets on the volatility of half a day, or even hourly volatility. This could also help with the above issue with the amount of data. The dataset could grow between two and eight times larger in size when using this method, or a variation of it and be more reactive to intra-day changes in market sentiment.

6 Conclusion

In drawing conclusions from our study, a key observation we have made is the superiority of more intricate models in successfully predicting stock volatility. The complexity of these models aligns with the inherent sophistication of volatility's nature, which is marked by intricate, non-linear patterns. Volatility, in its essence, resists simplistic predictions due to the multitude of variables that have a bearing on it, which these complex models are better equipped to handle.

However, it is important to note that this does not extend to excessively complicated models. When the complexity of a model tips over into excess, we observed an interesting trend - the out-of-sample prediction begins to converge towards a simple sample mean. There is a fine line between beneficial complexity and unnecessary complication, and striking the balance is key. It is of paramount importance for researchers venturing into this field to have a deep understanding of both the models they employ and the data they are examining. Comprehensive knowledge of these elements enables them to save significant time and effort. They can avoid pitfalls and inefficiencies by accurately selecting the tests, models, and hyperparameters that are most suited to their datasets right from the onset.

The role of Twitter sentiment in stock volatility also warrants mention. Our results indicate that the polarity of tweets does have some bearing on the prediction of stock volatility. While the predictive power unearthed in this study may not be game-changing, it does hint at a promising avenue for future research. Replicating this methodology with other, less volatile stocks, ETFs, or the market at large could potentially yield more impactful results. Larger databases of tweets and returns could also contribute to enhancing the precision and relevance of the findings. The high volatility of Tesla's stock presents a challenging environment for prediction, and even sophisticated models like LSTMs might struggle with such extreme values. However, with a focus on less volatile stocks, the task of predicting their behavior may be a touch easier, offering valuable insights for investors and researchers alike.

References

[1] T. Andersen and T. Bollerslev. "Answering the Skeptics: Yes, Standard Volatility Models do Provide Accurate Forecasts". In: (1998).

[2] *Apple Twitter Sentiment*. 2016. URL: https://data.world/crowdflower/apple-twitter-sentiment/workspace/file?filename=Apple-Twitter-Sentiment-DFE.csv.

[3] Usman Awan et al. "Predicting Stock Market Movements Using Network Science: An Information Theory Approach". In: *Information* 12.1 (2021), p. 34.

[4] F. Corsi. "A simple approximate long-memory model of realized volatility". In: *Journal of Financial Econometrics* 7 (2009), pp. 174–196.

[5] F. Corsi et al. "The volatility of realized volatility". In: *Econometric Reviews* 27 (2008), pp. 46–78.

[6] S. Degiannakis and G. Filis. "Forecasting oil price realized volatility using information channels from other asset classes". In: *Journal of International Money and Finance* 76 (2017), pp. 28–49.

[7] D. Dickey and W. Fuller. "Distribution of the Estimators for Autoregressive Time Series With a Unit Root". In: *Journal of the American Statistical Association* 74.366a (1979), pp. 427–431.

[8] J. D. Enders. "Granger Causality". In: *Journal of Agricultural and Applied Economics* 46.4 (2014), pp. 583–598.

[9] Xu Gong and Boqiang Lin. "Structural breaks and volatility forecasting in the copper futures market". In: *Journal of Futures Markets* (2018). DOI: https://doi.org/10.1002/fut.21867.

[10] C. W. J. Granger. "Investigating Causal Relations by Econometric Models and Cross-spectral Methods". In: *Econometrica* 37.3 (1969), pp. 424–438.

[11] J. Hagan and R. T. Henriksen. "News Sentiment in Volatility predictions". 2022.

[12] C. J. Hutto and E. Gilbert. "VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text". In: *Eighth International Conference on Weblogs and Social Media (ICWSM-14)*. 2014.

[13] J. Jakubik et al. "Incorporating financial news for forecasting Bitcoin prices based on long short-term memory networks". 2021.

[14] C. Jarque and A. K. Bera. "An Efficient Method of Testing Normality and Bias in Samples from a Normal Population". In: *Journal of the American Statistical Association* 75.371 (1980), pp. 575–579.

[15] H. Y. Kim and C. H. Won. "Forecasting the volatility of stock price index: A hybrid model integrating LSTM with multiple GARCH-type models". In: *Expert Systems with Applications* 103 (2018), pp. 25–37.

[16] Chao Liang et al. "Which sentiment index is more informative to forecast stock market volatility? Evidence from China". In: *Knowledge Based Systems* (2020).

[17] G. M. Ljung and G. E. P. Box. "On a Measure of Lack of Fit in Time Series Models". In: *Biometrika* 65.2 (1978), pp. 297–303.

[18] Srinivas Maguluri and Raghunathan Rengaswamy. "Stock Market Prediction Using Social Media Analysis". In: *International Journal of Information Engineering and Electronic Business* 12.1 (2020), pp. 36–42.

[19] Yash Mehta et al. "Stock Price Prediction Using Twitter Sentiment Analysis". In: *International Journal of Engineering and Advanced Technology* 10.2 (2021), pp. 267–271.

[20] Dang Vu Minh et al. "Deep Learning Approach for Short-Term Stock Trends Prediction Based on Two-Stream Gated Recurrent Unit Network". In: *IEEE Access* (2018). DOI: https://doi.org/10.1109/access.2018.2868970.

[21] D. N. Politis and J. P. Romano. "The stationary bootstrap". In: *Journal of the American Statistical Association* 89.428 (1994), pp. 1303–1313.

[22] M. Prokopczuk, L. Symeonidis, and C. Wese Simen. "Do jumps matter for volatility forecasting? Evidence from energy markets". In: *Journal of Futures Markets* 36 (2016), pp. 758–792.

[23] B. Sévi. "Forecasting the volatility of crude oil futures using intraday data". In: *European Journal of Operational Research* 235 (2014), pp. 643–659.

[24] N. Siami-Namini, M. Tavakoli, and A. Siami Namin. "The Performance of LSTM and BiLSTM in Forecasting Time Series". In: *International Journal of Computer Applications* 41.1 (2019), pp. 7–11.

[25] Bartosz Skuza and Andrzej Romanowski. "Sentiment Analysis of Twitter Data within Big Data Distributed Environment for Stock Prediction". In: *Foundations of Management* 7.1 (2015), pp. 171–182.

[26] F. Valencia, E. Munoz, and V. Chang. "Predicting cryptocurrency price bubbles using social media data and epidemic modelling". In: *Sustainability* 11.12 (2019), p. 3314.

[27] F. Viegas et al. "Exploiting semantic relationships for unsupervised expansion of sentiment lexicons". In: *Knowledge-Based Systems* 191 (2020), p. 105261.

[28] Frank Z. Xing, Erik Cambria, and Yue Zhang. "Sentiment-aware volatility forecasting". In: *Knowledge Based Systems* (2019).

[29] Chao Zhang et al. "Volatility Forecasting with Machine Learning and Intraday Commonality". In: *Journal of Financial Econometrics* (2023), pp. 1–39.

[30] Y. Zhang et al. "Forecasting oil price volatility: Forecast combination versus shrinkage method". In: *Energy Economics* 80 (2019), pp. 423–433.

Appendix

Table 1: Augmented Dickey-Fuller Test for TSLA Vol

ADF Statistic	-7.18327
p-value	2.61613e-10
Number of lags used	6
Number of observations	830
Critical Value (1%)	-3.43825
Critical Value (5%)	-2.86503
Critical Value (10%)	-2.56863

Table 2: Ljung-Box Test for TSLA Vol

Lag	Test Statistic	p-value
1	55.85	7.82179e-14
2	94.2892	3.35243e-21
3	124.57	7.99916e-27
4	144.568	2.96855e-30
5	164.104	1.32024e-33
6	183.069	7.56261e-37
7	204.649	1.18888e-40
8	217.894	1.07261e-42
9	233.361	3.22339e-45
10	247.644	1.69779e-47

Table 3: Augmented Dickey-Fuller Test for VADER Sentiment

ADF Statistic	-27.1734
p-value	0
Number of lags used	0
Number of observations	806
Critical Value (1%)	-3.43849
Critical Value (5%)	-2.86513
Critical Value (10%)	-2.56868

Table 4: Augmented Dickey-Fuller Test for TextBlob Sentiment

ADF Statistic	-28.6922
p-value	0
Number of lags used	0
Number of observations	806
Critical Value (1%)	-3.43849
Critical Value (5%)	-2.86513
Critical Value (10%)	-2.56868

Table 5: Augmented Dickey-Fuller Test for ABS VADER Sentiment

ADF Statistic	-27.3239
p-value	0
Number of lags used	0
Number of observations	806
Critical Value (1%)	-3.43849
Critical Value (5%)	-2.86513
Critical Value (10%)	-2.56868

Table 6: Augmented Dickey-Fuller Test for ABS TextBlob Sentiment

ADF Statistic	-28.6371
p-value	0
Number of lags used	0
Number of observations	806
Critical Value (1%)	-3.43849
Critical Value (5%)	-2.86513
Critical Value (10%)	-2.56868

Table 7: Granger Causality (RV - VADER Sentiment)

Lag Order	P-Value
1	0.576977

Table 8: Granger Causality (RV - TextBlob Sentiment)

Lag Order	P-Value
1	0.67911

Table 9: Granger Causality (RV - ABS VADER Sentiment)

Lag Order	P-Value
1	0.224792

Table 10: Granger Causality (RV - ABS TextBlob Sentiment)

Lag Order	P-Value
1	0.232034

Table 11: HAR Model with VADER Sentiment

Dep. Variable:	RV	**R-squared:**	0.070
Model:	OLS	**Adj. R-squared:**	0.065
Method:	Least Squares	**F-statistic:**	14.67
Date:	Tue, 30 May 2023	**Prob (F-statistic):**	1.45e-11
Time:	20:35:53	**Log-Likelihood:**	1181.5
No. Observations:	785	**AIC:**	-2353.
Df Residuals:	780	**BIC:**	-2330.
Df Model:	4		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
const	0.0444	0.005	9.530	0.000	0.035	0.054
RV_Daily	0.2234	0.035	6.402	0.000	0.155	0.292
RV_Weekly	0.1009	0.035	2.920	0.004	0.033	0.169
RV_Monthly	0.0332	0.034	0.974	0.330	-0.034	0.100
Pol_Daily	0.0016	0.012	0.137	0.891	-0.022	0.025

Omnibus:	1461.060	**Durbin-Watson:**	2.067
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	2201754.973
Skew:	12.598	**Prob(JB):**	0.00
Kurtosis:	261.225	**Cond. No.**	19.8

Table 12: HAR Model with TextBlob Sentiment

Dep. Variable:	RV	**R-squared:**	0.071
Model:	OLS	**Adj. R-squared:**	0.066
Method:	Least Squares	**F-statistic:**	14.94
Date:	Tue, 30 May 2023	**Prob (F-statistic):**	9.07e-12
Time:	20:47:51	**Log-Likelihood:**	1182.0
No. Observations:	785	**AIC:**	-2354.
Df Residuals:	780	**BIC:**	-2331.
Df Model:	4		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
const	0.0430	0.005	9.405	0.000	0.034	0.052
RV_Daily	0.2246	0.035	6.437	0.000	0.156	0.293
RV_Weekly	0.0990	0.035	2.865	0.004	0.031	0.167
RV_Monthly	0.0309	0.034	0.904	0.366	-0.036	0.098
Pol_Daily	0.0162	0.016	0.997	0.319	-0.016	0.048

Omnibus:	1457.864	**Durbin-Watson:**	2.066
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	2175363.406
Skew:	12.540	**Prob(JB):**	0.00
Kurtosis:	259.669	**Cond. No.**	19.7

Table 13: HAR Model with ABS VADER Sentiment

Dep. Variable:	RV	**R-squared:**	0.070
Model:	OLS	**Adj. R-squared:**	0.065
Method:	Least Squares	**F-statistic:**	14.67
Date:	Tue, 30 May 2023	**Prob (F-statistic):**	1.46e-11
Time:	20:47:20	**Log-Likelihood:**	1181.5
No. Observations:	785	**AIC:**	-2353.
Df Residuals:	780	**BIC:**	-2330.
Df Model:	4		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
const	0.0446	0.005	9.082	0.000	0.035	0.054
RV_Daily	0.2234	0.035	6.402	0.000	0.155	0.292
RV_Weekly	0.1010	0.035	2.925	0.004	0.033	0.169
RV_Monthly	0.0334	0.034	0.978	0.328	-0.034	0.100
ABS_Pol_Daily	0.0005	0.013	0.036	0.971	-0.026	0.027

Omnibus:	1461.115	**Durbin-Watson:**	2.067
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	2202573.557
Skew:	12.599	**Prob(JB):**	0.00
Kurtosis:	261.273	**Cond. No.**	19.9

Table 14: HAR Model with ABS TextBlob Sentiment

Dep. Variable:	RV	**R-squared:**	0.070
Model:	OLS	**Adj. R-squared:**	0.066
Method:	Least Squares	**F-statistic:**	14.75
Date:	Tue, 30 May 2023	**Prob (F-statistic):**	1.28e-11
Time:	20:48:55	**Log-Likelihood:**	1181.6
No. Observations:	785	**AIC:**	-2353.
Df Residuals:	780	**BIC:**	-2330.
Df Model:	4		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
const	0.0435	0.005	9.117	0.000	0.034	0.053
RV_Daily	0.2240	0.035	6.417	0.000	0.155	0.293
RV_Weekly	0.1005	0.035	2.911	0.004	0.033	0.168
RV_Monthly	0.0319	0.034	0.931	0.352	-0.035	0.099
ABS_Pol_Daily	0.0098	0.018	0.537	0.592	-0.026	0.046

Omnibus:	1459.069	**Durbin-Watson:**	2.066
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	2184821.701
Skew:	12.562	**Prob(JB):**	0.00
Kurtosis:	260.227	**Cond. No.**	19.7

Table 15: Configuration of the Simple LSTM

Configuration Point	**Simple LSTM**
Number of layers	1 (not bidirectional)
Number of cells	5
Dropouts	-
Batch size	128
Number of epochs	20
Bootstrapping	-
R^2_{OOS} (See corresponding figure)	-0.62 (Figure 8)

Table 16: Configuration of the Complex LSTM without bootstrapping

Configuration Point	**Complex LSTM (no bootstrap)**
Number of layers	8 (one bidirectional, the second one)
Number of cells	Between 1 and 6 per layer
Dropouts	2 (after the second and fifth layers)
Batch size	256
Number of epochs	20
Bootstrapping	-
R^2_{OOS} (See corresponding figure)	0.0859 (Figure 10)

Table 17: Configuration of the Complex LSTM with bootstrapping

Configuration Point	**Complex LSTM (with bootstrap)**
Number of layers	5 (one bidirectional, the second one)
Number of cells	Between 1 and 4 per layer
Dropouts	2 (after the second and fourth layers)
Batch size	256
Number of epochs	20
Bootstrapping	10 alternative time series obtained by block bootstrapping
R^2_{OOS} (See corresponding figure)	0.064 (Figure 12)

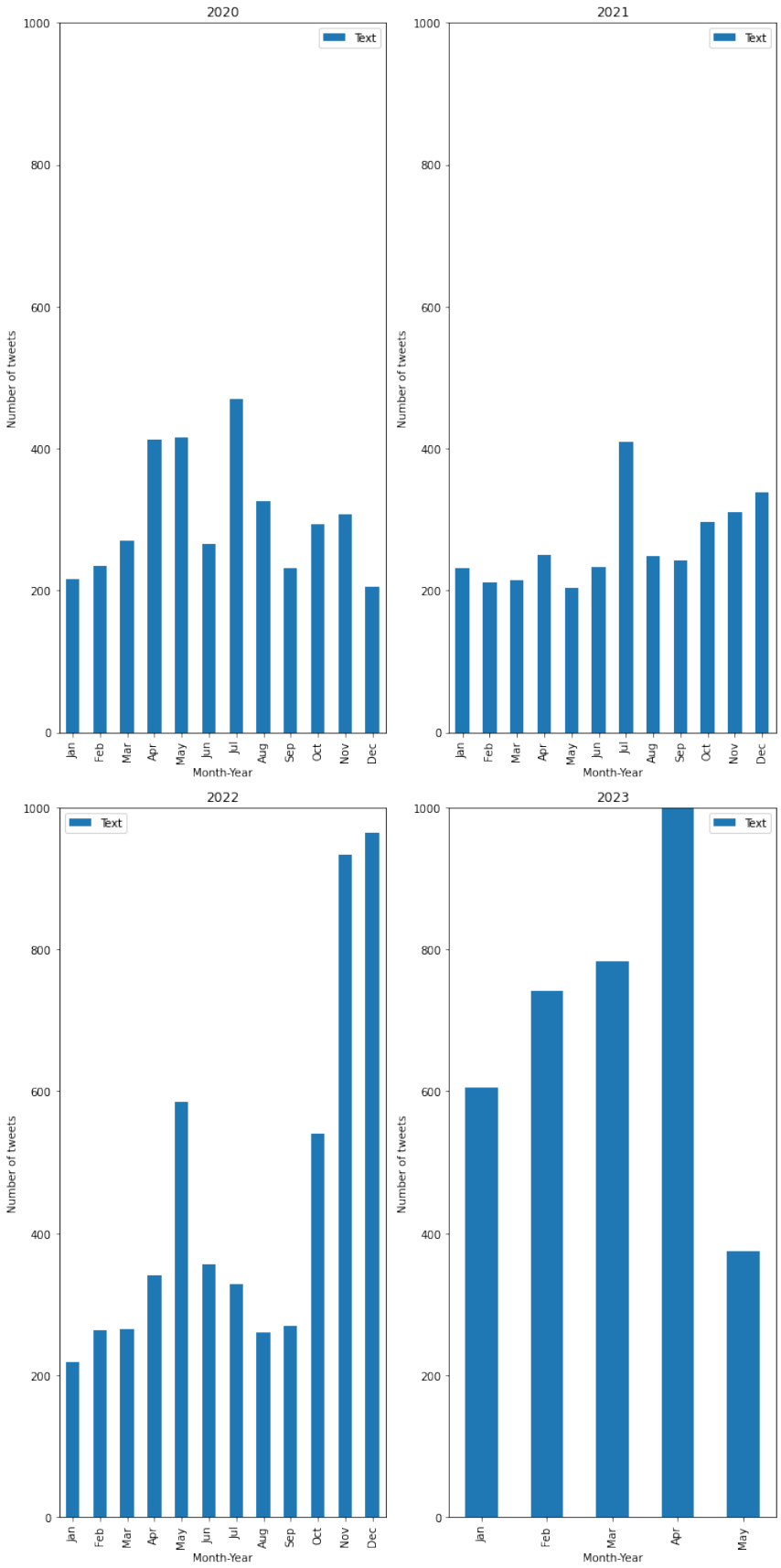

Figure 1: Monthly number of tweets for each year (2020-2023)

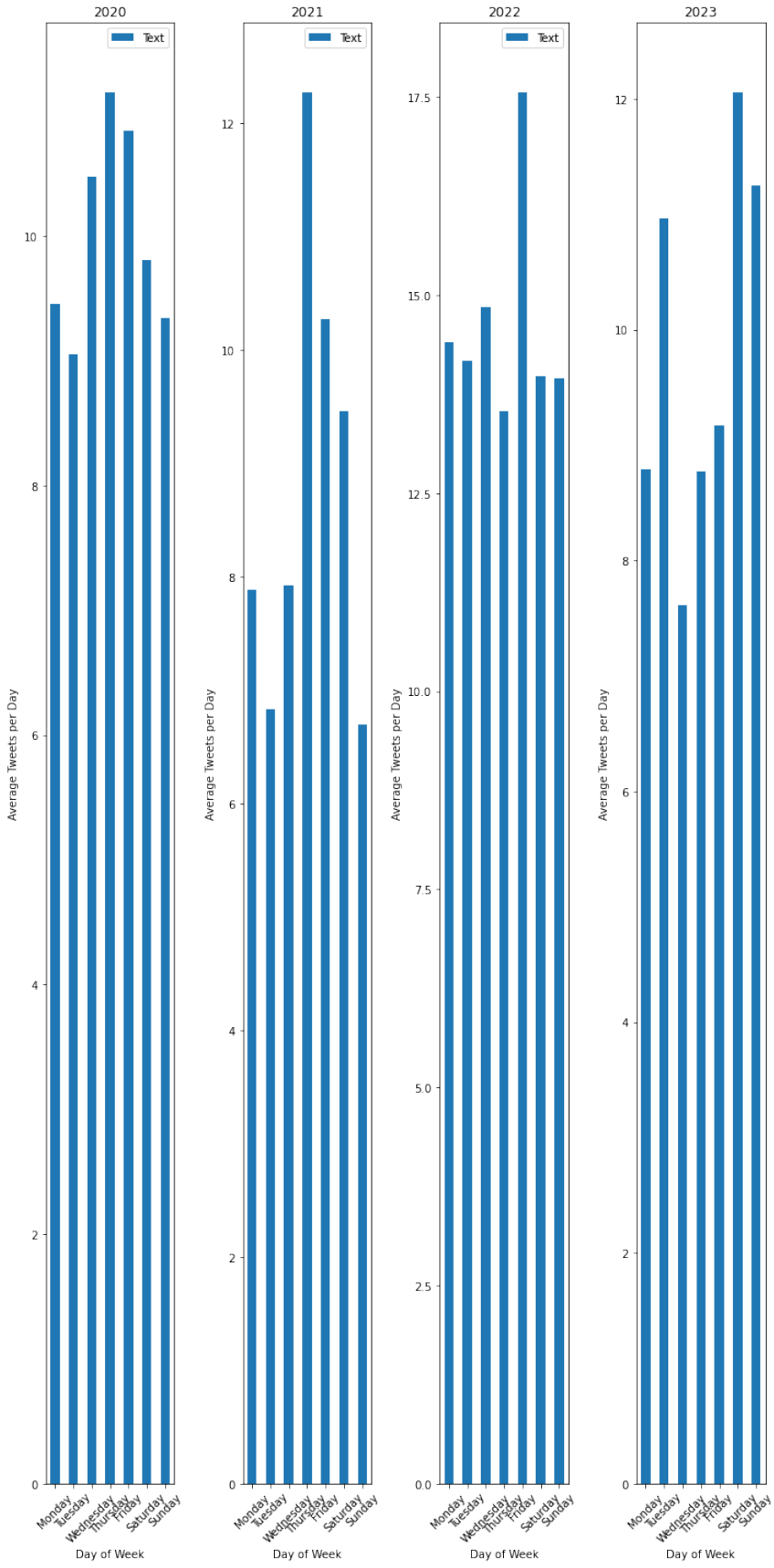

Figure 2: Average tweets per day for each year (2020-2023)

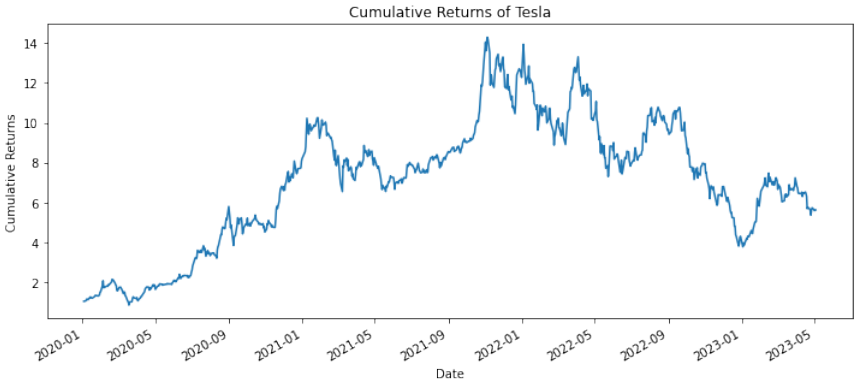

Figure 3: Cumulative Returns of Tesla for our time frame for analysis

Figure 4: Boxplot of daily Tesla's stock returns

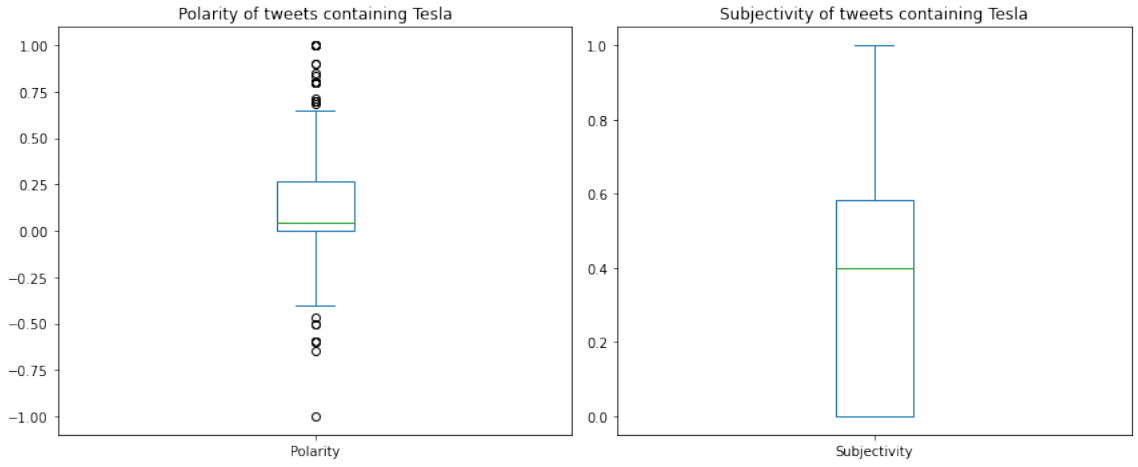

Figure 5: Boxplots for tweets' polarity and subjectivity values using Textblob

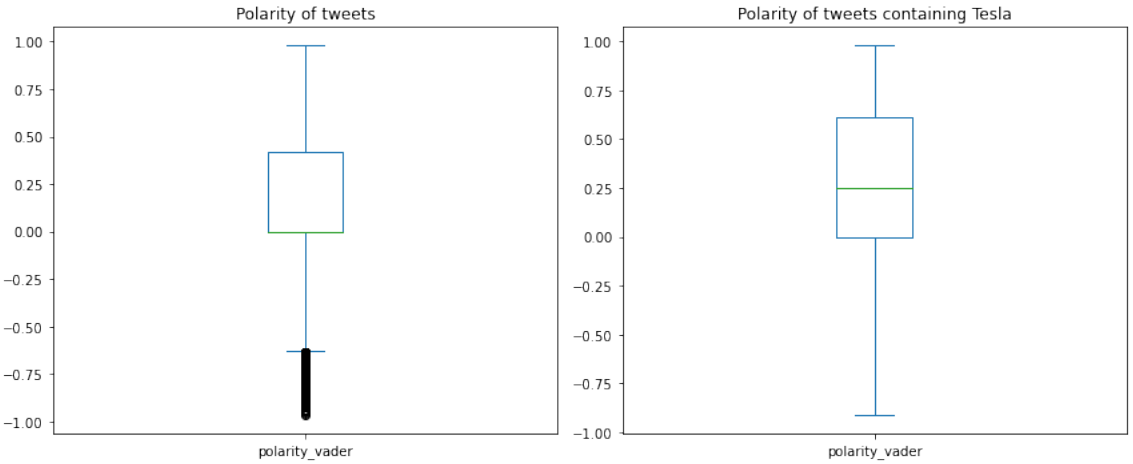

Figure 6: Boxplot for tweets' polarity values using VADER

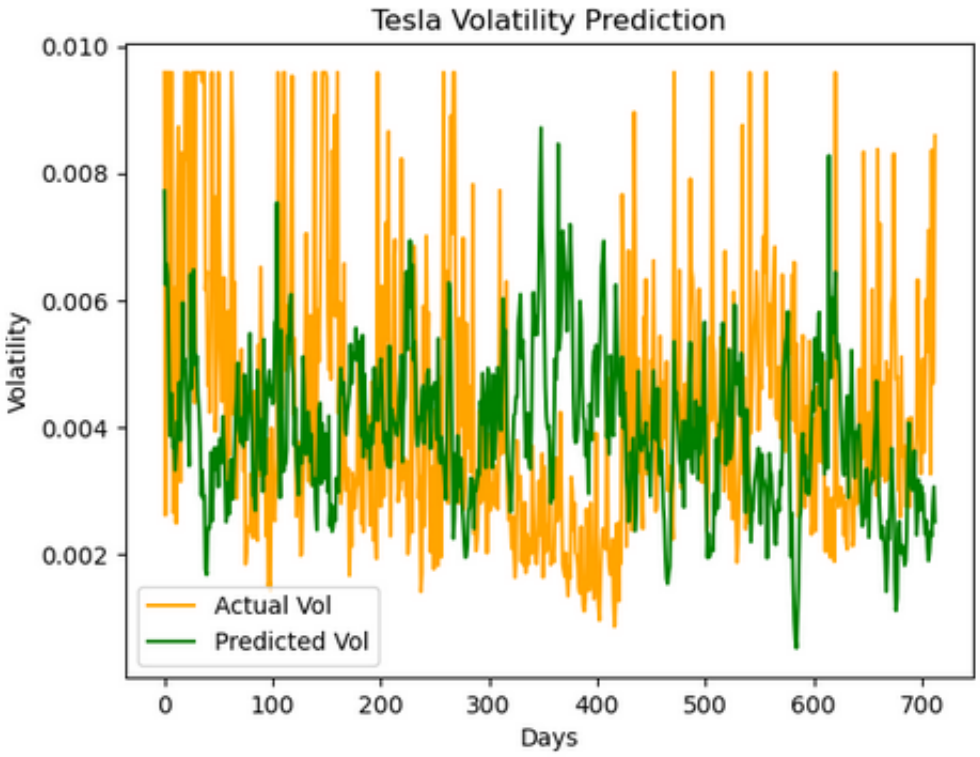

Figure 7: In-sample prediction with the simple model

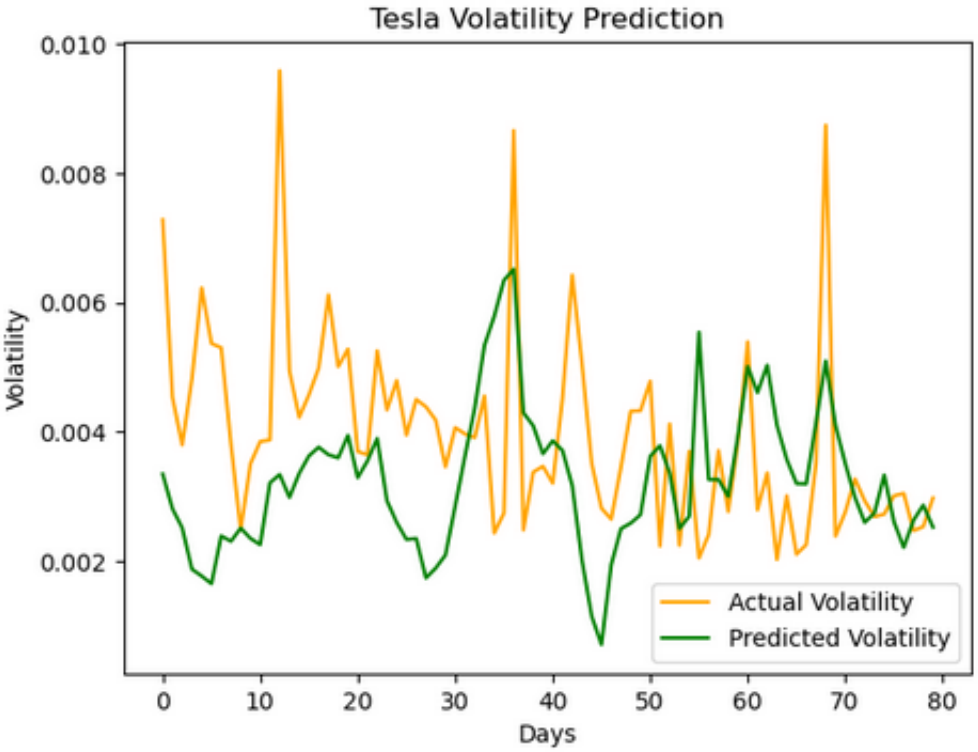

Figure 8: Out-of-sample prediction with the simple model

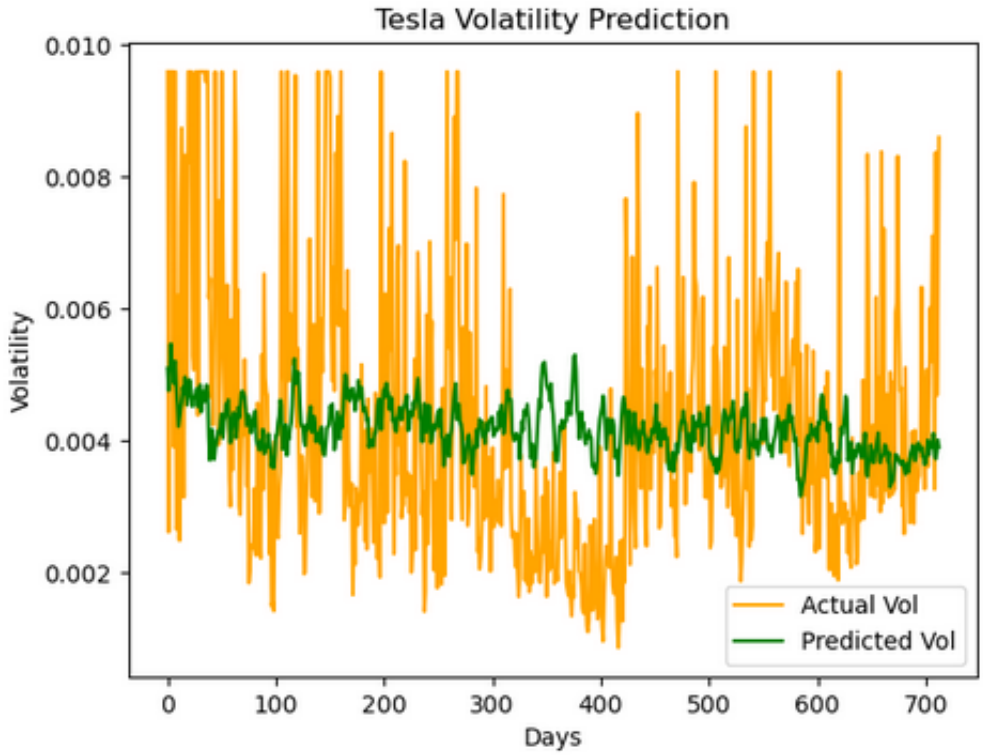

Figure 9: In-sample prediction with the complex model

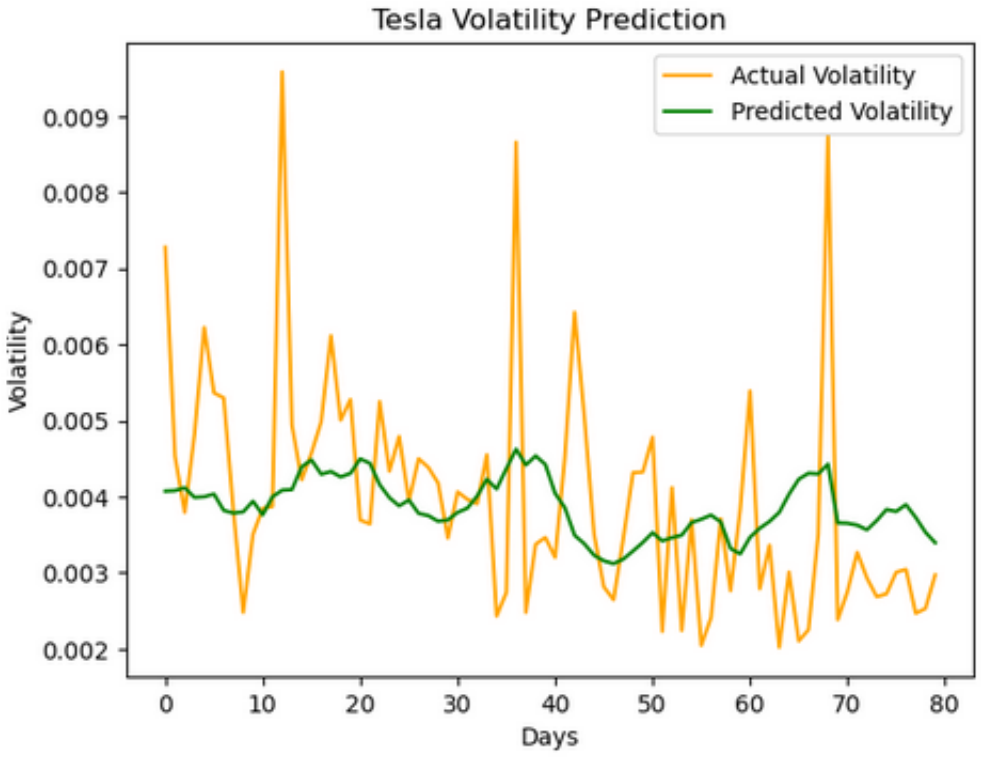

Figure 10: Out-of-sample prediction with the complex model

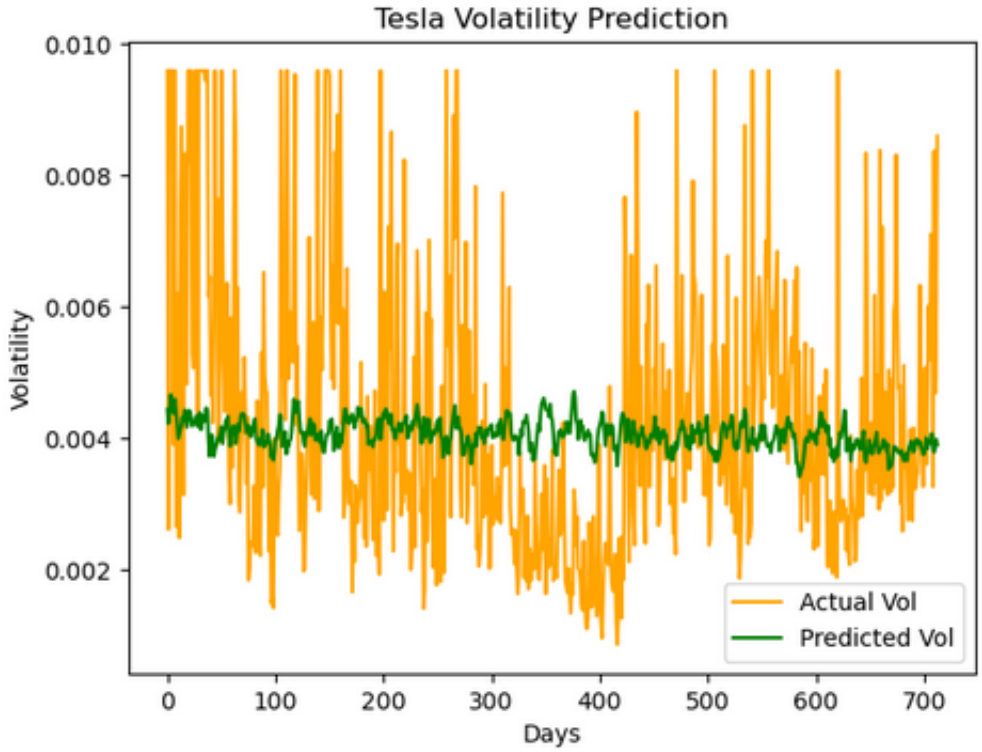

Figure 11: In-sample prediction with the complex boostrapped model

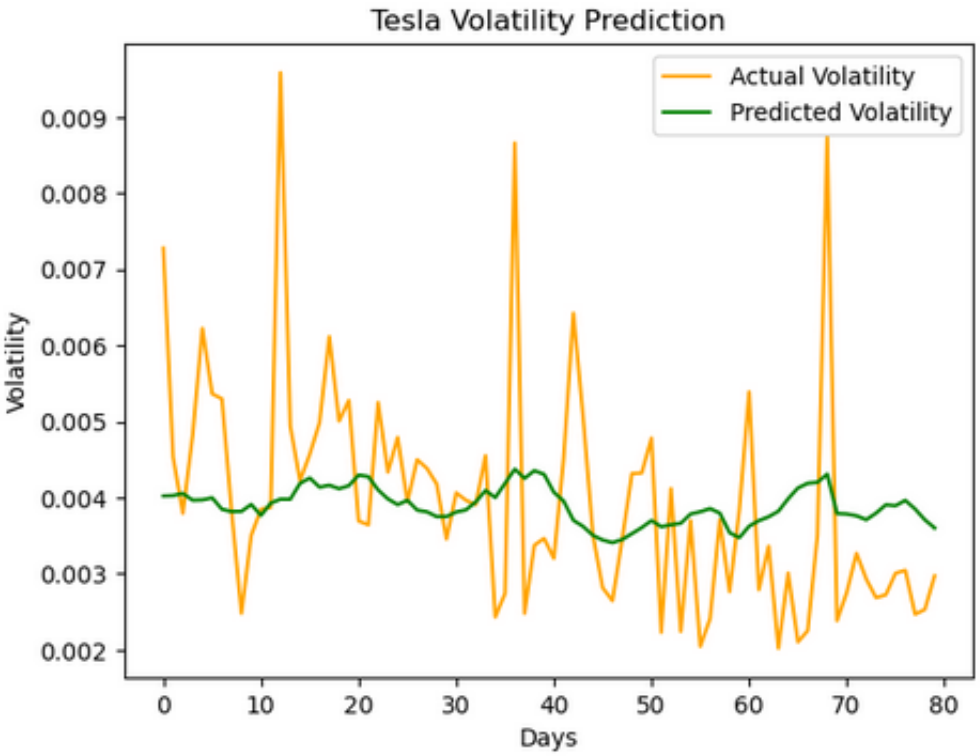

Figure 12: Out-of-sample prediction with the complex boostrapped model

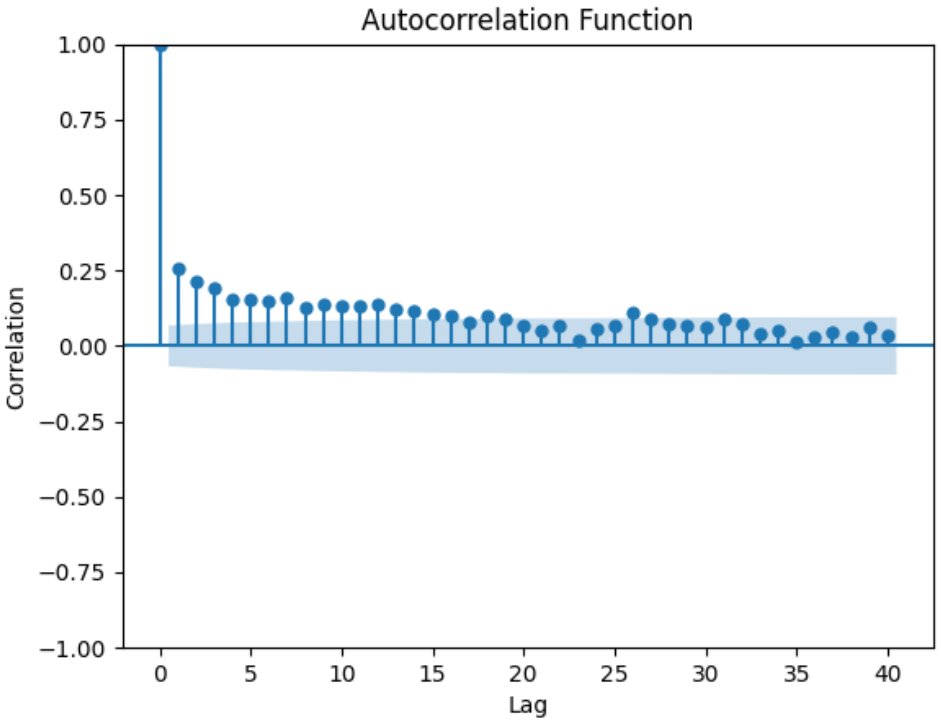

Figure 13: ACF Plot of RV

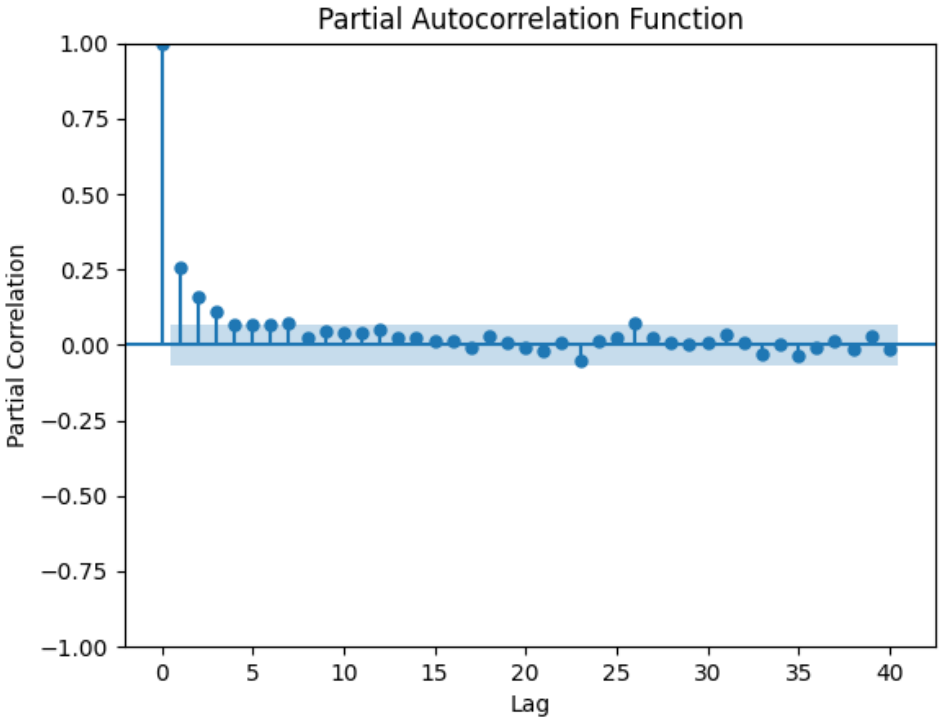

Figure 14: PACF Plot of RV