ERASMUS UNIVERSITY ROTTERDAM

Erasmus School of Economics

Master Thesis Econometrics and Management Science:
Business Analytics and Quantitative Marketing

Feasible Time Series Forecaster

Author:
Ziad Massali ([redacted])

Supervisor:
prof.dr. (Robin) RL Lumsdaine
Second Assessor:
dr. (Onno) O Kleen

15th July 2024

Abstract

This study aims to explore the efficiency of utilizing Artificial Intelligence (AI) and investor sentiment analysis from online platforms to predict stock market movements. To predict returns, a new model, TimeSeriesForecaster (TSF), which is a transformer-based model with a GPT-2 backbone, is introduced. The model's predictive ability is compared to traditional econometric models like Vector AutoRegression (VAR) and AutoRegressive Integrated Moving Average (ARIMA), as well as a newer AI-driven Long Short-Term Memory (LSTM). Also, bias identification techniques are explored to ensure transparent and ethical AI-driven predictions. The findings demonstrate that TSF has the capabilities to handle the dynamic and non-linear nature of financial markets. The contributions of this research are a comprehensive understanding of transformer-based models' potential in stock market prediction. The insights provided are valuable for investors, policymakers, and researchers. These findings set the stage for more accurate, computationally feasible, and transparent predictive modeling in financial markets.

1 Introduction

The fast-developing digital technologies and the exponential growth of online data have significantly influenced the dynamics of financial markets. This combination has led to intriguing developments regarding the improvement of predictive modeling. Currently, prediction models have the ability to predict stock market performance through the analysis of sentiment, particularly sentiments posted on online platforms. The aim to delve into the efficiency of leveraging Artificial Intelligence (AI) to predict stock market trends by utilizing analysis of consumer and investor sentiment can be divided into two aspects: firstly, an understanding of how AI can use the unstructured data of online sentiment to forecast stock market movements is gained; and secondly, the identification of the biases inherent in sentiment analysis to ensure integrity and consumer privacy. To address these aspects, the following three key questions are answered:

1. By using AI, how can the analysis of sentiment data from social media or online platforms improve the accuracy of forecasting models in predicting stock market performance?
2. How do traditional econometric models match up against AI-driven models, like transformer-based models, when it comes to forecasting stock market performance using sentiment analysis of social media data?
3. When implementing sentiment analysis for forecasting, how can businesses identify biases and what are the broader implications regarding market integrity?

The relevance of this research lies within the significant impact of accurately forecasting the stock market on economic planning, investment decisions, and the stability of financial markets. Investors and consumers often openly express their opinions, reactions, and expectations on recent market developments on social media. This digital footprint can offer unprecedented insights into public sentiment, potentially acting as a strong indicator of market movements. However, the challenging aspect of sentiment data is to accurately extract, analyze, and utilize it.

The existing knowledge within the field of predicting time series, is insufficient due to the limited amount of research on computationally feasible methods to combine textual data from social media with numeric stock data to predict stock prices. Current approaches mostly use a limited amount of stocks and social media data, lack interpretability to detect bias, or are computationally intensive.

To address this lack of knowledge, the development of an understanding of time series modeling is crucial in effectively designing a time forecaster that incorporates AI-driven models. The model proposed in this study, TimeSeriesForecaster (TSF), needs to uniquely combine textual analysis with predictive modeling. The combination of textual analysis and predictive modeling highlights the significant impact of real-time public opinions on market trends. Moreover, to identify biases within the data and the models, appropriate techniques that quantify the impact of each predictor, are essential.

Before the widespread use of the internet, time series forecasting mostly utilized traditional statistical methods such as the AutoRegressive Integrated Moving Average (ARIMA) model and Exponential Smoothing (ES) (Box and Jenkins (1970), McKenzie (1984)). These local univariate models performed exceptionally well during their time for their applications. They are widely used in financial and economic forecasting due to their performance and well-established methodologies (Hyndman & Khandakar, 2008). However, ARIMA and ES only consider a single time series, which potentially ignores the interaction with other correlated variables. Multivariate models such as the Vector AutoRegression (VAR) model, address this limitation by allowing multivariate interrelated input and prediction (Sims, 1980). These often outperform univariate models (Zivot & Wang, 2003). Despite this advantage, VAR models have a strong

linearity assumption that likely will be violated due to stock market returns exhibiting time-varying behavior (Kanas, 2005).

Lately, deep learning models have emerged on the frontline of time series prediction as a result of their ability to learn multiple levels of abstraction and handle high-dimensional data (Han, Zhao, Leung, Ma & Wang, 2019). Among these models, the Long Short-Term Memory (LSTM) model has gathered increasing attention in the prediction of time series due to its ability to better capture long-term dependencies (Y. Li, Zhu, Kong, Han & Zhao, 2018). Siami-Namini, Tavakoli and Siami Namin (2018) have shown that LSTM models can outperform traditional benchmarks such as ARIMA and machine learning models like the Random Forest, XGBoost, and Recurrent Neural Network (RNN) in predictions (Nabipour et al., 2020). However, there remains a lack of knowledge in the incorporation of social media data. Prior research suggests that the incorporation of social media data enhances stock price predictions (Nguyen & Shirai, 2015; Khan et al., 2020).

For the incorporation of social media data into the previously mentioned models, sentiment analysis is a crucial aspect. Sentiment analysis techniques have evolved through several models and methods. Recently, researchers' interests have shifted more toward the use of AI-driven models for sentiment analysis (Stine, 2019). Specifically, Generative Pre-trained Transformers (GPTs) show superior performance in complex sentiment classification through contextual understanding when compared to traditional techniques (Kheiri & Karimi, 2023). However, GPTs are computationally expensive, which leads to the consideration of an important balance between computational feasibility and performance.

While considering the balance between computational resources and performance, VADER (Valence Aware Dictionary and sEntiment Reasoner) emerges as an appropriate choice for the sentiment analysis (Hutto & Gilbert, 2014). VADER, which is a simple rule-based sentiment analysis model in a social media context, combines a sentiment lexicon with grammatical rules. A notable aspect is its performance which matches other sophisticated sentiment analysis techniques.

With sentiment analysis fundamentals in place, researchers have been increasingly recognizing the value of integrating textual data into predictive models. For example, Sadik, Date and Mitra (2019) have explored the use of quantified news sentiment to predict stock market prices using a GARCH model. Their findings support the current sentiment that the incorporation of news articles enhances the predictive ability of models (X. Li, Xie, Chen, Wang & Deng, 2014). Nonetheless, there has been limited notable research into the use of large social media data to enhance other traditional models like VAR or ARIMA. This gap is an opportunity for further investigation into the incorporation of social media data to enhance predictive performance.

One notable contribution to enhance prediction by using social media data, is the study of Ji, Wang and Yan (2021), which demonstrated the incorporation of social media data in a novel LSTM model. This novel method, called Doc-W-LSTM, outperforms both ARIMA and a standard LSTM that only utilizes financial data. Even though the results are promising, the study has a notable limitation due to its focus on a single stock, which raises concerns about the robustness of the findings.

Alongside these developments, Large Language Models (LLMs), which are AI models to understand human-like text, have shown an exceptional ability to accurately predict time series in a zero-shot fashion. Zero-shot forecasting is the prediction on unseen data without any specific training. Specifically, Gruver, Finzi, Qiu and Wilson (2023) introduced the LLMTime model, a forecaster utilizing LLMs, that achieves better performances than classical and machine learning-based estimators. LLMTime splits time series into strings of numbers. Therefore, it enables LLMs to predict future sequences without specific training. This capability of prediction

lies in their inherent ability to understand multimodal distributions, manage missing data, and utilize textual information. The notable limitations are the maximum of analyzed sequences and the reliance on a paid service.

Some of the limitations of LLMTime are addressed by the development of TimesFM (Das, Kong, Sen & Zhou, 2024). TimesFM seeks to mitigate issues inherent to the designs of LLMs, specifically concerning their processing of textual data. After training on a vast dataset of time series data, TimesFM demonstrated not only the ability to understand the underlying dynamics of time series more effectively but also achieved enhanced zero-shot performance at a smaller model size. Therefore, TimesFM achieves a better allocation of computational resources, in contrast to the computationally intensive LLMTime. Despite the impressive performance, TimesFM remains a zero-shot model and previous research has indicated that few-shot models, which are trained on a few examples, can outperform zero-shot models (Schick & Schütze, 2021; Yin, Rajani, Radev, Socher & Xiong, 2020). Also, TimesFM is not publicly available, which means that the extensive training phase poses a significant barrier.

As feasibility plays a big part in the development, the advancements made by Zhou, Niu, Wang, Sun and Jin (2023) are crucial to consider. The One Fits All (OFA) model addresses the challenge of limited computational resources by leveraging a GPT-2 model, which is an open-source LLM. OFA utilizes the original weights in GPT-2 to achieve enhanced performance in several time series analysis tasks compared to state-of-the-art forecasters. The results suggest that open-source transformers like GPT-2, offer an alternative to more computationally intensive models like the TimesFM.

To ensure valid results and interpretability, bias detection is a crucial aspect of this research due to the use of social media data that contains individuals' opinions. While the previously mentioned models excel in predictive accuracy, they often lack mechanisms for identifying biases. These biases could skew predictions and affect their applicability across different scenarios.

To address these concerns, eXplainable AI (XAI) techniques, which enhance the interpretability of AI's decisions (Adadi & Berrada, 2018), prove vital in identifying biases more effectively. There is a clear distinction between bias detection within LLMs and machine learning algorithms. Bias detection in LLMs is often related to natural language processing capabilities (Gallegos et al., 2023). Machine learning algorithms on the other hand focus on biases linked to feature importance.

A suitable technique for LSTM, is the Shapley values proposed by Štrumbelj and Kononenko (2013). These are derived from cooperative game theory to quantify the contribution of each variable to the predictions of machine learning models. Recent studies have adapted Shapley values within LSTM to assess and understand the predictions (Ibrahim, Mesinovic, Yang & Eid, 2020; Zou & Petrosian, 2020). However, these approaches do not take time series forecasting.

Both LLMs and TimesFM have implemented multi-head self-attention mechanisms, which allow the model to focus on relevant historical patterns and dependencies by computing multiple attention heads (weights) to attend to previous values (Vaswani et al., 2017). However, as the name suggests, these mechanisms only supply weights for the attention to each feature in their own feature context. To gain a better overall understanding of the importance of each feature within the transformer-based models, permutation importance is a suitable candidate (Breiman, 2001). Permutation importance shuffles the input data per feature and then computes the difference in performance with the original data. So, this technique quantifies the effect of each feature on the loss function.

The aim is to contribute to the academic field of predicting time series in multiple ways. Firstly, the computational and financial feasibility of time series forecasters is addressed. As discussed previously, LLMTime utilizes a paid service for its forecasts, which can be costly when forecasting

larger time series like stock prices. While TimesFM is more achievable, training a model on billions of data points requires large computational power. So, the first contribution is to propose a computationally feasible model by incorporating a pre-trained GPT-2 backbone into the TimesFM architecture. The aim is to adhere to the TimesFM's architecture while reducing the computational burden.

Secondly, the TimesFM and LLMtime are extended by exploring few-shot learning. Few-shot learning allows the model to adapt to new time series with only a small number of examples, which can be very useful in instances where gathering large amounts of data is unrealistic or expensive. Therefore, few-shot learning enhances the practicality of the models.

Thirdly, an objective is to bridge the gap between textual social media data and time series forecasting in transformer-based models. Even though previous research has utilized textual social media data to improve stock prediction, limited exploration has been done specifically in the context of using the combined data sources within transformer-based models for time series forecasting. The intention is to explore this area, which builds upon the finding that social media is a potent forecasting tool.

Lastly, the goal is to address the importance of bias explanation in time series forecasting. The understanding of these biases improves the reliability and accuracy of predictions. So, by using permutation importance, employing XAI for the LSTM, and testing model validity in the time series forecasters, these biases are identified. To the best of my knowledge, this research is the first to combine textual data with time series data to design an explained few-shot transformer-based model. Thus, this combination addresses the need for interpretability, feasibility, and bias identification in the novel architecture.

This research aims to contribute by providing a more nuanced comprehension of how AI can be used to predict stock market performance through sentiment analysis. It also strives to understand how to address the challenges of using social media data. Moreover, it looks to explore the implications of AI-driven market predictions on market integrity and consumer privacy. This study will offer valuable insights for investors, policymakers, and the academic community. It may spearhead the development of more accurate, computationally feasible, and transparent predictive modeling.

The remainder of this paper is structured as follows: Section 2 describes the data used in this study, as well as its preprocessing steps and some descriptive analysis. Section 3 elaborates on the methodology, which thoroughly explains the proposed TSF model, benchmark models, sentiment analysis techniques, bias identification methods, and the hyperparameter tuning technique. Section 4 presents the results of the study, where the performance of TSF is compared with benchmark models and the feature importance to identify potential biases are analyzed. Finally, Section 5 concludes this study with a summary of the key findings as well as a discussion on the implications and limitations of the research and the suggestion of avenues for future work.

2 Data

An extensive dataset that combines both numeric and textual data is examined to answer the central questions. This section provides an overview of the data, including sources and graphs. Furthermore, the cleaning process to prepare the data for analysis is described. This dataset consists of stock data, sentiment data, and Reddit posts and comments. This unique combination of data allows gauging an understanding of the influence sentiment has on the predictions of various models.

2.1 Numeric Data

2.1.1 Stock Data

The main focus is to predict stock prices. Thus stock data is the most important data source. 5 different stock indices are collected. These are the S&P 500, Russell 2000, Nasdaq Composite (Nasdaq), FTSE 100, and Dow Jones Industrial Average (Dow Jones). These indices were selected for diversification reasons. Firstly, the Dow Jones is an index of 30 large US companies and is one of the most watched indices globally. Then, to provide a broader representation of the US economy than the Dow Jones, the study also considers the S&P 500 index, which includes 500 of the largest publicly traded companies in the US. Next, to account for technology companies that are influential in the market dynamics, the Nasdaq is incorporated. Another important consideration is the inclusion of relatively smaller US companies, which is achieved by adding the Russell 2000 to the dataset. Lastly, to filter out US-specific effects, the British FTSE 100 is included as it is a widely recognized and significant non-US index. The selection of indices grants the ability for a more comprehensive analysis by capturing various segments. Even though the goal is to design a general forecaster, this study uses stock market indices to demonstrate the performance of complex and volatile time series data. The historical daily stock index data was sourced from Yahoo Finance. The data spans from January 1, 2010, to February 1, 2024. The intention is to focus on a specific time period based on the availability of textual data, which is discussed in the next subsection. The dataset contains the date, daily closing price, and volume. Due to the data's consistency and completeness, there is no need for imputation. The data on non-trading days (holidays and weekends) is not collected by Yahoo Finance. The handling of the data on non-trading days is discussed in the following sections.

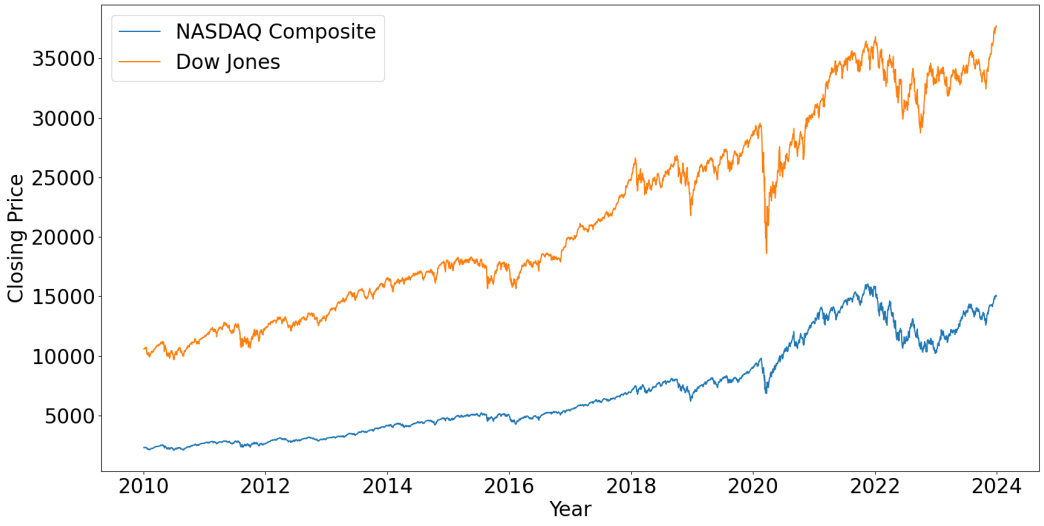

Figure 1: Closing Price of Dow Jones (Top) and Nasdaq (Bottom)

The daily closing prices of the indices are shown in Figure 1 and Figure 2. As a result of the COVID-19 pandemic, all indices crashed in March 2020. After this period, all indices experienced a strong recovery, where some of them even increased to record highs. Furthermore, all indices have shown significant growth from 2010 to 2024. Interestingly, Dow Jones, Nasdaq, Russell 2000, and S&P 500 have all grown more rapidly than the FTSE 100, which has lagged behind the US indices. This lack of growth might be partly attributed to the uncertainties of Brexit (Sampson, 2017). As the goal is to predict the future values of an index without using other indices, the use of the FTSE 100 without any modification is justified.

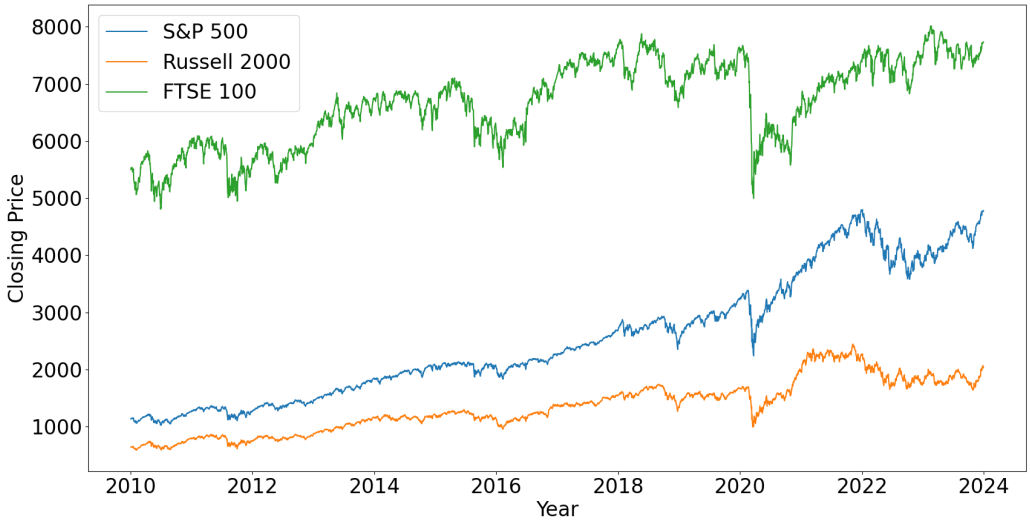

Figure 2: Closing Price of S&P (Middle), Russell 2000 (Bottom), and FTSE 100 (Top)

2.1.2 Consumer and Investor Sentiment

Another vital aspect of this study is the use of consumer and investor sentiment to predict stock prices. For this purpose, the Index of Consumer Sentiment (ICS) collected by the University of Michigan is included. Also, the AAII Sentiment Survey (AS) collected by the American Association of Individual Investors (AAII) is gathered. The ICS is derived from a monthly survey, which quantifies consumer confidence such as confidence in the economic performance of the US (Kellstedt, Linn & Hannah, 2015). This survey collects information about consumers' economic conditions and financial situations in the past, present, and future. The ICS possesses predictive power as it contains information that precedes shifts in consumer spending patterns (Lovell, 2001). The ICS is an index value, with a baseline of 100 set in 1966. So, a value above 100 indicates that consumers are more confident about the current economy than consumers were in 1966. Monthly ICS data spanning from January 2010 to February 2024 is obtained. In this case, due to the consistency and completeness, no preprocessing is required other than the alignment with the corresponding stock data.

The AS is an index derived from a survey conducted weekly among AAII members. It quantifies the expectations of individual investors regarding the stock market. The central question posed in this survey is: '*What direction do you feel the stock market will move over the next six months?*' Survey responses are categorized as bullish, bearish, and neutral. The respective proportions of each response category are computed and published on a weekly basis. The AS is considered a contrarian indicator, as the market typically reverses in cases of high percentages of bullish or bearish investors (Fisher & Statman, 2000). This phenomenon is attributed to the overreaction of investors towards a market sentiment. Like the ICS, the AS data spans from

January 2010 to February 2024, requiring only date alignment with the stock data. To avoid multicollinearity, the neutral proportion is dropped.

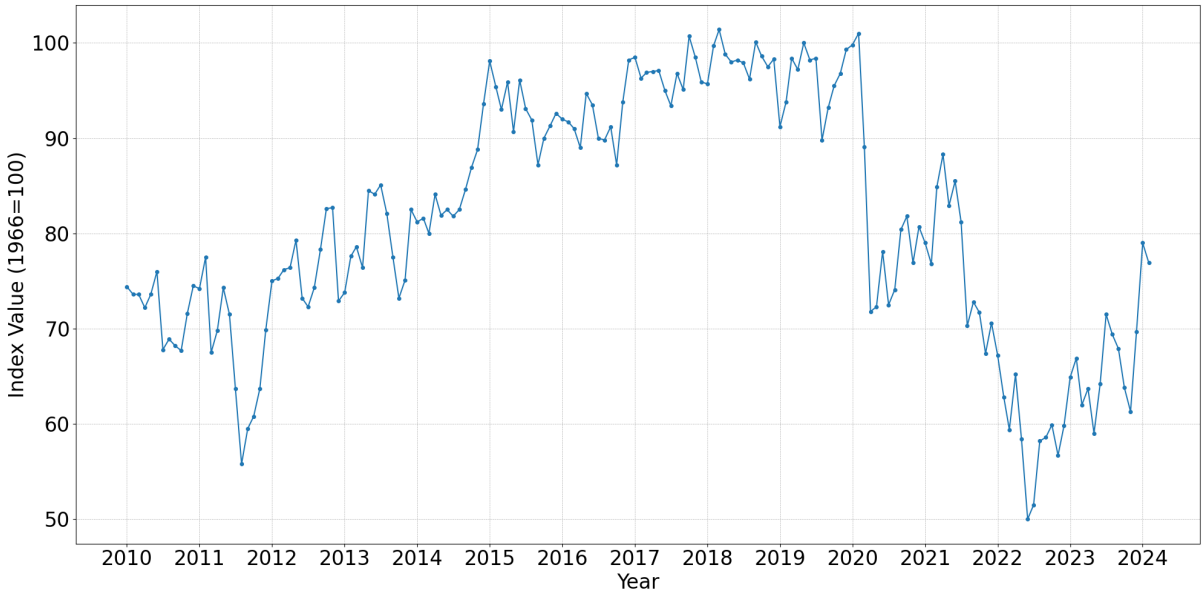

Figure 3: Index of Consumer Sentiment Across The Years

Figure 3 shows the monthly values of the ICS across the years. In February and March 2020, a sudden drop in confidence can be observed, due to the COVID-19 pandemic. Moreover, the ICS reached historically low levels in June and July of 2022, possibly caused by the highest Consumer Price Index since 1981 (Hobijn, Miles, Royal & Zhang, 2022). Interestingly, the ICS rarely exceeds 100 after 2010, which suggests a period of economic uncertainty for consumers.

To visualize the AS in Figure 4, the weekly proportions were aggregated into monthly averages and the difference between bullish and bearish sentiments is used for readability reasons. So, positive values indicate a more bullish sentiment in a given month, while negative values indicate a more bearish sentiment among investors. The AS plot demonstrates significant volatility in investor sentiment. In some instances, the plot supports the findings of Fisher and Statman (2000), where an overly bullish or bearish sentiment in a month tends to be corrected in the next months. The AS and ICS display some correlation, particularly after 2020. Specifically, significant increases in bearish investor sentiment coincide with declines in the ICS. Thereby this finding emphasizes a negative correlation between bearish investor sentiment and consumer sentiment.

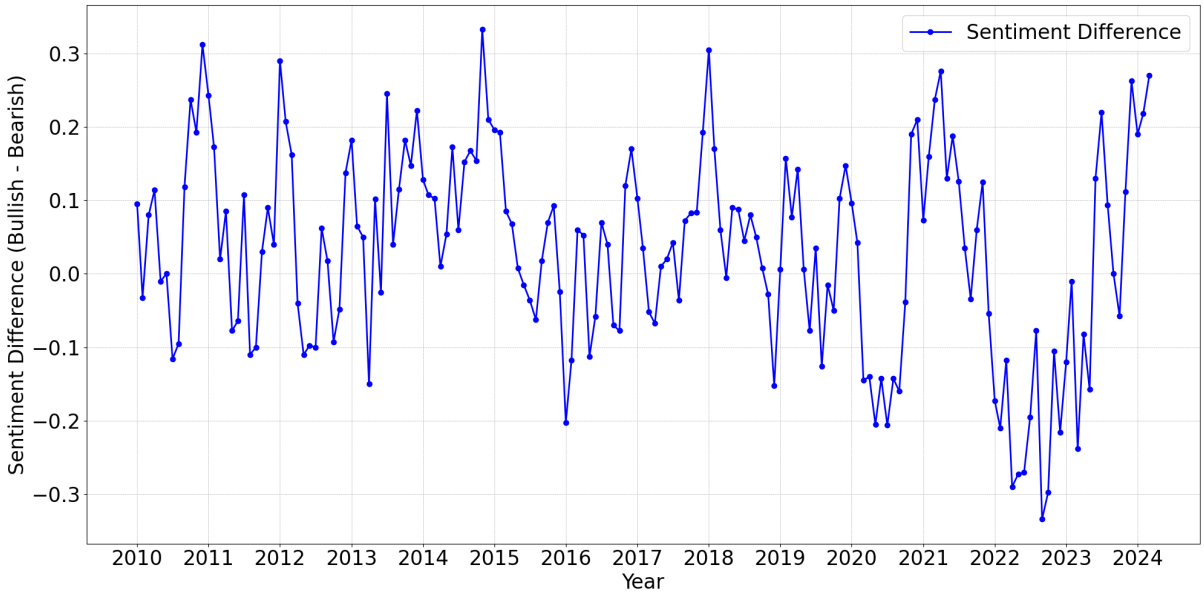

Figure 4: Investor Sentiment Across the Years Reported in Monthly Net Sentiment Difference (Bullish - Bearish Proportion of Investors).

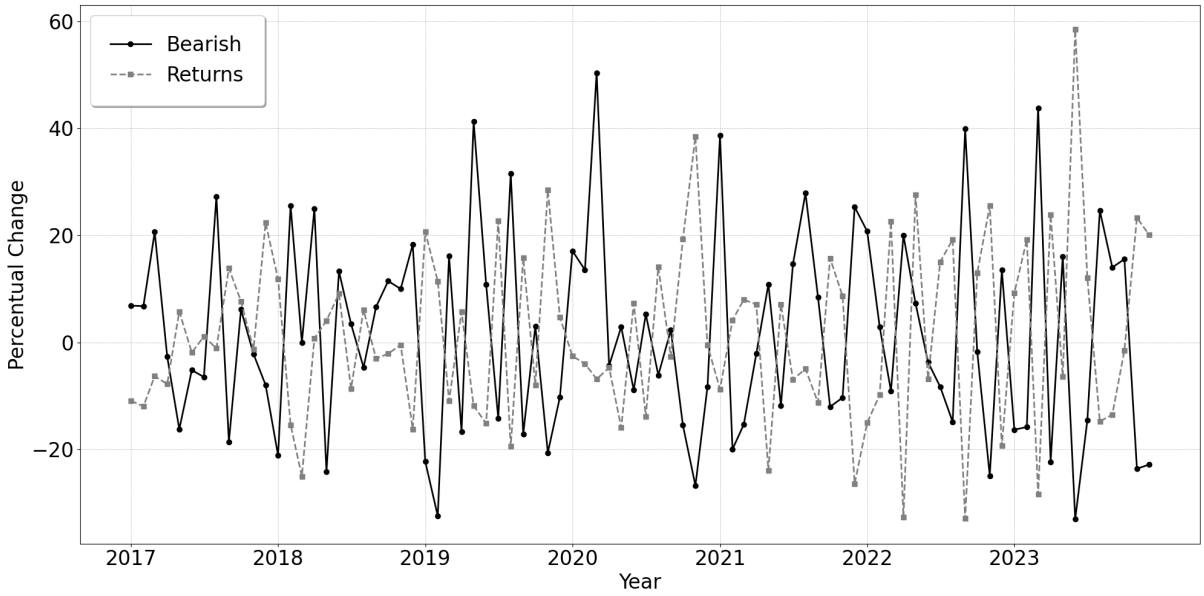

Figure 5: Percentage Change per Month in Bearish Sentiment and S&P 500 Returns

In Figure 5, the percentual change per month in bearish sentiment and S&P 500 returns are displayed. For readability reasons, only the years after 2017 are considered. In the figure, the contrarian movement of the bearish proportion of investors can be seen. A positive percentage change in the return of the S&P 500 is usually paired with a negative percentage change in the bearish proportion of investors and vice versa. These movements are mostly of the same scale, which indicates a strong negative correlation. In some cases, the bearish sentiment seems to lag behind the returns.

2.2 Textual Data

2.2.1 Reddit

To integrate social media data into this study, Reddit textual is used data. Reddit offers several advantages due to its architecture. Firstly, Reddit consists of specific communities known as subreddits. So, to limit the amount of noise, financial subreddits are selectively filtered. Secondly, even though the platform is moderated, Reddit contains unfiltered opinions providing genuine investor and consumer sentiment. Thirdly, contrary to other social media platforms, Reddit data is readily accessible for research. Specifically, the Pushshift Reddit dataset is of great interest (Baumgartner, Zannettou, Keegan, Squire & Blackburn, 2020). This dataset contains all submissions and comments of the top 40,000 most active subreddits. The dataset spans from June 2005 to February 2024. Therefore, this extensive dataset is invaluable for the goals of this study. Nonetheless, it is deemed essential to distill the relevant information given the size and variety of the data.

In this study, 25 finance-related subreddits are considered, which can be found in the Appendix. The selected subreddits aim to encompass a diverse range of financial sentiments and are based on activity levels. Firstly, the collection of subreddits contains the 5 biggest subreddits on general investing and stock market discussions, which are the investing, StockMarket, stocks, SecurityAnalysis, and investing_discussion subreddits. Then, to incorporate trading-focused sentiments, the Daytrading, algotrading, options, pennystocks, and thewallstreet subreddits are also considered in this study. Furthermore, to capture sentiments related to specific investment strategies, the ValueInvesting, dividends, and Bogleheads subreddits are included.

Next, all of the available wallstreetbets-related subreddits are considered due to their influence on the market dynamics by viewing trading as betting. This aspect is discussed later in this section. To consider a broader view of investment sentiments in different markets, the realestateinvesting, RealEstate, ETFs, and Wallstreetsilver subreddits are also included. Lastly, to incorporate sentiments on long-term investment strategies and economic outlook the financialindependence, fatFIRE, leanfire, and ExpatFIRE subreddits are evaluated as well. The selection of subreddits captures a multitude of investment perspectives that range from volatile short-term trading to long-term wealth accumulation. It is important to note that the selection of subreddits, which is based on activity within the subreddits, probably has a significant impact on the findings of this study. The exploration of additional subreddits falls outside the scope of this research due to time constraints.

The total file size of all subreddits amounts to 18.2 GB. Each subreddit has two separate files: a comments file and a submission file. Both files include the textual content, a username, a creation date, a title (for submissions), and a unique ID. The unique ID allows for data linkage. Furthermore, to minimize the potential influence of user identity, the usernames are deleted to ensure user privacy. This helps in shifting the focus toward the textual content.

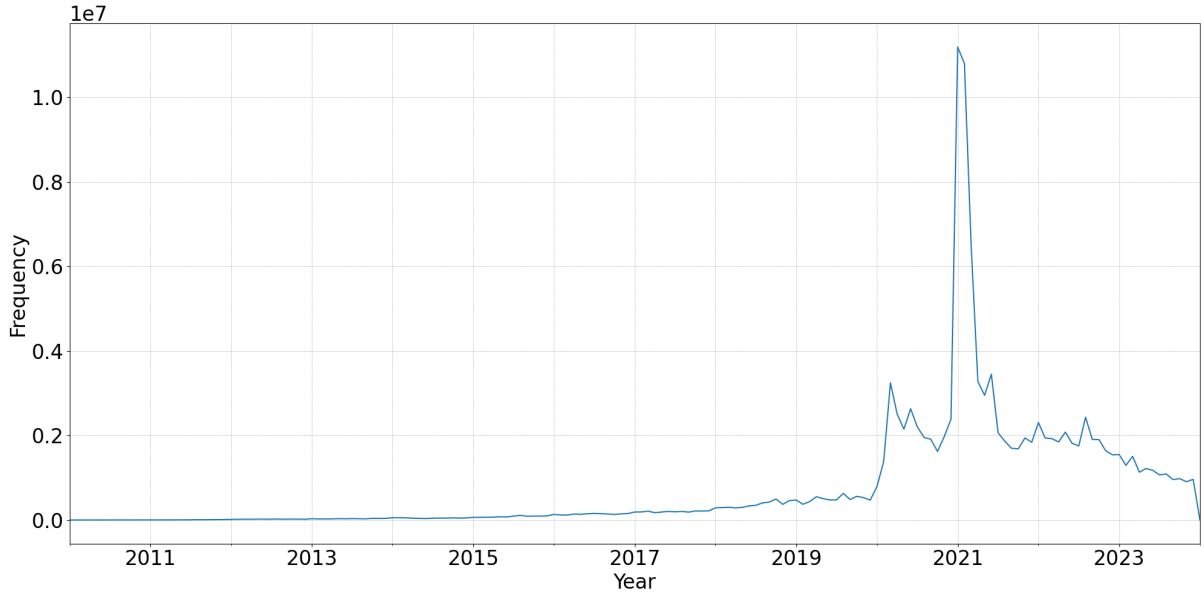

Figure 6: Monthly Reddit Posts and Comments Over The Years (Values Are in The Order of 10 Million)

Figure 6 illustrates that the highest amount of activity within the subset of subreddits occurred between 2020 and 2022. Therefore, this period is considered crucial for extracting the sentiment of users. Notably, subreddits with 'wallstreet' in their name, accumulated roughly 42 million comments and posts in 2021 alone. Since these subreddits often echo discussions about financial articles, the inclusion of financial articles is regarded as redundant. From 2010 to 2022, the total number of posts and comments on the selected subreddits is around 130 million, with the years 2020, 2021, and 2022 comprising 100 million comments and posts. Specifically, 2021 contains valuable data due to the GameStop incident that drove engagement. A group of investors on Reddit coordinated to drive up the stock price of GameStop to inflict losses on investments of hedge funds. This increased interest in trading this particular stock has led to a big shift in audience. The newer audience has a more novel view on trading and in expressing their sentiment. This provides a balance of sentiments of experienced and new investors. Therefore, the focus will be on the textual and numeric data collected between 2020 and 2022 given the activity of the users.

2.2.2 Preprocessing

The textual data underwent a thorough preprocessing procedure to ensure standardization and cleanliness before analysis. The procedure is designed specifically to handle noisy and unstructured social media data that contains HTML tags, URLs, special characters, and emojis. Initially, submissions and comments after January 1, 2020 are selected. Then, any comments that are linked to posts before the cut-off date are removed. This applies to submissions after January 1, 2020 without any comments as well. Submissions or comments removed by a moderator or the user, are also deleted as these do not contain usable textual information. After this, the text is cleaned by removing HTML tags, special characters, punctuation, emojis, and URLs. Next, contractions such as 'n't' and ''s' are expanded. Although emojis can convey sentiment, their representation varies across platforms. So, to standardize the text, these are removed. The text is subsequently standardized to lowercase. These steps focused on improving the learning ability and reducing variability. After cleaning, the text is tokenized into individual words by using the Natural Language Toolkit (NLTK) (Bird, Loper & Klein, 2009). This step prepares the text for computational processing by breaking the textual data into manageable chunks. Afterward, stop words(the, as, and with) that carry little sentiment, are deleted. The list of predefined stop

words from the NLTK library is utilized. To reduce the dimensionality of the vocabulary and normalize the text, lemmatization for the remaining words is employed. This reduces words to their base or dictionary form. Lastly, to simplify the identification of negation in the sentiment analysis, the negations are marked by adding a *'_NEG'* suffix to every word after a negation until the end of the negation scope (end of a sentence or a punctuation mark). For instance, *'I did not like the movie. It was bad'* will be transformed to *'I did not like_NEG the_NEG movie_NEG. It was bad'*. This step aids the algorithm in handling the impact of negations on the expressed sentiment in the text.

3 Methodology

The problem of forecasting is formalized by introducing notation adapted from Das et al. (2024). The goal is to develop a few-shot forecaster, TSF, which uses a historical sample t from the time series to amend the weights of the model and predict the next future values. The input of TSF is a multivariate time series, denoted by $\mathbf{Y}_{1:L} \in \mathbb{R}^{L \times k}$, where L is the length of the input sequence and k is the number of features. Specifically, $\mathbf{Y}_t$ contains both stock and sentiment data at time t. The prediction $\hat{Y}_{L+1} \in \mathbb{R}^m$ only encapsulates the numeric prediction of the closing price, where $m = 1$ in this case. TSF's goal is expressed as:

$$f : (\mathbf{Y}_{1:L}) \rightarrow \hat{Y}_{L+1}. \tag{1}$$

In this section, the architecture of TSF and the benchmarks are introduced. Furthermore, the methods for sentiment analysis and bias identification are thoroughly elaborated. Lastly, the hyperparameter tuning and evaluation metrics are discussed.

3.1 VAR

The Vector AutoRegressive model is a statistical framework that captures the dynamic relationship among multiple variables in multivariate time series data (Sims, 1980). By allowing multiple variables as input, it extends the univariate autoregressive model, which is restricted to a single variable. The VAR model generalizes to a system of equations, where each variable is modeled as a linear combination of its own lagged values as well as the lagged values of the other variables in the system. Therefore it accommodates an interdependent relationship between different variables. The order of the VAR model denoted as VAR(p), refers to the number of lagged values (p) used to predict the current value. In a system of k variables, the model can be represented by the following equation:

$$Y_t = c + A_1 Y_{t-1} + A_2 Y_{t-2} + \ldots + A_p Y_{t-p} + \epsilon_t, \tag{2}$$

where:

- t denotes the current time, ranging from 1 to T.
- Y_t is a $k \times 1$ vector of variables at time t.
- c is a $k \times 1$ vector of constants.
- $A_1, A_2, \ldots, A_p$ are $k \times k$ coefficient matrices.
- ϵ_t is a $k \times 1$ vector of error terms.

For the VAR model to be reliable, certain assumptions must be met. These assumptions are summarized in Lütkepohl (2005) and include:

1. Stationarity: The variables should be stationary, which means that mean, variance, and autocovariance remain constant over time.
2. Linearity: The utilized variables are assumed to have a linear relationship.
3. No perfect multicollinearity: The variables should have no perfect multicollinearity.
4. Sufficient lag length: An appropriate lag order is chosen to sufficiently capture the relationships between the variables.

All of these assumptions will be verified. To ensure stationarity, the ADF (Augmented Dickey-Fuller) test is employed. This assesses whether a unit root is present in the data (Dickey &

Fuller, 1979). Should the test results indicate non-stationary data, the data will be transformed into a stationary series by using the differencing method. Regarding the linearity assumption, this is tested through the BDS (Brock-Dechert-Scheinkman) test (Brock, Dechert, Scheinkman & LeBaron, 1996). Both tests will be conducted by using the standard settings of the corresponding functions in the statsmodels Python library and a 5% significance level (Skipper Seabold & Josef Perktold, 2010). If the results indicate non-linear relationships, the Threshold VAR (TVAR) is implemented. The TVAR model extends the VAR by allowing for regime changes with different coefficients. The transition between regimes is governed by a threshold variable, which will be determined in the following section. The K-means algorithm is utilized to identify the number of regimes (Hartigan & Wong, 1979).

To detect multicollinearity, the correlation matrix of the independent variables and the pairwise correlations are analyzed. In the case of a nearly perfect correlation, one of the variables will be randomly dropped. Lastly, the appropriate lag order will be determined by selecting the model that yields the lowest Mean Squared Error (MSE). This study does not consider the Bayesian Information Criterion due to the focus on predictive accuracy rather than balancing model fit with complexity.

The error terms must satisfy three conditions to ensure the validity of the VAR (Lütkepohl, 2005):

1. $E(\epsilon_t) = 0$: The expectation of the error terms should be zero.
2. $E(\epsilon_t \epsilon_t') = \Sigma_\epsilon$: The error terms should be homoscedastic.
3. $E(\epsilon_t \epsilon_s') = 0$ for $s \neq t$: The error terms should not be serially correlated.

These conditions will also be verified. The first condition will be validated by a t-test on the residual mean to verify whether the mean is significantly different from zero. The Breusch-Pagan (BP) test will be applied to assess whether the error terms are homoscedastic (Breusch & Pagan, 1979). Lastly, the absence of serial correlation will be checked by using the Ljung-Box (LB) test (Ljung & Box, 1978). The tests will be conducted using the standard settings of the statsmodels and SciPy libraries and a 5% significance level (Virtanen et al., 2020). If either correlated or heteroscedastic errors are detected, the Newey-West standard errors are employed (Newey & West, 1987). These errors are robust to both heteroscedasticity and autocorrelation. For simplicity, the parameters of the VAR model will be estimated by Ordinary Least Squares (OLS).

3.2 ARIMA

The AutoRegressive Integrated Moving Average (ARIMA) model is a statistical model that generalizes the AutoRegressive Moving Average (ARMA) (Box & Jenkins, 1970). The ARIMA fits both an AR and a MA model on the time series data. The AR models the variable of interest as a function of its lags, while the MA models the relationship as a function of past forecast errors. The integration part involves differencing the data one or more times to achieve stationarity. This improves the model's fit, and forecast accuracy compared to an ARMA model. The ARIMA(p,d,q) model refers to an ARIMA model with p order lags of the variable of interest, d order of differencing, and q order lags of the forecast errors. The model can be described by the following equation:

$$(1 - \sum_{i=1}^{p} \phi_i B^i)(1 - B)^d y_t = (1 + \sum_{i=1}^{q} \theta_i B^i)\epsilon_t, \tag{3}$$

where:

- y_t is the time series value at time t.
- B is a backshift operator, where $B^j y_t = y_{t-j}$.
- $(1 - \sum_{i=1}^{p} \phi_i B^i)$ denotes the autoregressive (AR) polynomial of order p.
- $(1 - B)^d$ represents the differencing operator of order d.
- $(1 + \sum_{i=1}^{q} \theta_i B^i)$ is the moving average (MA) polynomial of order q.
- ϵ_t is the error term at time t, which should be white noise.

If non-white noise residuals are detected, the Newey-West standard errors will be utilized. Some of the testing procedure is already discussed in Section 3.1. To ensure stationarity the ADF test is applied. In the event of non-stationary data, differencing is employed until the test results suggest that the data does not contain a unit root. Thus, the order of differencing is already determined by the ADF test. ARIMA models assume that the time series remains stable over time. If structural breaks or parameter shifts are detected through the Bai-Perron test (Bai & Perron, 1998), the model will be separately estimated for different regimes. However, this will only done if there is a sufficient amount of data.

The Bai-Perron test is implemented in R by using the strucchange package (Zeileis, Kleiber, Krämer & Hornik, 2003). Firstly, a supremum F test is applied to examine the presence of a break in the mean of each index by using a constant mean model. Then, the Bai-Perron breakpoint analysis is performed on the indices by again using a constant mean model. The optimal number and timing of structural breaks are determined by minimizing the Bayesian Information Criterion. The test is configured to guarantee a minimum segment size of 10% of the original data (roughly 50 observations) and allows multiple breaks. However, a segment of less than 100 observations is deemed impractical for the training procedure discussed in Section 3.7. To select the appropriate order of p, d, and q, the model configuration with the lowest MSE is chosen. Model estimation is done using the Powell estimation method to avoid convergence issues.

3.3 LSTM

Long Short-Term Memory (LSTM) is a type of neural network that can capture long-term dependencies (Hochreiter & Schmidhuber, 1997). LSTM addresses the limitations of traditional neural networks by providing short-term memory. In particular, the short-term memory helps in preventing the gradients from becoming very small during training, which stops the network from learning long-term dependencies effectively (Bengio, Simard & Frasconi, 1994). LSTM consists of memory cells that store information over several time intervals. Then, three types of gates control the flow of information within a memory cell: the forget gate, input gate, and output gate.

The forget gate determines the information of the previous cell (c_{t-1}) that should be forgotten or maintained by using a function that outputs values between 0 (completely forgotten) and 1 (completely maintained), which are stored in a forget vector f_t. The input gate identifies new information from the current input (x_t) and previous internal memory (h_{t-1}) that should be added to the current memory cell. Then, to update the cell (c_t), the input gate utilizes two parts that work together. The first part determines the updated information ($\tilde{c}_t$). The second part processes new information (i_t) that should be added to the current memory. Lastly, the output gate identifies the information from the current cell (h_t) that should be passed on to the next step.

The complete process of LSTM in mathematical formulation is denoted as:

$$
\begin{aligned}
f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f), \\
i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i), \\
\tilde{c}_t &= \tanh(W_c \cdot [h_{t-1}, x_t] + b_c), \\
c_t &= f_t \odot c_{t-1} + i_t \odot \tilde{c}_t, \\
o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o), \\
h_t &= o_t \odot \tanh(c_t),
\end{aligned}
\tag{4}
$$

where:

- W_f, W_i, W_c, W_o are the weight matrices for the forget gate, input gate, cell input, and output gate, respectively.
- b_f, b_i, b_c, b_o are the bias vectors for the forget gate, input gate, cell input, and output gate, respectively. These learnable parameters allow for additional control to add, forget, or maintain certain parts of information.
- σ is an activation function.
- tanh is a tangent activation function.
- o_t is the output gate's activation vector.

The default LSTM architecture consists of three types of layers. Firstly, the input layer that receives the input data. Secondly, the hidden layer can comprise multiple layers, which contain the memory cells. The number of hidden layers determines the depth of LSTM's network. However, by adding more hidden layers the risk of overfitting and the computation time increase. As a result, it is considered important to find an architecture that balances model performance and computational efficiency. Lastly, the output layer outputs the variable of interest.

So, due to both short-term fluctuations and long-term trends playing crucial roles in the stock market, LSTM can efficiently predict stock prices thanks to its architecture. In specific, the forget gate allows LSTM to forget irrelevant past information. Then, the input gate incorporates new important sentiment and stock market data. Lastly, the output gate enables LSTM to focus on the most relevant features for the predictions. This architecture grants LSTM the ability to adapt to the changing market dynamics and to remember long-term trends.

3.4 TSF

As mentioned in Section 1, LLMs possess the ability to predict time series. However, these LLMs are mostly paid services, which do not fit the feasibility frame. Therefore to obtain a feasible LLM time series forecaster, TSF uses an older LLM framework, that is free-of-use in the Huggingface's open-source transformer library (Wolf et al., 2020). Specifically, the GPT-2 backbone is utilized in TSF. With GPT-2 being the first capable LLM, it demonstrates the potential to predict time series accurately. Also, since predicting the next word boils down to predicting the next numbers in a token sequence, the GPT-2 backbone is deemed well-suited for the task. TSF essentially is a combination of TimesFM, designed by Das et al. (2024), with a GPT-2 backbone instead of expensive transformers to enhance computational feasibility while maintaining the original architecture to minimize knowledge loss. The layers of TSF are elaborated upon in the next sections. These layers can be compared to building blocks that each have their dedicated task to ultimately learn complex patterns for predicting accurately. TSF is illustrated in Figure 7.

It is important to note that TSF is designed to be a general time series forecaster which should be applicable across various domains. This study considers stock indices to evaluate its per-

formance. However, the choice of data does not limit TSF's potential applications. Due to their complexity and availability, stock indices were selected for the initial assessment. TSF's architecture is not specifically designed for the stock market. So, the use of stock market data merely serves as an example to demonstrate TSF's performance relative to other models.

3.4.1 Input Layer

The first building block is the input layer, which preprocesses the multivariate time series data into representations suitable for the next building block, the transformer layers. Firstly, to handle potential distribution shifts, which will complicate the training phase, reversible instance normalization is applied to the time series data (Kim et al., 2022). This helps in stabilizing the model's training across different datasets. In this study's case sudden shifts is indices are taken care off in this way.

The preprocessed data is segmented into fixed-size patches of size p, where p is a hyperparameter that balances the granularity of captured temporal patterns. Smaller patch sizes capture finer details of temporal patterns, whereas larger patch sizes capture higher-level patterns. Basically, a patch is a single unit of training data. Each patch is processed through a residual block. Residual blocks are Neural Network (NN) building blocks containing an MLP (Multi-Layer Perceptron) with a skip connection. An MLP with a skip connection is an NN that passes information by bypassing intermediate layers. So, these facilitate the learning of non-linear transformation while preserving the original information (Das et al., 2023). This block outputs patch embeddings, which capture local temporal patterns within each patch. Afterward, the positional encoding, which provides information about the order of the embeddings, allows the subsequent layer to better understand the relationship between the input (Vaswani et al., 2017).

To enable TSF to handle adaptable sequence length, padding is applied to ensure consistent input dimensions across batches. Padding is needed to guarantee that every patch has the same dimensions due to GPT-2 blocks' architecture needing a fixed-length input sequence. Padding entails appending zeros to the sequence until they reach a certain predetermined length, which is GPT-2's fixed-length input sequence (p) in this case. For the model to identify the padding tokens, a binary padding mask is defined to ensure the model only considers real values. For instance, consider a time series: [100, 102, 98, 105, 103]. The time series must be divided into non-overlapping patches of 3, which means that this sequence must be padded until its length is divisible by 3. So, the padded sequence would become [100, 102, 98, 105, 103, 0], where 0 is an appended value to assure a length that is divisible by 3. Then, the padded sequences are split into two patches: [100, 102, 98] and [105, 103, 0]. Afterward, a binary padding mask assigns a value of 1 to real values and 0 to padded values. For this example, the mask would be [1, 1, 1, 1, 1, 0]. The corresponding padding masks per patch are: [1,1,1] for the first patch and [1,1,0] for the second patch. This way, the mask assures that TSF focuses on real data points while ignoring the added values.

Then, the patches are processed through a residual block, which in this illustrative case adds 0.5 to every value, and are positionally encoded. This transforms the patches into [100.5 + PE1, 102.5 + PE2, 98.5 + PE3] and [105.5 + PE4, 103.5 + PE5, 0.5 + PE6], where PE denotes the positional encoding values. These representations not only allow the model to learn complex patterns within the stock market but also help TSF in distinguishing between different temporal patterns. Subsequently, these numerical representations of the input, are utilized in the next layer that consists of GPT-2 transformer blocks.

3.4.2 GPT-2 Transformer Blocks

The GPT-2 transformer blocks form the backbone of TSF, which uses the GPT-2 model proposed by Radford et al. (2019). GPT-2 is a language model trained on a large amount of text

data. GPT-2 consists of a stack of blocks where each block contains a multi-head self-attention mechanism and a position-wise Feed-forward Neural Network (FNN), which processes information in one direction in a neural network. TSF utilizes the pre-trained weights of these blocks, which define how the model attends to the input, to leverage and update the extensive knowledge GPT-2 has captured.

The multi-head self-attention mechanism within these blocks plays a vital part in ensuring TSF's ability to attend to various segments of the input sequence. The mechanism adjusts attention weights, which quantify the importance of every part of the input sequence for a prediction, for each head. Every head is a separate attention mechanism that tries to capture dependencies in the input sequence. This grants TSF the ability to effectively capture long-range dependencies and relationships between the different features. For instance, one head might focus on long-term dependencies, while another considers weekly stock market trends.

To obtain the weighted values from all attention heads, the mechanism creates Query (Q), Key (K), and Value (V) matrices. These matrices allow TSF to dynamically focus on different aspects of the historical stock and sentiment data. The Q matrix consists of the current queries the model aims to answer, which aids TSF in focusing on time periods that are relevant to the current prediction. In case TSF is too sensitive to Q, short-term fluctuations might be exaggerated, which will lead to volatile predictions. The K matrix comprises the knowledge of the model with different keys to retrieve information. The V matrix contains the actual value the model is looking for by using the key, which could be the actual stock data. If TSF's sensitivity to V is not correct, market trends will not be appropriately accounted for, which will lead to incorrect predictions. A scaled dot-product attention helps manage the sensitivity to the matrices:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{5}$$

where:

- $softmax$ denotes a function that converts real-valued scores to a probability distribution.
- Q, K, and V are the query, key, and value matrices, respectively.
- d_k is the dimension of the keys. The scaling of this aims to prevent the softmax from having small gradients when the input is large.
- T denotes the transpose sign.

In the previously mentioned example, the attention mechanism will allow TSF to focus on the last 2 observations (105 and 103), while also considering the overall trend from earlier. The resulting attention weights are directed as input for the FNN. The FNN consists of two connected layers, that are separated by a non-linear Gaussian Error Linear Unit (GELU) activation function (Hendrycks & Gimpel, 2023). This process allows TSF to capture non-linear transformations and complex patterns. The FNN operates independently across each position and might capture patterns that affect future prices.

TSF uses a technique called residual connection to ensure it learns effectively from all parts of the input data (He, Zhang, Ren & Sun, 2015). The use of residual connections, which are shortcuts in neural networks to prevent information loss, allows information to flow better through the network. Layer normalization is then employed to keep the data within a certain range as it moves through the neural network, which improves convergence (Ba, Kiros & Hinton, 2016).

To ensure that TSF can only attend to current and previous positions in the input, a causal mask is applied to the attention weights. A causal mask blocks out future values of the input sequence to prevent leakage of future information. Then, the causal mask is combined with

the previously mentioned padding mask to completely ignore padded values as well. When considering the first patch of [100, 102, 98], the model can only use information from 100 and 102 to predict the third value. Without the causal mask, TSF would be able to use the third value for its prediction of the third value. The number of transformer blocks depends on the complexity of the data. Therefore, the intent is to balance the computational feasibility and model performance by carefully selecting the number of transformer blocks based on the specific requirements of the time series. This is thoroughly discussed in Section 3.7

3.4.3 Output Layer

The objective of TSF is to forecast the future values in numeric time series data. The output layer maps the representations, that are generated by the GPT-2 blocks, into predictions. It leverages the information captured by the previous layers to generate the forecasted values. TSF outputs a prediction horizon that is of length p. However, in this study, the focus is fixated on one-day ahead forecasts.

The final transformer block outputs a sequence of representations, that captures the patterns and dependencies learned from the input. These representations are fed to a residual block. This is the same type of residual block that is used in Section 3.4.1. However, there is a small distinction in implementation to account for the dimensions. The final predictions are p-ahead predictions for all the input features.

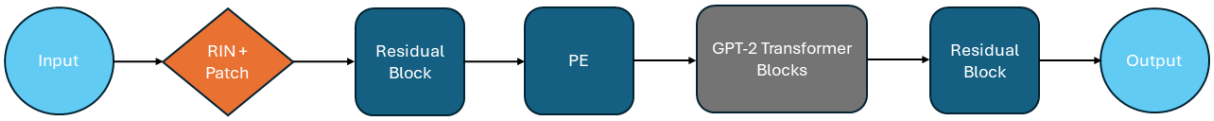

Figure 7: TSF's architecture. The multivariate input undergoes Reversible Instance Normalization (RIN), which keeps the values within a given range, and is divided into patches. Then, these patches are processed through a block that enhances learning. Next, Positional Encoding (PE) adds information about the position of each segment. Afterward, the data goes through a number of GPT-2 transformer blocks. These capture temporal dependencies and patterns. Lastly, the model maps the outputs that are obtained from the transformer blocks to predictions by processing these through another block that enhances learning.

3.4.4 Implicit Few-Shot Learning and Inference

One of the main contributions of this study is the incorporation of few-shot forecast abilities in TSF. This allows for the adaptation of new datasets with only a few labeled examples in training. To enable few-shot learning, the weights of the GPT-2 block are iteratively updated after seeing past data. The few-shot learning utilizes these weights of the GPT-2 as a strong prior, as this provides a solid foundation for the model to quickly adapt to the new dataset with limited data. To be precise, the model starts with the original GPT-2 weights for the multi-head self-attention mechanism, FNN, and layer normalization. Then, TSF generates a prediction for the first data it receives. The loss is calculated between the predicted value and the target value. Afterward, the gradients are backpropagated, which is an algorithm that computes the gradients of the loss function, to update the weights of the mentioned components. This is done for the whole training set and repeated until no improvement in loss is detected.

The weights are adjusted using backpropagation to minimize the loss. Specifically, the AdamW optimizer is used to update the weights during training. This design choice is made based on the fact that TSF overfits easily. AdamW includes weight decay regularization, which penalizes large weights during training (Loshchilov & Hutter, 2018). Therefore, AdamW is the better

choice for an adaptive learning rate optimizer, which automatically adjusts the learning rates during training, that combats overfitting. TSF is trained using the MSE. This metric measures the average squared difference between the predicted and actual numeric time series values. The model's performance is evaluated after the few-shot learning process by assessing its predictive power on a hold-out sample from the dataset.

Leveraging few-shot learning allows TSF to be adapted to a wide range of time series forecasting problems across different domains. Hence this enables an efficient adaptation to the specific requirements of the dataset. TSF generates the forecast in a rolling window manner. Thus, it predicts the future time series values sequentially. In each step, the forecast is generated using the most recent data.

TSF can beat the previously mentioned models as it has several features that the ARIMA, VAR, and LSTM do not possess. Firstly, TSF can handle multivariate data and capture complex non-linear relationships among variables. Thus it surpasses the ARIMA and VAR in that respect. Also, thanks to the attention mechanisms, TSF might be able to effectively capture long-term dependencies, which can be a limitation in the LSTM (Trinh, Dai, Luong & Le, 2018). Lastly, it is important to note that the performance observed with GPT-2 blocks on time series forecasting tasks represents a lower bound for the capabilities of LLMs, as newer LLMs significantly outperform GPT-2 in all types of tasks.

3.5 Sentiment Analysis

To incorporate social media data into the discussed models, the extraction of sentiment scores out of the textual data is crucial. Sentiment analysis is challenging in this study due to the size of the textual data. So, the need for an efficient sentiment analysis model is highlighted. VADER, proposed by Hutto and Gilbert (2014), is therefore leveraged due to its favorable trade-off between efficiency and accuracy.

VADER is a rule-based sentiment analysis model that utilizes heuristic rules and a sentiment lexicon to determine the sentiment of a text. Firstly, the input text is split into individual words and normalized. Each word is subsequently searched in the pre-defined lexicon and its corresponding sentiment value is retrieved. These sentiment values range from -4 (extremely negative) to 4 (extremely positive). Then, these scores are combined with a heuristic rule set to capture the sentiment in different grammatical constructs. The rules consider constructs such as negation, intensifiers, and punctuation. The final sentiment score s is calculated by using a weighted average, where the final score is between -1 (extremely negative) and 1 (extremely positive). Then, to determine the sentiment on a given day, the mean of all the scores on that given day is calculated.

It is important to note that the textual data from non-trading days must also be incorporated into the models. In order to align the non-trading day sentiment scores with the stock data, averaging is employed. Specifically, the sentiment scores of consecutive non-trading days and the first possible trading day are averaged. For instance, a given Saturday, Sunday, and Monday have sentiment scores of 1, 0.5, and 0.8, respectively. The sentiment score of Monday is recalculated to the average sentiment score over the weekend and Monday, which is roughly 0.77. This process is implemented across all indices, as the FTSE 100 (a British index) has trading calendars different from US indices.

3.6 Bias Identification

One of the objectives of this research is to identify bias in the predictions, as potential inaccuracies may influence investor decisions, policy, and the market's stability. However, the term bias has a different meaning in every model. In this section, the choices to identify bias are

discussed. Furthermore, the user's privacy is ensured by the preprocessing procedure discussed in Section 2.2.2. As a result, privacy is not in scope in this section. This section only focuses on the identification of bias, as mitigation is beyond the scope of this study due to the complexities of bias mitigation. These complexities include the trade-offs between fairness and accuracy, as well as defining fairness criteria (Corbett-Davies & Goel, 2018; Caton & Haas, 2020). Therefore, a dedicated study into bias mitigation is required, which is not feasible because of time constraints.

For the LSTM, Shapley values to quantify the contributions of individual input features toward the model's predictions are employed (Štrumbelj & Kononenko, 2013). These Shapley values provide an interpretation of feature importance by assigning values to each feature, based on its contribution to the prediction. This combination offers a powerful approach due to LSTM's ability to capture long-term dependencies and interpretability of feature importance. As a result, the contribution to the predictions across timesteps can be assessed. This leads to enhanced interpretability and transparency in decision-making. If the assessment detects that predictions are based purely on social media sentiment, this suggests that the data should be carefully curated, quantified, and analyzed. Next, bias in VAR and ARIMA refers to systematic mistakes in estimation. So, these biases are identified by examining the assumptions mentioned in the corresponding sections.

For TSF, even though the GPT-2 blocks include attention scores, the computation of these scores does not distinguish between time points of the same variable and time points of different variables. So, the attention scores only grant the ability to quantify the relevance of time points within the context of an individual variable. Consider the following example: GPT-2 blocks can capture the within and cross-dependencies of both past stock prices and sentiment scores to predict future stock prices and sentiment scores. However, the attention scores only quantify the effect of past stock prices on future stock prices. Therefore, this study focuses on a different technique to quantify feature importance.

In order to assess the feature importance, permutation importance is implemented (Breiman, 2001), which is a model-agnostic method. This algorithm performs several permutations of feature values of a given feature. Then, for each permutation, new predictions are made and a loss score is derived. This process is repeated a number of times. Then a mean loss is computed by averaging the loss scores over all the repetitions. Next, the feature score is calculated by subtracting the mean loss score from the baseline score, which is the loss score using the original data. This process is repeated for each feature. Finally, this results in an understanding of the relative contribution of each feature to the performance.

3.7 Hyperparameter Tuning

The hyperparameters are crucial for the model's performance and can significantly impact the predictive accuracy of the models (Hutter, Hoos & Leyton-Brown, 2014). Therefore, the tuning of these hyperparameters appropriately to enhance the predictive ability is essential. An 80% split of data for training and a 20% split for testing is considered. The training split is used to tune the hyperparameters using a 3-fold cross-validation approach. Then, the performance, while using these hyperparameters, is evaluated on the validation split. In this study, the Bayesian Hyperparameter Optimization (BHO) is leveraged to find the optimal combination of hyperparameters for the previously mentioned models (Wu et al., 2019). The steps for the BHO algorithm are given in Algorithm 1, following the same procedure as in Achahboun, Massali, Miron and Israeli (2024).

Algorithm 1: Bayesian Hyperparameter Optimization

Input: Hyperparameter space Θ, score function $s(\cdot)$ and maximum iterations i_{MAX}
Result: Optimal hyperparameters $\boldsymbol{\theta}^{opt}$ and optimal score s^{opt}
Initialize: Sample initial hyperparameters $\boldsymbol{\theta}_0 \in \Theta$ randomly
Evaluate the initial score $s_0 = s(\boldsymbol{\theta}_0)$
Set $\boldsymbol{\theta}^{opt} = \boldsymbol{\theta}_0$, $s^{opt} = s(\boldsymbol{\theta}_0)$ and $\mathcal{D}_0 = \{\boldsymbol{\theta}_0, s_0\}$
for $i = 1$ **to** i_{MAX} **do**
  Fit the probabilistic model $p(s|\boldsymbol{\theta}, \mathcal{D}_{i-1})$ using data $\mathcal{D}_{i-1}$
  Select $\boldsymbol{\theta}_i$ by maximizing the acquisition function a: $\boldsymbol{\theta}_i = \arg\max_{\boldsymbol{\theta}\in\Theta} a(\boldsymbol{\theta}|\mathcal{D}_{i-1})$
  Evaluate s at $\boldsymbol{\theta}_i$ to obtain the updated numeric score $s_i = s(\boldsymbol{\theta}_i)$
  Augment the data: $\mathcal{D}_i = \mathcal{D}_{i-1} \cup \{(\boldsymbol{\theta}_i, s_i)\}$
  if $s_i < s^{opt}$ **then**
    Update the optimal hyperparameters and score: $\boldsymbol{\theta}^{opt} = \boldsymbol{\theta}_i$, $s^{opt} = s_i$
  end
end
return $\boldsymbol{\theta}^{opt}$ *and* s^{opt}

The BHO technique utilizes a Bayesian inference to construct a probabilistic model to approximate an objective function for a certain combination of hyperparameters. In this case, the objective function is the MSE. This optimization of hyperparameters is done by iteratively selecting the next combination of hyperparameters to evaluate, based on a probabilistic model. One of the biggest benefits of using BHO is its computational feasibility due to the process of systematically selecting the next combination of hyperparameters based on certain parts of the hyperparameter combination that already yield good performance. Therefore it considers both untested areas and tested areas that might yield better performance. Consequently, this enables the BHO technique to find better combinations of hyperparameters significantly quicker than a grid search (Bergstra, Bardenet, Bengio & Kégl, 2011; Snoek, Larochelle & Adams, 2012). A traditional grid search explores all the possible predefined combinations of hyperparameters. Whereas, BHO uses a probabilistic approach by using prior information from earlier steps.

The Gaussian Process for the probabilistic model is selected in BHO due to its flexibility and closed-form computations of acquisition functions (Snoek et al., 2012). Therefore, the Expected Improvement (EI) is used for the acquisition function. The EI is a measure that estimates the potential improvement in the MSE for a certain combination of hyperparameters. The EI is defined as:

$$EI(\boldsymbol{\theta}) = \mathbb{E}\left[\max(0, s(\boldsymbol{\theta}) - s(\boldsymbol{\theta}^*))\right], \tag{6}$$

where $s(\boldsymbol{\theta})$ is the objective function, and $s(\boldsymbol{\theta}^*)$ is the best value of MSE found so far. The next combination of hyperparameters to evaluate is determined by selecting the combination that maximizes the EI.

Table 1: Hyperparameter Ranges

VAR	ARIMA	LSTM	TSF
p: $[1, 30]$	p: $[1, 30]$	*num_layers*: $[1, 6]$	*model_dim*: $[8, 128]$
trend: $[n, c]$	q: $[1, 30]$	*hidden_size*: $[1, 512]$	p: $[10, 50]$
	trend: $[n, c, t, ct]$	*dropout*: $[0.1, 0.5]$	*num_layers*: $[2, 20]$
		learning_rate: $[10^{-5}, 10^{-2}]$	*num_heads*: $[1, 8]$
		batch_size: $[32, 64, 128, 256, 512]$	*hidden_dim*: $[100, 1000]$
		num_epochs: $[100, 500]$	*dropout*: $[0.1, 0.5]$
			learning_rate: $[10^{-5}, 10^{-2}]$
			num_epochs: $[50, 100]$
			patience: $[3, 10]$

The hyperparameters and their respective ranges considered for tuning, are given in Table 1. Every index has its own separate combination of hyperparameters. Therefore, this study optimizes the performance of the models based on the considered index. For the VAR model, the lag-order p determines the number of previous timesteps to consider as predictors. The range of 1 to 30 allows the model to capture short-term and long-term dependencies, as well as strike a balance between model complexity and computational feasibility. The *trend* hyperparameter specifies a trend, which can only be no trend (n) or constant term (c) in this study's case.

For the ARIMA, the hyperparameter p has the same function as in the VAR. The order of the moving average term q, defines the size of the moving average window. The same range and logic apply to the ranges of the ARIMA. The *trend* has the same function as in the VAR model but has additional choices of a linear trend (t) and constant and trend (ct).

For the LSTM, the depth of the network is determined by the *num_layers* (number of layers). Next, *hidden_size* is the number of hidden units in each LSTM layer, which determines the capacity of the network. The ranges for these hyperparameters are deliberately small due to an exploratory study showing that large number of hidden states or layers cause overfitting. The dropout probability applied between the layers is determined by *dropout*. This helps in regularizing and prevents overfitting by dropping a fraction of units. The step size at which the model's weights are modified during training is *learning_rate*, which is logarithmically scaled. The *batch_size* is the number of samples utilized in each training batch. Therefore this affects memory usage and training speed. The *num_epochs* is the number of times the model uses the entire training data. Most of these ranges are typically utilized in deep learning. The Adam solver is used as a result of its efficiency and effectiveness in handling large datasets (Kingma & Ba, 2015).

For TSF, the *model_dim* is the dimensionality of the embeddings in the model. Preliminary exploration of TSF suggests that the range of this hyperparameter should be determined carefully due to overfitting risks. Therefore the range is set between 8 and 128 to prevent overfitting. The patch size p balances the granularity of captured temporal patterns, where smaller patches capture fine-grained details. The range between 10 and 50 grants the ability to create between 2 and 10 patches in training, where the training data consists of roughly 100 observations. Next, the number of transformer layers is specified by *num_layers*. The model's ability to attend to different aspects of the input is governed by *num_heads*. The capacity of the residual blocks is determined by *hidden_dim*. As TSF is a new model, the ranges of these parameters are kept broad. These specific ranges accommodate a trade-off between efficiency and computational feasibility, while keeping the overfitting risk in mind. Then, the step size at which the model's weights are modified during training is the variable *learning_rate*. These rates are sampled logarithmically to cover a wide range. The parameter *dropout* determines the proportion of neurons that are dropped during training. Lastly, *patience* monitors TSF's performance on the validation set to stop the process. This range allows for a reasonable amount of epochs to improve the performance. The model rarely exceeds 20 epochs. So the reported epoch range is only to ensure that there are enough epochs for training.

3.8 Evaluation Metrics

To assess and compare the performance of TSF with the benchmark models, four widely used evaluation metrics are employed: MSE, Mean Absolute Percentage Error (MAPE), Theil's U Statistic, and Directional Accuracy (DA).

MSE is a metric that quantifies the average squared difference between the predicted and actual values, defined as:

$$MSE = \frac{1}{T}\sum_{t=1}^{T}(Y_t - \hat{Y}_t)^2, \tag{7}$$

where Y_t is the actual value at time t, $\hat{Y}_t$ is the predicted value of the model, and T is the number of predictions. Due to the differences being squared, MSE penalizes larger errors more severely. It is important to punish inaccuracies heavily due to the subsequent financial implications of wrong decision-making. By using MSE, the overall accuracy of the models can be assessed and the model that minimizes the squared errors is identified.

MAPE is a measure that quantifies the average absolute error as a percentage of the actual values. So, this makes it easy to interpret the magnitude of the error relative to the actual data points. It is calculated as follows:

$$MAPE = \frac{100}{T}\sum_{t=1}^{T}|\frac{Y_t - \hat{Y}_t}{Y_t}|. \tag{8}$$

A lower MAPE indicates better forecasting accuracy due to the model's predictions being closer to the actual values in percentage terms.

Theil's U is a metric that compares the performance of the models to a naive forecast, which in this study is the last observed at time t. This simple and yet effective benchmark can be challenging to beat (Gilliland, Tashman & Sglavo, 2016). Theil's U statistic is defined as:

$$U = \frac{\sqrt{\frac{1}{T}\sum_{t=1}^{T}(Y_t - \hat{Y}_t)^2}}{\sqrt{\frac{1}{T}\sum_{t=1}^{T}(Y_t - Y_{t-1})^2}}, \tag{9}$$

where Y_{t-1} is the last observed value used as the naive prediction for time t. A test statistic less than 1 indicates that the model outperforms the naive forecaster, while a value greater than 1 suggests that the model performs worse than the naive forecaster. By using Theil's U, one can assess whether the proposed models provide better forecasts than a simple baseline.

Lastly, DA is a metric that quantifies the model's ability to correctly predict direction changes in a time series, which helps in constructing different types of portfolios. DA is the proportion of times the model has a correct directional prediction out of the total number of predictions. In case the data is differenced due to non-stationarity, 2 different DA proportions can be computed. One is calculated as previously described, while the other verifies whether the predicted returns are increasing or decreasing compared to the previously predicted returns. Both proportions will be reported, where DA RR indicates the DA of return changes. Consider the following example: at time t and $t+1$ the actual returns are 10 and 12, respectively. The DA RR is correctly identified if the predicted returns have an increasing trend. So, predicted returns of 5 at time t and 7 at $t+1$ result in a correct DA RR. However, predicted returns of 5 at time t and 4 at $t+1$ result in a correct DA but an incorrect DA RR. The DA RR not only provides an additional metric for model comparison but it also aids in assessing potential market trend changes.

The mentioned evaluation metrics will be computed on the test set for each model. By employing these metrics, the aim is to gain an understanding of the ability of TSF and the benchmark models to capture the complex dynamics of stock prices and leverage social media data for forecasting.

4 Results

In this section, a comprehensive analysis of the performances and their implications of the outlined methods are discussed. Firstly, a small study, which focuses on the sentiment analysis and their score is presented. Then, the model assumptions of the VAR and ARIMA are verified, and if needed, the models are modified. Afterward, the performances of the models are compared using the metrics mentioned in Section 3.8. Next, the feature importance of the LSTM and TSF are examined to identify potential biases. These scores also highlights the value of the inclusion of sentiment scores. Lastly, the implications and limitations of TSF are thoroughly discussed.

4.1 Sentiment Analysis

To gain an understanding of the sentiment across the subreddits, the sentiment scores are examined. The scores reveal a variety of sentiments among the subreddits. All of the average sentiment scores are larger than 0. This result indicates that to a certain extent, the subreddits have an overall positive sentiment. While some demonstrate positive sentiment, which may be recognized as bullish, others lean towards a more neutral sentiment, which indicates the tentative nature of these subreddits in terms of investing. The average overall sentiment score is 0.19, which is slightly positive. The average sentiment scores of each subreddit are reported in Table 2. The average score per month for each subreddit can be found in the Appendix.

Subreddit	Average Sentiment Score	Subreddit	Average Sentiment Score
algotrading	0.233	Bogleheads	0.240
Daytrading	0.212	dividends	0.273
ETFs	0.241	ExpatFIRE	0.264
fatFIRE	0.256	financialindependence	0.233
investing	0.176	investing discussion	0.254
leanfire	0.232	options	0.192
pennystocks	0.175	RealEstate	0.169
realestateinvesting	0.210	SecurityAnalysis	0.228
StockMarket	0.139	stocks	0.156
thewallstreet	0.132	ValueInvesting	0.276
wallstreetbets	0.050	WallStreetbetsELITE	0.084
Wallstreetbetsnew	0.067	wallstreetbetsOGs	0.079
Wallstreetsilver	0.171		

Table 2: Average Sentiment Scores of Reddit Subreddits

Subreddits such as algotrading, Bogleheads, dividends, investing_discussion, and ValueInvesting consistently have higher positive sentiment scores. It suggests a stronger sense of optimism within these communities. These subreddits likely have a community that focuses on a more optimistic outlook on the trading market. The positive sentiment in these subreddits could be attributed to their focus on long-term basic strategies. Specifically, algorithmic trading tends to be a less emotional approach to investing, which likely results in positive sentiment. Interestingly, for many subreddits, March 2020 has the lowest recorded sentiment scores, which coincides with the start of the COVID-19 pandemic. This period was marked by significant market volatility and uncertainty. These uncertainties possibly influenced the sentiment of the communities. However, in the following months, the sentiment seems to make a recovery.

Some subreddits, like stocks, StockMarket, and Wallstreetsilver, demonstrate a more neutral sentiment. These subreddits can be regarded as having more nuanced discussions when it comes to the stock market, which contributes to the balanced sentiment. On the other hand, the wallstreetbets subreddits display a more negative sentiment. As the name suggests, these subreddits

view investing more as betting. The communities in these subreddits engage in more speculative and high-risk trades, which could lead to a negative outlook. Furthermore, the sentiment scores are also negatively influenced by the use of profanity and the high-stress nature of the discussed trade strategies. It is important to note that the negative sentiment does not reflect the actual performance of the strategies.

Furthermore, a slight improvement in sentiment scores across most subreddits can be observed in 2021 compared to 2020. This might be due to the overall market recovery after the initial COVID-19 shock. These improved sentiment scores also reflect the growing optimism of investors during this time. In conclusion, the sentiment analysis of several financial-related subreddits offers valuable insights into the outlook of different financial communities. These quantified insights can be significant predictors alongside other sentiment indices, as both might contain information about the future decisions of investors.

4.2 Model Diagnostics VAR and ARIMA

The complete overview of all the tests for the VAR and ARIMA and their results can be found in the Appendix. One of the assumptions that both models have is the use of stationary data in the model. As mentioned before in Section 3.1, this assumption is verified by using the ADF test, which revealed that a unit root is present in all indices. This indicates that the indices are non-stationary. According to the test results, the ICS and the bearish proportion of investors per week also have unit roots, while the bullish proportion of investors does not. Therefore, the bullish proportion of investors appears to be stationary and is a good candidate for the threshold variable of the TVAR. Furthermore, only 7 of the sentiment scores of the subreddits exhibit a unit root. For all of the variables that contain a unit root, first differencing is applied to address potential non-stationarity. After first differencing, the ADF test indicated that all of the variables do not exhibit a unit root.

Both models also assume linearity. However, non-linearity is handled differently in each model due to ARIMA still providing useful approximations, if the non-linearity is not severe enough. For the VAR, nearly every variable is non-linear. Thus, the TVAR is implemented in Python by splitting the data into regimes based on the bullish proportion of investors, which serves as the threshold variable. For each regime, a separate VAR model is fitted. Afterward, the last known observation is used to determine the current regime and the corresponding VAR model is used for the one-day ahead prediction. The tuning range of the threshold variable is between 0.235 and 0.48, as these are the lower and upper bounds of this variable to ensure sufficient data per regime for prediction. TVAR's hyperparameter tuning process is similar to that of VAR, with an extra threshold variable to tune.

The number of regimes for TVAR was determined by analyzing an elbow plot, which is displayed in Figure 8. The plot illustrates the trade-off between the number of regimes and the inertia (total within-cluster sum of squares). The elbow is the point in the graph where adding more regimes does not result in a significant decrease in inertia. In this plot, a bend in the curve between 2 and 3 regimes can be seen. So, both a 2-regime and 3-regime TVAR can be justified. However, the interpretability of these regimes is deemed important. A 2-regime model aligns well with financial market intuition, as the interpretation of a bullish or non-bullish market is straightforward. Therefore, a 2-regime TVAR model is implemented. Lastly, the features do not display multicollinearity.

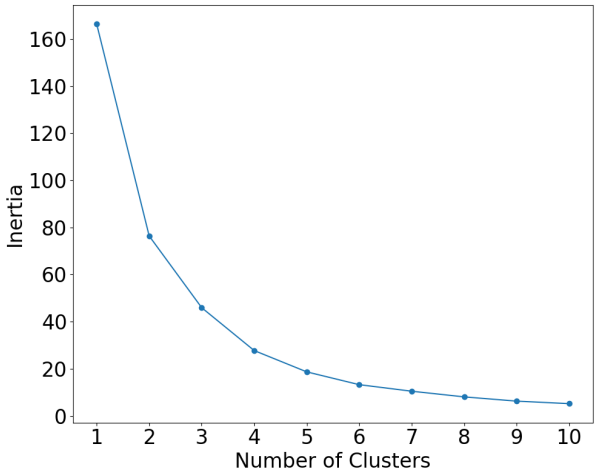

Figure 8: The elbow plot to determine the number of regimes (clusters) in the data. The x-axis denotes the number of regimes and the y-axis represents the inertia, which is the total within-cluster sum of squares. The elbow of the plot suggests an optimal number of clusters. The elbow is the point where the rate of decrease in inertia changes. In this plot, the elbow occurs between 2 and 3 clusters. Due to the focus being interpretability, the number of clusters used is 2.

After following the tuning procedure discussed in Section 3.7, the best possible TVAR and ARIMA models within the predefined ranges, are obtained. Then, the residual of these models are tested by performing the procedure outlined in Section 3.1. Firstly, for the TVAR, the mean of the residuals of several variables are significantly different from zero for the S&P 500, Russell 2000, and FTSE 100 model configurations. This result highlights the model's misspecification due to possibly missing variables or incorrect functional forms. The residuals of every model configuration exhibit heteroskedasticity and/or autocorrelation. Therefore, the Newey-West standard errors are computed and utilized for predictions.

For all ARIMA configurations, the mean residuals are not significantly different from 0, as the null hypothesis of the t-test is not rejected. Furthermore, none of the residuals for the different model configurations exhibit autocorrelation. However, all configurations exhibit heteroskedasticity, which is expected due to the different shocks in returns between 2020 and 2022. Lastly, the Bai-Perron test indicates several structural breaks for the FTSE 100, S&P 500, and Dow Jones in the first 3 months of 2020, which coincides with the COVID-19 pandemic. Despite the results, fitting 2 different ARIMA models in this case is impractical, as the smaller model does not contain sufficient data for this study's training procedure. Hence, the original models will still be employed in the original format.

4.3 Model Evaluation

In this subsection, a comprehensive analysis of TSF's performance across different stock indices is presented. As previously mentioned, the selection of indices represents a diverse range of market dynamics and sectors, which provides a robust testing context. Therefore, the insights gained from this section help in gauging the potential of TSF in different applications. The Theil's U and MSE values are presented in Table 3. These assess each model's ability to outperform a naive forecast. The left table contains the MSE values, where a lower value indicates better predictive accuracy. The right table reports the Theil's U values, which quantify the relative accuracy of the model compared to a naive forecaster. A value below 1 indicates that the model outperforms the naive forecaster.

LSTM cements itself as the top choice for predicting most indices, as can be concluded from Table 3. Specifically, LSTM beats the naive forecaster by roughly 30% for the S&P 500, Russell

Mean Squared Error						Theil's U					
Model	S&P	Russell	Nasdaq	FTSE	Dow	Model	S&P	Russell	Nasdaq	FTSE	Dow
LSTM	1389	820	35351	2668	80048	LSTM	0.689	0.660	0.800	0.638	0.713
TVAR	1812	1239	32995	4818	97632	TVAR	0.787	0.811	0.773	0.858	0.787
ARIMA	1515	834	27292	2872	88259	ARIMA	0.720	0.665	0.703	0.662	0.748
TSF	2086	1044	29674	3412	133827	TSF	0.845	0.745	0.733	0.722	0.922
NAIVE	2925	1883	55213	6547	157561						

Table 3: The Mean Squared Error (left table) and Theil's U (right table) values for each model across the indices. A lower MSE indicates better predictive accuracy. A Theil's U value smaller than 1 suggests that the model beats a naive forecaster, which uses the last observation as its forecast.

2000, FTSE 100, and Dow Jones, which can be seen in the Theil's U values. Furthermore, LSTM is the top-performing model for these indices, where it beats all the other models. This performance highlights LSTM's ability to capture complex temporal dependencies in time series.

As for ARIMA and TVAR, from the MSE and Theil's U values can be concluded that ARIMA beats TVAR in prediction across all indices. This outcome indicates that the detected non-linearity may not be as extreme as initially thought. Furthermore, ARIMA competes with LSTM in terms of predictive performance, where ARIMA is marginally outperformed in the Russell 2000 and FTSE 100. However, ARIMA comfortably beats all considered models in predicting the Nasdaq. Therefore, it highlights the relevance of traditional models in time series forecasting.

Noticeably, TSF's performance shows promising results when considering the Theil's U values. Specifically, the performance for the Russell 2000, Nasdaq, and FTSE 100, indicates promise as TSF beats the naive forecaster by roughly 25% for these indices. Also, TSF is marginally beaten by ARIMA for the Nasdaq. These results are noteworthy given the fact that TSF's architecture leverages older GPT-2 blocks, which suggests potential for improvement.

Lastly, the results reveal the significance of considering market-specific characteristics. In particular, the Theil's U values of the Dow Jones and Nasdaq indicate that all models are struggling to capture the temporal dependencies for these indices as all the values are larger than 0.7. The models struggle less with understanding the dependencies of the Russell 2000 and FTSE 100. This variability highlights the importance of considering different strategies across various markets.

Model	S&P	Russell	Nasdaq	FTSE	Dow
LSTM	3.913	1.127	8.885	1.093	3.453
TVAR	3.407	1.687	5.350	1.886	39.764
ARIMA	1.709	1.215	2.358	1.123	18.005
TSF	9.006	2.880	2.422	1.317	51.903
NAIVE	9.093	3.675	3.942	2.925	11.687

Table 4: The MAPE values for all the models across the indices. MAPE quantifies the average percentage difference between the predicted and actual values. So, a lower MAPE value indicates better predictive accuracy.

Table 4 reports the MAPE for all the different model configurations. To put these scores into perspective, the MAPE scores of the naive forecaster are also reported. MAPE measures the average percentage differences between predicted and actual values. So, a score of 10 indicates that, on average, the predictions deviate 10% from the actual values. The MAPE values also underscore the relevancy of traditional time series forecasters, as ARIMA outperforms all models

for the S%P 500 and Nasdaq. ARIMA's predictions deviate, on average, roughly 1.7% and 2.4% from the actual values, respectively. However, it is important to note that for the Dow Jones, ARIMA does not beat the naive forecaster. So, while ARIMA's MSE and Theil's U values indicate good performance, the MAPE values suggest the opposite. Therefore, this finding highlights the importance of using an appropriate metric for the corresponding task, as different metrics might lead to other outcomes.

As for the LSTM, it is the top-performing model for the FTSE 100 and Russell 2000, with very low MAPE values of around 1%. So, LSTM's predictions are on average, 1% off of the actual values. Furthermore, from the table, it can be concluded that LSTM is the only model that outperforms the naive forecaster for the Dow Jones, which supports the argument of accounting for market-specific characteristics. Another observation is that LSTM is not the best model for predicting the S&P values, according to the MAPE scores. This outcome further strengthens the argument for using appropriate metrics for different tasks. Then, the MAPE values for TVAR indicate respectable performances for the FTSE 100 (1.9), Russell 2000 (1.7), and S&P 500 (3.4). However, the model fails to grasp temporal dependencies for the Nasdaq and Dow Jones, which might be due to these indices exhibiting less severe non-linear returns.

The MAPE values of the TSF display impressive results. Firstly, TSF outperforms the naive forecaster in predicting all indices, except for the Dow Jones. Adding to this, TSF is marginally beaten by ARIMA for the Nasdaq where its prediction, on average, is off by 2.4%. Furthermore, the MAPE value for the FTSE 100 (1.3%) indicates excellent predictive ability. These findings highlight TSF's potential for point prediction, as these scores are outstanding.

Interestingly, the temporal dependencies of the Dow Jones were hard to grasp for the models, which is evident from the high MAPE values. Then, the Russell 2000 and FTSE 100 were easier to forecast for the models, with all models achieving relatively low MAPE values below 3%. These findings further evidence that unique market features should be considered when predicting time series.

Directional Accuracy						Directional Accuracy Change of Returns					
Model	S&P	Russell	Nasdaq	FTSE	Dow	Model	S&P	Russell	Nasdaq	FTSE	Dow
LSTM	0.515	0.495	0.505	0.646	0.475	LSTM	0.490	0.520	0.439	0.622	0.459
TVAR	0.535	0.505	0.444	0.444	0.505	TVAR	0.531	0.602	0.510	0.561	0.602
ARIMA	0.505	0.505	0.495	0.475	0.535	ARIMA	0.571	0.724	0.653	0.520	0.541
TSF	0.525	0.545	0.525	0.444	0.495	TSF	0.367	0.694	0.643	0.276	0.327
NAIVE	0.525	0.465	0.576	0.444	0.525	NAIVE	0.367	0.265	0.337	0.347	0.337

Table 5: This table displays the DA (left table) and DA RR (right table) scores for each model across the indices. The scores indicate the proportion of times a directional movement in the time series is correctly identified. Scores above 0.5 indicate a performance better than random guessing in predicting directional movements, with higher scores suggesting stronger predictive capability.

In Table 5 the DA and DA RR are reported. These provide crucial insights into the models' directional predictive capabilities, which are valuable for trading strategies and risk management in the financial markets. The scores indicate the proportion of correctly predicted directions. So, a score of 0.9 indicates that the model can identify 90% of the direction changes in the time series. A score under 0.5 implies that the model performs worse than random guessing. These results reveal the complexities of forecasting in a stock market context.

When considering the DA for S&P 500, all models beat the random guessing mechanism (0.5). TVAR is the top-performing model in this case, correctly identifying 54% of the direction changes. TSF's performance matches the native forecaster (0.525), which suggests TSF's potential as a directional forecaster, especially if newer transformer blocks are integrated into its

architecture. This claim can be substantiated by TSF's results for the Russell 2000 (0.545), as it outperforms random guessing and every other model. In this case, TSF accurately recognizes 55% of the direction changes. The other models do not or marginally beat the random guessing threshold with a score of 0.505.

As for the Nasdaq, the naive forecaster outperforms all the models with a score of 0.576. This underscores the challenge of forecasting technology-dominated markets, likely due to the volatility and sensitivity to real-time information. This underperformance in directional accuracy does not necessarily indicate a limitation in predicting this sector as the MAPE and Theil's U values are excellent. Next, for the FTSE 100, LSTM demonstrates exceptional results by correctly recognizing roughly 65% of the direction changes. This score significantly beats the other models and random guessing. Together with previous findings that considered the MSE and MAPE, LSTM seems as the best choice for the prediction of the FTSE 100. This performance might be attributed to LSTM's ability to effectively filter and retain long-term information. Lastly, the DA for the Dow Jones suggest that only ARIMA (0.535) narrowly beats the naive forecaster and random guessing. These findings together with the results of the MSE and MAPE, suggest that the considered models struggle with point and directional prediction for the Dow Jones.

The DA RR offers a different perspective of directional prediction due to the focus on changes in return magnitude. This capability is valuable in identifying market reversals. Firstly, ARIMA showcases its impressive abilities, with excellent scores for the Russell 2000 (0.724) and Nasdaq (0.653). The scores indicate ARIMA's capability to correctly identify about 70% of instances where returns increase or decrease the next trading day. These high DA RR scores narrowly beat TSF's scores for the Russell 2000 (0.694) and Nasdaq (0.643), which indicate that ARIMA and TSF can capture short-term changes in market dynamics. This finding demonstrates TSF's potential as an exceptional directional predictor of return changes. These indices are known to be volatile, which suggests that TSF may be specifically suited for capturing complex, non-linear relationships in more dynamic market segments. This claim is reinforced by TSF's performance when considering the MAPE scores.

When considering the DA RR scores for TVAR, the performance for the Dow Jones and Russell 2000 (both 0.602), indicates a respectable ability to correctly identify the magnitude of change in returns. Furthermore, TVAR beats the naive forecaster and random guessing for the S&P 500, Nasdaq, and FTSE 100. Then, LSTM displayed mixed results as it underperforms in predicting the directional change of returns in the S&P 500, Nasdaq, and Dow Jones. Specifically, LSTM's performance for these indices is worse than random guessing. LSTM outperforms all models in the FTSE 100 (0.622), further cementing itself as the top choice for predicting this index.

The analysis of DA and DA RR scores reveals the complexity of index price prediction across different markets. It is concluded that models that excel in predicting the direction of price movements may not necessarily perform well in predicting the direction of return changes, and vice versa. Thus, this underscores the importance of model choice for specific prediction tasks.

In this study, TSF displayed the potential to outperform other models in directional and point prediction. It is important to note that TSF uses GPT-2 blocks, which aren't even available anymore at OpenAI, from an open-source library. Therefore, these results should be considered a lower bound. Newer models, such as the GPT-4 or Claude 3 Opus, are superior to GPT-2. The metrics show that TSF outperformed or is narrowly outperformed by ARIMA and LSTM. The incorporation of newer models might lead to state-of-the-art performance. Therefore, these results might initiate the discussions of leveraging (smaller) LLMs for time series prediction.

Due to time and computational constraints, this study was not able to further robustify these findings by repeating the estimation several times with different model initializations. Ideally, the process would be repeated at least 100 times with different random seeds for model initial-

ization. This process allows for the calculation of a mean, standard deviation, and confidence interval, which offers a better understanding of the model's consistency and reliability. Furthermore, the tuning for TSF is computationally expensive, which limited the search for a better hyperparameter combination. So, there is more room to improve in terms of finding the best-performing model. The exploration of a wider range and a deeper search could lead to a more suitable combination of hyperparameters. Moreover, larger and more advanced models normally consist of many layers and heads, which improves the model's complexity and possibly forecasts (Das et al., 2024). These models are able to capture more intricate patterns and dependencies in the data. Furthermore, the utilization of different open-source transformers can be interesting, as every transformer handles token prediction differently. The exploration of alternative open-source transformers could potentially lead to further improvements.

4.4 Feature Importance

One of the key contributions of this study is the incorporation of social media sentiment data to predict returns. This section focuses on whether the inclusion of sentiment features improves the forecasts and their contribution to the forecasts. It is crucial to quantify the importance of each feature to ensure predictions are not solely based on the textual data. This would mean that the whole process of sentiment analysis should be investigated to identify and mitigate the biases associated with textual data such as selection bias and temporal bias. The bias analysis follows the methodology outlined in Section 4.4. In both models, the bold features in the figures indicate non-subreddit variables to improve readability.

4.4.1 LSTM

The mean absolute Shapley values for the different LSTM configurations are reported in this subsection. The values quantify the average magnitude of a feature's contribution to a prediction. So, a feature with a higher Shapley value has a greater impact on the model's predictions. Figure 9 displays the Shapley values for the Dow Jones model configuration. It is evident that the lagged returns nearly do not contribute to the Dow Jones return predictions. So, the predictions rely heavily on sentiment scores. Therefore, the importance of data collection, preprocessing, and sentiment analysis is highlighted, as these steps significantly influence the sentiment analysis. Specifically, the sentiment from the RealEstate subreddit is the most influential feature for Dow Jones return predictions, followed by dividends and ValueInvesting sentiment scores.

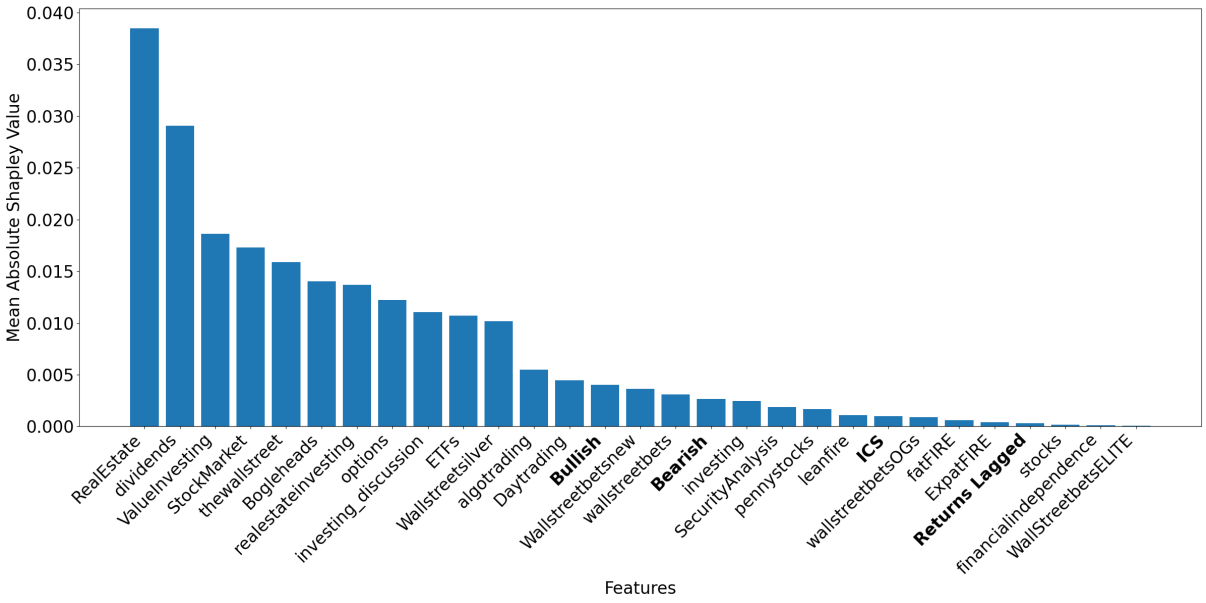

Figure 9: This figure reports the mean absolute Shapley values for features in LSTM's Dow Jones model configuration. Higher values indicate greater feature importance. Interestingly, lagged returns do not seem to contribute to the predictions as much as the sentiment scores.

Similarly, Figure 10 shows that the FTSE 100 return predictions are primarily driven by sentiment scores from the ValueInvesting, investing_discussion, and wallstreet subreddits. Again, the return lags contribute comparatively low to the predictions. The feature importance is more evenly distributed for the FTSE 100 compared to the Dow Jones. Interestingly, the ICS and investor (Bearish and Bullish in the figure) sentiment scores have low contributions to the predictions. The low contributions may be due to the discrepancy in sampling frequencies, as they are monthly and weekly sampling frequencies, respectively. Therefore, the model may not find a link between the ICS and investor sentiment scores and the fluctuations of the daily stock market data.

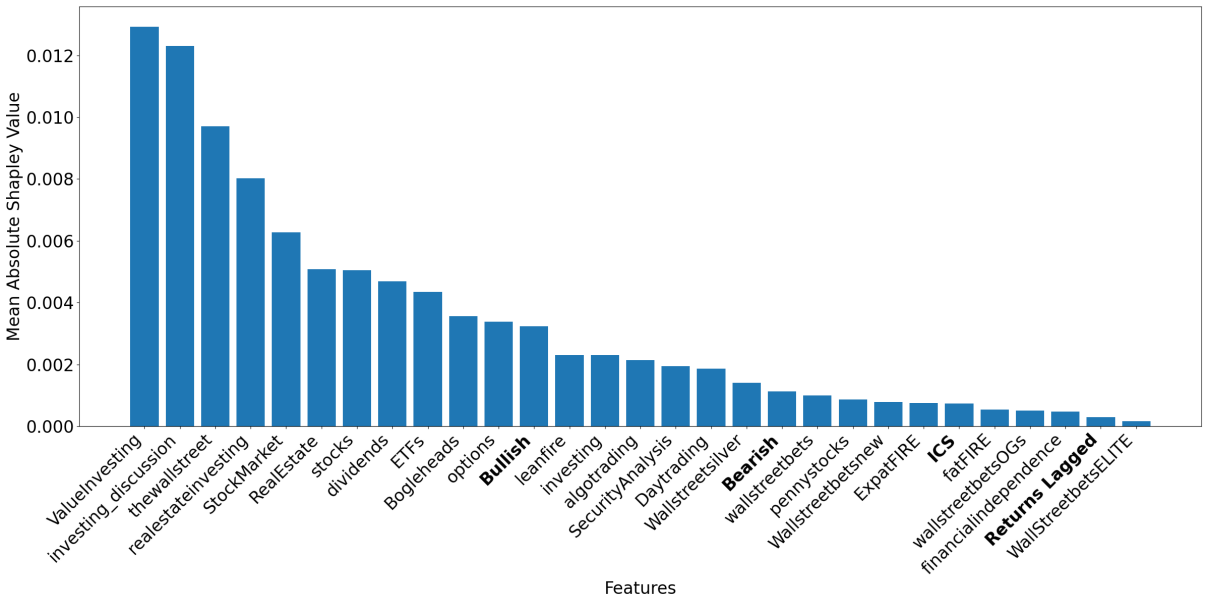

Figure 10: This figure reports the mean absolute Shapley values for features in LSTM's FTSE 100 model configuration. The Shapley values are more evenly distributed in feature importance compared to the Dow Jones. Again, the sentiment scores of ValueInvesting are relatively important to the model's predictions, whereas the lagged returns do not contribute as much.

For the S&P 500, the lagged returns have the lowest importance scores according to Figure 11. So, the lagged returns influence the predictions the least. Again, the RealEstate and ValueInvesting subreddits are the most influential features of the model's predictions. This repeated finding suggests that the sentiment regarding long-term alternative investments serves as a potent feature for prediction forecasts, even more than the lagged returns.

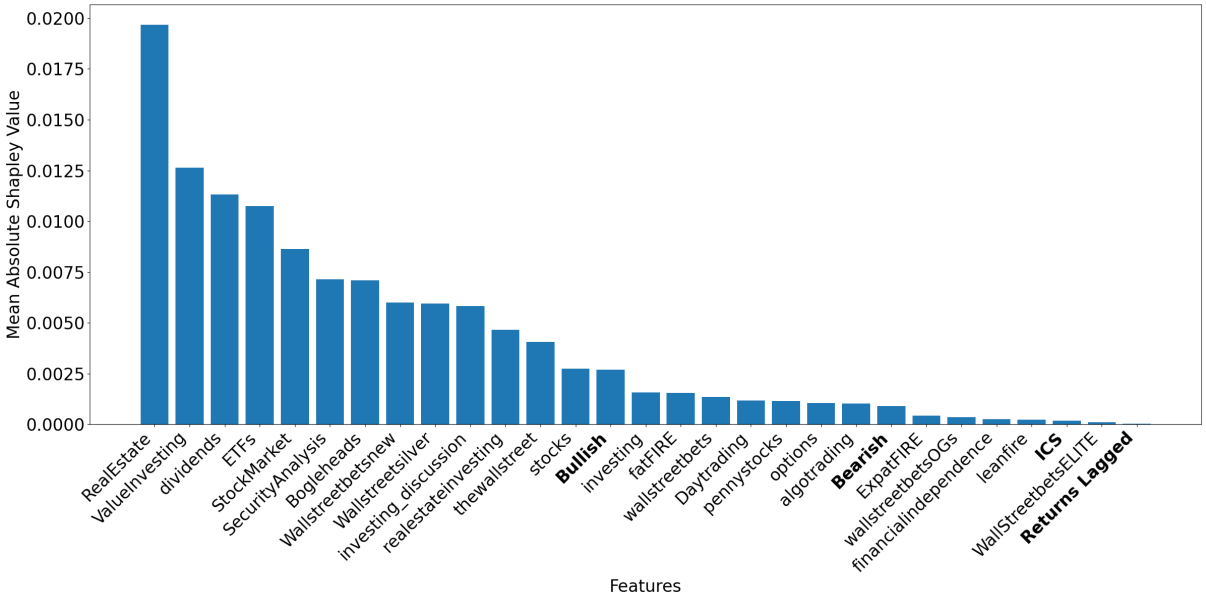

Figure 11: This figure reports the mean absolute Shapley values for features in LSTM's S&P 500 model configuration. The Shapley values show that the lagged returns contribute the least to the predictions. Again, the sentiment scores of RealEstate and ValueInvesting are the most important features to the model's predictions.

In the Nasdaq model configuration, the return lags are relatively important to the model, as

can be seen from the Shapley values displayed in Figure 12. This coincides with LSTM's worst relative performance across the indices, which might indicate that the lagged returns may not be very informative to the predictions. The Wallstreetbetsnew subreddit sentiment score is the most valuable predictor for Nasdaq returns by a margin.

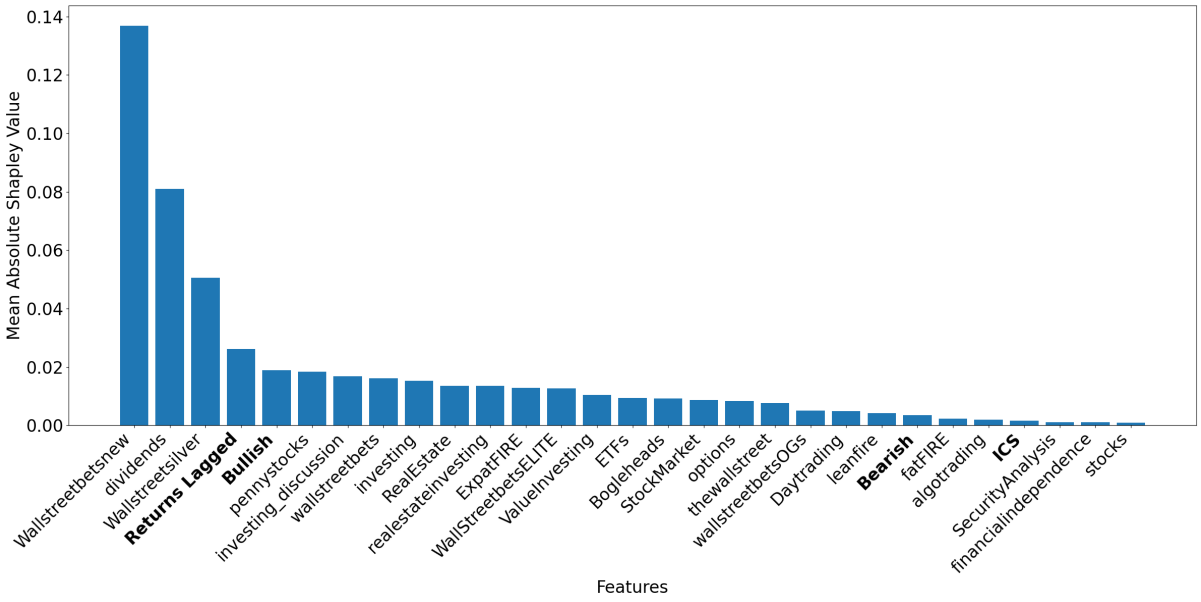

Figure 12: This figure reports the mean absolute Shapley values for features in LSTM's Nasdaq model configuration. The Shapley values show that the lagged returns are relatively important to the predictions. However, the sentiment scores of Wallstreetbetsnew is by far the most important to the model's predictions.

Lastly, for the Russell 2000 index, the sentiment score of the RealEstate subreddit has the highest feature importance, as can be concluded from Figure 13. Also, many of the sentiment scores repeatedly do not contribute significantly to the model's predictions, as well as the lagged returns. This repeated finding raises the question whether all variables should be included in the model.

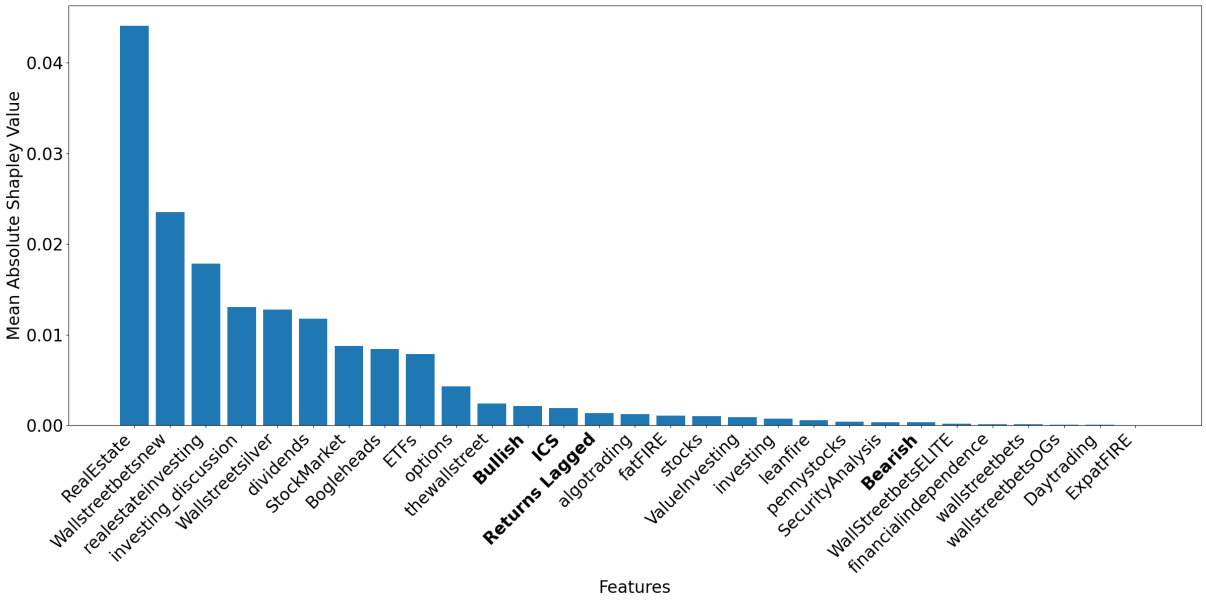

Figure 13: This figure reports the mean absolute Shapley values for features in LSTM's Russell 2000 model configuration. The Shapley values again show that the lagged returns are relatively unimportant to the predictions. Also, the sentiment scores of RealEstate is the most important to the model's predictions.

Interestingly, many models rely heavily on the sentiment scores of subreddits, especially the RealEstate subreddit, to make predictions. This finding suggests a strong link between real estate sentiment and market dynamics. The Shapley values across the model configurations indicate the importance of the sentiment scores in predicting the returns, which suggests that the sentiments from textual data have predictive potency. However, as earlier stated, the use of social media data risks the introduction of bias. Therefore, a comprehensive approach to address potential biases when using social media data, is necessary.

In specific, the sentiment analysis process involves 2 crucial steps that impact the LSTM predictions. First, the data preprocessing relies on predefined open-source libraries. As there is no single optimal preprocessing model, it is crucial to possess domain knowledge in preprocessing to avoid introducing bias. Secondly, the choice of preferred sentiment model is important. While this study employs VADER, other well-performing libraries like BERT exist. Social media data does significantly impact LSTM's predictions and therefore adequate steps to identify biases must be taken.

4.4.2 TSF

In this section, the permutation feature importance scores for TSF are examined. Permutation importance quantifies the feature importance by calculating the difference in performance after randomly shuffling the feature's values in the test set. The shuffling breaks the relationship between a feature and the returns. These scores can be interpreted as follows: a negative score indicates that the feature is important to the model and improves predictions, as shuffling its values increases the MSE compared to the baseline. On the other hand, a positive score suggests that permuting the feature values improves model performance. This means that the feature is detrimental to the model and should be removed or transformed. Furthermore, to gauge the impact, both the absolute and relative changes in MSE are reported. The relative change, which is expressed as a percentage, is calculated by using the MSE score of each index. The permutation importance helps in addressing biases by quantifying the influence of the sentiment scores on TSF's predictions.

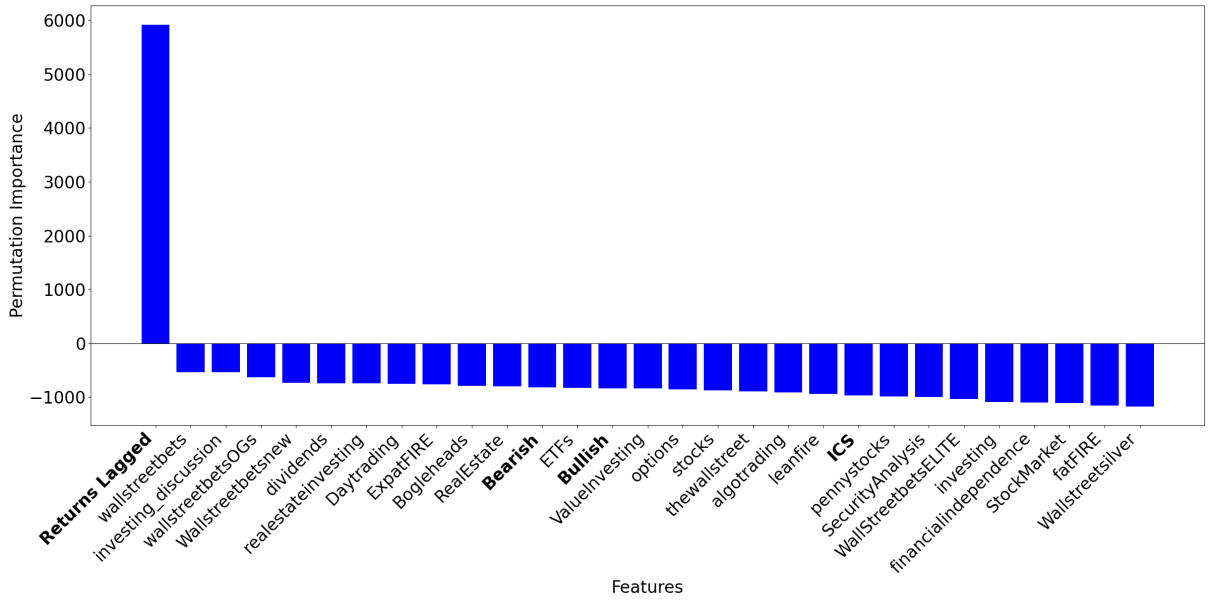

Figure 14: This figure reports the permutation importance of features in TSF's Dow Jones model configuration. The y-axis denotes the absolute value of the difference in MSE. Interestingly, the only feature that has a negative impact on the model is the lagged returns.

Figure 14 reports the permutation importance values for the Dow Jones model configuration. All features except the lagged returns positively impact model performance. The shuffling of the lagged returns decreases the MSE by approximately 6,000 units, which represents a 4.4% improvement compared to the baseline. So, if the current lagged returns were shuffled, TSF's model for the Dow Jones would be better by roughly 4.4%. This noteworthy improvement indicates a large detrimental effect of the lagged returns on the predictions, which raises questions about the model's ability to leverage historical returns. Also, the sentiment scores do not influence the model's prediction as much as the lagged returns.

Figure 15 reveals a more extreme case for the FTSE 100. In this case, the shuffled lagged returns improve the MSE by roughly 21% (approximately 700 units). This is a significant discrepancy that can not be ignored and requires further investigation into the model architecture and data. Furthermore, Figure 14 and Figure 15 illustrate that sentiment scores contribute positively to predictions, though these contributions are minimal. Specifically, the most influential sentiment score increases the MSE by 3% (roughly 100 points).

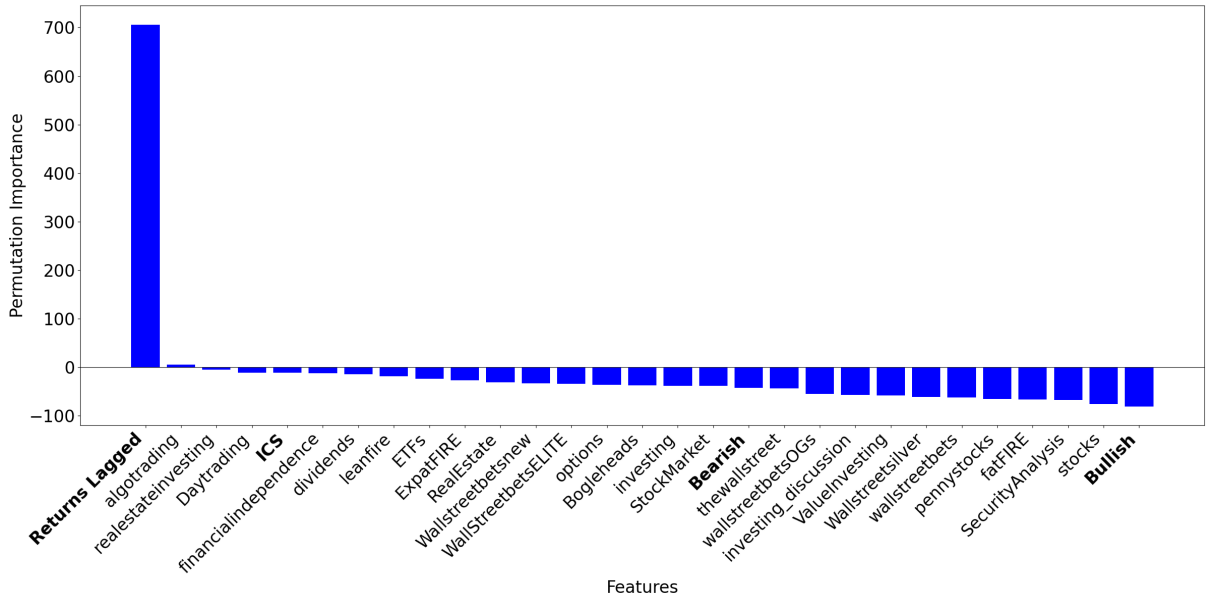

Figure 15: This figure reports the permutation importance of features in TSF's FTSE 100 model configuration. Also in this case, the lagged returns have by far the largest (negative) impact on the model. Furthermore, most of the sentiment scores have a positive effect on the predictions. However, these are overshadowed by the influence of the lagged returns.

The Nasdaq model configuration similarly shows that shuffling the lagged returns leads to a 10% improvement in MSE (around 700 points). Moreover, Figure 16 shows that the influence the sentiment scores have on the predictions, is negligible, with the most impactful sentiment score improving the model by roughly 0.7%. So, also in this case, the predictions are minimally influenced by the sentiment scores, while the lagged returns deteriorate the model.

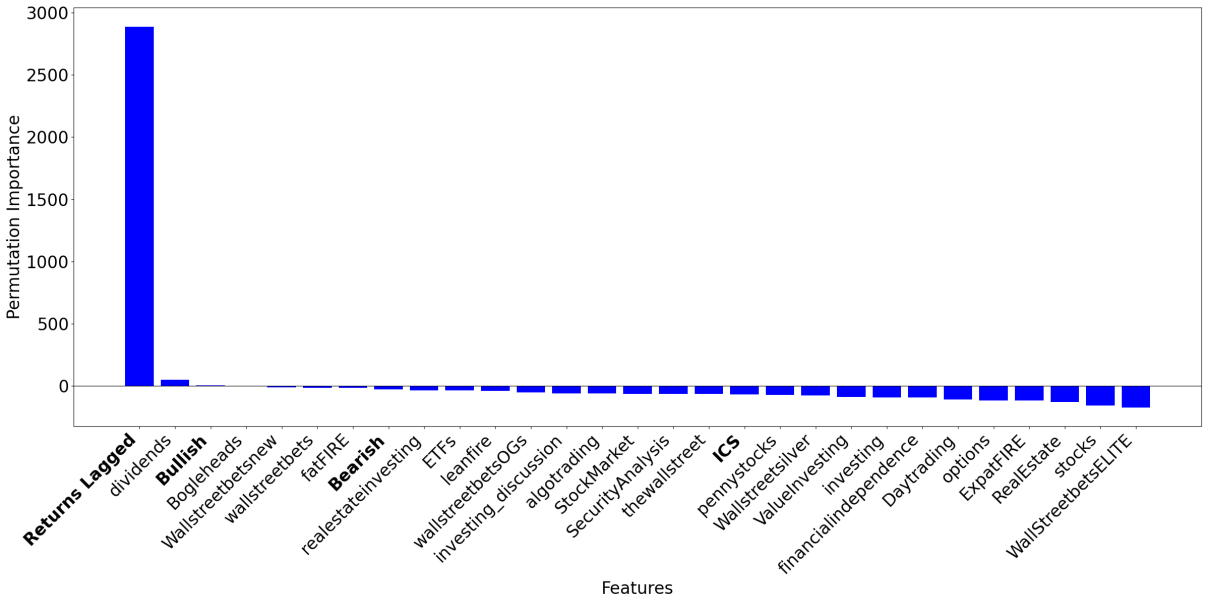

Figure 16: This figure reports the permutation importance of features in TSF's Nasdaq model configuration. Again, the lagged returns are the most influential to the TSF's predictions, while shuffling in the sentiment scores results in negligible changes.

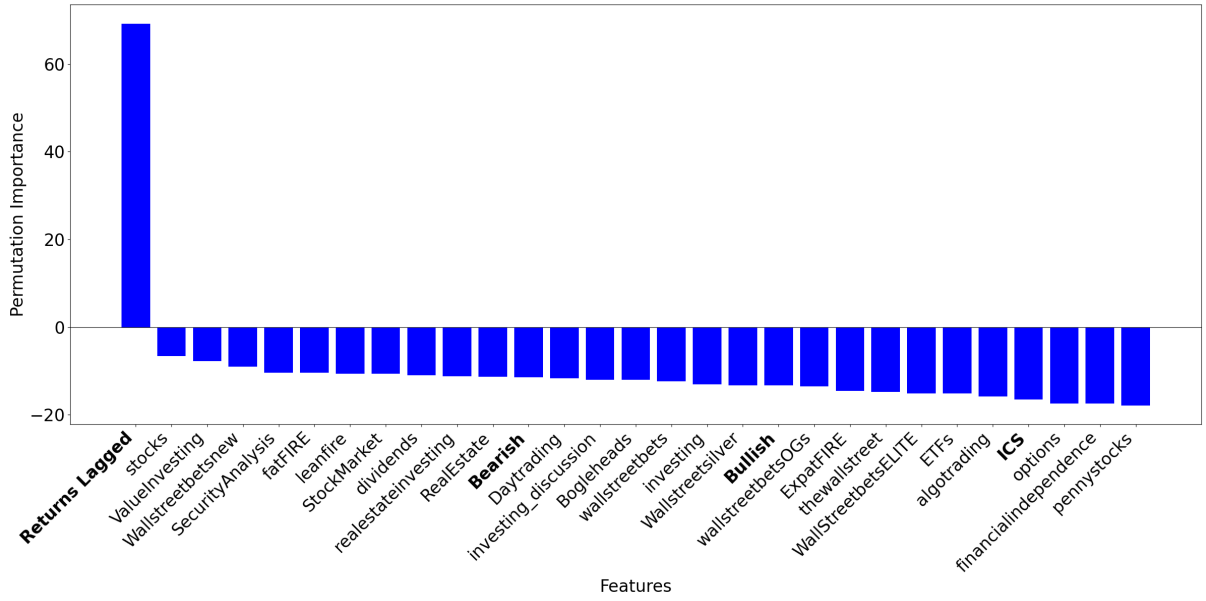

Figure 17: This figure reports the permutation importance of features in TSF's S&P 500 model configuration. Again, the only feature that has a negative impact on the model is the lagged returns. Also, the influence of the variables on the predictions are less pronounced than in other model configurations.

In Figure 17 the feature importance of the S&P 500 model configuration is reported. In the figure, it can be seen that the pattern is consistent with other indices. However, the effects of the lagged returns are less pronounced than before. The MSE change is relatively small, which is around 3.3% for the lagged returns (roughly 70 points). All sentiment scores contribute positively to the model's performance, which shows the value of textual data in forecasting. Lastly, Figure 18 shows similar observations for the Russell 2000 model configuration. The sentiment scores have a relatively larger impact on predictions compared to other indices, which is possibly due to Russell 2000's scale. The StockMarket subreddit sentiment score has nearly

as strong of an effect on the predictions as the lagged returns, with both changing the MSE by roughly 5% (approximately 50 points). Therefore, the sentiment scores can significantly impact the predictions of the TSF.

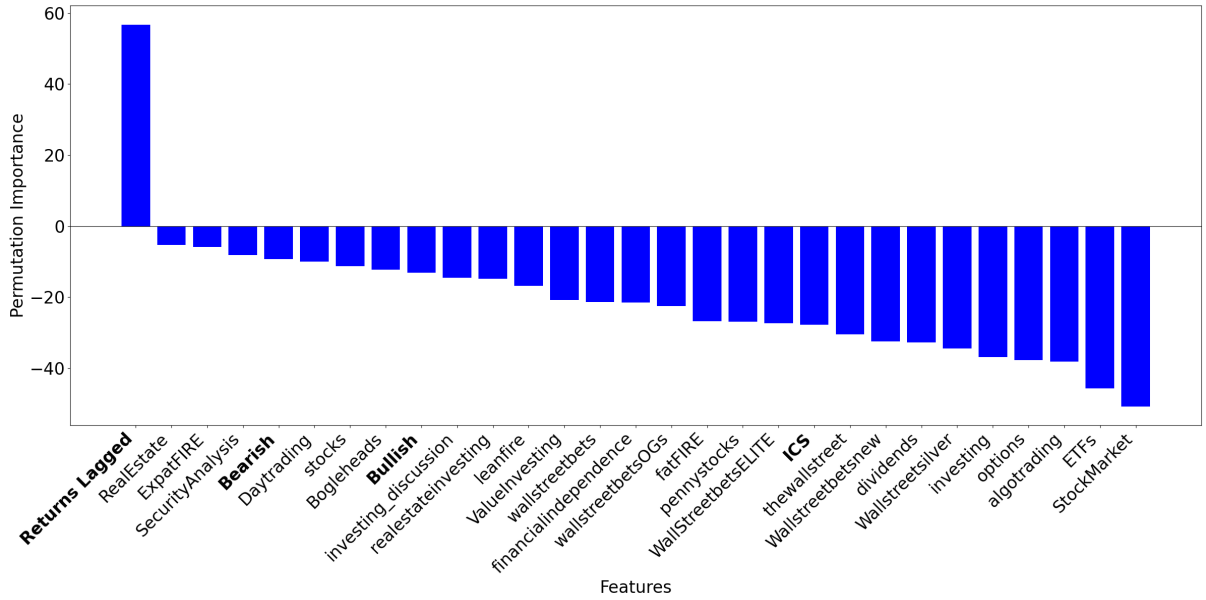

Figure 18: This figure reports the permutation importance of features in TSF's Russell 2000 model configuration. Again, the only feature that has a negative impact on the model is the lagged returns. However, the sentiment scores are as influential as the lagged returns in contributing to the predictions.

This feature importance analysis warrants a further investigation into TSF's prediction methods. It is unexpected that after shuffling the return lags, the predictions improve. This suggests that the lagged returns actually introduce noise into the model. This suggestion is supported by the feature importance analysis of the LSTM, as lagged returns were relatively unimportant in many model configurations. The scale of these noises varies for each model, with the largest observed differences in the Nasdaq and Dow Jones. It is hypothesized that TSF may be overfitting on the lagged returns and is learning the noise patterns in the training data. By shuffling the data its reliance on that pattern is diminished. Alternatively, the self-attention mechanism may play a significant role in assigning higher importance weights to the lagged returns, even if they are not the most informative. Therefore, further research into the attention weights to accurately pinpoint its significance is needed. However, the figures demonstrate that subreddit sentiment scores affect TSF's predictions positively, which emphasizes the importance of textual data analysis for the use of time series forecasting. Also, the bias identification methods show that a thorough analysis of the sentiment quantifying process is required to address biases in the LSTM and TSF when using textual social media data. Lastly, the feature importance scores in both the LSTM and TSF indicate that not all features contribute significantly to the estimation process. The consistent importance of the RealEstate and ValueInvesting sentiment scores across different indices suggests potential multicollinearity among these features. This redundancy could lead to an overestimation of their importance. A more rigorous feature selection for the subreddits and sentiment variables might yield better performances. Future research should address these issues for a more robust forecasting model.

4.5 Implications and Limitations

The development and evaluation of TSF has implications for different audiences, particularly the financial industry, policymakers, and investors. For the financial industry, TSF is a potent tool

for improving stock market predictions. The model's ability to capture temporal dependencies and incorporate sentiment analysis from textual data can provide financial institutions with a competitive edge in making informed investment decisions. By combining a transformer-based architecture with textual data, the financial industry can improve its risk assessment, portfolio management, and trading strategies.

Policymakers are able to also benefit from TSF due to the fact that accurate stock market predictions are crucial for maintaining financial stability and designing effective economic policies. This knowledge can inform regulatory decisions, help prevent market anomalies, and contribute to the overall stability of the financial system by the incorporation of investor sentiment to make predictions. This highlights the importance of monitoring and analyzing public sentiment and its impact on financial markets. For investors, TSF is a valuable tool to make informed investment decisions. The TSF's performance in point forecasting and directional forecasting can assist investors in identifying potential investment opportunities and strategies.

The implications of the results also reach the field of time series forecasting. The success of incorporating sentiment analysis and feasible transformer-based architectures in stock market prediction opens up new avenues for further research. Firstly, researchers can explore the application of TSF in forecasting different financial variables, such as exchange rates, commodities, and cryptocurrency. Additionally, TSF can also be adapted and utilized in different fields where time series forecasting is crucial, like energy demand forecasting.

However, it is essential to acknowledge the limitations and challenges associated with TSF and their implications. One significant limitation is the potential overfitting of the model to the lagged returns, as observed in the feature analysis. Hence, this highlights the necessity of careful model selection, regularization techniques, and robust evaluation to mitigate overfitting. Further, the robustification of the results will provide a better indication of the model's performances, as time constraints did not allow for a repeated experiment.

Another limitation is the bias that is introduced by the sentiment scores that are derived from social media data as can be concluded from Section 4.4. The sentiment expressed on social media platforms may not always reflect the actual market sentiment. This sentiment can be influenced by various factors such as demographics, opinions, and market conditions. Therefore, it is crucial to develop methods to mitigate biases in sentiment analysis to ensure fair and unbiased predictions.

Lastly, the scalability and computational feasibility of the TSF should be taken into account. Training transformer-based models is computationally intensive. This is specifically the case in this study, where the model deals with large datasets and repeated training for different indices. Future research should focus on optimizing the model's architecture. Particularly, it should explore efficient tuning ranges and incorporate the latest transformer models to ensure the practicality of deploying TSF in real-world scenarios.

In conclusion, TSF presents promising implications for not only a new approach to time series forecasting but also for the financial world. The ability of TSF to capture temporal dependencies, incorporate sentiment analysis, and provide valuable insights into the market benefits this world. However, it is important to address the limitations and challenges to unlock the full potential of transformer-based forecasters. By addressing these limitations, a more accurate and reliable time series forecaster can be designed, which leads to a better informed financial ecosystem.

5 Conclusions

The central aim of this study was to investigate the efficiency of leveraging artificial intelligence, specifically transformer-based models, to predict stock market trends by utilizing consumer and investor sentiment indices as well as sentiment analysis from social media data. The study proposed a novel time series forecaster, TSF, that combines the TimesFM architecture with a GPT-2 backbone, which enhances the computational feasibility while maintaining good predictive performance.

The key contributions of this research are fourfold. Firstly, it addressed the computational and financial feasibility of time series forecasters by incorporating an open-source LLM into TimesFM's architecture. Secondly, TimesFM and LLMtime are extended by exploring few-shot learning, which enhances the practicality of the models as they are trained for their application. Thirdly, it bridged the gap between textual social media data and time series forecasting in transformer-based models by incorporating sentiment scores derived from unstructured Reddit comments and posts into the TSF. Lastly, the importance of bias identification within these models is addressed by employing permutation importance and XAI techniques.

The results demonstrate the potential of transformer-based models as a competitive time series forecaster. Its performance is comparable to traditional models like TVAR, ARIMA, and LSTM in certain cases, where sometimes it even surpasses these models. The incorporation of sentiment scores from social media data significantly contributed to the predictive performance of the models. This highlights the value of leveraging textual data for stock market forecasting.

However, potential limitations and challenges are also identified. These are the overfitting of TSF on lagged returns, the biases introduced by sentiment scores derived from social media data, and the use of outdated GPT-2 transformer blocks. These findings underscore the importance of careful model selection, design, and robust evaluation to mitigate overfitting to ensure unbiased predictions.

The implications of this research go beyond the financial industry, policymakers, and investors. The development of reliable computationally feasible time series forecasters can help in making informed investment decisions. Furthermore, it can assist in designing effective economic policies and maintaining financial stability. Also, the study might initiate a discussion on applying transformer-based models to various financial variables and adapting TSF to other domains where time series forecasting is crucial.

To conclude, this study demonstrates the potential of combining transformer-based models with sentiment analysis from social media data for stock market prediction. The insights gained from TSF, contribute to the advancement of time series forecasting and its applications in the financial world, even with its limitations. Further research should be aimed at the identified limitations, the exploration of alternative architectures, and expansion to different markets.

References

Achahboun, N., Massali, Z., Miron, V. & Israeli, H. (2024). *Seminar in business analytics and quantitative marketing research team 07 Coolblue Energy: Predicting churn with a smile.* Erasmus School of Economics, Erasmus University Rotterdam. (Seminar paper)

Adadi, A. & Berrada, M. (2018). Peeking inside the black-box: A survey on explainable artificial intelligence (XAI). *IEEE Access*, *6*, 52138–52160.

Ba, J. L., Kiros, J. R. & Hinton, G. E. (2016). Layer normalization. In *NIPS 2016 Deep Learning Symposium* (p. 14).

Bai, J. & Perron, P. (1998). Estimating and testing linear models with multiple structural changes. *Econometrica*, *66*, 47–78.

Baumgartner, J., Zannettou, S., Keegan, B., Squire, M. & Blackburn, J. (2020). The pushshift Reddit dataset. In *Proceedings of the International AAAI Conference on Web and Social Media* (pp. 754–762).

Bengio, Y., Simard, P. & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, *5*, 157–166.

Bergstra, J., Bardenet, R., Bengio, Y. & Kégl, B. (2011). Algorithms for hyper-parameter optimization. In *Advances in Neural Information Processing Systems* (pp. 2546–2554).

Bird, S., Loper, E. & Klein, E. (2009). *Natural language processing with Python* (1st ed.). Sebastopol, CA: O'Reilly Media, Inc.

Box, G. E. P. & Jenkins, G. M. (1970). *Time series analysis: Forecasting and control* (1st ed.). San Francisco, CA: Holden-Day.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*, 5–32.

Breusch, T. S. & Pagan, A. R. (1979). A simple test for heteroscedasticity and random coefficient variation. *Econometrica*, *47*, 1287–1294.

Brock, W. A., Dechert, W. D., Scheinkman, J. A. & LeBaron, B. (1996). A test for independence based on the correlation dimension. *Econometric reviews*, *15*, 197–235.

Caton, S. & Haas, C. (2020). Fairness in machine learning: A survey. *ACM Computing Surveys*, *56*, 1–38.

Corbett-Davies, S. & Goel, S. (2018). The measure and mismeasure of fairness: A critical review of fair machine learning. *The Journal of Machine Learning Research*, *24*, 14730–14846.

Das, A., Kong, W., Leach, A., Mathur, S. K., Sen, R. & Yu, R. (2023). Long-term forecasting with TiDE: Time-series dense encoder. *Transactions on Machine Learning Research*, 21.

Das, A., Kong, W., Sen, R. & Zhou, Y. (2024). *A decoder-only foundation model for time-series forecasting.* (To be presented at the International Conference on Machine Learning, July 2024)

Dickey, D. A. & Fuller, W. A. (1979). Distribution of the estimators for autoregressive time series with a unit root. *Journal of the American statistical association*, *74*, 427–431.

Fisher, K. L. & Statman, M. (2000). Investor sentiment and stock returns. *Financial Analysts Journal*, *56*, 16–23.

Gallegos, I. O., Rossi, R. A., Barrow, J., Tanjim, M. M., Kim, S., Dernoncourt, F., ... Ahmed, N. (2023). Bias and fairness in large language models: A survey. *Computational Linguistics*, 79.

Gilliland, M., Tashman, L. & Sglavo, U. (2016). *Business forecasting: Practical problems and solutions* (1st ed.). Hoboken, NJ: Wiley.

Gruver, N., Finzi, M., Qiu, S. & Wilson, A. G. (2023). Large language models are zero-shot time series forecasters. In *Advances in Neural Information Processing Systems* (pp. 19622–19635).

Han, Z., Zhao, J., Leung, H., Ma, K. & Wang, W. (2019). A review of deep learning models for time series prediction. *IEEE Sensors Journal*, *21*, 7833–7848.

Hartigan, J. A. & Wong, M. A. (1979). Algorithm AS 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, *28*, 100–108.

He, K., Zhang, X., Ren, S. & Sun, J. (2015). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition* (p. 770-778).

Hendrycks, D. & Gimpel, K. (2023). *Gaussian error linear units (GELUs).* (Preprint)

Hobijn, B., Miles, R. A., Royal, J. F. & Zhang, J. (2022). *What is driving U.S. inflation amid a global inflation surge?* (Chicago Fed Letter No. 468). Federal Reserve Bank of Chicago. Retrieved from https://www.chicagofed.org/publications/chicago-fed-letter/2022/470

Hochreiter, S. & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*, 1735–1780.

Hutter, F., Hoos, H. & Leyton-Brown, K. (2014). An efficient approach for assessing hyper-parameter importance. In *Proceedings of the 31st International Conference on Machine Learning* (pp. 754–762).

Hutto, C. J. & Gilbert, E. (2014). VADER: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the International AAAI Conference on Web and Social Media* (pp. 216–225).

Hyndman, R. J. & Khandakar, Y. (2008). Automatic time series forecasting: The forecast package for R. *Journal of Statistical Software*, *27*, 1–22.

Ibrahim, L., Mesinovic, M., Yang, K.-W. & Eid, M. A. (2020). Explainable prediction of acute myocardial infarction using machine learning and Shapley values. *IEEE Access*, *8*, 210410–210423.

Ji, X., Wang, J. & Yan, Z. (2021). A stock price prediction method based on deep learning technology. *International Journal of Crowd Science*, *5*, 55–72.

Kanas, A. (2005). Nonlinearity in the stock price-dividend relation. *Journal of International Money and Finance*, *24*, 583–606.

Kellstedt, P. M., Linn, S. & Hannah, A. (2015). The usefulness of consumer sentiment: Assessing construct and measurement. *Public Opinion Quarterly*, *79*, 181–203.

Khan, W., Ghazanfar, M., Azam, M. A., Karami, A., Alyoubi, K. & Alfakeeh, A. (2020). Stock market prediction using machine learning classifiers and social media, news. *Journal of Ambient Intelligence and Humanized Computing*, *13*, 3433–3456.

Kheiri, K. & Karimi, H. (2023). *SentimentGPT: Exploiting GPT for advanced sentiment analysis and its departure from current machine learning.* Retrieved from https://arxiv.org/abs/2307.10234

Kim, T., Kim, J., Tae, Y., Park, C., Choi, J.-H. & Choo, J. (2022). Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations* (p. 25).

Kingma, D. P. & Ba, J. (2015). Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations* (p. 15).

Li, X., Xie, H., Chen, L., Wang, J. & Deng, X. (2014). News impact on stock price return via sentiment analysis. *Knowledge-Based Systems*, *69*, 14–23.

Li, Y., Zhu, Z., Kong, D., Han, H. & Zhao, Y. (2018). EA-LSTM: Evolutionary attention-based LSTM for time series prediction. *Knowledge-Based Systems*, *181*, 104785.

Ljung, G. M. & Box, G. E. (1978). Measure of lack of fit in time series models. *Biometrika*, *65*, 297–303.

Loshchilov, I. & Hutter, F. (2018). *Fixing weight decay regularization in Adam.* Retrieved from https://openreview.net/forum?id=rk6qdGgCZ

Lovell, M. C. (2001). The predictive power of the index of consumer sentiment. *Brookings Papers on Economic Activity*, *2001*, 175–207.

Lütkepohl, H. (2005). *New introduction to multiple time series analysis* (1st ed.). Berlin, Germany: Springer Berlin.

McKenzie, E. (1984). General exponential smoothing and the equivalent ARMA process. *Journal of Forecasting*, *3*, 333–344.

Nabipour, M., Nayyeri, P., Jabani, H., Mosavi, A., Salwana, E. & S., S. (2020). Deep learning for stock market prediction. *Entropy*, *22*, 840.

Newey, W. K. & West, K. D. (1987). A simple, positive semi-definite, heteroskedasticity and autocorrelation consistent covariance matrix. *Econometrica*, *55*, 703–708.

Nguyen, T. H. & Shirai, K. (2015). Topic modeling based sentiment analysis on social media for stock market prediction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing* (pp. 1354–1364).

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D. & Sutskever, I. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, *1*, 9.

Sadik, Z. A., Date, P. & Mitra, G. (2019). News augmented GARCH(1,1) model for volatility prediction. *IMA Journal of Management Mathematics*, *30*, 165–185.

Sampson, T. (2017). Brexit: The economics of international disintegration. *Journal of Economic Perspectives*, *31*, 163–184.

Schick, T. & Schütze, H. (2021). It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (pp. 2339–2352).

Siami-Namini, S., Tavakoli, N. & Siami Namin, A. (2018). A comparison of ARIMA and LSTM in forecasting time series. In *17th IEEE International Conference on Machine Learning and Applications* (pp. 1394–1401).

Sims, C. A. (1980). Macroeconomics and reality. *Econometrica*, *48*, 1–48.

Skipper Seabold & Josef Perktold. (2010). Statsmodels: Econometric and statistical modeling with Python. In *Proceedings of the 9th Python in Science Conference* (pp. 92–96).

Snoek, J., Larochelle, H. & Adams, R. P. (2012). Practical Bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems* (pp. 2960–2968).

Stine, R. (2019). Sentiment analysis. *Annual Review of Statistics and Its Application*, *6*, 287–308.

Trinh, T., Dai, A., Luong, T. & Le, Q. (2018). Learning longer-term dependencies in RNNs with auxiliary losses. In *Proceedings of the 35th International Conference on Machine Learning* (pp. 4965–4974).

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems* (pp. 5998–6008).

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., ... SciPy 1.0 Contributors (2020). SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nature Methods*, *17*, 261–272.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., ... Rush, A. M. (2020). Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations* (pp. 38–45).

Wu, J., Chen, X.-Y., Zhang, H., Xiong, L.-D., Lei, H. & Deng, S.-H. (2019). Hyperparameter optimization for machine learning models based on Bayesian optimization. *Journal of Electronic Science and Technology*, *17*, 26–40.

Yin, W., Rajani, N. F., Radev, D., Socher, R. & Xiong, C. (2020). Universal natural language processing with limited annotations: Try few-shot textual entailment as a start. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing* (pp. 8229–8239).

Zeileis, A., Kleiber, C., Krämer, W. & Hornik, K. (2003). Testing and dating of structural changes in practice. *Computational Statistics & Data Analysis*, *44*, 109–123.

Zhou, T., Niu, P., Wang, X., Sun, L. & Jin, R. (2023). One fits all: Power general time series analysis by pretrained LM. In *Advances in Neural Information Processing Systems* (pp. 43322–43355).

Zivot, E. & Wang, J. (2003). Vector autoregressive models for multivariate time series. In *Modeling Financial Time Series with S-Plus* (pp. 369–413). New York, NY: Springer New York.

Zou, J. & Petrosian, O. (2020). Explainable AI: Using Shapley value to explain complex anomaly detection ML-based systems. *Machine Learning and Artificial Intelligence*, *332*, 152–164.

Štrumbelj, E. & Kononenko, I. (2013). Explaining prediction models and individual predictions with feature contributions. *Knowledge and Information Systems*, *41*, 647–665.

Appendix

A.1 TVAR and ARIMA Tests Results

A.1.1 Brock-Dechert-Scheinkman Test Results

Column	Test Statistic	P-value
Close S&P 500	8.358	0.000
Close Russell 2000	5.048	0.000
Close Nasdaq	5.089	0.000
Close FTSE 100	4.541	0.000
Close Dow Jones	9.268	0.000
ICS	-0.789	0.430
Bullish	75.458	0.000
Bearish	-2.223	0.026
algotrading	0.294	0.769
Bogleheads	4.648	0.000
Daytrading	2.846	0.004
dividends	6.947	0.000
ETFs	5.060	0.000
ExpatFIRE	9.850	0.000
fatFIRE	1.919	0.055
financialindependence	13.822	0.000
investing	7.601	0.000
investing_discussion	-0.622	0.534
leanfire	3.601	0.000
options	8.854	0.000
pennystocks	8.789	0.000
RealEstate	8.277	0.000
realestateinvesting	6.457	0.000
SecurityAnalysis	3.812	0.000
StockMarket	19.471	0.000
stocks	23.364	0.000
thewallstreet	0.936	0.349
ValueInvesting	6.397	0.000
wallstreetbets	2.672	0.008
WallStreetbetsELITE	12.609	0.000
Wallstreetbetsnew	0.608	0.543
wallstreetbetsOGs	6.828	0.000
Wallstreetsilver	14.720	0.000

Table 6: This table presents the Brock-Dechert-Scheinkman test results. The test is conducted by using the standard settings of the corresponding function in the statsmodels and SciPy Python libraries and a 5% significance level. The null hypothesis states that the considered time series is linear and independently and identically distributed. The alternative hypothesis states that the considered time series is not linear. The two columns report the test statistic and p-value of the test.

A.1.2 Augmented Dickey-Fuller Test Results

Column	Test Statistic	P-value	Differenced Test Statistic	Differenced P-value
Close S&P 500	-0.235	0.934	-6.494	0.000
Close Russell 2000	-0.760	0.830	-7.174	0.000
Close Nasdaq	-0.659	0.857	-6.939	0.000
Close FTSE 100	-1.947	0.310	-7.740	0.000
Close Dow Jones	-0.720	0.841	-6.691	0.000
ICS	-2.424	0.135	-22.426	0.000
Bullish	-3.125	0.025	-12.424	0.000
Bearish	-2.223	0.198		
algotrading	-19.321	0.000		
Bogleheads	-8.293	0.000		
Daytrading	-17.037	0.000		
dividends	-5.239	0.000		
ETFs	-6.729	0.000		
ExpatFIRE	-1.655	0.455	-9.902	0.000
fatFIRE	-5.744	0.000		
financialindependence	-4.544	0.000		
investing	-2.523	0.110	-6.278	0.000
investing_discussion	-7.771	0.000		
leanfire	-17.686	0.000		
options	-6.705	0.000		
pennystocks	-2.712	0.072	-9.184	0.000
RealEstate	-3.528	0.007		
realestateinvesting	-4.352	0.000		
SecurityAnalysis	-20.248	0.000		
StockMarket	-3.269	0.016		
stocks	-2.975	0.037		
thewallstreet	-3.437	0.010		
ValueInvesting	-3.947	0.002		
wallstreetbets	-2.276	0.180	-6.310	0.000
WallStreetbetsELITE	-2.127	0.234	-8.061	0.000
Wallstreetbetsnew	-3.211	0.019		
wallstreetbetsOGs	-2.284	0.177	-10.893	0.000
Wallstreetsilver	-0.579	0.876	-5.245	0.000

Table 7: This table presents the Augmented Dickey-Fuller test results. The test is conducted by using the standard settings of the corresponding function in the statsmodels and SciPy Python libraries and a 5% significance level. The null hypothesis states that the considered time series contains a unit root and is non-stationary. The alternative hypothesis states that the considered time series does not contain a unit root. The first two columns report the test statistic and p-value of the original series. If the test fails to reject the null-hypothesis, the series is differenced. Subsequently, presence of a unit root is checked by conducting the same test again. These test results can be found in the last two columns.

A.1.3 TVAR Residual Tests

For the residual tests of the TVAR, it is important to note that some models do not possess enough observations or variation for the tests to be conducted. A minus sign (-) denotes that the test could not be conducted.

Variable	LB Test Statistic	LB P-value	BP Test Statistic	BP P-value	T Statistic	T P-value
Close	5.306	0.021	12.405	0.003	2.014	0.060
ICS_ALL	4.846	0.028	15.347	0.001	2.617	0.018
Bullish	5.186	0.023	13.689	0.002	2.141	0.047
Bearish	0.146	0.702	-	-	-1.633	0.121
algotrading	5.265	0.022	13.600	0.002	2.113	0.050
Bogleheads	5.094	0.024	13.358	0.002	2.137	0.047
Daytrading	5.204	0.023	13.722	0.002	2.148	0.046
dividends	5.239	0.022	13.934	0.002	2.141	0.047
ETFs	5.232	0.022	13.696	0.002	2.149	0.046
fatFIRE	5.220	0.022	13.527	0.002	2.130	0.048
financialindependence	5.184	0.023	13.756	0.002	2.142	0.047
investing	1.715	0.190	-	-	1.958	0.067
investing_discussion	5.217	0.022	14.244	0.002	2.244	0.038
leanfire	5.128	0.024	13.216	0.002	2.121	0.049
options	5.224	0.022	13.435	0.002	2.116	0.049
pennystocks	6.068	0.014	-	-	-2.297	0.035
RealEstate	5.181	0.023	13.292	0.002	2.110	0.050
realestateinvesting	5.177	0.023	13.543	0.002	2.150	0.046
SecurityAnalysis	5.218	0.022	13.996	0.002	2.119	0.049
StockMarket	5.250	0.022	13.897	0.002	2.174	0.044
stocks	5.202	0.023	13.664	0.002	2.145	0.047
thewallstreet	5.279	0.022	14.141	0.002	2.173	0.044
ValueInvesting	5.235	0.022	13.527	0.002	2.137	0.047
wallstreetbets	4.325	0.038	-	-	2.313	0.033
WallStreetbetsELITE	1.217	0.270	-	-	1.892	0.076

Table 8: This table presents the t-test, Ljung-Box (LB) and Breusch-Pagan (BP) test results for the first TVAR model of the S&P 500 configuration. The tests are conducted using the standard settings of the corresponding functions in the statsmodels and SciPy Python libraries with a 5% significance level. For the Ljung-Box test, the null hypothesis states that the residuals are independently distributed. The alternative hypothesis states that the presence of serial correlation in the residuals. For the Breusch-Pagan test, the null hypothesis states that the residuals are homoscedastic. The alternative hypothesis states that the residuals are heteroscedastic. For the t-test, the null hypothesis states that the mean of the residuals is zero. The alternative hypothesis states that the mean is non-zero. The first two columns report the p-values for the Ljung-Box test, the middle two columns report the p-values for the Breusch-Pagan test, and the last two columns show the p-values for the t-test. A p-value below 0.05 suggests rejection of the null hypothesis for the respective test.

Variable	LB Test Statistic	LB P-value	BP Test Statistic	BP P-value	T Statistic	T P-value
Close	0.123	0.726	34.807	0.000	0.114	0.910
ICS_ALL	0.001	0.976	360.514	0.000	-0.123	0.902
Bullish	0.015	0.904	1.149	0.284	0.103	0.918
Bearish	0.066	0.797	20.512	0.000	-0.077	0.939
algotrading	2.085	0.149	1.394	0.239	0.382	0.703
Bogleheads	0.436	0.509	60.371	0.000	0.288	0.774
Daytrading	0.955	0.329	0.141	0.708	0.293	0.769
dividends	0.828	0.363	0.511	0.475	0.072	0.942
ETFs	0.090	0.764	6.034	0.014	0.184	0.854
ExpatFIRE	11.438	0.001	13.938	0.000	-0.021	0.983
fatFIRE	4.168	0.041	5.624	0.018	0.362	0.718
financialindependence	4.845	0.028	1.567	0.211	0.085	0.933
investing	5.920	0.015	1.863	0.173	0.079	0.937
investing_discussion	0.012	0.911	0.100	0.752	0.237	0.813
leanfire	0.028	0.868	2.117	0.146	0.135	0.892
options	0.584	0.445	16.472	0.000	0.044	0.965
pennystocks	0.751	0.386	1.788	0.182	-0.026	0.979
RealEstate	10.968	0.001	0.820	0.366	0.322	0.748
realestateinvesting	3.007	0.083	0.203	0.653	0.114	0.909
SecurityAnalysis	0.108	0.743	50.999	0.000	0.023	0.982
StockMarket	7.594	0.006	4.311	0.039	-0.079	0.937
stocks	4.329	0.037	0.012	0.914	0.017	0.987
thewallstreet	0.041	0.839	59.943	0.000	0.267	0.789
ValueInvesting	1.011	0.315	15.148	0.000	0.189	0.850
wallstreetbets	3.112	0.078	0.243	0.622	0.028	0.978
WallStreetbetsELITE	10.742	0.001	14.415	0.000	0.017	0.986
Wallstreetbetsnew	2.188	0.139	197.983	0.000	-0.068	0.946
wallstreetbetsOGs	10.044	0.002	611.068	0.000	0.150	0.881
Wallstreetsilver	1.512	0.219	11.596	0.001	0.000	1.000

Table 9: This table presents the t-test, Ljung-Box (LB) and Breusch-Pagan (BP) test results for the second TVAR model of the S&P 500 configuration. The tests are conducted using the standard settings of the corresponding functions in the statsmodels and SciPy Python libraries with a 5% significance level. For the Ljung-Box test, the null hypothesis states that the residuals are independently distributed. The alternative hypothesis states that the presence of serial correlation in the residuals. For the Breusch-Pagan test, the null hypothesis states that the residuals are homoscedastic. The alternative hypothesis that the residuals are heteroscedastic. For the t-test, the null hypothesis states that the mean of the residuals is zero. The alternative hypothesis states that the mean is non-zero. The alternative hypothesis states that the mean is non-zero. The first two columns report the p-values for the Ljung-Box test, the middle two columns report the p-values for the Breusch-Pagan test, and the last two columns show the p-values for the t-test. A p-value below 0.05 suggests rejection of the null hypothesis for the respective test.

Variable	LB Test Statistic	LB P-value	BP Test Statistic	BP P-value	T Statistic	T P-value
Close	0.069	0.793	0.041	0.843	0.277	0.785
ICS_ALL	2.508	0.113	32.507	0.000	1.452	0.165
Bullish	2.330	0.127	24.808	0.000	2.766	0.013
Bearish	1.903	0.168	-	-	-1.874	0.078
algotrading	1.735	0.188	16.317	0.001	2.282	0.036
Bogleheads	2.511	0.113	17.728	0.001	2.182	0.043
Daytrading	2.700	0.100	17.364	0.001	1.826	0.085
dividends	1.866	0.172	18.368	0.001	2.373	0.030
ETFs	2.517	0.113	13.711	0.002	2.022	0.059
fatFIRE	2.323	0.127	16.888	0.001	2.060	0.055
financialindependence	2.348	0.125	15.147	0.001	2.146	0.047
investing	0.580	0.446	-	-	0.826	0.420
investing_discussion	2.916	0.088	21.840	0.000	1.890	0.076
leanfire	2.397	0.122	12.319	0.003	2.256	0.038
options	2.651	0.103	16.344	0.001	2.128	0.048
pennystocks	4.320	0.038	-	-	-1.346	0.196
RealEstate	2.375	0.123	9.957	0.006	1.697	0.108
realestateinvesting	2.225	0.136	18.855	0.001	2.090	0.052
SecurityAnalysis	2.291	0.130	18.916	0.000	1.936	0.070
StockMarket	2.132	0.144	25.039	0.000	2.066	0.054
stocks	2.432	0.119	14.152	0.002	1.927	0.071
thewallstreet	0.759	0.384	0.000	1.000	2.795	0.012
ValueInvesting	0.734	0.392	0.000	1.000	-1.385	0.184
wallstreetbets	3.122	0.077	-	-	-1.137	0.271
WallStreetbetsELITE	0.871	0.351	0.000	1.000	-0.291	0.775

Table 10: This table presents the t-test, Ljung-Box (LB) and Breusch-Pagan (BP) test results for the first TVAR model of the Russell 2000 configuration. The tests are conducted using the standard settings of the corresponding functions in the statsmodels and SciPy Python libraries with a 5% significance level. For the Ljung-Box test, the null hypothesis states that the residuals are independently distributed. The alternative hypothesis states that the presence of serial correlation in the residuals. For the Breusch-Pagan test, the null hypothesis states that the residuals are homoscedastic. The alternative hypothesis that the residuals are heteroscedastic. For the t-test, the null hypothesis states that the mean of the residuals is zero. The alternative hypothesis states that the mean is non-zero. The alternative hypothesis states that the mean is non-zero. The first two columns report the p-values for the Ljung-Box test, the middle two columns report the p-values for the Breusch-Pagan test, and the last two columns show the p-values for the t-test. A p-value below 0.05 suggests rejection of the null hypothesis for the respective test.

Variable	LB Test Statistic	LB P-value	BP Test Statistic	BP P-value	T Statistic	T P-value
Close	0.013	0.911	33.213	0.000	0.067	0.947
ICS_ALL	0.005	0.946	365.999	0.000	-0.124	0.901
Bullish	0.005	0.946	1.671	0.197	0.104	0.917
Bearish	0.090	0.764	20.877	0.000	-0.077	0.939
algotrading	2.279	0.131	0.950	0.330	0.381	0.704
Bogleheads	0.544	0.461	59.960	0.000	0.292	0.771
Daytrading	0.930	0.335	0.170	0.680	0.295	0.768
dividends	0.809	0.368	0.372	0.542	0.079	0.937
ETFs	0.099	0.753	6.222	0.013	0.183	0.855
ExpatFIRE	11.463	0.001	13.499	0.000	-0.023	0.981
fatFIRE	4.504	0.034	5.803	0.016	0.363	0.717
financialindependence	4.831	0.028	1.809	0.179	0.084	0.933
investing	5.982	0.014	1.741	0.188	0.080	0.936
investing_discussion	0.020	0.888	0.146	0.703	0.238	0.812
leanfire	0.005	0.946	2.044	0.154	0.135	0.893
options	0.577	0.448	18.372	0.000	0.049	0.961
pennystocks	0.686	0.408	1.851	0.174	-0.025	0.980
RealEstate	10.917	0.001	1.234	0.267	0.321	0.749
realestateinvesting	3.129	0.077	0.160	0.690	0.113	0.910
SecurityAnalysis	0.112	0.738	51.055	0.000	0.024	0.981
StockMarket	8.423	0.004	3.761	0.053	-0.076	0.939
stocks	4.624	0.032	0.075	0.784	0.020	0.984
thewallstreet	0.042	0.838	57.415	0.000	0.268	0.789
ValueInvesting	0.957	0.328	15.108	0.000	0.189	0.851
wallstreetbets	3.559	0.059	0.032	0.858	0.035	0.972
WallStreetbetsELITE	11.077	0.001	15.170	0.000	0.019	0.985
Wallstreetbetsnew	2.028	0.154	196.813	0.000	-0.066	0.947
wallstreetbetsOGs	10.424	0.001	619.204	0.000	0.149	0.882
Wallstreetsilver	1.496	0.221	11.841	0.001	0.001	0.999

Table 11: This table presents the t-test, Ljung-Box (LB) and Breusch-Pagan (BP) test results for the second TVAR model of the Russell 2000 configuration. The tests are conducted using the standard settings of the corresponding functions in the statsmodels and SciPy Python libraries with a 5% significance level. For the Ljung-Box test, the null hypothesis states that the residuals are independently distributed. The alternative hypothesis states that the presence of serial correlation in the residuals. For the Breusch-Pagan test, the null hypothesis states that the residuals are homoscedastic. The alternative hypothesis that the residuals are heteroscedastic. For the t-test, the null hypothesis states that the mean of the residuals is zero. The alternative hypothesis states that the mean is non-zero. The alternative hypothesis states that the mean is non-zero. The first two columns report the p-values for the Ljung-Box test, the middle two columns report the p-values for the Breusch-Pagan test, and the last two columns show the p-values for the t-test. A p-value below 0.05 suggests rejection of the null hypothesis for the respective test.

Variable	LB Test Statistic	LB P-value	BP Test Statistic	BP P-value	T Statistic	T P-value
Close	0.035	0.851	29.760	0.000	0.104	0.917
ICS_ALL	0.004	0.951	655.246	0.000	-0.124	0.901
Bullish	0.002	0.965	8.327	0.004	0.114	0.909
Bearish	0.063	0.801	21.653	0.000	-0.124	0.901
algotrading	1.251	0.263	0.027	0.870	0.360	0.719
Bogleheads	0.079	0.779	60.568	0.000	0.242	0.809
Daytrading	0.374	0.541	0.025	0.874	0.241	0.810
dividends	0.048	0.827	1.136	0.287	0.035	0.972
ETFs	0.058	0.810	3.020	0.083	0.155	0.877
ExpatFIRE	8.496	0.004	0.291	0.590	-0.034	0.973
fatFIRE	2.471	0.116	3.930	0.048	0.310	0.757
financialindependence	4.944	0.026	6.425	0.012	0.074	0.941
investing	1.969	0.161	0.140	0.709	0.086	0.932
investing_discussion	0.000	0.994	0.123	0.726	0.178	0.859
leanfire	0.089	0.766	3.142	0.077	0.127	0.899
options	1.374	0.241	17.198	0.000	0.096	0.924
pennystocks	0.735	0.391	1.026	0.312	-0.005	0.996
RealEstate	8.079	0.004	1.656	0.199	0.294	0.769
realestateinvesting	2.263	0.132	0.003	0.956	0.078	0.938
SecurityAnalysis	0.111	0.740	57.588	0.000	0.035	0.972
StockMarket	4.983	0.026	0.721	0.397	-0.101	0.919
stocks	3.834	0.050	0.235	0.628	0.027	0.978
thewallstreet	0.028	0.867	32.655	0.000	0.225	0.822
ValueInvesting	0.743	0.389	13.423	0.000	0.147	0.883
wallstreetbets	0.817	0.366	1.027	0.312	0.025	0.980
WallStreetbetsELITE	10.275	0.001	18.378	0.000	0.046	0.963
Wallstreetbetsnew	0.252	0.615	331.900	0.000	-0.030	0.976
wallstreetbetsOGs	7.435	0.006	612.040	0.000	0.128	0.898
Wallstreetsilver	1.163	0.281	22.173	0.000	-0.036	0.971

Table 12: This table presents the t-test, Ljung-Box (LB) and Breusch-Pagan (BP) test results for the first TVAR model of the Nasdaq configuration. The tests are conducted using the standard settings of the corresponding functions in the statsmodels and SciPy Python libraries with a 5% significance level. For the Ljung-Box test, the null hypothesis states that the residuals are independently distributed. The alternative hypothesis states that the presence of serial correlation in the residuals. For the Breusch-Pagan test, the null hypothesis states that the residuals are homoscedastic. The alternative hypothesis that the residuals are heteroscedastic. For the t-test, the null hypothesis states that the mean of the residuals is zero. The alternative hypothesis states that the mean is non-zero. The alternative hypothesis states that the mean is non-zero. The first two columns report the p-values for the Ljung-Box test, the middle two columns report the p-values for the Breusch-Pagan test, and the last two columns show the p-values for the t-test. A p-value below 0.05 suggests rejection of the null hypothesis for the respective test.

Variable	LB Test Statistic	LB P-value	BP Test Statistic	BP P-value	T Statistic	T P-value
Close	0.048	0.827	15.619	0.000	0.037	0.970
ICS_ALL	0.113	0.736	18.688	0.000	0.044	0.965
Bullish	0.063	0.802	4.248	0.043	0.063	0.950
Bearish	0.030	0.862	31.392	0.000	-0.045	0.964
algotrading	2.395	0.122	11.007	0.001	0.040	0.968
Bogleheads	0.029	0.864	0.311	0.579	0.075	0.941
Daytrading	0.127	0.721	0.191	0.664	0.104	0.918
dividends	0.292	0.589	0.000	1.000	0.045	0.965
ETFs	0.643	0.423	1.581	0.213	0.002	0.999
ExpatFIRE	0.003	0.957	3.738	0.057	0.003	0.998
fatFIRE	0.357	0.550	8.105	0.006	0.087	0.931
financialindependence	0.520	0.471	0.098	0.755	0.034	0.973
investing	1.656	0.198	0.213	0.646	0.067	0.947
investing_discussion	1.291	0.256	12.136	0.001	0.050	0.960
leanfire	0.003	0.956	7.281	0.009	0.014	0.989
options	0.251	0.617	0.000	0.989	-0.016	0.988
pennystocks	0.551	0.458	0.351	0.556	-0.038	0.970
RealEstate	3.642	0.056	4.260	0.043	0.061	0.952
realestateinvesting	2.476	0.116	0.118	0.733	0.025	0.980
SecurityAnalysis	0.002	0.963	1.720	0.194	-0.005	0.996
StockMarket	3.436	0.064	1.637	0.205	0.004	0.997
stocks	1.377	0.241	3.725	0.057	0.046	0.963
thewallstreet	1.368	0.242	2.707	0.104	0.044	0.965
ValueInvesting	0.319	0.572	6.649	0.012	0.134	0.894
wallstreetbets	1.012	0.314	0.258	0.613	0.023	0.982
WallStreetbetsELITE	0.016	0.900	17.991	0.000	-0.065	0.949
Wallstreetbetsnew	0.023	0.879	47.244	0.000	-0.048	0.962
wallstreetbetsOGs	0.011	0.916	4.408	0.039	0.026	0.979
Wallstreetsilver	0.064	0.800	0.059	0.809	-0.013	0.990

Table 13: This table presents the t-test, Ljung-Box (LB) and Breusch-Pagan (BP) test results for the second TVAR model of the Nasdaq configuration. The tests are conducted using the standard settings of the corresponding functions in the statsmodels and SciPy Python libraries with a 5% significance level. For the Ljung-Box test, the null hypothesis states that the residuals are independently distributed. The alternative hypothesis states that the presence of serial correlation in the residuals. For the Breusch-Pagan test, the null hypothesis states that the residuals are homoscedastic. The alternative hypothesis that the residuals are heteroscedastic. For the t-test, the null hypothesis states that the mean of the residuals is zero. The alternative hypothesis states that the mean is non-zero. The alternative hypothesis states that the mean is non-zero. The first two columns report the p-values for the Ljung-Box test, the middle two columns report the p-values for the Breusch-Pagan test, and the last two columns show the p-values for the t-test. A p-value below 0.05 suggests rejection of the null hypothesis for the respective test.

Variable	LB Test Statistic	LB P-value	BP Test Statistic	BP P-value	T Statistic	T P-value
Close	0.494	0.482	2.861	0.109	-0.405	0.690
ICS_ALL	0.439	0.508	5.226	0.035	-0.669	0.512
Bullish	0.507	0.476	2.839	0.110	-0.396	0.696
Bearish	0.386	0.534	-	-	0.491	0.629
algotrading	0.507	0.476	2.784	0.114	-0.386	0.704
Bogleheads	0.509	0.476	2.911	0.106	-0.406	0.689
Daytrading	0.508	0.476	2.827	0.111	-0.393	0.699
dividends	0.511	0.475	2.869	0.109	-0.404	0.691
ETFs	0.505	0.477	2.793	0.113	-0.391	0.701
fatFIRE	0.506	0.477	2.882	0.108	-0.402	0.693
financialindependence	0.508	0.476	2.867	0.109	-0.402	0.693
investing	0.312	0.576	4.591	0.047	-0.449	0.659
investing_discussion	0.497	0.481	2.827	0.111	-0.388	0.702
leanfire	0.508	0.476	2.908	0.106	-0.407	0.689
options	0.510	0.475	2.834	0.111	-0.395	0.698
pennystocks	0.028	0.868	-	-	-2.498	0.022
RealEstate	0.508	0.476	2.860	0.109	-0.400	0.694
realestateinvesting	0.508	0.476	2.903	0.107	-0.407	0.689
SecurityAnalysis	0.503	0.478	2.939	0.105	-0.415	0.683
StockMarket	0.507	0.477	2.831	0.111	-0.390	0.701
stocks	0.502	0.479	2.839	0.110	-0.395	0.698
thewallstreet	0.504	0.478	2.886	0.108	-0.405	0.691
ValueInvesting	0.515	0.473	2.774	0.114	-0.383	0.706
wallstreetbets	0.468	0.494	-	-	-0.765	0.454
WallStreetbetsELITE	0.219	0.640	0.000	1.000	2.322	0.032

Table 14: This table presents the t-test, Ljung-Box (LB) and Breusch-Pagan (BP) test results for the first TVAR model of the FTSE 100 configuration. The tests are conducted using the standard settings of the corresponding functions in the statsmodels and SciPy Python libraries with a 5% significance level. For the Ljung-Box test, the null hypothesis states that the residuals are independently distributed. The alternative hypothesis states that the presence of serial correlation in the residuals. For the Breusch-Pagan test, the null hypothesis states that the residuals are homoscedastic. The alternative hypothesis that the residuals are heteroscedastic. For the t-test, the null hypothesis states that the mean of the residuals is zero. The alternative hypothesis states that the mean is non-zero. The alternative hypothesis states that the mean is non-zero. The first two columns report the p-values for the Ljung-Box test, the middle two columns report the p-values for the Breusch-Pagan test, and the last two columns show the p-values for the t-test. A p-value below 0.05 suggests rejection of the null hypothesis for the respective test.

Variable	LB Test Statistic	LB P-value	BP Test Statistic	BP P-value	T Statistic	T P-value
Close	0.004	0.951	22.742	0.000	0.119	0.905
ICS_ALL	0.027	0.870	376.964	0.000	-0.136	0.892
Bullish	0.010	0.922	1.790	0.182	0.084	0.933
Bearish	0.104	0.747	19.112	0.000	-0.067	0.947
algotrading	1.618	0.203	0.344	0.558	0.350	0.727
Bogleheads	0.066	0.797	59.162	0.000	0.269	0.788
Daytrading	0.921	0.337	0.457	0.499	0.271	0.787
dividends	0.885	0.347	0.017	0.895	0.063	0.949
ETFs	0.092	0.762	5.208	0.023	0.200	0.842
ExpatFIRE	11.634	0.001	11.998	0.001	-0.020	0.984
fatFIRE	3.589	0.058	5.213	0.023	0.322	0.748
financialindependence	3.664	0.056	2.464	0.117	0.065	0.948
investing	5.009	0.025	0.141	0.707	0.074	0.941
investing_discussion	0.002	0.968	0.130	0.719	0.262	0.794
leanfire	0.021	0.885	4.783	0.029	0.110	0.912
options	0.194	0.660	20.942	0.000	0.036	0.971
pennystocks	0.285	0.593	1.648	0.200	-0.050	0.960
RealEstate	9.604	0.002	6.457	0.011	0.314	0.754
realestateinvesting	2.023	0.155	0.017	0.897	0.097	0.922
SecurityAnalysis	0.107	0.744	52.797	0.000	0.023	0.981
StockMarket	9.634	0.002	1.164	0.281	-0.057	0.954
stocks	5.364	0.021	0.287	0.592	-0.002	0.998
thewallstreet	0.024	0.878	84.004	0.000	0.256	0.798
ValueInvesting	1.108	0.293	15.829	0.000	0.184	0.854
wallstreetbets	1.807	0.179	0.014	0.907	0.007	0.994
WallStreetbetsELITE	9.506	0.002	35.357	0.000	-0.035	0.972
Wallstreetbetsnew	11.286	0.001	79.502	0.000	0.031	0.976
wallstreetbetsOGs	10.549	0.001	612.424	0.000	0.158	0.874
Wallstreetsilver	1.837	0.175	1.185	0.277	0.020	0.984

Table 15: This table presents the t-test, Ljung-Box (LB) and Breusch-Pagan (BP) test results for the second TVAR model of the FTSE 100 configuration. The tests are conducted using the standard settings of the corresponding functions in the statsmodels and SciPy Python libraries with a 5% significance level. For the Ljung-Box test, the null hypothesis states that the residuals are independently distributed. The alternative hypothesis states that the presence of serial correlation in the residuals. For the Breusch-Pagan test, the null hypothesis states that the residuals are homoscedastic. The alternative hypothesis that the residuals are heteroscedastic. For the t-test, the null hypothesis states that the mean of the residuals is zero. The alternative hypothesis states that the mean is non-zero. The alternative hypothesis states that the mean is non-zero. The first two columns report the p-values for the Ljung-Box test, the middle two columns report the p-values for the Breusch-Pagan test, and the last two columns show the p-values for the t-test. A p-value below 0.05 suggests rejection of the null hypothesis for the respective test.

Variable	LB Test Statistic	LB P-value	BP Test Statistic	BP P-value	T Statistic	T P-value
Close	0.188	0.665	15.652	0.000	0.139	0.890
ICS_ALL	0.003	0.954	636.817	0.000	-0.122	0.903
Bullish	0.000	0.996	7.459	0.007	0.120	0.904
Bearish	0.070	0.791	20.678	0.000	-0.130	0.897
algotrading	1.257	0.262	0.078	0.780	0.355	0.723
Bogleheads	0.092	0.762	60.397	0.000	0.243	0.809
Daytrading	0.580	0.446	0.021	0.884	0.245	0.806
dividends	0.227	0.634	0.557	0.456	0.040	0.968
ETFs	0.106	0.745	2.945	0.087	0.151	0.880
ExpatFIRE	9.427	0.002	0.038	0.845	-0.033	0.974
fatFIRE	2.410	0.121	4.355	0.038	0.309	0.758
financialindependence	4.916	0.027	5.156	0.024	0.080	0.937
investing	2.035	0.154	0.202	0.653	0.088	0.930
investing_discussion	0.001	0.982	0.093	0.761	0.177	0.860
leanfire	0.030	0.862	2.965	0.086	0.131	0.896
options	1.522	0.217	16.012	0.000	0.096	0.923
pennystocks	0.661	0.416	1.008	0.316	-0.001	0.999
RealEstate	7.686	0.006	1.591	0.208	0.290	0.772
realestateinvesting	2.215	0.137	0.004	0.949	0.075	0.940
SecurityAnalysis	0.156	0.693	57.270	0.000	0.036	0.971
StockMarket	5.508	0.019	0.762	0.383	-0.097	0.923
stocks	3.615	0.057	0.313	0.577	0.029	0.977
thewallstreet	0.021	0.884	32.120	0.000	0.226	0.822
ValueInvesting	0.799	0.371	13.436	0.000	0.147	0.883
wallstreetbets	0.638	0.424	1.119	0.291	0.028	0.977
WallStreetbetsELITE	10.532	0.001	17.890	0.000	0.047	0.962
Wallstreetbetsnew	0.229	0.632	335.421	0.000	-0.026	0.979
wallstreetbetsOGs	7.764	0.005	607.783	0.000	0.131	0.896
Wallstreetsilver	1.120	0.290	22.582	0.000	-0.034	0.973

Table 16: This table presents the t-test, Ljung-Box (LB) and Breusch-Pagan (BP) test results for the first TVAR model of the Dow Jones configuration. The tests are conducted using the standard settings of the corresponding functions in the statsmodels and SciPy Python libraries with a 5% significance level. For the Ljung-Box test, the null hypothesis states that the residuals are independently distributed. The alternative hypothesis states that the presence of serial correlation in the residuals. For the Breusch-Pagan test, the null hypothesis states that the residuals are homoscedastic. The alternative hypothesis that the residuals are heteroscedastic. For the t-test, the null hypothesis states that the mean of the residuals is zero. The alternative hypothesis states that the mean is non-zero. The alternative hypothesis states that the mean is non-zero. The first two columns report the p-values for the Ljung-Box test, the middle two columns report the p-values for the Breusch-Pagan test, and the last two columns show the p-values for the t-test. A p-value below 0.05 suggests rejection of the null hypothesis for the respective test.

Variable	LB Test Statistic	LB P-value	BP Test Statistic	BP P-value	T Statistic	T P-value
Close	0.031	0.860	0.848	0.360	0.015	0.988
ICS_ALL	0.966	0.326	4.639	0.035	0.044	0.965
Bullish	0.080	0.778	5.137	0.026	0.066	0.948
Bearish	0.006	0.940	31.586	0.000	-0.045	0.964
algotrading	0.494	0.482	7.287	0.009	0.056	0.956
Bogleheads	0.041	0.840	0.014	0.905	0.079	0.937
Daytrading	0.000	0.986	0.020	0.888	0.110	0.913
dividends	0.531	0.466	0.002	0.967	0.044	0.965
ETFs	0.677	0.411	2.460	0.121	0.001	0.999
ExpatFIRE	0.002	0.964	4.373	0.040	0.000	1.000
fatFIRE	0.425	0.514	9.936	0.002	0.091	0.928
financialindependence	0.471	0.492	0.037	0.848	0.032	0.974
investing	1.736	0.188	0.042	0.839	0.065	0.949
investing_discussion	1.253	0.263	13.049	0.001	0.047	0.963
leanfire	0.001	0.979	6.812	0.011	0.018	0.986
options	0.383	0.536	0.005	0.941	-0.011	0.991
pennystocks	0.552	0.457	0.438	0.510	-0.038	0.970
RealEstate	3.939	0.047	4.531	0.037	0.060	0.952
realestateinvesting	4.029	0.045	1.248	0.268	0.027	0.978
SecurityAnalysis	0.001	0.982	0.886	0.350	-0.008	0.994
StockMarket	3.369	0.066	1.996	0.162	0.008	0.994
stocks	1.304	0.253	4.441	0.039	0.052	0.959
thewallstreet	0.619	0.431	0.170	0.682	0.045	0.964
ValueInvesting	0.534	0.465	6.736	0.011	0.135	0.893
wallstreetbets	1.267	0.260	0.337	0.563	0.022	0.982
WallStreetbetsELITE	0.011	0.916	17.657	0.000	-0.065	0.948
Wallstreetbetsnew	0.002	0.967	50.299	0.000	-0.052	0.958
wallstreetbetsOGs	0.025	0.875	3.105	0.082	0.028	0.978
Wallstreetsilver	0.089	0.766	1.721	0.194	-0.016	0.987

Table 17: This table presents the t-test, Ljung-Box (LB) and Breusch-Pagan (BP) test results for the second TVAR model of the Dow Jones configuration. The tests are conducted using the standard settings of the corresponding functions in the statsmodels and SciPy Python libraries with a 5% significance level. For the Ljung-Box test, the null hypothesis states that the residuals are independently distributed. The alternative hypothesis states that the presence of serial correlation in the residuals. For the Breusch-Pagan test, the null hypothesis states that the residuals are homoscedastic. The alternative hypothesis that the residuals are heteroscedastic. For the t-test, the null hypothesis states that the mean of the residuals is zero. The alternative hypothesis states that the mean is non-zero. The alternative hypothesis states that the mean is non-zero. The first two columns report the p-values for the Ljung-Box test, the middle two columns report the p-values for the Breusch-Pagan test, and the last two columns show the p-values for the t-test. A p-value below 0.05 suggests rejection of the null hypothesis for the respective test.

A.1.4 ARIMA Residuals Tests

Index	LB Test Statistic	LB P-value	BP Test Statistic	BP P-value	T Statistic	T P-value
S&P 500	1.511	0.912	59.588	0.000	-0.163	0.870
Russell 2000	0.148	0.700	56.585	0.000	0.006	0.995
Nasdaq	0.006	0.940	74.241	0.000	0.029	0.977
FTSE 100	0.341	0.997	56.881	0.000	-0.649	0.517
Dow Jones	0.523	0.998	35.452	0.000	0.790	0.430

Table 18: This table presents the t-test, Ljung-Box (LB) and Breusch-Pagan (BP) test results for all ARIMA model configurations. The tests are conducted using the standard settings of the corresponding functions in the statsmodels and SciPy Python libraries with a 5% significance level. For the Ljung-Box test, the null hypothesis states that the residuals are independently distributed. The alternative hypothesis states that the presence of serial correlation in the residuals. For the Breusch-Pagan test, the null hypothesis states that the residuals are homoscedastic. The alternative hypothesis that the residuals are heteroscedastic. For the t-test, the null hypothesis states that the mean of the residuals is zero. The alternative hypothesis states that the mean is non-zero. The alternative hypothesis states that the mean is non-zero. The first two columns report the p-values for the Ljung-Box test, the middle two columns report the p-values for the Breusch-Pagan test, and the last two columns show the p-values for the t-test. A p-value below 0.05 suggests rejection of the null hypothesis for the respective test.

A.2 Subreddits

In this study, the following subreddits are considered: algotrading, Bogleheads, Daytrading, dividends, ETFs, ExpatFIRE, fatFIRE, financialindependence, investing, investing_discussion, leanfire, options, pennystocks, realestateinvesting, RealEstate, SecurityAnalysis, StockMarket, stocks, thewallstreet, ValueInvesting, WallStreetbetsELITE, Wallstreetbetsnew, wallstreetbetsOGs, wallstreetbets, Wallstreetsilver. The average monthly sentiment score per subreddit is reported below in Table 19 and Table 20. The minus sign in the tables denotes that the corresponding subreddit did not exist in that point in time.

	Jan 20	Feb 20	Mar 20	Apr 20	May 20	Jun 20	Jul 20	Aug 20	Sep 20	Oct 20	Nov 20	Dec 20
algotrading	0.240	0.240	0.198	0.217	0.218	0.230	0.228	0.237	0.230	0.238	0.235	0.247
Bogleheads	0.229	0.233	0.206	0.262	0.253	0.256	0.256	0.241	0.254	0.247	0.242	0.266
Daytrading	0.199	0.199	0.188	0.201	0.207	0.223	0.219	0.216	0.212	0.211	0.221	0.209
dividends	0.276	0.302	0.213	0.275	0.251	0.265	0.253	0.255	0.269	0.267	0.284	0.295
ETFs	0.198	0.235	0.211	0.230	0.248	0.271	0.274	0.254	0.253	0.256	0.250	0.259
ExpatFIRE	-	-	-	-	-	-	-	0.167	0.205	0.305	0.193	0.272
fatFIRE	0.245	0.251	0.214	0.255	0.268	0.269	0.266	0.257	0.266	0.256	0.259	0.259
financialindependence	0.227	0.228	0.163	0.208	0.237	0.242	0.240	0.241	0.226	0.233	0.241	0.240
investing	0.176	0.137	0.083	0.145	0.156	0.155	0.174	0.187	0.185	0.199	0.196	0.194
investing_discussion	0.276	0.277	0.213	0.277	0.225	0.249	0.272	0.228	0.260	0.247	0.201	0.247
leanfire	0.215	0.220	0.195	0.197	0.236	0.244	0.249	0.232	0.243	0.241	0.231	0.233
options	0.201	0.187	0.141	0.169	0.194	0.199	0.193	0.199	0.191	0.206	0.222	0.207
pennystocks	0.173	0.173	0.142	0.147	0.156	0.138	0.145	0.143	0.152	0.159	0.158	0.172
RealEstate	0.193	0.181	0.153	0.156	0.181	0.187	0.178	0.177	0.178	0.172	0.170	0.180
realestateinvesting	0.233	0.227	0.164	0.179	0.214	0.223	0.223	0.225	0.212	0.215	0.233	0.224
SecurityAnalysis	0.215	0.220	0.179	0.193	0.210	0.249	0.219	0.255	0.231	0.230	0.229	0.224
StockMarket	0.145	0.149	0.083	0.112	0.138	0.138	0.151	0.163	0.163	0.166	0.193	0.196
stocks	0.161	0.143	0.088	0.129	0.151	0.154	0.156	0.179	0.162	0.175	0.181	0.182
thewallstreet	0.133	0.140	0.111	0.121	0.135	0.147	0.151	0.143	0.138	0.128	0.132	0.137
ValueInvesting	0.274	0.326	0.291	0.294	0.283	0.259	0.301	0.307	0.306	0.275	0.270	0.283
wallstreetbets	0.044	0.038	0.011	0.020	0.034	0.034	0.049	0.054	0.041	0.044	0.054	0.057
WallStreetbetsELITE	-	-	-0.011	0.046	-0.017	0.002	0.028	-0.024	-0.014	0.014	-0.019	0.075
Wallstreetbetsnew	-	-	-0.015	-0.016	-0.014	0.010	-0.015	0.000	-0.027	0.000	0.005	-0.030

Table 19: Average Sentiment Scores of Reddit Subreddits in 2020. The sentiment scores range from negative (-1) to positive (1).

	Jan 21	Feb 21	Mar 21	Apr 21	May 21	Jun 21	Jul 21	Aug 21	Sep 21	Oct 21	Nov 21	Dec 21
algotrading	0.227	0.247	0.241	0.233	0.243	0.241	0.245	0.217	0.226	0.254	0.225	0.238
Bogleheads	0.250	0.245	0.254	0.247	0.260	0.245	0.239	0.229	0.217	0.207	0.209	0.222
Daytrading	0.214	0.230	0.226	0.211	0.212	0.218	0.210	0.200	0.203	0.203	0.234	0.227
dividends	0.291	0.325	0.317	0.300	0.276	0.284	0.297	0.268	0.266	0.255	0.218	0.252
ETFs	0.267	0.252	0.237	0.236	0.226	0.249	0.222	0.226	0.228	0.227	0.224	0.213
ExpatFIRE	0.273	0.278	0.255	0.274	0.260	0.265	0.246	0.261	0.238	0.251	0.244	0.232
fatFIRE	0.252	0.255	0.250	0.249	0.259	0.258	0.267	0.253	0.260	0.252	0.252	0.263
financialindependence	0.246	0.245	0.241	0.249	0.230	0.242	0.246	0.232	0.238	0.234	0.229	0.233
investing	0.191	0.212	0.209	0.216	0.181	0.186	0.175	0.173	0.166	0.165	0.171	0.185
investing_discussion	0.215	0.258	0.266	0.256	0.241	0.254	0.238	0.259	0.308	0.313	0.238	0.259
leanfire	0.259	0.233	0.250	0.244	0.225	0.225	0.227	0.212	0.241	0.231	0.246	0.230
options	0.198	0.200	0.205	0.195	0.183	0.202	0.192	0.185	0.194	0.181	0.187	0.176
pennystocks	0.177	0.212	0.194	0.189	0.199	0.207	0.199	0.201	0.196	0.186	0.197	0.195
RealEstate	0.188	0.184	0.173	0.171	0.163	0.161	0.156	0.149	0.159	0.146	0.146	0.148
realestateinvesting	0.216	0.216	0.216	0.200	0.211	0.203	0.210	0.197	0.201	0.201	0.201	0.204
SecurityAnalysis	0.230	0.262	0.237	0.269	0.232	0.271	0.177	0.197	0.202	0.217	0.215	0.248
StockMarket	0.173	0.148	0.138	0.141	0.129	0.126	0.099	0.131	0.110	0.123	0.112	0.109
stocks	0.175	0.181	0.164	0.169	0.156	0.173	0.148	0.152	0.136	0.154	0.144	0.144
thewallstreet	0.136	0.148	0.126	0.131	0.132	0.135	0.129	0.111	0.115	0.133	0.119	0.127
ValueInvesting	0.284	0.283	0.280	0.267	0.241	0.255	0.227	0.253	0.246	0.251	0.226	0.219
wallstreetbets	0.062	0.069	0.070	0.075	0.066	0.090	0.059	0.059	0.051	0.044	0.048	0.038
WallStreetbetsELITE	0.087	0.077	0.104	0.107	0.111	0.093	0.079	0.080	0.078	0.092	0.103	0.075
Wallstreetbetsnew	-0.020	0.083	0.104	0.094	0.096	0.085	0.067	0.090	0.089	0.071	0.075	0.085
wallstreetbetsOGs	0.023	0.082	0.054	0.054	0.052	0.097	0.081	0.086	0.092	0.092	0.085	0.078
Wallstreetsilver	0.011	0.143	0.178	0.177	0.188	0.197	0.182	0.160	0.147	0.161	0.171	0.175

Table 20: Average Sentiment Scores of Reddit Subreddits in 2021. The sentiment scores range from negative (-1) to positive (1).