Analyzing Retail Sentiment & Detecting Trading Signals

Haozhe Zeng | Cornell University | hz657@cornell.edu

Zixiao Wang | Cornell University | zw699@cornell.edu

https://github.com/howie-zeng/Analyzing-the-Correlation-Between-Retail-Traders--Sentiments-and-Equity-Market-Movements

Abstract

This research investigates the influence of retail trader sentiment on equity market dynamics, leveraging cutting-edge Natural Language Processing (NLP) models, notably BERT, to analyze discourse on social media platforms such as Twitter and Reddit. By defining the parameters of this relationship, the study progresses to develop sophisticated machine learning models that integrate daily stock data with insights derived from sentiment analysis. This approach aims to unearth potential trading signals, tailored to adapt to market volatility. The emphasis lies on harnessing the collective sentiment's predictive power alongside technical market indicators, thereby offering a novel perspective for anticipating stock price movements. Furthermore, the project includes the creation of a user-friendly interface designed to graphically represent the generated buy and sell signals, enhancing interpretability and usability of the findings.

Introduction

The recent years have seen a paradigm shift in the financial landscape, primarily fueled by the digitization of trading platforms. This technological revolution has democratized access to stock trading, empowering a wave of retail traders who were previously overshadowed by institutional investors. Characterized by their swift decision-making and collective actions, these retail traders have emerged as significant influencers in the equity markets.

A quintessential example of this new era of retail trading is the WallStreetBets forum on Reddit. This platform has evolved into a pivotal hub for retail traders, where they exchange insights, strategies, and sentiments about various stocks. The impact of such collective sentiment was strikingly demonstrated in events like the GameStop short squeeze, where coordinated actions, driven by discussions and emotions on the forum, led to extraordinary stock price fluctuations, surprising many institutional investors.

Despite the visibility of events like the GameStop, a comprehensive analysis examining the broader influence of such forums on the equity market remains unexplored. Our research seeks to fill this void by conducting an in-depth analysis of the sentiments expressed on these platforms and their correlation with market movements. Our objective extends beyond identifying superficial correlations; we aim to discern the depth and endurance of sentiment's influence on stock prices.

Furthermore, we investigate whether these collective sentiments can be harnessed as a predictive tool for market trends. To this end, the study will not only provide comprehensive insights into the relationship between retail trader sentiments and market behavior but will also explore the feasibility of developing a time series machine learning model. This model will aim to predict stock returns and prices by integrating sentiment analysis, thereby offering a novel approach to understanding and forecasting market dynamics in the digital age.

Comparison with Related Work

The interplay between retail sentiment and stock market dynamics has been the subject of various studies, with a consensus pointing towards a positive correlation. Concurrently, the application of machine learning in forecasting stock prices has piqued the interest of researchers. Yet, the literature reveals a scarcity in attempts to synthesize sentiment analysis with machine learning for the purpose of predicting market movements.

Predominant models in existing research tend to diverge into two distinct streams. One stream focuses on leveraging machine learning for long-term stock price forecasting, often marginalizing the role of sentiment analysis. The other stream neglects the sentiment dimension altogether, which may lead to a myopic understanding of market dynamics. This dichotomy represents a critical gap, given the multifaceted and dynamic nature of the stock market, which is influenced by a complex array of factors beyond historical price trends.

Acknowledging this, our approach advocates for a more nuanced and adaptive methodology. We propose the integration of sentiment analysis into a machine learning framework, capitalizing on the predictive power of retail sentiment as a contemporaneous market indicator. Moreover, we introduce the dynamic decision making as a core component of our model. This method facilitates periodic retraining and recalibration of the model, aligning it with the latest market data. Such a strategy ensures that the model remains responsive to market fluctuations, thereby enhancing its predictive accuracy and robustness over time.

In essence, our approach is designed to capture the dynamic interplay between market sentiment and price movements, offering a more holistic and agile forecasting tool that aligns with the ever-evolving landscape of the stock market.

Problem Statement:

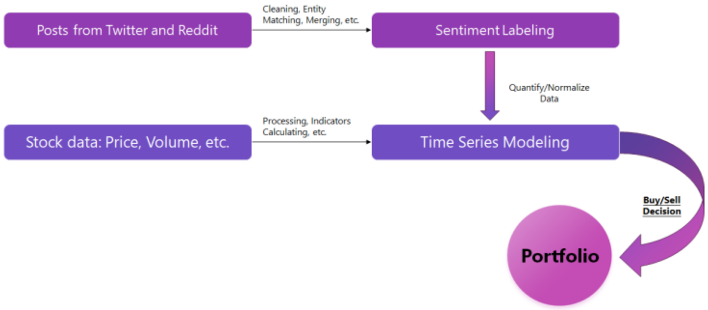
Given a stock market with a set of stocks $S = \{s_1, s_2, \dots, s_n\}$, where each stock s_i has a daily closing price $p_i(t)$ at time t, and a corresponding set of daily retail sentiment scores $R = \{r_1(t), r_2(t) \dots, r_n(t)\}$, the problem is to construct a predictive model M that forecasts the future price $\hat{p}_i(t + \Delta t)$ of stock s_i at a future time $t + \Delta t$.

The model M aims to leverage the sentiments extracted from social media platforms, quantified as sentiment scores $r_i(t)$, to predict the impact on the future stock prices. The sentiment scores are derived from the analysis of textual data using Natural Language Processing (NLP) techniques, capturing the collective mood and opinions of retail traders.

The predictive model M will be evaluated based on its accuracy in forecasting the price $\hat{p}_i(t + \Delta t)$, using standard metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE). The model will also be assessed for its robustness across different stock categories and market conditions.

The ultimate goal is to determine the efficacy of retail sentiment scores as predictors of stock market behavior and to establish a reliable method for stock price prediction that can aid in investment decision-making processes.

General Structure of Our Framework


Posts from Twitter and Reddit
Cleaning, Entity Matching, Merging, etc.
Sentiment Labeling
Quantify/Normalize Data
Stock data: Price, Volume, etc.
Processing, Indicators Calculating, etc.
Time Series Modeling
Buy/Sell Decision
Portfolio

NLP

Data Preparation

Gathering relevant, high-quality data presented challenges due to API rate limits on platforms like Reddit and Twitter. We explored alternatives by sourcing data from Kaggle, open-source datasets, and Twitter posts, including informal language. We also incorporated labeled news headlines to capture real-time market influences.

Our decision to diversify data sources was driven by unique advantages:

1. Kaggle and research paper data offer specialized and well-curated content for sentiment analysis in the financial domain.

2. Twitter posts provide real-time, colloquial insights from market participants, helping us address the noise in social media discussions.

3. Labeled news headlines enable our model to respond dynamically to breaking news, mirroring market reactions.

4. We used ChatGPT to generate diverse conversational-style data, which was labeled with sentiments for a well-rounded training dataset.

To bolster the quality of our training data, a variety of data augmentation techniques were employed.

1. Synonym Replacement: This technique replaces words in a sentence with synonyms to add variety. For instance, changing "Worried about the recent drop in the price of gold" to "Concerned about the recent decline in the value of gold."

2. Back Translation: This involves translating text into another language and then back into the original language, introducing subtle phrasing changes. For example, "AAPL's product launch was underwhelming, considering selling our shares" might become "AAPL's product launch was disappointing; thinking about divesting our shares."

3. Paraphrasing: It provides alternative sentence structures and expressions. For instance, "Just sold our Amazon shares; they've become too expensive" could be paraphrased as "We've recently disposed of our Amazon holdings as they've become unaffordable."

4. Oversampling/Undersampling: These techniques address class imbalances, ensuring equal representation of sentiment categories. If there's an imbalance, oversampling duplicates examples from the minority class, while undersampling reduces examples from the majority class. Here, we oversampled the negative data to create a balanced training set.

One pressing issue we faced was dataset bias, which can affect model performance and fairness. To mitigate bias, we used ChatGPT's natural language processing to balance the data and ensure diverse perspectives. This helped us create a more equitable dataset, forming a strong foundation for our project.

Our labeled dataset is divided into three subsets:

1. Training Dataset (8,925 rows): Used for model training.
2. Validation Dataset (2,232 rows): For hyperparameter tuning and model evaluation.
3. Test Dataset (876 rows): Reserved for final model performance evaluation.

We also collected unlabeled comment posts in three subsets based on their characteristics:

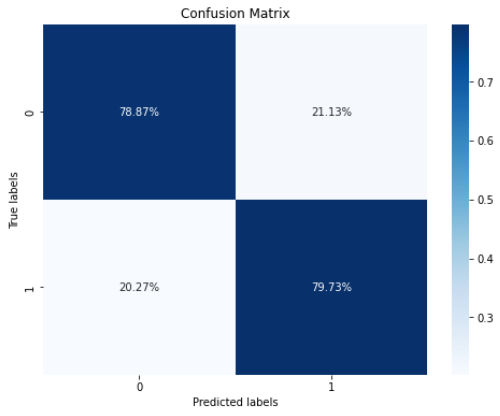
1. Tweet_filtered_TSLA Dataset (1,123,262 rows): Tweets with dates and stock mentions.
2. stock_tweets_filtered_TSLA Dataset (37,422 rows): Focused on TSLA-related tweets.
3. tweets_remaining_filtered_TSLA Dataset (60,836 rows): Additional TSLA-related tweets.

We chose not to combine labeled and unlabeled datasets due to significant differences in their origins and potential data variations. Unlabeled posts will undergo labeling through natural language processing, making them ready for integration into our stock price prediction model.

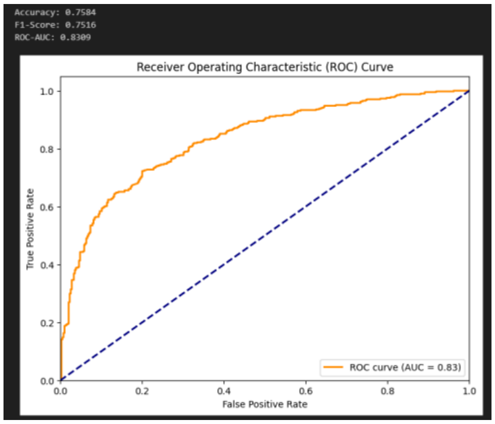
Model Testing

FastText & Word2Vec Embedding Model:

We began with a dataset of over 1.6 million Twitter posts, which covered a wide range of topics beyond the equity market. In contrast, equity-specific datasets were smaller, typically around 8,000 entries. We used the larger dataset for training and the smaller, equity-specific ones for validation. Due to the larger dataset's non-specific nature and many irrelevant posts, our model achieved 80% training set accuracy but dropped to approximately 60% accuracy on the test set, as shown in the confusion matrices below.

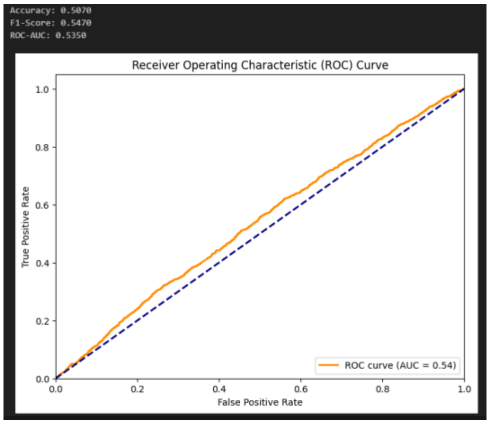


Regarding FastText embeddings, we trained the model on the initial dataset for three epochs, resulting in an ROC-AUC score of 0.83. This score indicates effective discrimination between positive and negative cases within the training data. FastText's ability to capture subword information proves valuable, especially for languages with complex morphology and out-of-vocabulary words. It provides richer and more context-aware representations compared to Word2Vec, which we previously experimented with.

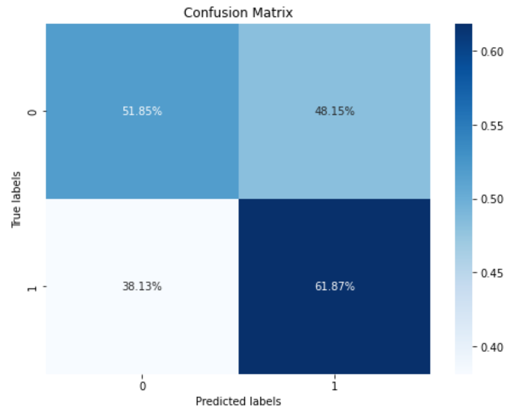


```
Accuracy: 0.7584
F1-Score: 0.7516
ROC-AUC: 0.8309
```

However, when the same model is subjected to testing using an entirely unseen dataset, a disparity emerges. The model performs suboptimal on this new data, as indicated by an ROC-AUC score of only 0.535, which is significantly lower than the training performance. This discrepancy, where the model's performance regresses when applied to unseen data and its ROC-AUC score falls closer to the baseline value of 0.5, is far from ideal. It suggests that the model might not generalize well to new, unseen instances and may need further refinement or adjustments to enhance its predictive capabilities on diverse datasets.



```
Accuracy: 0.5070
F1-Score: 0.5470
ROC-AUC: 0.5350
```



LSTM Model

The initial training and testing accuracy using the RNN LSTM neural network displayed a noticeable disparity, with a commendable 0.79 on the validation set but a less satisfactory 0.58 on the testing set. This discrepancy raised concerns, as the model appeared to perform exceptionally well within the known confines of the training data but struggled when presented with new, unseen data. This disparity prompted the comprehensive evaluation of the data and model, necessitating an exploration into potential improvements in the training dataset composition and, perhaps, model architecture, to achieve a more balanced and consistent performance.

When confronted with the drop in testing accuracy, we initiated a comparative study involving different machine learning models. The objective was to discern if the discrepancy in performance was a result of the training data's quality or if it stemmed from the chosen model's limitations.

Naive Bayes, Random Forest, XGBoost Models

For a thorough comparison, we employed a variety of machine learning models, each known for specific strengths in sentiment analysis. Our ensemble included Naive Bayes for simplicity, Random Forest for complex relationships, and XGBoost for versatility. This allowed us to investigate if the dip in testing accuracy was due to model intricacies or data composition.

Naive Bayes yielded a training validation accuracy of 0.76 and a testing accuracy of 0.54, similar to our initial neural network approach. This implies that the issue may not solely stem from the choice of model but could be influenced by dataset challenges. These findings stress the importance of addressing data quality and diversity.

Similarly, the XGBoost experiment resulted in a training validation accuracy of 0.68 and a testing accuracy of 0.61, matching the outcomes of the initial neural network. This consistency across different methods highlights persistent dataset challenges. It underscores the need for further data preprocessing, feature engineering, or exploration of alternative data sources.

Thus, the testing accuracy from these alternative models didn't substantially improve upon the initial results obtained with the more complex RNN LSTM neural network. This suggests that model complexity may not be the primary bottleneck in this scenario.

BERT Model

To address issues in our training dataset, including unrelated Twitter posts, we improved model performance using FinBERT, a specialized NLP model designed for financial text and sentiment analysis. FinBERT is fine-tuned for finance using a vast financial corpus, including the Financial PhraseBank dataset. More details can be found in the paper "FinBERT: Financial Sentiment Analysis with Pre-trained Language Models" and our Medium blog post.

We deployed the model through the Hugging Face Query API, using the repository "tarnformnet/Stock-Sentiment-Bert." The model surpassed expectations, achieving a test dataset accuracy of 0.68. Additionally, we explored an alternative variant, ProsusAI/finbert, which provides softmax outputs for three sentiment labels: positive, negative, and neutral.

Accuracy			0.68	50
Macro Avg	0.47	0.46	0.46	50
Weighted Avg	0.74	0.68	0.71	50

However, given the binary nature of our testing dataset, the accuracy is not good on the testing data with accuracy rate only 0.27. So, we endeavored to further fine-tune the model to align it with the specific requirements of our dataset.

Accuracy			0.27	100
Macro Avg	0.48	0.21	0.28	100
Weighted Avg	0.84	0.27	0.40	100

We trained the BERT model from scratch using the 'bert-base-uncased,' the original uncased base model, in combination with the newly acquired financial data. This method offered the advantage of full control and customization over the training process, enabling us to align the model precisely with our specific requirements.

To prevent overfitting, dropout layers are included, randomly setting some input units to zero during training (with a 0.1 probability in this case). The "classifier" is a linear layer with 768 input features (matching BERT's output size) and 2 output features for binary classification, which can be adjusted for different tasks.

BERT's strengths include bidirectionality, pre-training on a large text corpus for rich language representations, and its large-scale architecture. However, it requires significant computational resources.

We attempted to enhance our model by exploring the "yiyanghkust/finbert-tone" repository for sentiment analysis improvements but faced compatibility issues in our environment.

Undeterred, we chose to train a BERT model from scratch, utilizing the 'bert-base-uncased' model as a foundation due to its reputation and adaptability. We augmented the model with an additional dataset for financial sentiment analysis.

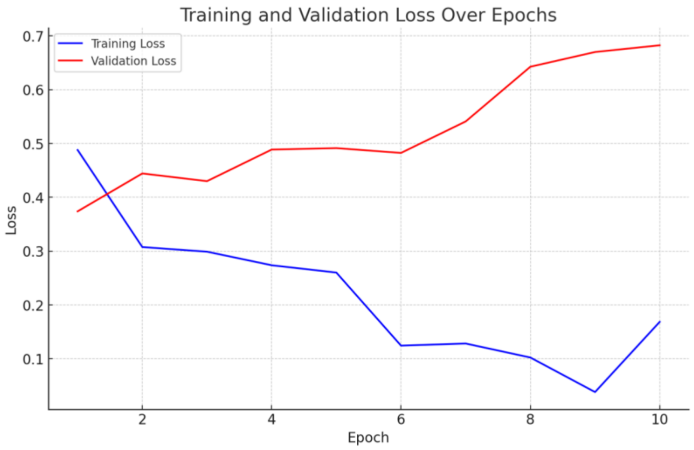
In the next phase, we trained the new BERT-based model using the augmented dataset, targeting a minimum 80% accuracy benchmark. We employed a comprehensive pipeline involving various natural language processing stages, including tokenization with BertTokenizer, setting training parameters with TrainingArguments, and ensuring uniform sequence lengths with DataCollatorWithPadding. Training occurred on available GPU resources in Google Colab.

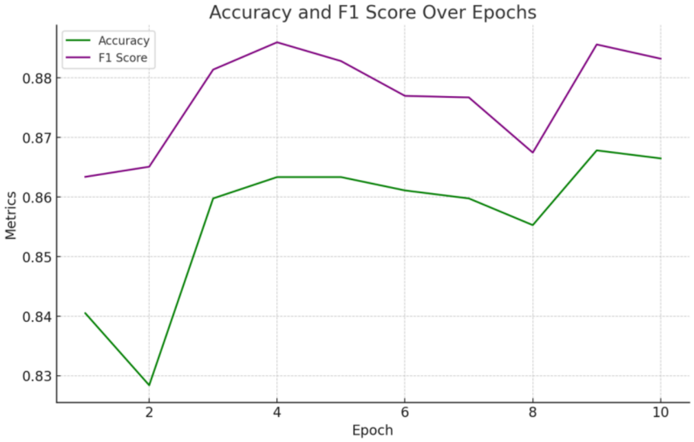
A Trainer instance from transformers managed the training loop, with post-training evaluation utilizing standard metrics from scikit-learn, providing valuable insights into its effectiveness for the sequence classification task.

During the ten training epochs, we monitor key metrics like Training Loss, Validation Loss, Accuracy, and F1 Score to assess the model's learning and generalization. Each epoch represents a training phase, revealing how the model adapts and optimizes predictions over time. These metrics are crucial for diagnosing issues like overfitting and fine-tuning parameters, providing a balanced assessment of the model's performance.

F1 Score: Balances precision and recall, showing stability with a slight overall increase from 0.863392 to 0.883229. The model maintains a good balance throughout training, even with the increasing Validation Loss.

Epoch	Training Loss	Validation Loss	Accuracy	F1
1	0.4879	0.373879	0.840502	0.863392
2	0.3075	0.444309	0.828405	0.865093
3	0.2989	0.429997	0.859767	0.881394
4	0.2737	0.48869	0.863351	0.885981
5	0.2601	0.491391	0.863351	0.882828
6	0.1243	0.482482	0.861111	0.876984
7	0.1282	0.540972	0.859767	0.876723
8	0.1022	0.642647	0.855287	0.86746
9	0.038	0.670001	0.867832	0.885615
10	0.1685	0.6825	0.866487	0.883229





Pseudocode the NLP training model is as follows:

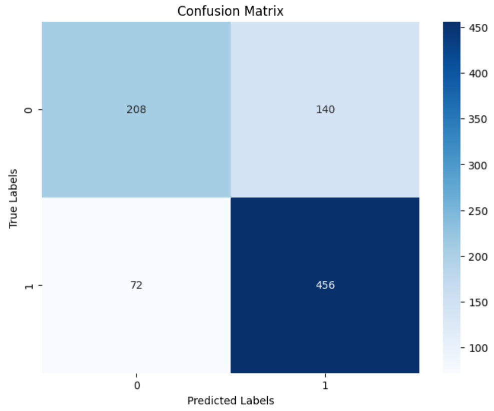
```
1. Model Configuration and Training
    - Import and configure BERT model and tokenizer for sequence classification
2. Model Evaluation
    - Generate a performance report
    - Plot accuracy over time periods
    - Compute and plot ROC curve; calculate AUC
3. Model Application and Prediction
    - Load a saved model and tokenizer
    - Make predictions on a test dataset
    - Process and output predictions (probabilities and labels)
    - Save predictions to a CSV file
```

Model Prediction

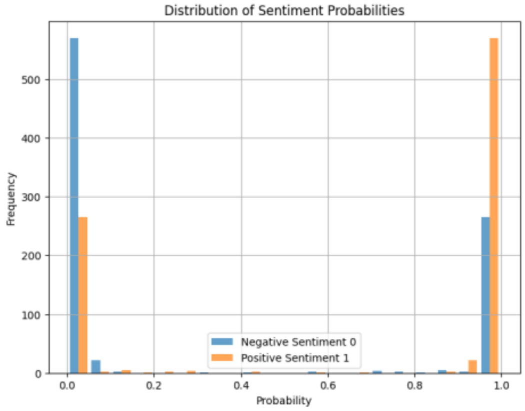
Using the trained model, we make predictions on an unknown dataset and further analyze the results.

The confusion matrix highlights the classifier's strong ability to correctly identify both classes. It shows higher numbers of true positives (456) and true negatives (208) compared to false positives (140) and false negatives (72). This indicates the model's effectiveness in identifying both positive and negative classes, with relatively fewer false predictions.

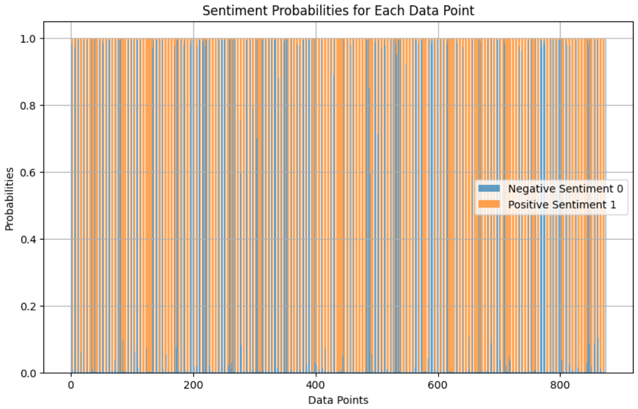
The lower occurrence of false predictions suggests the model's accuracy and promising generalization capability to classify new, unseen data. This also implies a balanced sensitivity and specificity, which is desirable in predictive models, especially in applications where both types of classification errors have significant consequences.



In the bar chart below, we see two clear peaks: one with high confidence predictions for negative sentiment (0) near probability 0, and another for positive sentiment (1) near probability 1. This distribution signifies strong and confident predictions, as most are concentrated at the extreme ends of the probability scale. It highlights a well-performing model with high certainty in its classifications.



The plot below displays two overlapping series of data points representing negative (label 0) and positive (label 1) sentiment probabilities. It reveals a model that predicts both sentiments across the dataset without bias. The intermingling of blue and orange lines indicates balanced probability assignments, showcasing a well-calibrated model. This even distribution ensures the model is equally likely to predict positive or negative sentiment, preventing skewed interpretations.



After the predicted results analysis, we are confident that the model has a good performance.

Predictions to be used in Stock Model

Once we achieved the desired accuracy, we integrated the prediction model's numeric outputs into our final stock price prediction model. This fusion allowed us to assign precise values to labels, enhancing the precision and data richness of our model. It was a significant step in rigorously testing our hypothesis and exploring the relationship between market sentiment and stock prices.

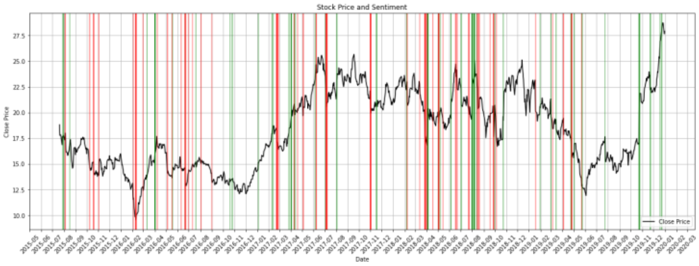
We chose numeric values over categorical values to increase information richness and flexibility. Numeric values offered a wide range of data for deeper insights and statistical analysis, including means, maximums, minimums, and variances. This approach enabled us to comprehensively study the interplay between market sentiment and stock prices, uncovering subtleties that impact stock price movements.

In essence, using numeric values provided the precision and adaptability needed to explore the complex dynamics of financial markets thoroughly. It empowered us to extract valuable insights for future strategies and decisions, enhancing our ability to navigate the financial landscape.

Tesla Sentiment Analysis

The image depicts sentiment analysis on Tesla from 2015 to 2020, where the color intensity reflects the sentiment's deviation from historical levels over a past

period of time (rolling window) — darker shades indicate a greater deviation.
5.5 cm



According to the data, there appears to be a positive correlation between daily positive sentiment and Tesla's stock price, while daily negative sentiment exhibits a negative correlation. Interestingly, periods of extreme negative sentiment, indicated by dark red colors, have been found to correlate positively with subsequent stock returns, implying a contrarian relationship during times of heightened negative sentiment. The logistic regression on return of the stock analysis provides the following statistically significant results:

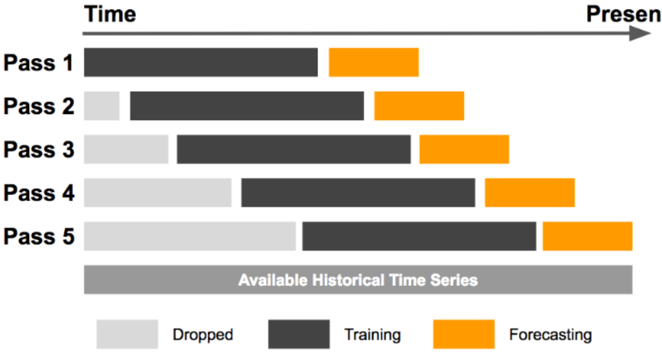
Feature	Coef	P-val
daily_positive_sentiment	0.0035	0.012
daily_negative_sentiment	-0.0039	0.009
z_score_positive	-0.8931	0.006
z_score_negative	0.6232	0.006

The statistical significance of these findings is supported by the z-scores and corresponding p-values. The coefficients for daily positive and negative sentiments, though small, are statistically significant and have opposing effects. The significant negative z_score_positive and significant positive z_score_negative further suggest that extremely negative sentiment could indeed be a predictive indicator of an upcoming rise in Tesla's stock, offering a contrarian viewpoint to traditional sentiment interpretation. These results could imply that market overreactions to negative news might provide buying opportunities, as the sentiment eventually corrects and the stock price rebounds.

Trading Signal Prediction

The approach towards trading signal prediction in this study is crafted to address the dynamic and rapidly changing nature of the stock market. Traditional methodologies often utilize a single model for forecasting stock returns over extended periods, sometimes up to a hundred days. This strategy, while valid in certain contexts, might not be fully equipped to handle the market's volatility and frequent shifts.

To counter this limitation, our proposed model adopts a more flexible and responsive strategy. We suggest a daily recalibration of the model, utilizing the latest available data for each day's predictions. This method involves forecasting returns for the upcoming day or week and then incorporating the actual return data for that day into the model. This rolling window approach allows for continuous refinement of predictions, aligning the model closely with the current market conditions. By focusing on daily predictions and updating the dataset after each forecast, our model remains attuned to the market's dynamics, thereby potentially enhancing the accuracy and relevance of its predictions.



Pseudocode

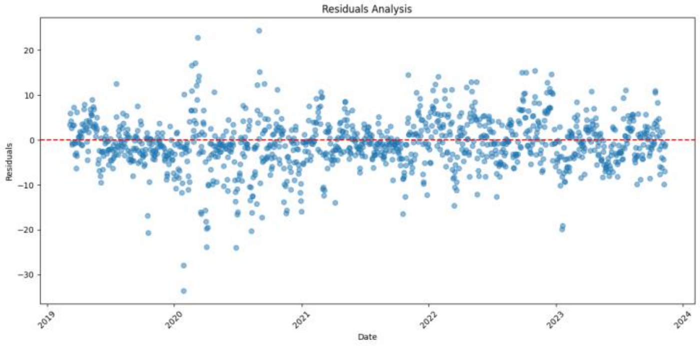
```
Initialize rolling_window_start, rolling_window_end to the first index of the time series data
Initialize predictions, validation_scores, test_scores to empty lists
while rolling_window_end is less than the length of the time series data do
    Set training_set to the subset of data between rolling_window_start and rolling_window_end
    Set training_labels to the subset of labels between rolling_window_start and rolling_window_end
    model ← train_model(training_set, training_labels)
    forecast_start ← rolling_window_end + 1
    forecast_end ← forecast_start + FORECAST_HORIZON – 1
    if forecast_end is within the validation set indices then
    validation_set ← get_subset(data, forecast_start,
    forecast_end)
            validation_prediction ← model.predict(validation_set)
            Append validation_prediction to validation_scores
        else if forecast_end is within the test set indices then
            test_set ← get_subset(data, forecast_start,
            forecast_end)
            test_prediction ← model.predict(test_set)
            Append test_prediction to test_scores
  Increment rolling_window_start and rolling_window_end by
  FORECAST_HORIZON
  end while
  Evaluate model using validation_scores
  Evaluate model using test_scores
  Return predictions for the test set
```

Objective: Nonstationary Vs. Stationary

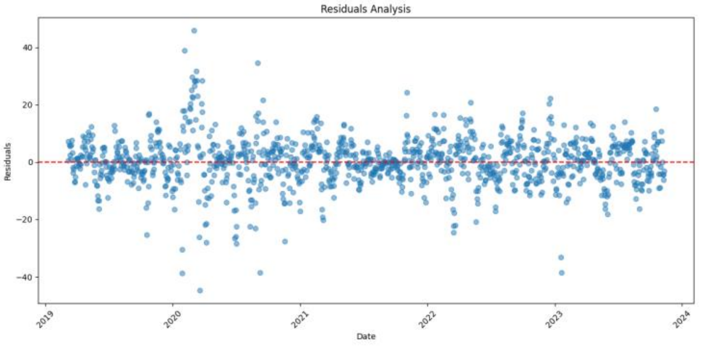
An important aspect to address before developing our models is the objective we aim to achieve. In traditional time series modeling, Return is often chosen as a key variable due to its stationarity, which provides robustness over time. However, our study diverges from this norm by focusing on short-term trends, specifically those within a timeframe of less than two weeks. In this context, using Price as the primary variable has been shown to yield better outcomes than using Return. This is because, within such a short-term focus, the price movements are more indicative of immediate market reactions and trends.

Although transforming all variables into a stationary format could lead to lower loss in the model, as depicted in the two accompanying images, it has been noted that stationary models fall short in capturing market trends, despite their efficacy in reducing loss. In contrast, nonstationary models tend to yield greater gains in portfolios, suggesting their advantage for our study's objectives. Emphasizing this point, the upcoming images showcase the use of a Random Forest model, where residuals are converted into percentage returns.

Objective being Price and using Nonstationary features



Objective being Return and using Stationary features



Feature Engineering & Selection

In our study, we utilized a dataset from Yahoo Finance that includes basic daily pricing and volume information about stocks, such as Close, High, and Volume. Leveraging this data, we conducted a series of transformations and calculations of indicators to create variables that capture the stock's momentum, as well as its overbought or oversold levels, thereby aiding in the identification of market trends. Additionally, we incorporated indicators such as the one-month Treasury rate and sector data to gauge the current economic climate and industry-specific trends. This comprehensive approach refines our dataset to include variables that are most pertinent and influential in understanding the stock market's dynamics. These carefully engineered features will serve as essential inputs for our predictive models, enhancing their ability to analyze and forecast market movements with greater accuracy and relevance.

Price Related Features	Close, Hight, Lagged Close, etc.
Volume Related Features	Volume, Lagged Volume, etc.
Price Related Indicators	Moving Averages (MAs), RSI, etc.
Volume Related Indicators	CM Williams Vix Fix, Volume Price Trend (VPT), etc.
Economic Indicators	Sector Performance (SPY), One Month Treasury Rate (Interest Rate)
Sentiment Score	Sentiment Score from sentiment analysis

Rolling Window Size Selection

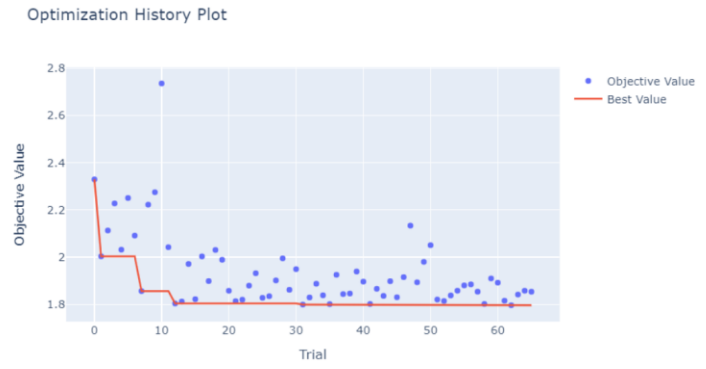
The concept of a rolling window is pivotal in our model, as it determines the amount of historical data the model considers for making predictions. This window size is crucial since it not only influences the accuracy of the model but also affects its runtime. After completing Feature Selection & Engineering, it is essential to define an appropriate window size. Our experimentation with various window sizes [5, 10, 20, 40, 80, 160, 320, 640] yielded the following outcomes with XGBoost model:

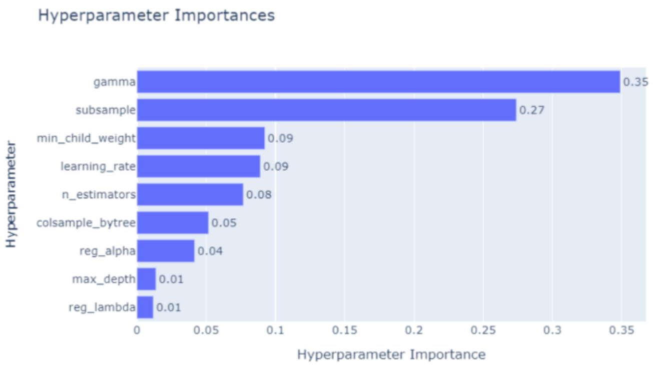
Window Size (Days)	Mean Square Error	Mean Absolute Error	Processing Time
5	7.5503	2.0919	03:47
10	6.5008	1.9264	03:59
20	6.7170	1.9485	04:00
40	6.2436	1.8836	04:23
80	6.2501	1.8715	05:09
160	7.9529	2.0051	06:46
320	6.9490	1.9744	09:21
640	7.0130	1.9748	11:05

These results indicate a trade-off between accuracy (as measured by MSE and MAE) and computational efficiency. A smaller window size results in quicker processing but higher errors in general, whereas a larger window size can potentially decrease error at the cost of increased processing time. The choice of the window size should thus be a balance between desired accuracy and feasible runtime, and a window size of 40 is selected for future usage.

Hyperparameter Tuning

In our study, we have harnessed the capabilities of Optuna for hyperparameter optimization within our Python models. This sophisticated framework conducts an automated and intelligent search to pinpoint the most impactful hyperparameters, thereby augmenting the model's performance. Initially, we define a broad hyperparameter search space. Optuna then iteratively evaluates different hyperparameter combinations through its trial-and-error approach. Each trial's performance informs the subsequent search direction. Optuna's pruning feature is particularly noteworthy; it discontinues trials that are not promising, optimizing computational efficiency. Employing intelligent search strategies, such as Bayesian optimization and the Tree-structured Parzen Estimator, Optuna efficiently homes in on the most promising hyperparameters, outperforming traditional search techniques like grid or random search. Following a comprehensive sequence of trials, Optuna finalizes the hyperparameters that yield the optimal model performance, thereby fine-tuning our model with precision and streamlining the entire optimization cycle.





Dynamic Decision Making

In the field of financial forecasting, a model predicting weekly outcomes is not enough; it's the translation of these forecasts into actionable strategies that counts. We've crafted a robust approach that marries predictive analytics with historical stock price data to forge a dynamic trading strategy.

Forecast Generation: We generate predictions for each stock entry based on the established window size, comparing predicted returns against actual returns over a subsequent 5-day window.

Portfolio Initialization: The strategy begins with an initial capital allocation and no stock positions. We continuously track the portfolio's cumulative value, which includes both the available capital and the market value of any held stocks.

Trading Logic—The Core Mechanism: Central to our strategy is the adaptive trading logic, which uses the previous window size real returns to set buy and sell thresholds based on the 75th and 25th percentiles, respectively. This generates trading signals that prompt buying when exceeded or selling when dipped below.

a) Acquisition Strategy: When at least three strong buy signals occur within the predicted range and there's sufficient capital, we authorize a stock purchase.

b) Divestment Strategy: Conversely, when three or more sell signals appear and stocks are in the portfolio, we proceed with selling.

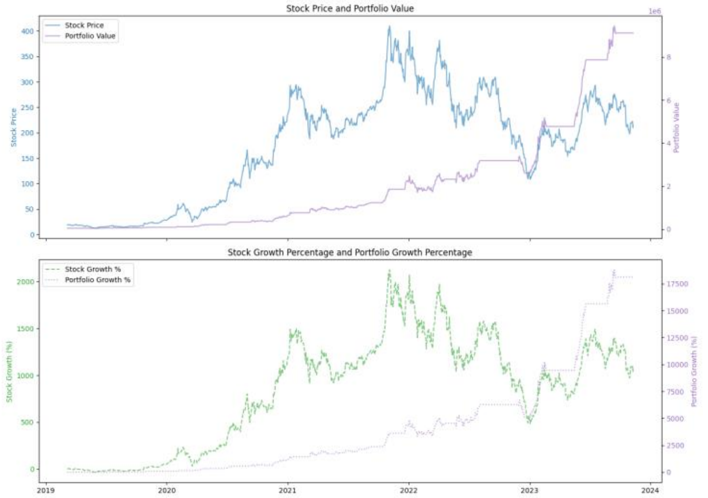
Model Selection

Having completed the optimization of features, window sizes, hyperparameters, and our trading strategy, we proceeded to evaluate the efficacy of different predictive models. The focus of our analysis lies on the Random Forest and XGBoost models due to the non-comparability of LSTM's stationary normalization process with these models.

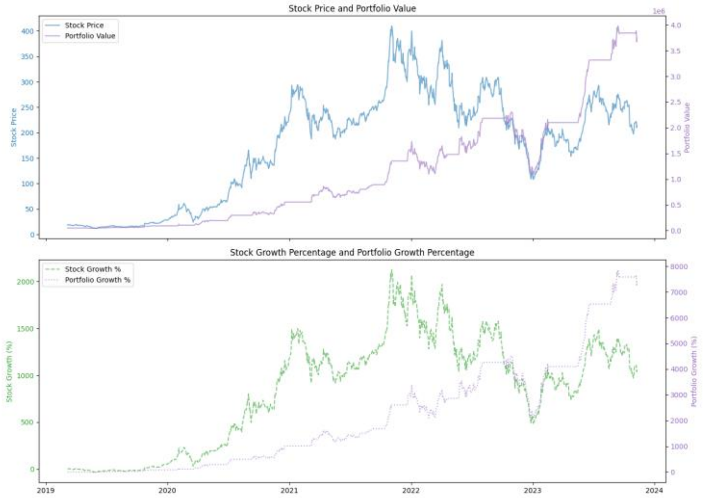
The two plots under consideration here illustrate the gains of our Portfolio versus the gains of the Tesla Stock itself. The results are remarkable—XGBoost has outperformed Tesla stock by approximately 1700%, while Random Forest has exceeded it by about 800%. However, a closer examination of the plots suggests that XGBoost is particularly adept at navigating markets characterized by high volatility and fluctuations. On the other hand, Random Forest seems less capable of managing in highly volatile conditions. This observation is crucial as our models are designed to capitalize on short-term fluctuations, aligning with our objective of capturing short-term trends.

Moreover, XGBoost demonstrates efficiency in computational time, taking only 1 minute and 57 seconds to process data spanning from 2019-01-01 to 2023-11-21, whereas Random Forest requires 3 minutes and 17 seconds for the same task. Given XGBoost's robust performance in volatile markets and its computational efficiency, we have selected it as the model of choice for our future use and demonstrations.

XGBoost on Tesla



Randon Forest on Tesla

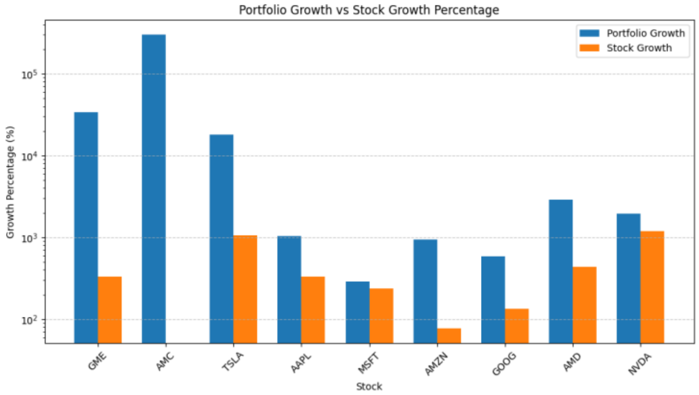


Robustness Test

Upon calibrating our model with data from Apple, we have scrutinized the efficacy of the XGBoost model utilizing Tesla's stock data, which yielded commendable outcomes. It is imperative to extend this evaluation to encompass an array of stocks, encapsulating both technology sector mainstays and so-called 'meme' stocks. The model's robustness has been previously affirmed in a bullish market spanning from 2019 to 2022, with an enhanced performance observed amidst the heightened volatility of a bearish market.

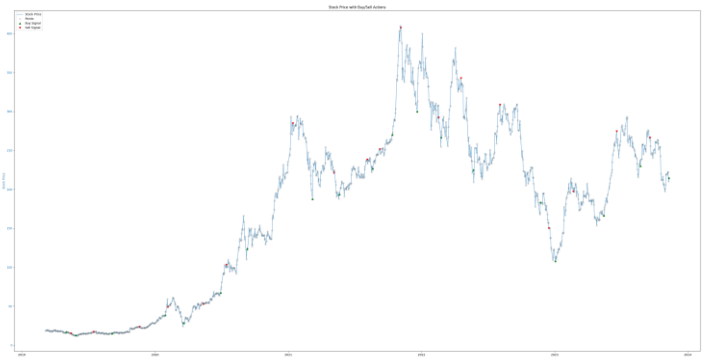
We now endeavor to analyze the model's performance across a selection of stocks, specifically: GME, AMC, TSLA, AAPL, MSFT, AMZN, GOOG, AMD, and NVDA. The graph presented employs the logarithmic scales to illustrate the comparative growth of our portfolio against the growth of these stocks. The visual evidence indicates that, in all examined scenarios, our model has consistently achieved superior performance relative to the broader market.

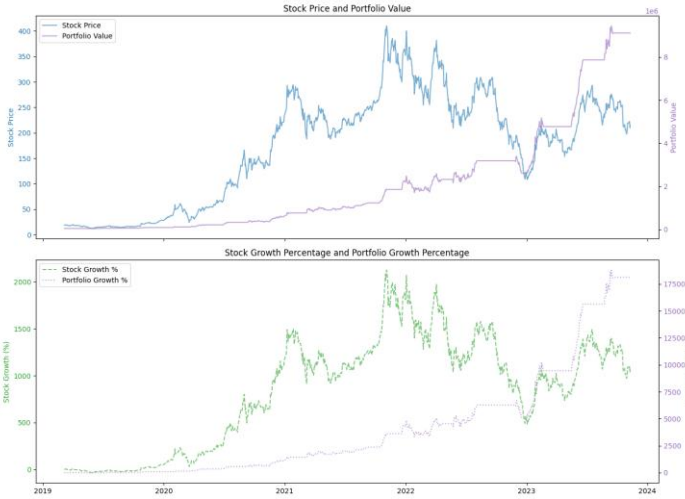
It is important to highlight that AMC's stock growth is not depicted due to its negative trajectory. This underscores the versatility of our model; it adeptly navigates both winning and losing assets, excelling particularly in volatile environments as evidenced by stocks like GME, AMC, and TESLA, which are characterized by their substantial market volatility.



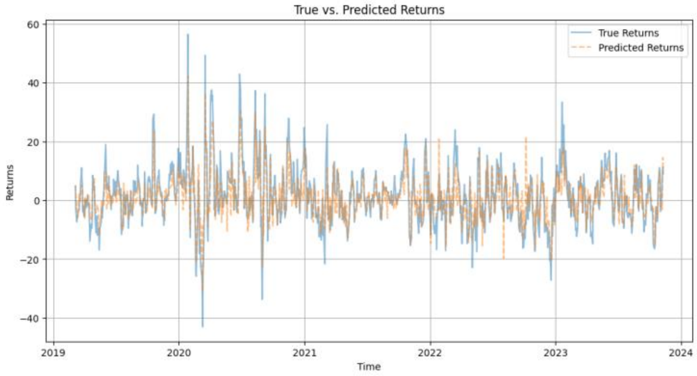
Performance of XGBoost on Tesla

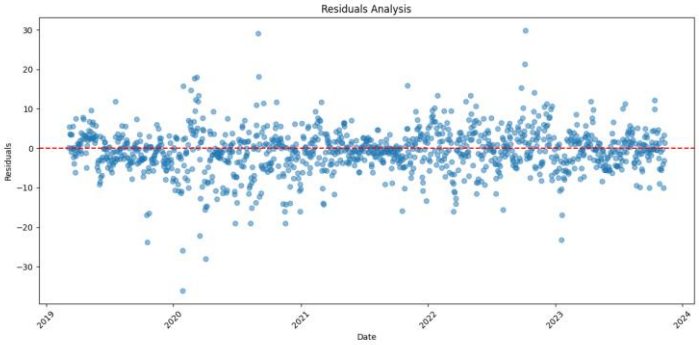
The empirical analysis of our model showcases a commendable 90% accuracy in executing profitable trades. Despite occasional misses in capturing a buying opportunity or prematurely selling a winning trade, the bulk of its decisions are well-judged and successful. The model's proficiency is not limited to a specific timeframe or stock type; it demonstrates versatility and effectiveness across various stocks and market conditions.





From the 'True vs. Predicted Returns' plot, it is evident that our model adeptly tracks most atypical trends, indicating a keen capture of market anomalies. Furthermore, the residual analysis presents a mean close to zero and a consistently stable variance, underscoring the model's reliability and robustness. These findings affirm the model's predictive strength and validate its strategic prowess in financial market forecasting.

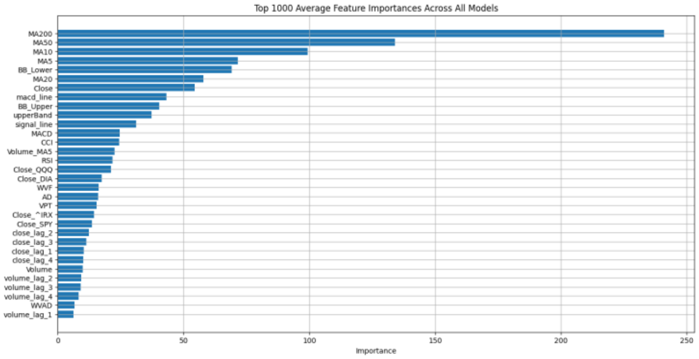




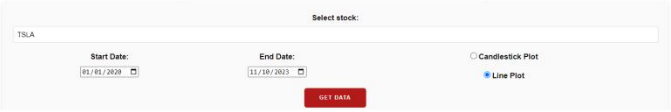
Potential issues our current model

The feature importance plot yields a surprising insight: 'Volume' and 'Close price' are deemed to be of low importance in our model's decision-making process, which initially appears counterintuitive given their perceived significance in stock valuation. Upon closer examination, the likely explanation is that the information typically conveyed by the closing price is already explained within other indicators such as Moving Averages (MAs) and Bollinger Bands, which are calculated based on closing prices over time and thus may overshadow the predictive power of the raw closing price itself.

As for volume, despite rigorous attempts to refine its relevance through various methods, indicators, normalizations, and transformations, it consistently ranks low in importance. This suggests that the current model does not heavily weigh volume in its predictive algorithm. This aspect of the model's feature weighting may point towards an area for potential enhancement, as conventional market theories often attribute significant interpretive value to volume data.



UI Design:


Select stock:

TSLA

Start Date: 01/01/2020

End Date: 11/10/2023

○ Candlestick Plot

● Line Plot

The user interface (UI) displayed in the image features several distinct elements:

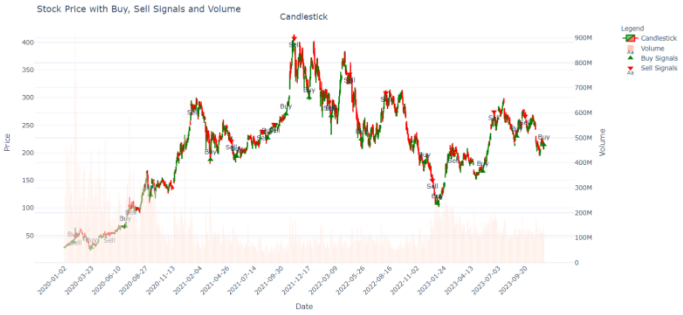
- Stock Selection: Users can choose a stock via a dropdown menu, indicated by the placeholder "TSLA" for Tesla Inc.

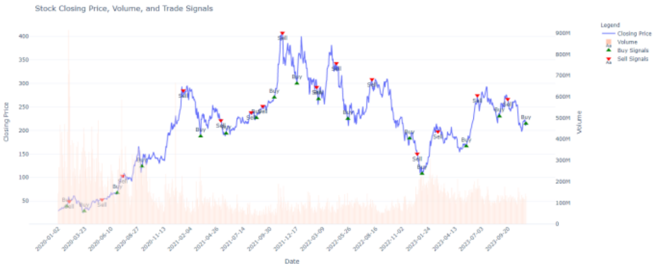
- Date Range: Input fields for "Start Date" and "End Date" allow users to select a specific period for analysis.

- Graph Type: Options to display data as either a "Candlestick Plot" or "Line Plot" are available, with the "Line Plot" selected in this instance.

- Data Retrieval: A "GET DATA" button suggests functionality to fetch and display stock data based on the chosen parameters.

The line chart visualizes the closing price and trading volume of a stock, with added "Buy" and "Sell" signals based on algorithmic analysis, while the candlestick chart provides a detailed representation of daily price movements and trading volume. Both charts span from the selected date range and include a legend for clarity. The interface is minimalistic, focusing on essential features to enhance user experience in stock analysis.





Conclusion:

This report documents an innovative blend of machine learning for time series analysis and natural language processing to predict stock market movements, culminating in a user-friendly interface designed to democratize access to these advanced techniques. The fusion of cutting-edge analytical methods with a distinctive trading strategy represents a significant leap beyond conventional financial analysis tools.

Our empirical assessments underscore the exceptional performance of our models, notably those employing BERT for sentiment classification, which have shown marked accuracy and precision in financial contexts. These gains are complemented by the models' swift processing capabilities and their adaptability to a range of stocks and market conditions. Notably, our time series model and strategic decision-making framework have outperformed the market significantly, achieving over an 800% gain in just four years, and have proven effective across various tech and meme stocks, as well as across different temporal spans.

Despite these successes, we recognize the potential for further refinements. With the constraints of time, numerous dimensions from data sourcing to model robustness present opportunities for enhancement in future research.

Future Works:

NLP

Incorporate Expert Analysis: We can integrate insights from institutional traders and professional analysts, enriching our sentiment analysis with expert opinions and market forecasts. This will enhance the model's ability to interpret complex market sentiments.

Diversify Data Sources: By expanding our data sources to capture a broader demographic, we can make our sentiment analysis more comprehensive and representative of the wider trading community.

Advanced NLP Technologies: We can adopt the latest NLP advancements, potentially leveraging transformer-based models that specialize in financial discourse, to refine sentiment analysis accuracy.

Stock

Sector Diversification: We can extend our analysis to include a variety of sectors, allowing for a more diversified and robust understanding of market trends.

Data Enrichment: Adding in-depth financial metrics and economic indicators can create a richer dataset, leading to more precise market predictions.

Stock Selection Algorithm: Developing an advanced framework to identify stocks with high return potential will be a priority, aiming to optimize stock picks based on a range of financial indicators.

Portfolio Optimization: We can explore sophisticated portfolio optimization techniques to balance returns and risk, tailoring portfolios to align with individual risk tolerances and market conditions.

These future directions indicate a concerted effort to refine and expand the system’s capacity for sentiment analysis and stock market predictions. By embracing a wider array of data, extending analytical breadth, and utilizing advanced techniques, we aspire to elevate the system's predictive performance and contribute to the innovation in financial technology.

Citations:

Stock

Bao, W., Yue, J., & Rao, Y. (2017). A deep learning framework for financial time series using stacked autoencoders and long-short term memory. PLoS ONE, 12(7): e0180944. https://doi.org/10.1371/journal.pone.0180944

Relation: Bao, Yue, and Rao present a deep learning framework utilizing stacked autoencoders and long-short term memory for analyzing financial time series. Notably, they introduce concepts of buy and sell signals based on predicted prices, resonating with our exploration into machine learning-driven financial predictions.

Differentiation: While they lay the groundwork in understanding financial time series through deep learning, our research extends this by incorporating contemporary machine learning methodologies to forecast stock returns over shorter durations. Additionally, we delve into portfolio management through our trading strategy, a topic not explored in their paper.

Dash, R., & Dash, P. K. (2016). A hybrid stock trading framework integrating technical analysis with machine learning techniques. The Journal of Finance and Data Science, 2(1), 42-57. https://doi.org/10.1016/j.jfds.2016.03.002

Relation: This paper delves into trading signals and the intricacies of implementing a comprehensive

trading strategy. Its content is rich in explaining how trading decisions can be informed and executed.

Differentiation: Unlike the paper's emphasis on broader sectors like SPY, our approach zeroes in on individual stocks. Our research also capitalizes on a myriad of indicators, dedicating significant effort to feature selection and engineering, aspects that weren't as extensively addressed in the referenced paper.

Pezim, B. (2018). How To Swing Trade. Preface by A. Aziz. ISBN: 9781726631754.

Relation: The book provides an extensive overview of swing trading strategies and market dynamics, setting the stage for our exploration of stock market behaviors.

Differentiation: Our project enhances these basic principles with state-of-the-art machine learning techniques to forecast stock market returns, delivering a modern, technology-enhanced viewpoint.

NLP

Loughran, T., & McDonald, B. (2011). When is a liability not a liability? Textual analysis, dictionaries, and 10-Ks. The Journal of Finance, 66(1), 35-6

https://papers.ssrn.com/sol3/papers.cfm?abstract_id=1331573

Relation: This paper explores the nuances of financial terminology and how it can be manipulated or misinterpreted, shedding light on the challenges of textual analysis in the financial domain. The work is highly relevant for our understanding of the linguistic subtleties in financial reporting.

Differentiation: In contrast to the paper by Loughran and McDonald, our research takes a different angle in the field of textual analysis within finance. While their work focuses on the challenges and complexities of financial terminology and reporting, our research places an emphasis on the practical application of textual analysis within financial modeling. We delve into the specific application aspect of textual analysis. This differentiation is important as it narrows down the scope of our investigation, allowing for a deeper dive into the intricacies of applying textual data to financial models.

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Bidirectional Encoder Representations from Transformers. arXiv preprint arXiv:1810.04805.

https://doi.org/10.48550/arXiv.1810.04805

Relation: The paper presents a groundbreaking model known as BERT, which stands for Bidirectional Encoder Representations from Transformers. BERT revolutionized the way researchers and practitioners approached various NLP tasks by pre-training a transformer-based neural network on large text corpora. BERT's bidirectional context and contextual embeddings have made it a pivotal milestone in NLP research, and its techniques have been widely adopted in numerous NLP applications.

Differentiation: In contrast to the paper, our research takes a more specific focus on the practical applications and fine-tuning of the BERT model. While the original paper introduces the model and its pre-training techniques, our work capitalizes on BERT's capabilities and explores its adaptability to specific NLP sentiment analysis. Our research goes beyond the model's introduction to demonstrate how BERT can be effectively employed and fine-tuned for particular tasks, thus providing valuable insights into the practical implementation of this transformative NLP technology.

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). FastText: Enriching Word Vectors with Subword Information. arXiv preprint arXiv:1607.04606.

https://doi.org/10.48550/arXiv.1607.04606

Relation: The paper introduces FastText, a novel approach for word embeddings. FastText differs from traditional word embeddings like Word2Vec by considering subword information, which allows it to represent words as combinations of character n-grams. This approach has gained widespread recognition in NLP for its ability to capture the morphological and semantic properties of words efficiently. Our research is related to this seminal work, as we build upon the concepts and techniques introduced in FastText to address specific challenges or applications in the field of NLP.

Differentiation: Our research takes a more focused approach by investigating the application and adaptation of FastText embeddings to specific NLP tasks or domains. While the foundational paper introduces the FastText model and its capability to enrich word vectors with subword information, our work delves deeper into the practical implementation and fine-tuning of FastText embeddings. Our research contributes by showcasing how FastText can be effectively harnessed for particular NLP challenges, demonstrating its versatility and utility in real-world applications.

Yang, Y., Uy, M. C. S., & Huang, A. (2020). Finbert: A pretrained language model for financial communications. arXiv preprint arXiv:2006.08097.

https://doi.org/10.48550/arXiv.2006.08097

Relation: The paper introduces Finbert, a pre-trained language model specifically designed to understand and analyze financial communications. Finbert is tailored to the unique linguistic characteristics and terminology used in the financial sector, making it a valuable resource for financial sentiment analysis, document classification, and other applications. Our research is related to this paper as it leverages Finbert's capabilities and may explore its use in specific financial NLP tasks.

Differentiation: In our study, we took a more specialized approach by applying and customizing the Finbert model for specific financial NLP tasks or domains. Unlike the foundational paper that primarily introduces the Finbert model and its adaptation for financial communications, we conducted comparative experiments with other models to showcase Finbert's effectiveness in these particular financial tasks and its potential to enhance decision-making and analysis in the financial sector.