École Polytechnique Fédérale de Lausanne

FIN-407 - Financial Econometrics

# Sentiment analysis on r/Wallstreetbets

Students :

- Matteo Ferrazzi
- Matteo Sansonetti
- Umberto Maria Ciceri
- Charles Gendreau

Professor: Prof. Elise Gourier
Teaching Assistant: Goutham Gopalakrishna

Academic Year 2022-2023

# Contents

# 1 Introduction

This research focuses on analyzing the sentiment of Wallstreetbets submissions and investigating the potential for constructing a high-frequency trading strategy based on the sentiment information. WallStreetBets is a subreddit, i.e. an online community on Reddit, that primarily focuses on discussing financial investments, trading, and market strategies. This subreddit stands out for its casual, humorous, and often provocative approach to investing. Users, which are mainly small retail investors, share their experiences, strategies, and opinions on stocks and financial markets.

The study involves scraping and analyzing the submissions of the last year about S&P500 firms. The sentiment analysis is conducted using three different models: TextBlob, VADER, and BERT. These models provide insights into the sentiment polarity of the submissions, ranging from positive to negative. The results show that the majority of submissions have a positive sentiment, and there is a high agreement between VADER and TextBlob in sentiment classification.

To explore the predictive power of the sentiment information we constructed a high-frequency trading strategy. The strategy leverages the sentiment scores derived from the sentiment analysis models to make trading decisions. The performance of the strategy is evaluated, and its profitability is assessed.

The findings indicate that there is some potential in using sentiment information from Wallstreetbets submissions for constructing a high-frequency trading strategy. Although, the strategies show results only when considering high-frequency trading (daily) which is exposed to high transaction costs. Therefore, it is not possible to assess certainly that they would be profitable. In addition, further research is needed to validate and refine the strategy, including using labeled datasets, optimizing hyperparameters, and addressing limitations such as data limitations and tokenization issues.

# 2 Data Extraction

The first thing we had to do was to obtain the data of our interest from the Reddit comments. We decided to use the PRAW library to scrape Wallstrett bets. Although, access to Reddit archives is very limited by the website. Indeed, we experienced many difficulties in scrapping multiple comments, and due to legal issues with the API it was not possible to access the archive of Reddit and get all the submissions. We were only able to have access to the ones that can be searched manually on the website through the search bar.

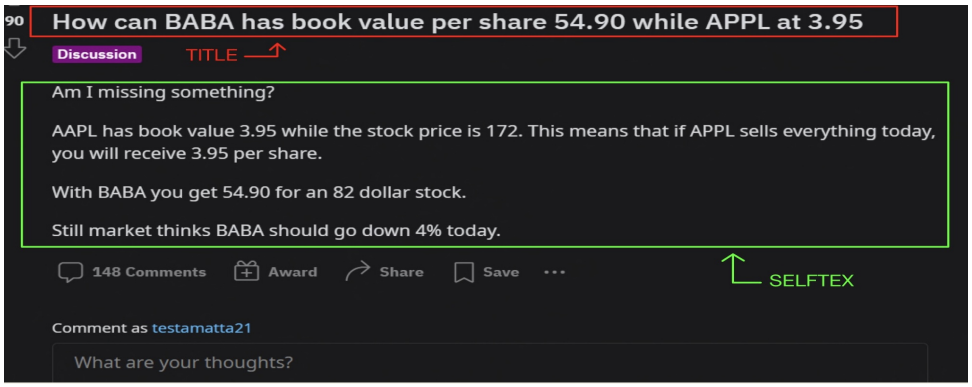



Figure 1: Example of a Reddit submission

With the PRAW library, we were able to search comments that were maximum one-year-old. The period we scrapped was 05/08/2022 - 05/08/2023. We decided to focus on the S&P500 stocks to have enough submissions since they are the most famous and interest-attracting companies. When searching for submissions related to a firm, in order to get all of them, we searched by both the company's name and its ticker. Sometimes, when the ticker was just a letter or a combination of letters like 'ON' or 'MMM', we had to search using only the full name, since by using the ticker we were finding random submissions containing those "words". From every post, we scrapped only the title and the so-called "selftext" (i.e. description). We decided to ignore the comments and the number of reactions and likes as they could have been published anywhere in time after the submissions. Since they could have been published after something happened in the market that at the time of the submission was unknown, we believe to be the information not valid for a project whose aim is trying to construct a reliable trading strategy.

## 2.1 Summary statistics

Overall, we scrapped 46701 submissions related to the S&P 500. The following table gives information on the comments we scraped. These statistics were obtained with the non-cleaned comments.

| Statistic | Value |
|---|---|
| Mean | 252 |
| Minimum | 0 |
| 25% Percentile | 7 |
| 50% Percentile | 70 |
| 75% Percentile | 310 |
| Maximum | 3756 |

Table 1: Summary statistics for the number of words in the submissions

It is possible to notice a significant difference between the average and the median. This is mainly due to two factors. Firstly, the presence of submissions with a very long selftext (the longest has 3756 words). Secondly, there were a lot of submissions from which we scrapped very few words. Indeed, a quarter of the comments had less than 7 words. These submissions probably had a title and an explanatory image,

such as a graph or a meme. This format is indeed common on Reddit and aligned with its humorous nature.

## 2.2 Distribution of submissions over time

We can see from the graphs below that submissions are not constant over time. We have very few submissions at the beginning of the period analyzed (this might be due to the fact that the old comments are always less relevant and hard to find on Reddit) while there are peaks when something happens in the market. Indeed, in August 2022 there was a drop in the S&P 500, and we can see a peak of submission corresponding to that in all the industries. [1] The second big increase in comments is in March 2023, when SVB collapsed. Indeed we can see, from the second graph that the increase was especially in the financial sector. Same story for April 2023 when the Credit Suisse acquisition happened.

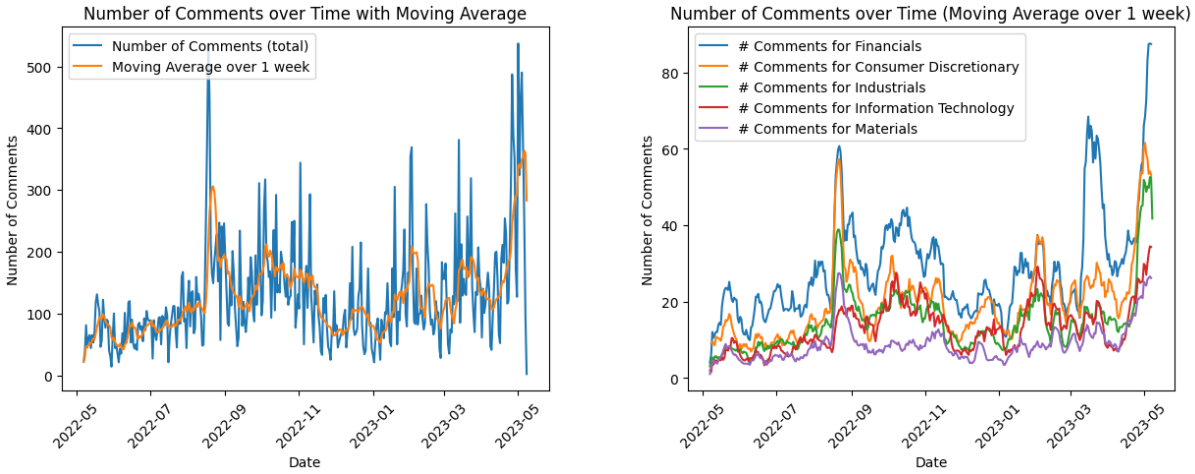



Figure 2: Number of Comments over Time

[1] In graph 2 we have plotted the number of comments for five of the industries in which the S&P 500 is divided. The GICS are: Energy, Materials, Industrials, Consumer Discretionary, Consumer Staples, Healthcare, Financials, Information Technology, Telecommunication Services, Utilities, Real Estate.

# 3 Sentiment analysis

According to NVIDIA Glossary, Sentiment Analysis is the automated interpretation and classification of emotions (usually positive, negative, or neutral) from textual data such as written reviews and social media posts [2]. This analysis involves Natural Language Processing (NLP) techniques and machine learning algorithms to extract and interpret the emotions expressed in the text. We evaluated three models: VADER, TextBlob, and BERT.

## 3.1 TextBlob

TextBlob is a Python library that provides a simple and intuitive interface for common natural language processing (NLP) tasks. It can be used for complex analysis and works with textual data. TextBlob comes equipped with a pre-trained sentiment analysis model that can classify text as positive, negative, or neutral based on its emotional content. We used the default algorithm Pattern Analyzer giving us the predetermined rules-based model. TextBlob is a Lexicon-based method. Lexicon-based methods use pre-defined sentiment lexicons or dictionaries that associate words with a specific sentiment score. When a sentence is passed into Textblob it gives two outputs, which are polarity and subjectivity. Polarity is the output that lies between [-1,1], where -1 refers to negative sentiment and +1 refers to positive sentiment. Subjectivity is the output that lies within [0,1] and refers to personal opinions and judgments. We also tried to use TextBlob with the Naive Bayes algorithm, but it did not give any relevant results. Naive Bayes is a probabilistic classifier that assumes independence between the features. Although, TextBlob may struggle with understanding and accurately analyzing complex language constructs, such as sarcasm, irony, or ambiguous sentences it provides an easy-to-use algorithm. It relies on rule-based heuristics and simple machine-learning algorithms, which may not effectively capture the intricacies of language, so its accuracy may not be on par with more advanced and specialized models. Furthermore, it is interesting to note that when a negation is found, the sentiment score is multiplied by -0.5. This explains why TextBlob gives scores very close to 0, as we will see later. [1]

## 3.2 VADER

VADER is a lexicon sentiment analysis method specifically designed for analyzing sentiment in social media and micro-blog texts. For this reason, it is able to deal with the unique characteristics of social media language, including slang, abbreviations, and emoticons. As TextBlob, it is a rule-based model with predetermined rules which gives a sentiment score from -1 to 1. Overall, VADER is an improved version of TextBlob: it takes into consideration the linguistic structure, the punctuation and capital letters (which is not the case with TextBlob). [2]

## 3.3 VADER and TextBlob

VADER and TextBlob have analogies, but also their own peculiarities. Since they are lexicon-based methods they are both easy to use. Moreover, the intensity in Text-Blob and the Compounded Score in VADER range between -1 and 1, and the score represents the polarity. They are models with predetermined rules [3], but VADER is an improved version of TextBlob analysis optimized for social media. Indeed, VADER relies on a predefined sentiment lexicon specifically designed for social media text, Text-Blob, which is trained on movie reviews, does not. In both of them, there is a sentiment lexicon orientation that classifies words with either a positive or negative sentiment (for instance "wonderful" has positive sentiment and "worst" has a negative one). Additionally, and as shown in Table 2 with the word "good", repeated words have an influence on the intensity of the VADER score while not (or almost not) on the TextBlob one. Moreover, VADER has a better overall understanding of the context. This is especially true when there are negations. As shown in Table 2, when there is a negation word next to a sentiment word, the score is multiplied by -0.5 [3]. However, when the negation word and the word with sentiment are separated by a neutral word (e.g. "that" in "not that good"), the negation has no

[2]Source: NVIDIA Glossary

[3]For TextBlob we used the default Pattern Analyzer algorithm

effect and the final sentiment of the sentence is wrong. This issue does not occur with VADER. Finally, VADER takes into account more punctuation signs. If they both take into consideration exclamation points, VADER also considers other types of punctuation such as ellipsis. This is quite useful as ellipsis are often a sign of sarcasm. In VADER, an ellipsis tends to give a neutral score when set after a word with positive or negative sentiment as seen in the last two lines of Table 2.

| Text | VADER | TextBlob |
|---|---|---|
| This movie was good | 0.44 | 0.70 |
| This movie was GOOD | 0.56 | 0.70 |
| This movie was GOOD!! | 0.64 | 1.00 |
| This movie was good, good, good | 0.83 | 0.70 |
| This movie was not good | -0.34 | -0.35 |
| This movie was not that good | -0.34 | 0.70 |
| I do not understand why people do not like this movie! | -0.34 | 0.00 |
| People say this movie is good... | 0.00 | 0.70 |
| People say this... movie is good | 0.00 | 0.70 |

Table 2: Examples of VADER & TextBlob Sentiment Scores

After conducting the sentiment analysis with VADER and TextBlob we obtained that only 25% of the submissions have negative sentiment and that TextBlob scores are almost all close to 0. In addition, VADER and TextBlob classified 69% of the submissions with the same sentiment[4].

| Statistic | Value |
|---|---|
| Mean | 0.05 |
| Standard Deviation | 0.16 |
| Minimum | -1.00 |
| Maximum | 1.00 |
| 25% Percentile | 0.00 |
| 50% Percentile | 0.04 |
| 75% Percentile | 0.11 |

Table 3: Summary Statistics for TextBlob Sentiment Scores

| Statistic | Value |
|---|---|
| Mean | 0.31 |
| Standard Deviation | 0.64 |
| Minimum | -1.00 |
| Maximum | 1.00 |
| 25% Percentile | 0.00 |
| 50% Percentile | 0.36 |
| 75% Percentile | 0.96 |

Table 4: Summary Statistics for VADER Sentiment Scores

We can clearly see how the mean, median, and standard deviation of VADER sentiment scores are significantly higher than the ones of TextBlob. We see from the tables that TextBlob sentiment scores are much closer to zero compared to the Vader scores. Indeed, for TextBlob scores, the mean is 0.05 with a standard deviation of 0.16, while for Vader the mean is 0.31 with a standard deviation of 0.64. TextBlob tends to give a neutral score compared to Vader. As mentioned above, when a negation is found, the TextBlob sentiment score is multiplied by -0.5. This explains why TextBlob has scored very close to 0. With both TextBlob and Vader, only 25% of the comments have a negative sentiment score.

[4]I.e. both positive or both negative

Figure 3: Text-Blob Scores

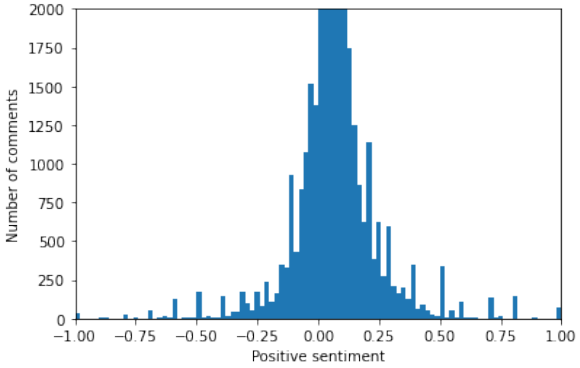



Figure 4: VADER Scores

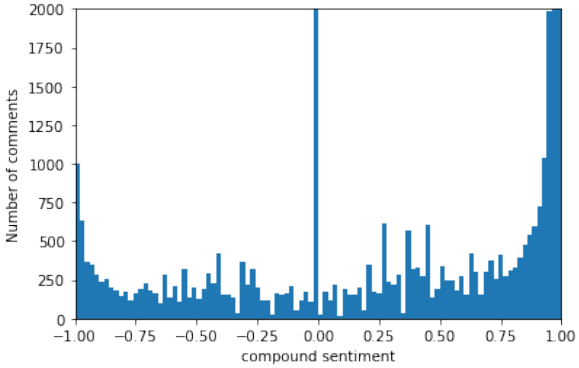



Figure 5: Stationary of time series

From the two graphs above, it's possible to see how VADER is able to give scores that are more evenly distributed than TextBlob. Since VADER doesn't have a penalty on negation, the scores it gives are more polarized, with a lot of extremely positive/negative scores and a lot of neutrals. On the other hand, the scores of TextBlob are compressed near zero, and very few submissions satisfy $|score| > 0.25$.
Lastly, we can observe a very low positive correlation. This is certainly due to the large difference in intensity between the two methods.

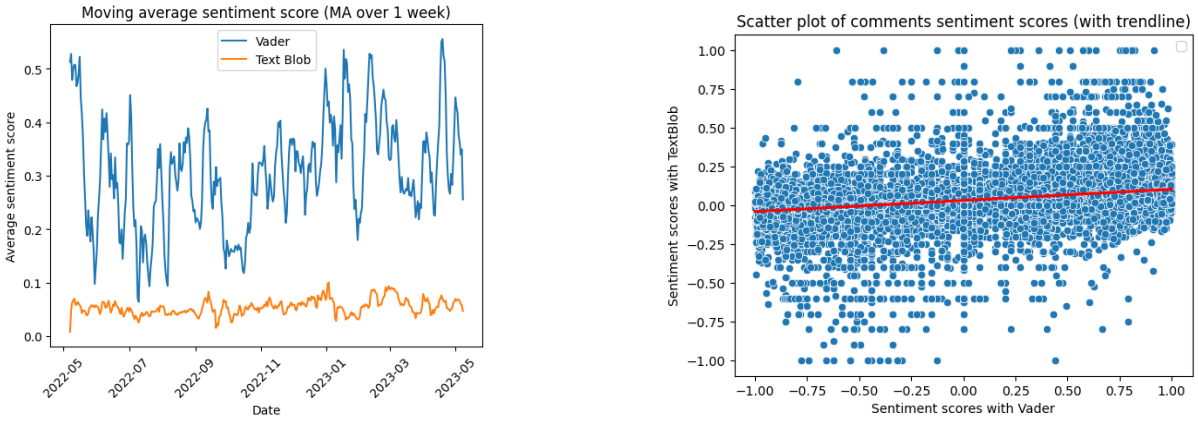



Figure 6: Comparison between VADER and TextBlob sentiment scores

The comments tend to have positive sentiment as shown by the figure on the left. The average score of VADER is more significant probably due to the fact that it gives more extreme scores.

## 3.4 BERT

BERT is an open-source machine learning framework to perform NLP. Being a Machine Learning model, in contrast to language models that process text in a left-to-right or right-to-left manner, BERT utilizes a transformer-based architecture that incorporates bidirectional context. This method allows one to better understand and decode the meaning of words, resulting in a deeper understanding of language semantics.
BERT relies on transformers which are a specific type of neural network architecture designed to process sequential data, for example, a natural language text. They are deep models, with many layers and nodes, however, they present 2 main parts:

- Encoder: starting from tokens, the encoder's aim is to generate contextualized representations for each word. In particular, the encoder consists of two parts:

  - The self-attention layer is up to contextualize and recognize patterns between words including itself, and then to compute a weighted representation that is transformed in a linear model to the following layers.
  - The feed-forward Neural allows for non-linear transformations of the representations generated by the self-attention layer. This refinement process gives the possibility to model to analyze more complex features in the data, enhancing a more precise representation.

- Decoder: starting from the output of the encoder, it predicts the output in the context, it also consists of three parts:
  - Self-attention which takes as input the output of the decoder and uses the same technique to recognize interdependencies among words
  - Encoder-decoder Attention which allows to dynamically decode the input received by the encoder
  - Feed-forward neural network which gives back the required output, in our case the polarity score.

The complex Neural network allows BERT to perform relational analysis, considering the context of each word rather than analyzing them independently as in previous algorithms which is essential in a natural language text. BERT is pre-trained on the entirety of the English Wikipedia and the Brown Corpus. We decided to use the most general form of BERT as our training rather than more specific dictionaries such as FinBERT WHICH is a pre-trained NLP model to analyze the sentiment of financial text because we wanted to have a broader approach, we wanted to incorporate all the possible news regarding the company, for example, high tech companies' stocks, can be influenced by announcement relative to new discoveries, hence we want to keep a broader vocabulary rather than focusing only on the financial lexicon.
During pre-training, BERT model learns to predict missing words or sentences within a given context. This process allows BERT to create a comprehensive internal representation of language and the associations between words, resulting in a deep understanding of linguistic nuances and connections. Indeed, one of the key strengths of BERT is its ability to capture contextualized word embeddings. Rather than representing each word with a fixed vector, BERT generates contextualized word embeddings that vary based on their context within a sentence. This contextual information enhances the model's understanding of word meaning and disambiguation. Word embeddings are a way to use an efficient, dense representation of a text in which similar words have a similar encoding. Each word embedding has its own unique method to assign values to the words.
To better understand how BERT works we had to build a pipeline. Building a pipeline for comment processing consists of a series of consecutive steps that are applied to the text to make them more machine understandable. To achieve this task we achieved the library SpaCy. Be careful that each word embedding is unique, hence Bert has its own type of word embeddings, however, we choose Spacy to have a general idea of the process works. The steps we followed are:

- Sentence splitting and tokenization: in SpaCy, tokenization is the process of splitting a text into individual units called tokens. Tokens can represent words, punctuation marks, or other meaningful elements in the text.
- Part of speech tagging: make words and expressions more recognizable for the computer
- Named entity recognition: model works better with noun-based phrases, hence it is essential to recognize names in the text.
- Removing stop words: stop words are words that do not add much meaning to the text, they can be noisy, especially because they are frequently repeated, hence is better to get rid of them.
- Lemmatization: a linguistic technique that aims to transform words into their base or dictionary forms, which are referred to as lemmas. Lemmatization allows for the normalization and grouping of related word forms by providing a standardized representation.
- Chunking (shallow parsing): group and label neighbouring words within a sentence, aiming to extract significant phrases or chunks.

- Dependency parsing: identifying the syntactic relationships or dependencies between words in a sentence. These dependencies are represented as directed links or arcs, which highlight how one word depends on or relates to another one.
- Counting word occurrences: it is important to have an idea about which words are the most popular

## 3.5 BERT implementation

We use the pre-trained model from BERT only for comments from the last 3 months. Implementation:

- Create BERT embeddings standardizing via padding and truncation
- Calculate the average embedding vector for each submission
- Tokenize the comments, each submission can't have more than 512 tokens
- Convert each comment into a BERT tensor representation
- Apply the BERT model
- Compute the score as the cosine similarity between the output of the BERT model and the average embedding vector

To notice that "def outp" makes a batch dimension, as many deep learning models, such as BERT, expect the input to be in a batch format. This is needed for several reasons:

- Parallel processing: Deep learning models often benefit from parallel processing, which becomes faster
- efficient vectorized computations
- regularization and generalization avoid overfitting

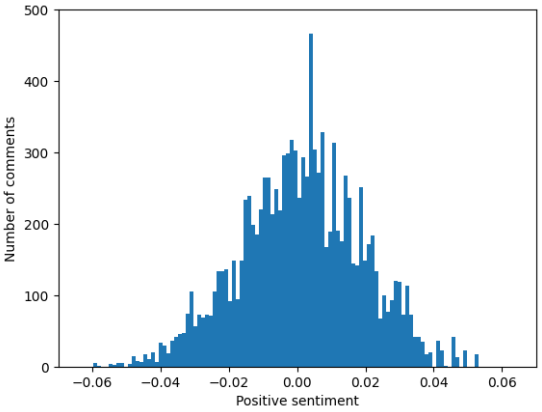



Figure 7: BERT scores

It is possible to notice how BERT gives a distribution similar to the one of Text-Blob, however, the long tails seem to give more weight to extreme values, which could be an improvement thanks to a more accurate tuning of the hyperparameters. [4] [5]

## 3.6 Possible improvements

Even if BERT appears to be a better model for our cause than the other two, we still have some limits and drawbacks such as having a no labelled dataset, limited tokenization and the use of the mean of embedding scores rather than all values. Indeed, BERT is a very complex model which is difficult to set because of the variety of parameters it has. Here there is some suggestions for improving the model:

- Use TF-IDF approach where TF measures the frequency of a term in a document while IDF measures how often a term across these different documents appears with while IDF measures how often a term across these different documents appears with the following formula: TF − IDF = TF(term, document) ∗ IDF(term)s
- Use a labelled data set that can better exploit the NN predictive power
- Better training of transformers' hyper-parameters
- RAM limitations caused a split of the data

# 4 Strategies

## 4.1 Rationale

After scrapping the submissions and doing the sentiment analysis on them, it's now time to try to see if it is possible to build a trading strategy based on Wallstreetbets posts. We have decided to start constructing a high-frequency trading strategy. Indeed, by reviewing the literature our attention was attracted by the article "Informational role of social media: Evidence from Twitter sentiment" by Chen Gu and Alexander Kurov. In their work, they found that it's possible to build a trading strategy only at high frequencies. High-frequency trading (HFT) is an important and prominent aspect of today's financial markets. Indeed, HFT firms account for a significant portion of trading volume in many major financial markets.
After constructing the HTF trading strategy, we tried to decrease the frequency and see if the results still hold. Furthermore, we also wanted to investigate whether or not there is a predictable power in Reddit submissions. Lastly, the performance of our strategy will also be a test for our score since with unlabeled data we can not test them.
One needs to be careful since if a strategy does not perform well this could be due to the score not being precise. But, on the other hand, it may be impossible to implement a strategy based on Reddit comments since they do not contain enough meaningful information. [6]

## 4.2 Data

We downloaded daily and weekly returns of S&P500 stocks. Then we shifted the date to the day/week before in order to use them with the submission of the previous day/week. In particular, Monday's return lagged to the Friday before since the stock market is closed during the weekend. We also took into account comments made over the weekend for next Monday return.
We then downloaded the weights of each stock in the S&P500. We started with a daily frequency and then we tried to increase it looking for the strategy that performs the best.
To compare the strategies we focus on their information ratio (IR). This ratio is used for evaluating the performance of an investment strategy compared to a benchmark or market index. It provides insights into the strategy's risk-adjusted returns. To calculate the information ratio, you divide the excess return of the investment strategy by its tracking error.
The IR expresses the strategy's risk-adjusted performance relative to the benchmark. It indicates how much excess return the strategy has generated per unit of risk taken (as measured by tracking error). It helps investors assess whether the strategy's returns are due to good decision-making or simply to higher exposure to risk. A higher information ratio suggests that the strategy has been successful in generating excess returns while managing risk effectively. In the context of high-frequency trading (HFT), the information ratio can be used to evaluate the performance of HFT strategies relative to market excess returns. HFT involves rapid trading and exploiting short-term market inefficiencies, so evaluating its performance based on risk-adjusted returns becomes crucial. To calculate the IR for our strategies, firstly, we regress their excess return over the market excess return. We decided to use the S&P500 as a market portfolio. Then, the IR is computed by dividing the $\alpha$ of the regression over the standard deviation of the residuals. We calculated the IR by doing a linear regression of our strategy over a constant and the excess return of the market. We computed this excess return as the difference between the return of the S&P500 and the risk-free. For the risk-free, we always used the 3-month T-Bills.

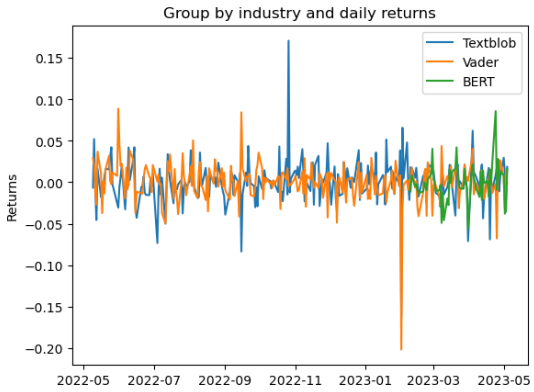
## 4.3 Group by industry and day

The first strategy we constructed is a daily frequency strategy with the stocks grouped by industry. Here is the construction:

- Group the stocks by the industry they belong to according to S&P500 with a daily frequency
- Compute the average score[5] for each industry in a given day
- Rank the industries based on their average score

[5]For all the scores used, i.e. TextBlob, VADER, BERT

- Create a long-short portfolio:
  - Long in the stocks of the industry with a higher average score
  - Short in the stocks of the industry with a lower average score
  - The weights of both portfolios are rescaled to sum up to 1




(a) Returns grouping by industry and day

| | |
|---|---|
| 1 | 0.2558 |
| 0.2558 | 1 |

(b) Correlation matrix between VADER and TextBlob

Figure 8: Portfolio performance and correlation matrix (Strategy 4.3)

| | Mean | Standard deviation | Information Ratio |
|---|---|---|---|
| VADER | 0.1045 | 0.3786 | 0.0774 |
| TextBlob | 0.2626 | 0.4215 | 0.5164 |
| Word embeddings + BERT | -0.1688 | 0.3530 | X |

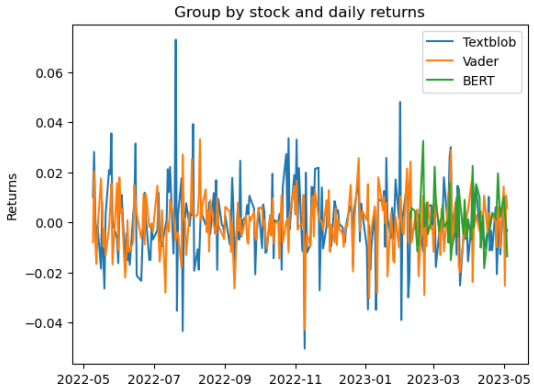
Table 5: Performance summary

It is possible to notice from the graph that there are two big spikes, one positive for TextBlob and one negative for VADER. These spikes explain the difference in the IR between VADER and TextBlob. Overall, TextBlob appears to have the highest average return and relatively higher risk-adjusted performance, as indicated by the information ratio. VADER also demonstrates positive average returns but with a lower risk-adjusted performance. However, the performance of Word embeddings + BERT is negative. This could be due to the fact that we considered BERT only in the last three months and, therefore, the strategy could lack robustness but it could be also due to the fact that we can not train the model to achieve better scores.
We also noticed that the two strategies based on VADER and TextBlob are positively correlated.

## 4.4 Group by stock and day

We then decided to consider single stocks.

- Compute the average score (for all the scores used) for each stock on a given day
- Rank the stocks based on their average score
- Create a long-short portfolio:
  - Long in stocks of the top decile (based on average score)
  - Short in stocks of the bottom decile (based on average score)
  - The weights of both portfolios are rescaled to sum up to 1




(a) Returns grouping by stock and day

| | |
|---|---|
| 1 | 0.1896 |
| 0.1896 | 1 |

(b) Correlation matrix between VADER and TextBlob

Figure 9: Portfolio performance and correlation matrix (Strategy 4.4)

| | Mean | Standard deviation | Information Ratio |
|---|---|---|---|
| VADER | -0.1191 | 0.1722 | X |
| TextBlob | 0.1202 | 0.2337 | 0.3159 |
| Word embeddings + BERT | 0.4728 | 0.1524 | 3.1271 |

Table 6: Performance summary

Here we see a big positive peak for TexBlob, and a negative one for VADER. TextBlob and BERT have positive average returns. Word embeddings + BERT has a higher average return, lower volatility, and so a significantly higher information ratio, indicating stronger risk-adjusted performance. However, also in this case we should be careful about the performance of the strategy based on Word embeddings + BERT scores since we have considered it only in a 3-months period. Again, we see that the strategies based on VADER and TextBlob are positive correlated.

## 4.5 Group by industry and week

We then considered a lower frequency to see if the results were still valid.

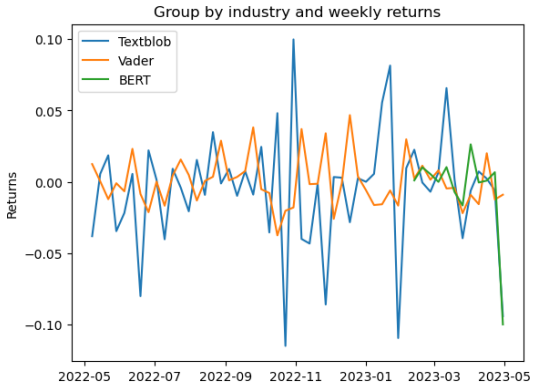
- Group the stocks by the industry they belong to according to S&P500 with a weekly frequency
- Compute the average score (for all the scores used) for each industry in a given week
- Rank the industries based on their average score

We then created a long-short portfolio:

- Long in the stocks of the industry with a higher average score
- Short in the stocks of the industry with a lower average score
- The weights of both portfolios are re-scaled to sum up to 1

| | Mean | Standard deviation | Information Ratio |
|---|---|---|---|
| VADER | 0.0031 | 0.1260 | X |
| TextBlob | -0.3124 | 0.3010 | X |
| Word embeddings + BERT | -0.2740 | 0.2275 | X |

Table 7: Performance summary




(a) Returns grouping by industry and week

| | |
|---|---|
| 1 | -0.1256 |
| -0.1256 | 1 |

(b) Correlation matrix between VADER and TextBlob

Figure 10: Portfolio performance and correlation matrix (Strategy 4.5)

In this case, we see that the strategy based on TextBlob and Word embeddings + BERT have a negative mean but also the mean of the strategy-based VADER is just a little bigger than 0 but not enough to provide a positive information ratio. However, the negative performance of Word embeddings + BERT could be due to the fact that we considered BERT only in the last three months Furthermore, now that we have decreased the frequency the strategies based on VADER and TextBlob are no longer positively correlated.

## 4.6 Analysis of all the S&P500 with daily returns

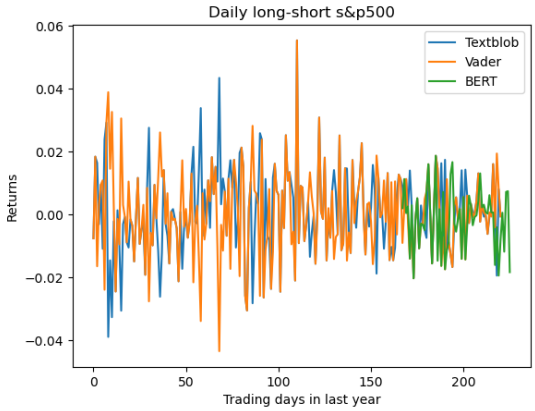
With this last strategy, we wanted to try to see if all the comments in a day can predict whether the S&P500 would have a positive return the day after. The main steps of the strategy are:

- Compute the average score (for all the scores used) over all the stocks in a given day
- Create a long short portfolio:
  - Long in the S&P500 and short the risk-free rate (3M T-bill) if the average score is higher than the mean of the average score over the year (we do not use average score higher/lower than 0 since it is higher almost all the day)
  - Short in the S&P500 and long the risk-free rate (3M T-bill) if the average score is lower than the mean of the average score over the year

| | Mean | Standard deviation | Information Ratio |
|---|---|---|---|
| VADER | 0.1399 | 0.2178 | 0.6471 |
| TextBlob | 0.2789 | 0.2173 | 1.3028 |
| Word embeddings + BERT | -0.4104 | 0.1508 | X |

Table 8: Comparison of the strategies returns

Figure 11: Portfolio performance and correlation matrix




(a) Returns on going long or short in whole S&P500

| | |
|---|---|
| 1 | 0.1600 |
| 0.1600 | 1 |

(b) Correlation matrix between VADER and TextBlob

TextBlob method appears to have the highest information ratio, indicating the strongest risk-adjusted performance among the three methods. Also, VADER shows a good performance. Also in this case, the performance of Word embeddings + BERT is negative and possibly due to the smaller period taken into consideration. We have that the strategies based on VADER and TextBlob are positively correlated.

## 4.7 Issues and possible improvements

High-frequency trading often involves the frequent rebalancing of portfolio weights, which can result in high transaction costs. These costs can eat into the overall performance of the strategy. Furthermore, when analyzing sentiment from social media platforms like Reddit, there may be periods with too few comments, a problem that we experienced due to the scarcity of submissions during the first part of the period we analyzed. This scarcity of data can limit the effectiveness of the strategy and make it challenging to draw meaningful conclusions.

However, one potential solution to improve the strategy is to increase the number of comments used for analysis. A larger sample size of comments can provide a more comprehensive view of sentiment and potentially lead to a more accurate strategy. In particular, focusing on individual stocks rather than broad industry sentiments may offer more specific insights and potentially enhance the strategy's performance especially if it is possible to have quite a lot of comments for each stock in order to guarantee a robust strategy. It is important to note that while a strategy that considers industry sentiments may provide a more robust approach, incorporating individual stock sentiments can offer a deeper understanding of market dynamics. By considering both approaches, investors can potentially strike a balance between robustness and accuracy in their sentiment-based trading strategy.

In our opinion, finding the optimal balance between frequency, transaction costs, and data availability is crucial in designing a successful sentiment-based trading strategy. While high-frequency trading may offer the advantage of capturing short-term market inefficiencies, it is essential to carefully evaluate the impact of transaction costs and ensure that they do not erode potential profits. Although the problem here is that decreasing the frequency leads to a reduction in Reddit submission's predictive power. Moreover, incorporating a sufficient number of comments and considering individual stock sentiments can provide a more detailed and accurate picture of market sentiment, leading to more informed investment decisions. Striking the right balance between industry-level and stock-specific sentiments can offer a well-rounded strategy that combines robustness and accuracy. Ultimately, the effectiveness of a sentiment-based trading strategy relies on thorough analysis, continuous refinement, and adaptation to market conditions. It is important to monitor performance, evaluate the impact of transaction costs, and adjust the strategy as needed to maximize the potential benefits of sentiment analysis in trading decisions.

# 5 Conclusions

Based on our analysis, we have observed some interesting findings regarding the relationship between sentiment of comments and stock returns. It appears that there is potential predictability in returns based on the sentiment of comments, although only with a higher frequency of trading. This suggests that sentiment analysis can offer insights into market dynamics, even though it might be unexpected considering that the majority of Reddit users are likely uninformed retail investors.

Despite the presence of uninformed users, comments still exhibit some level of explanatory power in relation to stock returns. This finding highlights the potential usefulness of sentiment analysis as a tool for gaining an informational edge in the market. It suggests that even though the majority of Reddit users may not possess expert knowledge, the collective sentiment expressed in their comments can still provide valuable insights for trading strategies. Moreover, strategies that aggregate sentiment across different stocks appear to be more robust compared to focusing solely on individual stocks. By considering sentiment at an industry or market level, the strategy can capture broader market trends and potentially reduce the impact of idiosyncratic factors. This diversification of sentiment signals can enhance the overall performance and reliability of the strategy.

We also had problems in using more sophisticated algorithms for sentiment analysis. One notable issue is the unavailability of labeled data for training the model. The absence of labeled data can limit the algorithm's ability to learn and accurately classify sentiments. Additionally, the use of a shorter time period on which we used the model may further impact its performance and effectiveness. Furthermore, it would be essential to continuously evaluate and refine the sentiment-based trading strategy, taking into account changes in market conditions, data availability, and the evolving landscape of social media platforms. By leveraging sentiment analysis alongside other fundamental and technical factors, investors can potentially gain a more comprehensive understanding of market sentiment and make more informed investment decisions.

Overall, while sentiment analysis has shown promise in explaining stock returns, it is essential to approach it with caution, acknowledging the limitations and complexities involved. Combining sentiment analysis with other analytical tools and a rigorous risk management framework can help maximize the potential benefits and navigate the challenges associated with incorporating sentiment into trading strategies.

# References


[1] Matt Loria and contributors. Textblob 0.15.3 documentation, 2021.

[2] C. Gilbert and C.J. Hutto. Vader: A parsimonious rule-based model for sentiment analysis of social media text. *Proceedings of the 8th International Conference on Weblogs and Social Media*, 2014.

[3] Anuj Sharma. Sentiment analysis with textblob and vader, 2021.

[4] Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. Bert: Pre-trained models for natural language processing, 2021.

[5] Elise Gourier. EPFL FIN-407 Financial Econometrics Lecture Notes Spring 2023.

[6] Eugene F. Fama and Kenneth R. French. A five-factor asset pricing model. *SSRN Electronic Journal*, 2018.