BACHELOR'S DEGREE THESIS

Degree in Statistics

Title: Search Engines in the use of Financial Sentiment Analysis

Author: Ramon Coroando Montoro

Advisor: Salvador Torra I Porras

Department: Econometria, Estadística i Economia Aplicada

Academic year: 2023/2024

UNIVERSITAT DE BARCELONA

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat de Matemàtiques i Estadística

This thesis is submitted to the Applied Mathematics Department, Universitat Politècnica de Catalunya in fulfilment of the requirements for the degrees in Statistics and Economics.

Ramon Coronado Montoro, June 2024

Acknowledgements

I would like to extend my sincere gratitude to Dr. Salvador Torra for his invaluable guidance and support throughout this research endeavor. His expertise has been instrumental in shaping both the technical and conceptual aspects of this work.

I also wish to acknowledge the significant influence of two notable figures in the field of quantitative finance: Ivan Atienza known for his insightful contributions to the dissemination of financial knowledge, and Marcos López de Prado, a leading quantitative and renowned professor at Cornell University. In today's age of readily available information, thanks to open access, learning from experts like Ivan and Marcos has become more accessible than ever. Their insights have provided me with invaluable perspectives and deepened my understanding of quantitative finance.

Furthermore, I am grateful for the unwavering support of my friends and family, whose encouragement has been a constant

Abstract

Ramon Coronado Montoro

Search Engines in the use of financial sentiemnt analysis

Financial market predictions often rely on historical and numerical data, but recent advancements in large language models encourage the use of alternative datasets like financial news text. However, this methodology often faces limitations due to the scarcity of extensive datasets that combine both quantitative and qualitative sentiment analyses. To address this gap, we used the Bing Search API to build a dataset comprising over 100.000 financial news articles from more than 90 websites. Our work aims to illuminate the process of building a dataset using search engines, demonstrating that the use of keywords to collect "custom" data from the vast Internet is an effective alternative for data collection. We evaluated the dataset using a sentiment index, which we later compared with the S&P 500 stock index. We concluded that while news sentiment may not immediately reflect price variations, it can effectively indicate broader market trends.

Key words: Big data, search engine, web scraping, sentiment analysis, Standard and Poor's 500 index.

AMS classification: 91-00 - General reference works (handbooks, dictionaries, bibliographies, etc.) pertaining to game theory, economics, and finance.

Contents

List of Figures

List of Abbreviations

ABSA	Aspect-Based Sentiment Analysis
API	Aplication Programming Interface
DNS	Domain Name System
DOM	Document Object Model
FSA	Financial Sentiment Analysis
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
IP	Internet Protocol address
JSON	JavaScript Object Notation
NLP	Natural Language Processing
REST	Representational State Transfer
SA	Sentiment Analysis
SE	Search Engine
SEO	Search Engine Optimization
SERPs	Search Engine Result Pages
TCP	Transmission Control Protocol
ToS	Terms of Service
URI	Uniform Resource Identifier
URL	Uniform Resource Locators

URN	Uniform Resource Names
VADER	Valence Aware Dictionary and Sentiment Reasoner
VPN	Virtual private network
W3C	World Wide Web Consortium
WWW	World Wide Web

Chapter 1

Big Data

Data, as defined by Merriam-Webster Dictionary[1], *"encompasses actual information such as measurements or statistics, utilized as a foundation for reasoning, discussion, or calculation"*. Remarkably, the concept of data predates the digital era, with ancient civilizations like the Sumerians recording information on cuneiform tablets, documenting livestock and property ownership over five thousand years ago[2].

Although data is commonly associated with numbers, especially in the financial environment, its richness is not limited to numerical formats. Data can also be found in other forms such as texts, where new digital forms of communication—websites, blog posts, tweets—differ significantly from traditional sources. Notably, the growth of digital data far surpasses its non-digital counterpart as data permeates all disciplines. The resources dedicated to supporting the study of philology, texts, dictionaries, and quotas, among others, have increased. This includes the availability of open software and resources that facilitate these studies.

For example, we will later use the Loughran-McDonald Master Dictionary with Sentiment Word Lists, derived from release 4.0 of the 2of12inf baseline dictionary, now including words from 10-K documents and earnings calls not found in the original 2of12inf list. The 2of12inf dictionary comes from the free project SCOWL (Spell Checker Oriented Word Lists) and Friends[3]. The database includes word frequency, spelling differences between English dialects, spelling variants, and basic part-of-speech and inflection information.

[1] Merriam-Webster Dictionary: *Data*

[2] The Origins of 'Big Data': An Etymological Detective Story by Steve Lohr

[3] SCOWL (Spell Checker Oriented Word Lists) and Friends is a database of information on English words useful for creating high-quality word lists: WordlList.net

The widespread adoption of the internet has made it accessible to around 5 billion people worldwide, with nearly 4.6 billion using it monthly. This connectivity has led to an unprecedented surge in data traffic, reaching over 588 exabytes in 2020, according to Ericsson's Mobile Data Traffic Outlook. To put this in perspective, all the information in US academic research libraries totals about 2 petabytes. Estimating the global volume of data requires breaking it down into smaller increments. Each day, the world generates about 2.5 quintillion bytes, or 1,000 petabytes. In 2020, the global datasphere reached 64 zettabytes, with IoT data being the fastest-growing segment, followed by social media data. The enterprise datasphere is expected to grow twice as fast as the consumer environment, driven by the increasing role of cloud technology. Therefore, 64 zettabytes is a conservative estimate, representing the lower bound of the world's data volume.[4]

This exponential surge underscores the necessity of focusing on digital data in the contemporary paradigm. Notably, the top-ranking companies by market capitalization, such as Google, Apple, Meta (formerly Facebook), and Nvidia, wield considerable influence in shaping and expanding internet services and data topics.

The term "Big Data" does not have a precise scientific origin. Initially, it was used to describe various data types handled in new ways, not just large volumes of data. During the 1990s, John Mashey mentioned that it captures the evolving scope of computing. As a tool, has now permeated every ecosystem, with organizations endeavoring to utilize as much data as possible for machine learning, predictive modeling, and advanced analytics. On platforms like Google Cloud Analytics, companies observe consumer behavior to offer personalized retail recommendations, integrate data to optimize last-mile delivery strategies, and harness AI-driven technologies to analyze unstructured medical data, advancing treatment protocols and enhancing patient care.

The three fundamental characteristics of big data, often referred to as the "3 Vs of big data," are volume, velocity, and variety, as originally defined by Gartner in 2001. [5]

• Volume: This characteristic primarily emphasizes the vast amount of data generated from various sources and devices continuously. Big data is characterized by its sheer volume, reflecting the immense scale of information available for collection and analysis.

[4]Source: Big data statistics: How much data is there in the world? Kevin Bartley,texRivery.iot
[5]Source: What is Big Data? Oracle 2024

• Velocity: Big data velocity signifies the speed at which data is produced. In contemporary times, data is frequently generated in real-time or near-realtime, necessitating rapid processing, access, and analysis to derive meaning

• Variety: Data exhibits heterogeneity, stemming from its diverse origins and formats. While traditional structured data, such as that found in spreadsheets or relational databases, remains prevalent, it is now complemented by unstructured data like text, images, audio, and video files, as well as semi-structured formats such as sensor data. This variety underscores the multifaceted nature of big data, challenging conventional data management approaches and necessitating adaptable strategies for organization and analysis. We will see more in the chapter 3.1.1.2 on the use of unstructured data in machine communication.

In recent years, two additional "Vs" have surfaced: value and veracity, signaling the risks that have accompanied the proliferation of data. While data inherently possesses value, its effectiveness hinges on our capacity to uncover and evaluate its accuracy and reliability. Extracting meaningful insights from big data necessitates a thorough discovery process beyond mere analysis, asking relevant questions, discern patterns, make informed assumptions to avoid erroneous conclusions. Moreover, society has been inundating with copious amounts of information that can sometimes blur the distinction between truth and falsehood. This deluge of data, coupled with the anonymity provided by the internet, presents both opportunities and challenges. Despite the significant value that big data offers, concerns regarding its authenticity and associated risks cannot be disregarded.

1.1 Big data challenges

1.1.1 *The elephant in the room*

Data storage and processing alone isn't enough; the value of data lies in its curation. This involves cleaning and organizing data to enable meaningful analysis, a process that consumes 50 to 80 percent Figure 1.1 of a data scientist's time. Extracting value from big data entails managing exponentially growing volumes of data. Poor data quality, termed the "dark side" of big data, hampers insight discovery, with only a third of business leaders trusting the information they use for decision-making Sadiq and Papotti (2016).

This crucial aspect of data processing is typically addressed during the Exploratory Data Analysis (EDA). EDA is paramount because it informs the decisions that

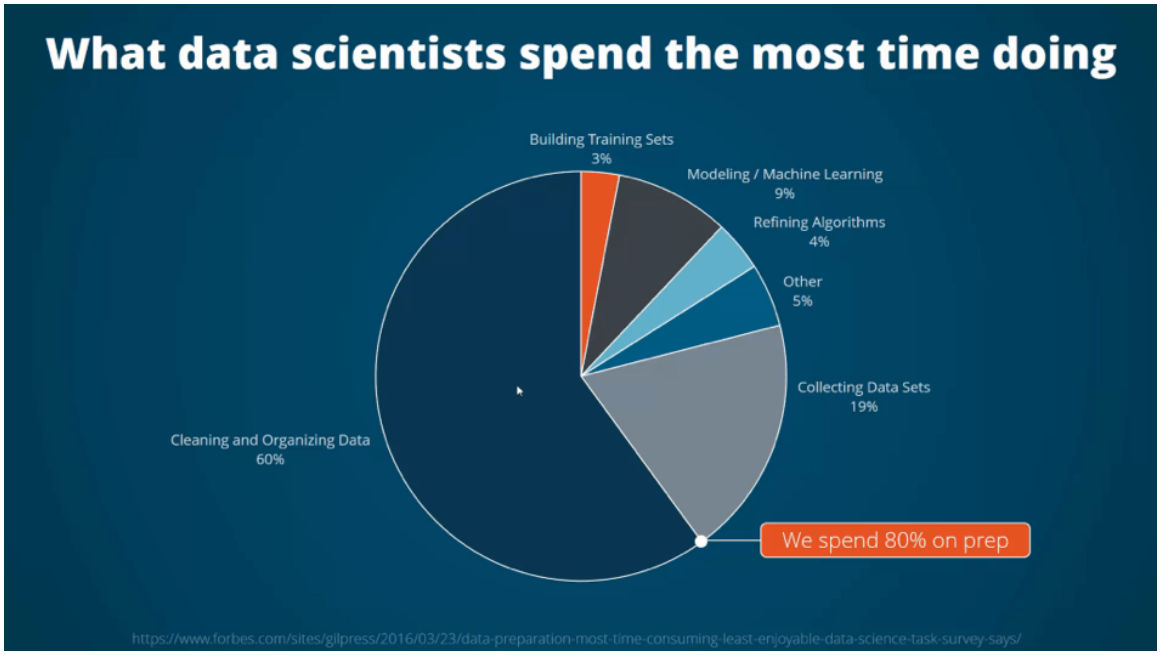

Figure 1.1: Time spent in a Data Science project *Source:* Medium

shape the results, beyond the choice of model. Decisions such ass whether to eliminate outliers, replace missing values with the mean or mode, eliminate correlated variables, grouping the dataset or treating temporal data are essential. The complexity of corner data cases complicates analysis, exacerbated by the absence of a definitive guide or reference manual for navigating such scenarios. Big Data projects diverge from traditional statistical analyses because statistical analyses are often conducted in controlled laboratory settings of data collection and sampling,while large data analysis transcends mere science, demanding a blend of scientific rigor and artistic finesse. During this project, there are hints about things we could explore more, but we've made sure to explain why we made each decision.

1.1.2 Overfitting in finance

Data curation in finance, particularly for constructing investment portfolios, demands heightened sensitivity. Within an environment where the adage "Past returns do not guarantee future returns" [6] holds his way, a dedicated science exists for evaluating the feasibility of deploying strategies based on historical data, known as backtesting. Overfitting, wherein a model fits the training data too closely, risking poor performance on unseen data, poses a significant challenge,

[6]Fundsmith Equity Fund slogan, one of the most successful value investing equity founds to date.

especially in contexts where temporary leakage is difficult to eradicate despite attempts at control Bailey et al. (2015).

In mathematical finance, a ”backtest” evaluates a trading strategy's performance using historical market data. Modern computer systems can efficiently explore numerous strategy variations, selecting the best performer based on the input dataset. However, this ”optimal” strategy frequently falters when applied to new datasets, a consequence of overfitting the strategy's parameters to the input data, termed ”backtest overfitting”Bailey et al. (2014).

The U.S. Securities and Exchange Commission found hedge funds using tricky tactics of “selection bias.” to make their performance look better than it is. Meanwhile, Staff (2015) discovered lots of examples where financial studies didn't properly test their findings, leading them to believe many claims in finance might be wrong. This leads these authors to conclude that “most claimed research findings in financial economics are likely false”. Furthermore, in the context of this project, which aims to describe new techniques for obtaining news data and formulating sentiment indices, it's crucial to acknowledge that the conclusions drawn are not prospective, as there's no intent to predict market indices. Rather, the focus lies on comparative analysis between metrics formulated, emphasizing the importance of discerning between correlation and causation in interpreting findings.

1.1.3 Sentiment data in finance

Economic data has improved greatly in recent years, offering detailed insights from both public and private sources. Modern techniques like machine learning can identify complex relationships within this data. In finance, using extra information about competitors can yield significant returns and so, financial firms are eager to develop computer-assisted decision-making. Still, the complexity of financial systems often exceeds the modeling capability of these traditional quantitative methods. Moreover, interesting datasets like satellite images and news articles are difficult to analyze using conventional econometrics López de Prado (2019).

To better understand the types of datasets used in economic applications, we can classify them into three categories:

- **Data-Rich Applications**: These applications benefit from having an abundance of data. Examples include sentiment analysis, credit ratings, and Big

Data applications, which often involve large datasets with millions of examples.

- **Experimental Applications**: These allow researchers to conduct randomized controlled experiments to establish causal mechanisms. Examples include execution and sentiment extraction, where we might reword a news article and compare a machine learning model's prediction with a human's conclusion or experiment with different execution algorithms to observe market reactions.

- **Data-Free Applications**: These applications do not require any data. Examples include risk analysis, portfolio construction, outlier detection, feature importance, and bet sizing methods, which are developed based on appealing mathematical properties.

In this project, we focus on experimental applications, specifically mining controlled financial news using search engines for textual sentiment analysis. The analyzed texts typically fall within these categories:

- **Corporate Sentiment**: Sentiment expressed by corporations through public filings (10-Ks, 10-Qs), press releases, or conference calls.

- **Media Sentiment**: This involves sentiment expressed by the media, such as news articles or analyst reports.

- P**ublic Sentiment**: This encompasses sentiment expressed by the general public through internet blogs or social media.

1.1.4 Textual Datasets in Finance

Recent advancements in large language models highlight the importance of integrating sentiment data and numerical factors for financial analysis. However, this approach often faces limitations due to the lack of comprehensive datasets that combine both quantitative and qualitative sentiment data. Since previous works have demonstrated that sentiment analysis's accuracy for various ML models is highly dependent on the amount of training data and the quality of the training data Ibrahim and Yusoff (2017) there is a real need to continue expanding the literature on textual data sets and explore the best ways to contribute to the ecosystem. To address this gap, the financial dataset landscape is evolving, with several works

have developed financial news datasets using diverse sources and methodologies, summarized in 1.1.

Table 1.1: Financial News existing Datasets. *Source:* Dong et al. (2024)

Name	FNSPID	Reuters	Benzinga	Bloomberg	Lenta	Lutz's	Farimani's	SemEval	SEntFiN 1.0
Time Stamp	Yes	Yes	Yes	Yes	Yes	No	No	No	No
Text Type	Article	Article	Article	Article	Article	Sentence	Sentence	Headline	Headline
Number of News	15698563	8556324	3252885	447341	800974	1000	21867	1142	10753
Symbol	Yes	No	Yes	No	No	No	No	No	No
Summarization	Yes	No	No	No	No	No	Yes	No	No
Sentiment Score	Integer	-	-	-	-	Integer	-	Real	Integer
URL	Yes	No	Yes	No	No	No	No	No	No
Language	Many	Eng	Eng	Eng	Ru	Eng	Eng	Eng	Eng
Stock Price	Yes	No	No	No	No	No	Yes	No	No

The table compares various financial news datasets, highlighting differences in attributes such as the type of text (article or sentence), the number of news items, language, and the presence of additional features like summarization, sentiment score, and stock prices. Each dataset serves different approaches or purposes, making direct comparisons not appropriated at all. However, it is clear that FNSPID offers the most comprehensive features, including many news articles, multiple languages, and additional attributes such as stock prices and sentiment scores.

Philippe's dataset, sourced from Bloomberg and Reuters, offers a large collection of financial news time series for analysis. Yutkin's dataset, which includes news from Lenta and contributions from sources like Benzinga. Lutz provides a dataset that categorize financial news as positive or negative, along with textual representations. Farimani introduced a dataset that combines latent economic concepts, news sentiment, and technical indicators, where all the data is provided in time series. Cortis provided a dataset for fine-grained sentiment analysis of financial microblogs and news, including sentiment scores and lexical/semantic features. Sinha et al.'s SEntFiN 1.0 dataset, notable for its entity-sentiment annotations and extensive database of financial entities. Lastly, the most recent a robust dataset, FNSPID is a comprehensive resource encompassing financial news in Englishfrom 1999 to 2023 with more than 20 GB of news data. The dataset exclusively sources information from trusted financial news platforms like NASDAQ. Additionally, the researches prepared a programming tool that allow us to scrape today's data from NASDAQ, further enhancing the dataset's utility and longevity.

As shown in Table 1.2, previous research on Financial Sentiment Analysis (FSA) has explored a variety of data sources, including news, blogs, tweets, and company disclosures.

Table 1.2: Main Data Types in FSA

Data Type	**Length**	**Objectiveness**	**Frequency**
News	Variable	Variable: according to the source	High
Corporate disclosures	Long	Subjective	Low
Social Media	Variable	Subjective	High

Despite advancements, the quest for robust datasets remains v bastion of exploration, especially as ML models become more standardized and unique data becomes scarcer. When building such type of data resource, we have to consider the following:

- Lack of detailed company financials: The dataset omits detailed company financials, potentially limiting in-depth analysis for users requiring company-specific financial data.
- Absence of entities for target sentiment analysis: Potentially impacting sentiment analysis forecasting accuracy by limiting granularity in analyzing sentiment towards specific entities or companies.
- Limited news volume and proprietary sentiment scoring: The dataset's limited news volume and proprietary sentiment scoring raise concerns about its reliability and accuracy.
- Absence of timestamps: Pose challenges in aligning sentiment data with price data.
- Reliance on short headlines: Providing lack context for accurate sentiment analysis, potentially limiting the dataset's effectiveness.
- Data quality concerns: Source data from trusted sources to address concerns about fake news.

Analyzed the current news data landscape and considering that many previously open resources have been restricted from public access, we prompted exploration into search engines, leveraging the trust and reputation of platforms like Bing and Yahoo to augment our news data sources. Furthermore, during dataset formation, decisions were made that persistently fail to address certain shortcomings observed in previous datasets. This path remains open for further exploration in new research endeavors.

Chapter 2

Sentiment Analysis

2.1 Introduction

"Within our mandate, the ECB is ready to do whatever it takes to preserve the euro. And believe me, it will be enough." - Mario Draghi, 2012 [1]

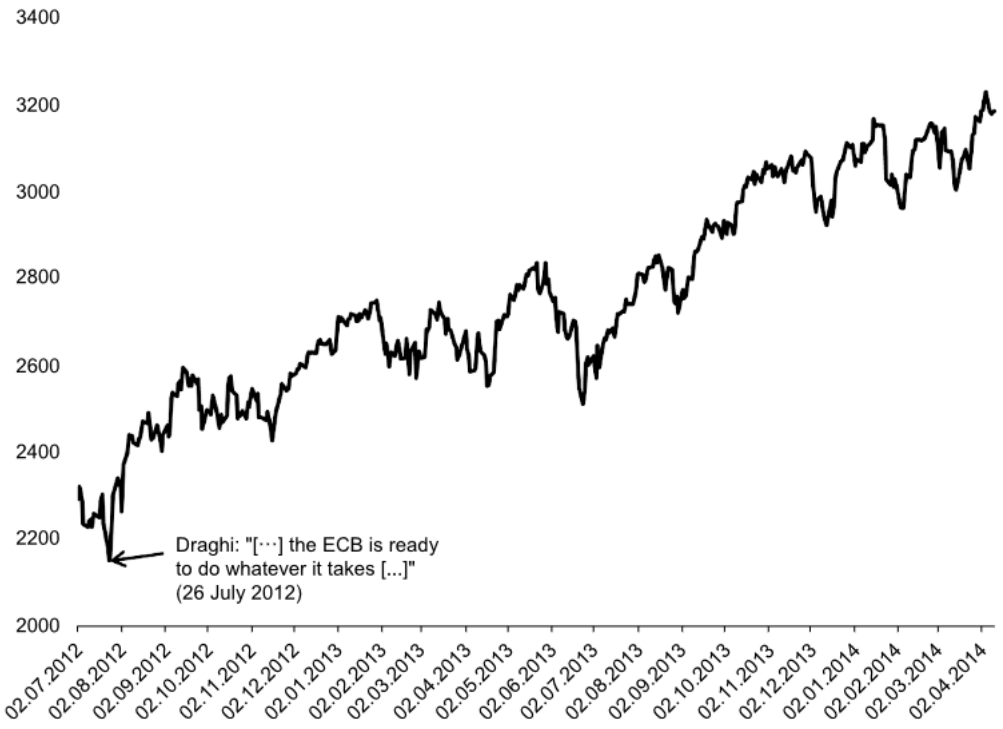

Figure 2.1: The EuroStoxx after Draghis Speech on July 26, 2012

In 2012, when the Euro teetered on the brink of bankruptcy, Mario Draghi's resolute words brought confidence to the financial markets, as seen in figure 2.1. Just as

[1] Speech by Mario Draghi, President of the European Central Bankat at the Global Investment Conference in London, July 2012 *ECB*

Draghi vowed to save the Euro "whatever it takes," sentiment analysis in financial news decodes the emotions driving market reactions. By capturing the market's mood—whether fear, greed, or relief—these insights become valuable data on retrieving expectations Uhl et al. (2015). On this chapter we will review the concept of sentiment expressed in text, techniques to analyze it and its impact on markets through financial news.

Sentiment analysis (SA) is a method used to extract emotions or opinions from text on a specific topic, allowing us to understand the attitudes, opinions, and emotions expressed. This study analyzes textual data, but sentiment is not limited to text; it can also be applied to audio, images, videos and as reviewed in 1.1.3, the grown in this alternative data make sentiment analysis an important field of research nowdays.

Sentiment analysis can be applied in many **fields**, such as s healthcare, crime, finance, government intelligence (social media), and in academia (Jain et al. (2021)) among others. Most research articles focus on classifying the sentiment of text as positive, neutral, or negative but we can also find ranking the attitude of the text or detect the target, the entity and the type of the actitude. Sinha et al. (2022) remarked that most financial news datasets are only effective when a single entity is mentioned in the headlines. When multiple entities are involved, capturing their interactions and sentiment expressions is necessary for effective sentiment extraction. In financial news, these entities are typically companies or organizations mentioned in the text.

SA models are built to perform various **tasks**, including opinion mining [2], sentiment classification, and opinion summarization (field of research in natural language processing (NLP)). Although we could extend futher the complexity of the analysis of financial news, on this project we limit our scope on polarity classification.

In analyzing sentiment, the message is broken down into three **attributes**: the holder, the target, and the type of attitude. The holder is the source of sentiment, the target is the object towards which the sentiment is directed and the type of attitude refers to the nature of the sentiment, such as love, hate, positive, or negative. For example, in the phrase "John loves the new movie," John is the holder, the new movie is the target and the type of attitude is positive, specifically love.

[2]Opinion mining and SA usually are considered synonyms across research standards. Still, Microsoft Azure AI makes the sutil difference where Opinion mining refers to automatically extract granular information about the opinions while SA refers to mere classification based on sentiment of polarity.

The analysis scope of SA can be primarily divided into three levels based on the text inputs **types of text**: document, sentence, and aspect levels Behdenna et al. (2018).

- **Document Level Analysis:** This level focuses on determining the overall opinion of a document, treating each document as an independent object with a single sentiment polarity, making it coarse-grained. It assumes the document discusses one topic and expresses opinions on a single entity, thus not suitable for documents evaluating multiple entities. Wen et al. (2020) proposed a speculative SA model, suggesting that similar reviews are likely written by users with similar sentiments, utilizing similar documents to improve SA accuracy.

- **Sentence Level Analysis:** At this level, the task is to determine whether each sentence expresses a positive, negative, or neutral opinion. It involves subjectivity classification, distinguishing between factual (objective) and opinion-based (subjective) sentences. The process includes two steps: identifying if the sentence carries an opinion and assessing if it is positive or negative. A key challenge is that objective sentences can sometimes carry an opinion, although they typically do not convey any opinion.

- **Aspect Level Analysis:** This analysis is based on identifying the polarity (positive or negative) and the target of the opinion. It involves two main steps: identifying the entity and its aspects, and then evaluating the opinion on each aspect. Aspect-level SA is finer-grained and models the relationship among the aspect term, aspect category, opinion term, and sentiment polarity Yang et al. (2018). For instance, in the sentence "This restaurant's steak is delicious," "steak" is an aspect term under the category "food," "delicious" is the opinion term, and the sentiment polarity is positive. This model, fitting into aspect-based sentiment analysis (ABSA), recognizes entities and extracts sentiments related to them.

This distinction is important since it influence the flow in which the data will be collected, its form and type. Given the available data 5 and the vast number of terms and entities associated with "SP500," establishing a comprehensive dictionary of entities is challenging. Therefore, we will focus on classifying sentiment at the sentence level. With this approach, this study aims to capture the overall trend of market sentiment, even if it means losing some precision.

We can break down the sentence into three attributes in we might associate sentiment: the holder, the target, and the type of attitude. The holder is the source of sentiment in the message. The target is the object towards which the sentiment is directed. The type of attitude refers to the nature of the sentiment, such as love, hate, positive, or negative.

2.2 Literatirue review Sentiment Analysis

Different models and techniques have emerged to analyze textual sentiment. The Sentiment Analysis Baseline Algorithm, developed by Pang et al. (2008), is considered foundational in the field of SA modeling and has set a precedent for future studies. This algorithm comprises three main **steps**:

- **Tokenization:** In this phase, the message is segmented for proper analysis. The data collected from websites for sentiment analysis contains HTML and XML markup, stop words, capitalization, and numbers, all of which should be removed during this phase.
- **Feature Extraction:** When the input data is too large to be processed, transforming it into a set of features is called feature extraction.[3]
- **Classification:** Before obtaining cured text, it is time to retrieve sentiment. Below is described the classification techniques landscape.

SA techniques can be broadly categorized into four main approaches: lexicon-based, traditional machine-learning, deep-learning, and hybrid approaches Mao et al. (2024). Sentiment analysis approaches are depicted in 2.2.

Lexicon-based Approach The lexicon-based approach utilizes a sentiment lexicon that assigns scores to the collected tokens. It is an unsupervised technique and is domain-reliant, as the same word can have different sentiments in different contexts. For example, in the sentences "The project is taking too long" and "Her fingers are long and slender," the term "long" is negative in the first statement due to the preference for short project times, but positive in the second sentence as longer fingers are often considered more attractive. This problem can be mitigated by adopting a dictionary adaptation technique. Lexicon-based approaches

[3]Most used techniques of feature extraction are Bag of Words, TF-IDF (Term Frequency-Inverse Document Frequency), word embedding, and NLP based. Feature extraction is the process where raw text is transformed into a structured representation (points in a vector space, where each word is a point) that machine learning algorithms can work with effectively (in other words, numbers).

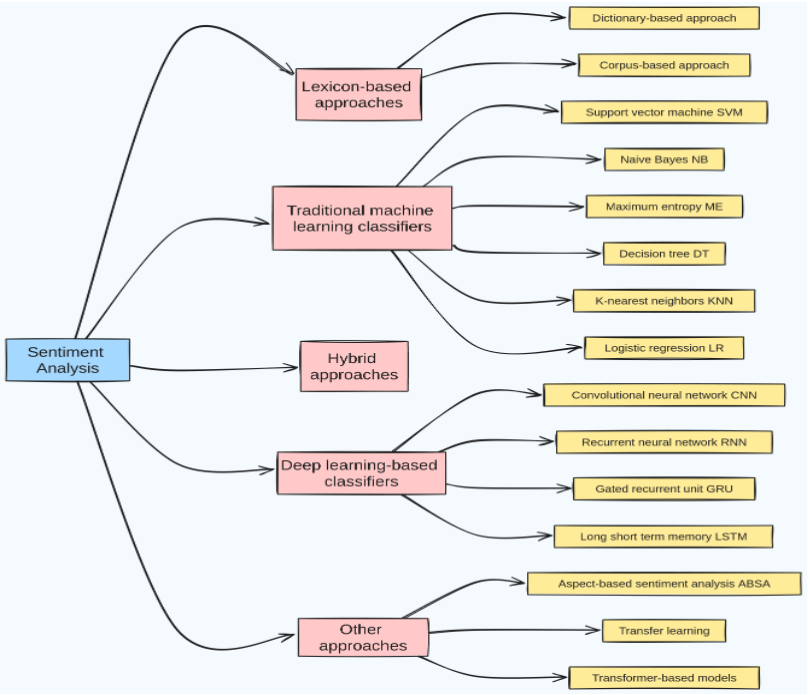

Figure 2.2: Sentiment Analysis models landscape, *Source:* Mao et al. (2024)

are usually divided into two methods: dictionary-based and corpus-based methods:

- Dictionary-based Methods: These methods rely on pre-defined sentiment lexicons and are relatively simple to implement. However, they may lack coverage for domain-specific terms.
- Corpus-based Methods: These methods build sentiment lexicons dictionary from a corpus of text, allowing for better coverage of domain-specific vocabulary. Require a sufficiently large and representative corpus but can be implemented with the help of pre-defined lexicons.

Machine learning approach Machine learning methods separate datasets into training datasets and test datasets. The models learn from the training datasets, acquiring knowledge about known information, and the test datasets are then used to evaluate the performance of the models. Conventional machine learning classifiers commonly used for sentiment classification include:

- **Naive Bayes (NB)**: This classifier assumes that the features are independent given the class label. It is simple and efficient but may struggle with complex data structures.

- **Support Vector Machines (SVM)**: SVM finds the hyperplane that best separates the data into classes. It is effective for high-dimensional data but can be computationally intensive.
- **Decision Trees (DT)**: DT splits the data into subsets based on feature values, creating a tree-like model of decisions. They are easy to interpret but can become overfitted, especially with noisy data.

Deep learning-based classifiers Deep learning is a branch of machine learning that uses neural networks to model and understand complex patterns in data. These classifiers can process large amounts of data and capture intricate linguistic nuances. By employing these deep learning-based classifiers, sentiment analysis can achieve higher accuracy and handle more complex linguistic structures, making it a powerful tool for understanding sentiment in text.

- **Convolutional Neural Networks (CNN)**: CNNs are feed-forward neural networks that utilize convolutional computations and pooling operations (Chen, 2015; Li et al., 2021b). Originally developed for the computer vision field, CNNs have been extended to many areas, including Natural Language Processing (NLP). Chen (2015) proposed a notable CNN sentiment analysis method built on word2vec for sentence-level sentiment categorization, which outperformed competing approaches. This work demonstrated the practicality of pre-training word embeddings in deep learning.
- **Recurrent Neural Networks (RNN)**: RNNs are widely used in sentiment analysis due to their ability to capture and remember information over long sequences. RNNs leverage prior knowledge to remember previous information, making them particularly effective for sequential data which is usefull for sentence sentiment analysis.
- **Long Short-Term Memory (LSTM) Networks**: LSTMs incorporate a gating mechanism to address long-distance dependencies that standard RNNs cannot handle.

Transformer-Based Models[4] Transformers utilize the self-attention mechanism for modeling, enabling them to encode long text sequences, unlike LSTM networks which have memory and computational limitations. By replacing LSTM

[4]Refer to A for an example.

networks with a complete Attention structure, Transformers address the sequence-to-sequence problem more effectively, achieving superior results and reducing computational complexity.

A special Transformer model named BERT (Bidirectional Encoder Representations from Transformers) was proposed by Google Research in 2018. BERT consists of Transformer Encoder layers, supplemented by word and positional encoding, to encode the syntax and semantics of text comments and uses contextual information to produce token-level representations. Moreover BERT infuses auxiliary sentiment knowledge by incorporating sentiment contextual information into language representation models. The contextual word embedding of this language model includes inter-sentence relationships and understanding the context of the entire comment, preserving the semantic meaning of words across various domains.

Table lists the advantages and disadvantages of each model. In summary, while lexicon-based methods provide a straightforward approach, machine learning and deep learning techniques, particularly SVM and Bi-LSTM, offer higher accuracy and better handling of complex text data. Transformer-based models like BERT represent a significant advancement, integrating deeper contextual understanding and adaptability.

In 2.1 its reviews advantages and disadvantages of the explained models.

Table 2.1: Comparison of sentiment analysis methods.

Technique	**Advantage**	**Disadvantage**
Lexicon-based approach	• No need for trained data. • Quickly access word definitions in the vocabulary. • Better results when domains are different.	• Opinions have a specific content orientation. • Performance varies due to the lexicon's wide range. • Difficult to provide comprehensive texts that cover every text term.

Continued on next page

Table 2.1 – continued from previous page

Technique	Advantage	Disadvantage
SVM	• Most famous SA algorithm. • Achieve good accuracy for a huge datasets.	• Model fine tuning is quite time-consuming and challenging. • Long training time are required for large datasets.
NB	• Simple to Implement. • Fewer training data are needed. • Less data and training time are needed compared with other methods.	• Assuming that features are mutually independent. • May experience a zero frequency issue. • Limited by imbalanced data categories.
DT	• Simple to build. • Less training time. • Training does not require a large datasets.	• Over-fitting is more likely in models. • The domain-oriented model will be built.
KNN	• Non-linear decision boundaries can be constructed. • Without explicit training, data can be continuously added throughout time.	• More datasets and dimensions result in more complex predictions. • All features are given equal weight.

Continued on next page

Table 2.1 – continued from previous page

Technique	Advantage	Disadvantage
CNN	• Higher Accuracy. • Faster Training.	• Need a large amount of train datasets and train time. • Pooling layers may result in the feature losing its position or order.
RNN	• Enable to remember long distance relationships between sequential data. • High reliability.	• Compared to other models, train more slowly. • Costly in terms of computing and complicated.
LSTM	• Better than RNN. • Can capture long term dependencies.	• Extremely complex model. • High training time.
Transformer	• Self-attention models are used to identify dependencies. • Concentrates only on the sentence's key points.	• Require huge data.

This study has chosen VADER for its superior performance. The survey of SA on lexicon-based approaches compared the SA tools, including Natural Language Toolkit (NLTK), Text blob, and Valence Aware Dictionary and Esntiment Reasone (VADER), to find that VADER outperforms other tools (Bonta and Janardhan, 2019). Lexicons are among the most commonly used techniques in Financial Sentiment

Analysis. Our work does not aim to revolutionize the landscape of sentiment analysis models but rather to study its application within the context of search engines.

2.2.1 Limitations

Birjali et al. (2021) reviewed SA approaches, challenges and trends. In its findings, it is essential to establish robust datasets in many languages. These datasets should be well-annotated, finely graded, and comply with ethical standards, making them widely available in the public domain to facilitate better research. Addressing the co-reference resolution problem and detecting hidden emotions, irony, and sarcasm remain open research questions in sentiment analysis (SA). Feature extraction in SA encounters several issues, such as context dependency, high dimensionality, redundancy, and slang words. Additionally, ongoing research topics in SA include handling multilingual data, improving cross-domain accuracy, cross-dataset sentiment analysis, and implicit sentiment analysis using contextual backgrounds.

2.2.2 Financial Sentiment Analysis

Keynes (1936) suggested that investors' "animal spirits" could justify wild stock market price movements. However, the standard finance model's assumption that unemotional investors force capital market prices to equal the rational present value of expected cash flows is a poor fit to historical stock market patterns. Investor sentiment is a key driver of asset prices and is based on noise traders with random beliefs and rational arbitrageurs with Bayesian beliefs. These traders create a downward-sloping demand for risky assets, leading to an equilibrium where noise traders can influence prices.

In the financial industry, two mature securities market analysis methods are Fundamental Analysis and Technical Analysis. Fundamental Analysis evaluates firms' value and predicts mid- and long-term securities price trends using economic principles, macroeconomic indicators, policy, and industry trends. Technical Analysis examines historical transaction data, such as liquidity supply and demand, stock prices, and volumes. Due to the unstructured and scattered nature of many macroeconomic factors, NLP techniques have become valuable supplements in fundamental analysis. Researchers have found that market sentiment influences price trends, trading volumes, volatility, and potential risks Kazemian et al. (2016).

Consequently, some trading strategies are now based on financial sentiment analysis.

The tasks of Financial Sentiment Analysis (FSA) differ from traditional sentiment analysis, often used in user-product scenarios. Firstly, in financial texts, the sentiment reflects market participants' expectations. For example, a text with positive sentiment suggests optimism about a company's future. Secondly, financial texts are more implicit in sentiment compared to product reviews. They can be verbose or densely packed with information and often lack a clear linguistic structure. This complexity arises from the use of technical terms, domain-specific knowledge, and numerous statistics. Thirdly, financial texts encompass various perspectives such as macroeconomic, microeconomic, event-oriented, and company-specific views, with the same word potentially having different sentiments in different contexts. Thus, specific sentiment analysis techniques are needed.

The debate on whether investor sentiment affects stock markets has been prominent in behavioral finance. Previous studies have shown that investor sentiment explains stock returns, has a larger influence on stocks with subjective valuations, and is subject to reversals. However, defining a good measure of investor sentiment remains unresolved. Various attempts have been made to quantify investor sentiment and evaluate the effectiveness of available measures in explaining and predicting stock market activity. These indicators have been based on different methodologies, data sets, sources, and have targeted different retail investors over various time periods.

As shown in Table 2.2, methodologies used in Financial Sentiment Analysis (FSA) are categorized into unsupervised, supervised, and semi-supervised. The main approaches are listed below.

Table 2.2: Three Methodologies of FSA

Type	**Characteristics**
Unsupervised	Use dependency parsing to determine the polarity and do not need prior or specific training
Supervised	Label data to train classifiers, useful, time-consuming, depend on expertise knowledge
Semi-supervised	Use numerous unlabeled samples to facilitate the supervised classifier

Existing literature on financial text mining often uses simple textual representations like bag-of-words, based on dictionaries or word frequency from message

corpora. Common methods include TF-IDF and minimum word occurrence. These basic approaches overlook more complex features that capture text semantics and use robust selection procedures based on market feedback. Most research uses proprietary methods and datasets, making results hard to compare. For better benchmarking, this study rebuilds previous approaches using corporate disclosures from two sources, focusing on firm-value relevant facts. Despite advancements in deep learning, recent surveys still focus on traditional machine learning models. The table 2.3 summarizes related work on FSA.

Table 2.3: Summary of related work (ordered by relevance to our work).

	Data set		Text mining - feature processing			Machine Learning	
Author	Text base	Effect	Feature type	Selection method	Market feedback	Method	Accuracy
Schumaker et al	US financial news	Stock prices (intraday)	Noun phrases	Minimum occurrence per document	No	SVM	58.2%
Schumaker et al	US financial news	Stock prices (intraday)	Noun phrases	Minimum occurrence per document	No	SVR	59.0%
Groth et al	German adhoc announcements	Stock prices (daily)	Bag-of-words	Only stopword removal	No	SVM	56.5%
Mittermayer	US financial news	Stock prices (intraday)	Bag-of-words	TF IDF: selecting 1000 terms	No	SVM	–
Wüthrich et al	Worldwide general news	Index prices (daily)	Bag-of-words	Pre-defined dictionaries	No	K-nn, ANNs, naïve Bayes	Not comparable
Li 2010	US corporate filings	Stock prices (daily)	Bag-of-words	Pre-defined dictionaries	No	Naive Bayes	Not available

Chapter 3

Web scrapping

Data acquisition is the starting point of any data science project, with inputs obtainable from private sources like internal company data or a wide array of public sources. Public sources include official data from government institutions, journals, or any data found on the web, whether open access or paid. There are three primary data sources methodologies: utilizing pre-formed datasets (1.1.4), leveraging APIs (Application Programming Interfaces), and employing web scraping techniques. While pre-formed datasets provide structured and curated data, APIs offer dynamic access to real-time information, and web scraping allows for the collection of data directly from web pages. This chapter will focus on an in-depth review of APIs and web scraping methods, assessing their effectiveness and implementation in financial news data extraction.

Web scraping, also referred to as Screen Scraping, Web Data Extraction, or Web Harvesting, is a technique to extract data from the World Wide Web (WWW) and save it to a file system or database for later retrieval or analysis. This is accomplished either manually by a user or automatically by a bot or web crawler. This process aims at converting unstructured data into organized databases. Common techniques are utilizing Hypertext Transfer Protocol (HTTP) or a web browser, like parsing Document Object Model (DOM) or HyperText Markup Language (HTML), to retrieve data (Zhao (2017)).

Most of the produced data is stored on private servers, but a considerable part is made publicly available across the 1.83 billion websites available online that are now up for grabs for researchers equipped with basic web-scraping skills[1]. While new data types and data analysis tools offer new opportunities and one can expect

[1]See Internet Live Stats a live-streaming website that collects web usage for numerous topics.

a significant increase in use of video, voice, or picture analysis in the future, web scraping is used predominantly to collect text from websites.

Self-built Web Scrapers require advanced knowledge of programming, but nowadays, we encounter Cloud, Pre-Build or Browser extensions scrappers that offers the user an interface to interact with[2]. We cannot understand web scraping without the context of web browser and the process behind the scenes on a web-page.

3.1 Web Browser

Web scraping involves using software to extract data by fetching and parsing web pages. Typically, users do not interact directly with the elements behind a web page, they use Web Browsers, which translates the HTML (and more data) into a visual and interactive format that can be easily navigated. Web browsers applications, such as Google Chrome, Mozilla Firefox, Microsoft Edge, and Apple Safari, allow users to enter web addresses (URLs) to visit specific websites, bookmark favorite pages, and use search engines (SE)4 like Google, Yahoo, or Bing to find information across the internet (we will review them in the next chapter). They support various web technologies and standards, ensuring compatibility and functionality of web pages, and often include features like tabs, history, and extensions to enhance the user experience.

3.1.1 Identifying webs and its contents

The primary role of a browser is to provide the content of a website to the user in a user-friendly manner. To display the website data, the user specifies its resource location using a URI (Uniform Resource Identifier). A URI is a string of characters used to identify a resource on the internet. URIs can be further categorized into subsets, such as URLs (Uniform Resource Locators) and URNs (Uniform Resource Names). For example, entering a URL into the browser's address bar allows the browser to locate and display the specified web resource, providing a seamless browsing experience.

The primary role of a browser is to provide the content of a website to the user in a user-friendly manner. An indispensable component of the browser is the SE, which allows the user retrieve and display web resources, typically websites but can also include PDFs, images, and other content types. This component also generates

[2]See Serverless architecture for a web scraping solution, AWS

document summaries, known as snippets, and utilizes data from the document store to enhance search results.

The user specifies the resource location using a URI (Uniform Resource Identifier). URI is a string of characters used to identify a resource on the internet. URIs can be further categorized into subsets, such as URLs (Uniform Resource Locators) and URNs (Uniform Resource Names).

- **URLs**: These are the most common type of URI. They specify the location of a resource and how to access it. For example, `https://www.example.com/page.HTML` is a URL that points to a web page located at the domain "example.com." We can parse URLs.
- **URNs**: Unlike URLs, URNs are persistent identifiers for resources but don't necessarily provide information on how to locate them. An example could be an ISBN for a book.

To scrap an HTML page first is need to identify its URL. A dynamic URL is usually made up of two main parts, with the first one being the base URL, which lets the web browser know how to access the information specified in the server. The next part is the query string that usually follows a question mark. An example of a news URL is NasdqNasdq: where the base part is `https://www.nasdaq.com/news-and-insights`, and the query string part is `?page=1&rows_per_page=10`, which consists of specific queries to the website: displaying news in a list view (`rows_per_page=10`), showing 10 products in each page, and loading the first page (`page=1`) of the News Headlines. The regular structure of URLs is another feature that makes web scraping easy. Changing the page number (the last digit in the preceding URL) from 1 to 2 will display the subsequent 20 results of the query. This process can continue until the last query result is reached. Thus, it is easy to extract all query results with a simple loop.

Once identified a desired URL, to parse the web we need to understand its structure and contents. Parsing a document means translating it to a structure the code can use.

Web content is semi-structured and displayed using HTML, which defines the elements and layout of web pages. When a web page is loaded, the browser parses the HTML to create the Document Object Model (DOM), a tree-like structure representing the page's content and elements, allowing for dynamic manipulation and interaction. HTML's vocabulary and syntax, detail the tree order using labels and

tags to illustrate the different nested levels of the webpage source code. The DOM then create a tree of nested elements that can be interpreted by the browser, so the user can interact with the page[3]. The interactive process between the user's actions and the website's responses is managed by scripts, written in JavaScript, which are also pulled when the user requests the URL content (Berners-Lee et al. (2023))

HTML is a universal language used to create web pages, compatible with various devices like Computers and handheld devices. It enables users to publish online documents with diverse content, such as headings, text, tables, images, hyperlinks, videos, sounds, and forms for searching or ordering. An HTML document consists of elements that act as labels for different content types, guiding the web browser on how to display them. For instance, the `<title>` element contains the page title, the `<body>` element holds the main content, the `<img>` element includes images, and the `<a>` element creates links. To view HTML in any browser, right-click and select the Inspect (or Inspect Element) option. In the following images, there is an example for Nasdq News, its browser content and its HTML content:

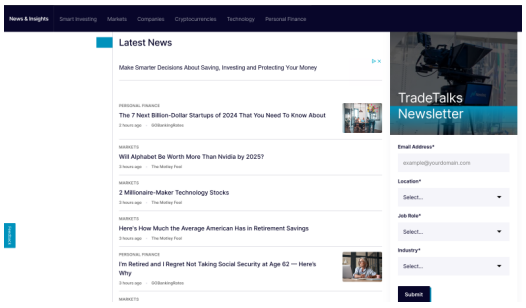

Figure 3.1: Nasdaq News and Insights, 2024.06.22

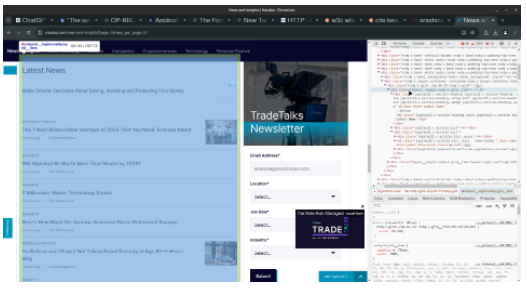

Figure 3.2: Nasdaq News and Insights HTML content

A part from rendering contents, browsers include more features, which are not case of study of this project. The standar key components are[4]:

- **User Interface (UI)**: Including elements like the address bar, navigation buttons, and bookmarking menu, everything except the main window where the webpage is displayed.
- **Rendering Engine**: Responsible for interpreting and displaying requested content, parsing HTML and CSS to render the webpage.

[3]Refer to B for more information about the rendering process from HTML to DOM
[4]Refer to the resource Web Browser article for extended information about web browsers

- **Networking**: Handles network calls such as HTTP requests, with different implementations across platforms.
- **UI Backend**: Draws basic interface elements like dropdown menus and windows, utilizing platform-specific methods at the operating system level.
- **Data Storage**: Provides a persistence layer for storing various types of data locally, including cookies and other forms of browser storage like localStorage, IndexedDB, WebSQL, and FileSystem.

3.1.1.1 HTML througt python

To extract HTML data from a URL directly without relying on web browsers is needed to bypass the typical user interface experience of web browsing. To request a web resource, a socket in the computer interacts with the HTTP protocol by creating a TCP connection to the server's IP address on port 80 for HTTP or 443 for HTTPS. The Domain Name System (DNS) translates domain names into IP addresses. After entering a URL, DNS provides the IP address, and your computer sends an HTTP request (e.g., GET or POST) to the server. The server processes this request and returns an HTTP response with a status code and the requested content, such as HTML, or images. A successful request (status code 200) includes the webpage's HTML content in the response body[5]. All this process is summarized in Figure 3.3.

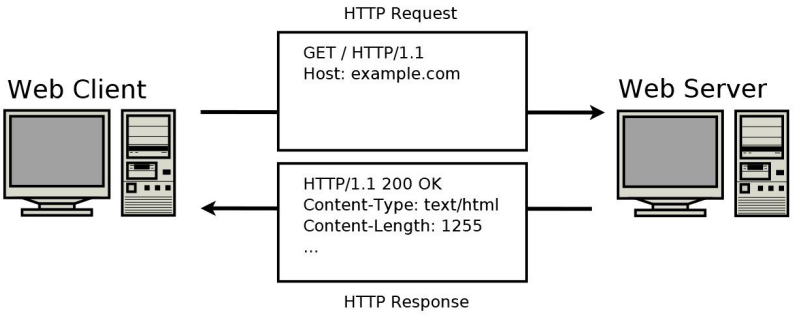

Figure 3.3: HTTP @Request Protocol *Source:* PACKTUB

Several libraries have been developed in many programming languages and environments to help users parsing HTML code. In Python 3.[6], the most relevant and

[5]Internet protocols are a major area of study, standardized globally for decades. For more details, refer to: `https://www.w3.org/Protocols/`, defined by the World Wide Web Consortium (W3C) organization.

[6]Python is a high-level, interpreted programming language. For more information on Python

widely used is BeautifulSoup. Amongst other capabilities, BeautifulSoup provides ways of navigating, fetching and transforming the parse tree. Its main functionality lies in the fact that the gap between HTML and XML is not always savable. At times, HTML is broken in ways that will cause an XML parser to reject the entire website and, in this respect, BeautifulSoup tolerates flawed HTML and still lets the user extract the data easily.

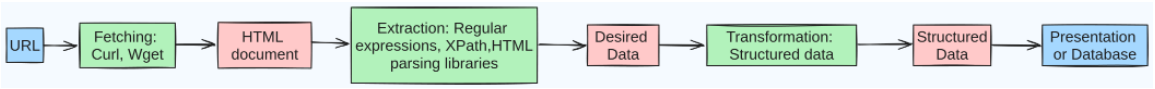

Figure 3.4: Web-Scraping process *Source:* Khder (2021)

One of the most popular web-scraping techniques is using Requests and BeautifulSoup libraries in Python, which are available for both Python In addition to having Python installed, it is required to install necessary libraries such as bs4 and requests. The next step is to build a web scraper that sends a request to the website's server asking the server to return the content of a specific page as an HTML file. The requests' module in Python enables the performance of this task. For example, one can retrieve the HTML file from Wordery your online bookshop with the following codes:

3.1.1.2 The role of Unstructured Data

In the field of Browsers and during this study, we will work with unstructured data. Unstructured data play a key role on internet information transmissions due to its velocity and flexibility. While structured data adheres to predefined table formats and conforms and facilitates analysis and modeling, unstructured data, with its dynamic and varied nature, presents challenges like categorization but also harbors valuable insights (Amazon, Unestructured Data).

Common unstructured data formats include JSON (JavaScript Object Notation), XML (eXtensible Markup Language), and HTML (HyperText Markup Language), amongst others. In table 3.1 we found most common Unstructured data types.

For a dummy example of unstructured data in the financial news context, consider the following JSON schema. It lists the headline, author, content, and quote

3.0 and to see the current versions, visit Python 3.0 Release. The current versions are releases of Python 3.x.

Table 3.1: Common Unstructured Data Formats

Format	Description
JSON	Widely used for data interchange due to its simplicity and human-readable format, often utilized in web APIs.
XML	Commonly employed for representing and exchanging structured data between different systems, with a hierarchical structure and tags.
HTML	The standard markup language for creating web pages, structures content with tags but lacks strict validation.

as string elements; the date as an integer (epoch-based time[7]); and the related tickers and tags as lists of strings. This means that for the first JSON element, and for the keys `related_tickers` and `tags`, the JSON will provide a list of elements instead of just one element as in typical tables.

```
{
  "headline": "Tech Stocks Surge",
  "date": 1719110535,
  "author": "John Doe",
  "content": "Tech stocks soared on positive economic news. Apple,
   Amazon, and Google led the gains.",
  "related_tickers": ["AAPL", "AMZN", "GOOGL"],
  "quote": "This indicates market stabilization.",
  "tags": ["tech stocks", "market"]
}
```

3.1.2 Regulations and *Robots.txt*

Although web scraping is a powerful technique in collecting large data sets, it is controversial and may raise legal questions related to copyright, terms of service (ToS), and "trespass to chattels[8]" (O'Reilly (2006)).

A web scraper can legally copy data from web pages as copyright typically doesn't cover mere data. Court rulings suggest users can use web crawlers on public sites without agreeing to Terms of Service. However, sites often restrict detailed data to registered users who agree to ban automated scraping.

[7]Epoch time, also known as Unix time or POSIX time, is the number of seconds that have elapsed since 00:00:00 Coordinated Universal Time (UTC), Thursday, 1 January 1970. It is used widely in computing as a standard way to represent points in time.

[8]Trespass to chattels is a legal term for a tort where one party intentionally or negligently interferes with another person's lawful possession of movable personal property (chattel).

Web scraping involves sending requests to a host server, which processes these requests and sends back responses. This process often includes repeatedly querying and loading numerous web pages in a short timeframe. Such activity can overwhelm the server, causing traffic surges, system overload, and potential damage to both the server and its users. This strain can be akin to a denial-of-service attack, as it affects the physical web server owned by the web application provider.

An ethical web scraping tool maintains reasonable request frequency and ensures data privacy and confidentiality. Before scraping, one should check the target website for restrictions, typically specified in the Robots Exclusion Protocol (robots.txt file). For example, the robots.txt file of the Nasdaq website, available at Nasdaq Robots.txt, states that:

```
#
User-agent: *
Crawl-delay: 30

# Allow
Allow: /profiles/*.png
Allow: /profiles/*.svg

# Files
Disallow: /README.txt
Disallow: /web.config

# Paths (clean URLs)
Disallow: /admin/
Disallow: /comment/reply/
Disallow: /search?q=
Disallow: /user/register/
Disallow: /user/password/
Disallow: /user/login/
Disallow: /user/logout/
```

The User-agent specifies the bot and its rules. If not provided, rules under User-agent: * apply. Allow and Disallow indicate URLs bots can and cannot access, respectively. For instance, Nasdaq website disallows pages such as “search?q” /comment/reply/,” /user/login/,” and /user/logout/”.

If the rules are not followed, most web servers will automatically block the user IP, preventing further access to its pages. Websites prefer serving content to real users on actual web browsers—except when it comes to Search Engines (SE), as these sites generally accommodate SE scrapers because they want to appear in their search results. Multiple walk arounds strategies emerge to avoid being blocked, taking in to account that webpage developers work to patch them. There are two

main strategies: request as a Web Browsers (and not as a simple HTTP requst) and emulating human behavior.

Simple HTTP requests omit key identification components created by web browsers. Without this context, websites can easily identify us as bots and restrict our access. To emulate a web browser, we can follow these guidelines[9]:

- **Use Proxies**:
  - Utilizing proxies allows you to make requests from various IP addresses, preventing websites from detecting and blocking your single IP.
- **Understand Browser Fingerprinting**:
  - Browser fingerprinting involves identifying unique browser characteristics and behaviors. Websites use this to verify the authenticity of a browser.
- **Set Request Headers and Change Your User Agent**:
  - HTTP headers, especially the "User-Agent" header, are crucial in web scraping as they convey browser and device information to the server.
    1. *Set a Popular User Agent*: Use a string from a well-known browser like Chrome, Safari, or Firefox, or utilize tools like Fake-Useragent for Python to generate one.
    2. *Keep User Agents Updated*: Regularly update user agents to match the latest browser versions, preventing detection.
    3. *Rotate User Agents*: Rotate through multiple user agents to avoid patterns and make your requests appear more like regular user activity.
- **Use a CAPTCHA Solving Service**:
  - CAPTCHAs are designed to differentiate humans from bots, often appearing for suspicious IP addresses. CAPTCHA solving services automate this process, allowing your scraper to bypass these challenges and continue extracting data seamlessly.
- **Randomize Your Request Rate**:

[9]See ScrapingBee for more information

  - Consistent request intervals, such as one-second intervals, are easily detectable as scraping behavior. Instead, vary your request intervals randomly to mimic human browsing patterns and avoid detection by anti-scraping technologies.

- **Consider Your Location**:

  - Websites often target specific regions, and accessing them from IPs outside their main service area can appear suspicious. Using proxies or VPNsdescription from the same geographic location as the target site can help blend in with typical user traffic and reduce the likelihood of being blocked.

- **Manage Cookies**:

  - Cookies store user-specific data, helping websites track sessions and user activity. Scrapers should handle cookies like a real browser to maintain session continuity and avoid detection.

- **APIs**:

  - Use API as alternative if the resource provides it.

3.1.3 Application Programming Interfaces

Recognizing the need for users to collect information, many websites also make their data available and directly retrievable through an open API. Specifically, they determine the types of requests that can be made, how to make them, the data formats that should be used, and the rules to adhere to. Instead of collecting the resource data from its website, we collet it directly from the database (In figure 3.5 we distinct the key API architecture `Client - Server`). The service provider manages the database backing the API. This can result in differences between the data available from the API and the website, specially for sensitive or private data, we may refer to the API contract[10].

Numerous companies and organizations offer free public APIs. If APIs are available, it is usually much easier to collect data through an API than through web

[10] APIs are thought of as contracts, with documentation that represents an agreement between parties: If party 1 sends a remote request structured a particular way, this is how party 2's software will respond. Note that APIs don't directly deal with the technicalities of connecting two machines. Instead, they focus on defining how information should be requested and what responses to expect.

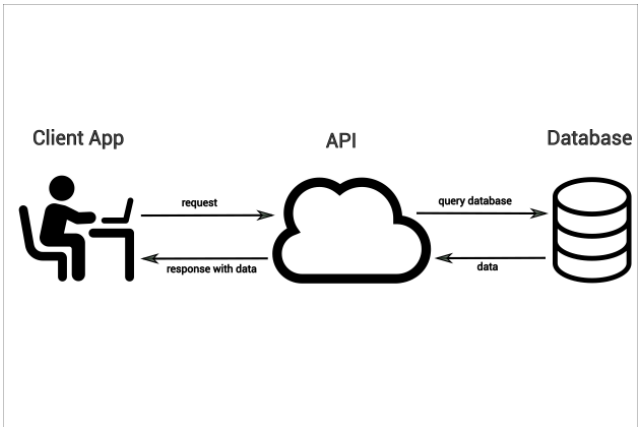

Figure 3.5: API Flow

scraping. Furthermore, data scraping with API is completely legal, the data copyrights remain with the data provider.

Here are common API[11] features: (A): Rate limiting, (B): Authentication mechanisms (API keys, OAuth tokens[12]), (C): Metadata provision, (D): Support for various data formats (JSON, XML), (E): Error handling, (F): Error handling, (G): Status codes, (H): Versioning, (I): Documentation and API explorer, (J): Callback mechanisms for asynchronous operations.

APIs use status codes to indicate the result of a request. These codes are grouped into categories based on their type, as seen in Table 3.2:

Status Code	Description
1xx	Informational
2xx	Success
3xx	Redirection
4xx	Client Error
5xx	Server Error

Table 3.2: API Status Code Categories

[11] APIs come in different flavors, such as REST, SOAP, GraphQL, and gRPC. REST (Representational State Transfer) is popular for its simplicity and use of standard HTTP methods, and we will use it during this study. For more details, visit Red Hat's API guide.

[12] OAuth is an open standard for access delegation, commonly used as a way to grant websites or applications limited access to a user's information without exposing passwords; for more information, see OAuth.net.

Chapter 4

Search Engines

At this point, we have gained the know-how to extract text data using scrapers. However, unlike other studies that typically scrape only one domain1.1.4, such as Reuters, Nasdaq News, or Bloomberg, we will not limit ourselves in this way. Instead, this study brings a novel approach by utilizing Search Engines (SE). This allows us to obtain information from a variety of sources over time. On this chapter we will explore, the advantages, disadvantages, and possible implementations of this approach.

We consider that SE are powerful source for financial information due to their ability to provide constant, real-time updates, and they have shown high predictive capacity for economic events. Moreover, we believe that users typically leverage search engines to gather information from various sources rather than visiting specific newspapers in the case of financial data. Furthermore, as mentioned, we consider that the fact of defining the Keywords for scraping information is a key differentiator from other data collection methods. Establishing various keywords we can create a network of domains to scrape information. Automated scraping tools can continuously gather the latest news and trending articles, adding trust and credibility by incorporating news from reputable publications and minimizing the risk of 'fake news'.

SE are tools for listing database content based on keywords and often are managed within a web browser or a mobile app. They allow users to input specific terms, which the engine uses to retrieve relevant information from its indexed database. For example, when searching for a product on Amazon, the search engine processes the keywords and lists products matching the criteria. Behind the scenes, a ranking process occurs, where indexed elements in the database are sorted based on relevance and other factors to present the most pertinent results. There are numerous search engines available on the internet that cater to specific

types of content, making it easier for users to find relevant information within particular domains. Some examples include[1]:

- **Education**:
  - Google Scholar
  - Internet Archive Scholar
  - Library of Congress
  - Semantic Scholar
- **Genealogy**:
  - Mocavo.com
- **Job**:
  - Indeed (US)
  - Yahoo! HotJobs (Countrywise subdomains, International)
- **Medical**:
  - CiteAb
  - Searchmedica
  - WebMD

In this study, we will focus on general SE, which provide hyperlinks to web pages and other relevant information on the Web in response to a user's query. The search results are often a list of hyperlink, but the user also have the option of limiting the search to a specific type of results, such as images, videos, or news.

Although we could extract SA from many unstructured data, in this study we decided to limit the search for news resources, which offers several advantages. The texts provided by websites for a general search are typically descriptive, objective, and without sentiment bias. For instance, searching "SP500 Bullish" in Google on 06/18/2024 yielded the link S&P 500 (SPX) Chart and Forecast Today, which discussed the positive performance of the index that day (+0.48%, or +26.16 points). Another link, What Is the Bullish Percent Index?, explained how to calculate the Bullish Percent Index (BPI), an objective article with limited relevance to the day's

[1] *Source:* Wikipedia

index performance. Furthermore, the date linked to the new and its sentiment is of vital importance, since we will later be able to compare the performance of the index that day with the sentiment result.

There have been many search engines since the dawn of the Web in the 1990s, but Google Search became the dominant one in the 2000s and has remained sowith a 91% global market share. In table 4.1 is listed the current SE market share distribution.

Table 4.1: Search Engine Market Share Worldwide - May 2024. *Source:* StatCounter

Search Engine	**Market Share (%)**
Google	90.8
Bing	3.72
Yandex	1.58
Yahoo!	1.19
Baidu	0.92
DuckDuckGo	0.56

4.1 How Search Engine works

The term 'engine' in this context primarily refers to the searching component, which is built upon vast databases containing indexes that list websites and internet information. In essence, SE serve as the gateway to navigating and accessing the vast expanse of the web. SE architectures are incredibly complex systems that must navigate non predefined spaces to search while meeting user expectations by delivering relevant and timely results. This study will cover only basic concepts of SE architecture.

For an in-depth understanding, we refer readers to the official Google SE paper Brin and Page (1998) and its patent Inc. (2003).

Search engines serve two primary functions: indexing and querying. The indexing process constructs the necessary data structures for searching, while the query process utilizes these data structures and the user's query to generate a ranked list of documents. Essentially, there is a continuous flow of information being indexed, which is then stored and organized for efficient retrieval when users perform searches.

4.1.1 Flowing in information: Collect and Sotre

The first stage, `crawling`, is performed by 'spiders' or 'crawlers'. A Web crawler starts with a list of URLs to visit, the *seeds*. As the crawler visits these URLs it identifies all the hyperlinks in the retrieved web pages[2] and adds them to the list of URLs to visit, called the *crawl frontier* and it copies and saves the information as it goes. This proces creates a vertigenous amount of number of pages to visit, so it needs to prioritize its download, Edwards et al. (2001). The follow image 4.1 describes a crwaling process.

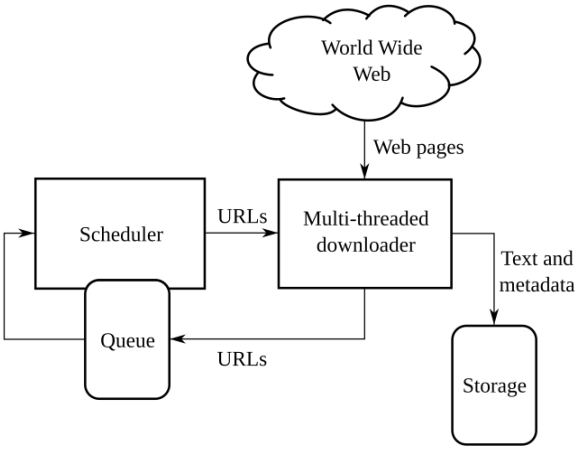

Figure 4.1: Web Crawling architecture. *Source:* Wikipedia WebCrawl

They continuously revisit sites to detect changes, typically every month or two. The spiders check for a robots.txt file, which contains directives about which pages to crawl or avoid. Not all sites are crawled equally; some are crawled exhaustively, while others only partially, due to practical constraints like infinite websites and spam. The spider sends certain information back to be indexed depending on many factors, such as the titles, page content, JavaScript, Cascading Style Sheets (CSS), headings, or its metadata in HTML meta tags

All this crawling process is used to build an index, or catalog, that is a comprehensive collection of all the web pages the spider has found. This index is continuously updated to reflect any changes or new pages detected by the spider. However, there can be delays in adding new pages or changes to the index. Until a page is indexed, it is not searchable.

When users perform a search, they are not searching the live web but an indexed snapshot of it, which may be slightly outdated. Webmasters can expedite the indexing of new content by notifying search engines directly.

[2]HTML links are stored under the tagg `<a href="URL">` easily to identify

4.1.2 Stock of information

Finally the user triggers the third part, the search engine software. This is the program that sifts through the millions of pages recorded in the index to find matches to a search and rank them in order of what it believes is most relevant. For each search query search engines typically do most or all of the following:

- Accept the user inputted query, checking to match any advanced syntax and checking to see if the query is misspelled to recommend more popular or correct spelling variations.
- Check to see if the query is relevant to other vertical search databases (such as news search or product search) and place relevant links to a few items from that type of search query near the regular search results.
- Gather a list of relevant pages for the organic search results. These results are ranked based on page content, usage data, and link citation data.
- Request a list of relevant ads to place near the search results.

A SE's usefulness hinges on the relevance of its results, especially top few results (Jakob Nielsen), assessed by user acceptance. It determines relevance by analyzing indexed information using key indicators in its algorithm, such as meta tags, headings, and page content, to identify key phrases, images, and information that gauge the webpage's quality.

When users search for topics like "bank merger" or "bank takeover," the search engine needs algorithms capable of comparing the text of the queries to the content of the stories to determine relevance. However, unlike structured data like account numbers, understanding and comparing the meaning of words, sentences, paragraphs, and entire news stories is complex. This complexity underscores the importance of understanding how people compare texts and designing computer algorithms that can accurately perform this comparison, forming the essence of information retrieval. Three types of considerations are explained regarding search engines.

- **Relevance**: Relevance determines if a document matches a user's query, considering more than just text matching. Algorithm design must account for this complexity, differentiating between topical and user relevance, which includes factors like recency and language, to optimize the search experience.

- **Evaluation**: The quality of document ranking is measured by how well it meets user expectations. Evaluation methods include precision (proportion of retrieved documents that are relevant) and recall (proportion of relevant documents retrieved).

- **Information retrieval**: Users judge search quality, leading to studies on user interactions and query refinement techniques. Unlike database requests, text queries are often vague. Techniques like query suggestion and relevance feedback refine initial queries based on user interaction and context to improve search results.

4.1.3 Keywords

Typically when a user enters a query into a search engine it is a few keywords. Search keyword, also called search query, is the text that a user enters into Internet search engine to satisfy his or her information needs. The index already has the names of the sites containing the keywords, and these are instantly obtained from the index. Then the top search result item requires the lookup, reconstruction, and markup of the snippets showing the context of the keywords matched.

Andrei Broder authored A Taxonomy of Web Search [PDF], notes that most searches fall into the following 3 categories:

- **Informational** - seeking static information about a topic.

- **Transactional** - shopping at, downloading from, or otherwise interacting with the result.

- **Navigational** - send me to a specific URL.

This unique capability - search huge vast of data in just defined terms - make Search Engine powerful tools to collect data. Specially, we believe that combining SE with SA is a perfect suit for assessing market overall conditions and may outperform other financial indicators. SE enable us to develop a strategy moving from broad to specific terms to gather comprehensive market information about one topic.

For example, let's build a strategy to determine the market sentiment of the European gas market using news. We can start by listing all companies involved in the gas market in Europe, including producers, explorers, and distributors. Then we can search for their news and collect their sentiment. Additionally, we may

think macroeconomic factors are key to understand this market drivers, such as the European Central Bank's interest rates, the eurozone's average growth, deficit or politics. After building the appropriate list, we can also aggregate this macro-sentiment to the micro-company sentiment. Furthermore, we can account that the Ukrainian conflict may affect directly to this market. Again, we can seek for real-time information on the conflict's status combining SA and SE. This approach allows us to refine our sentiment index by considering various elements, ensuring a thorough analysis of the market's sentiment.

Still, building this keywords list is not trivial, and the literature behind it is scarse. For example, Peng et al. (2017) asses that different queries may indicate different trends and behaviors, affecting their predictive ability, an issue often overlooked. The authors introduce a new method, the Hurst Exponent (HE) Time Difference Correlation (TDC) screening method, to select relevant keywords to perform queries. The HE-TDC method combines HE and TDC analysis, requiring keywords to have high correlation and fluctuation memorability similar to the target series. The Hurst coefficient measures the long-term memory of time series and was originally used in hydrology.

4.2 Search engine bias

All crawler-based search engines have the basic parts described above, but there are differences in how these parts are tuned. In this sense, the same search on different SE may produce different results. To begin with, some SE index more web pages than others or more often. The result is that no search engine has the exact same collection of web pages to search through. The user's specific settings also come into play in determining which pages show up in the SERPS, such as their location, their search history and any restrictions and/or provisos they have placed on the search. This is linked to the concept we explain on 3.1.3 in which we are always dependent on the data provided by the server behind the service.

On the other side, SE are commercial products designed to enhance user experience. Most web search engines generate revenue through advertising, with some allowing advertisers to pay for higher rankings in search results. The practice of improving website visibility in search results is known as search engine optimization (SEO). SEO primarily focuses on Google, the dominant player in the search engine market. Companies invest in SEO to ensure their websites rank higher in

organic search results, which can significantly increase traffic and potential revenue. Effective SEO involves optimizing various elements of a website, including content, meta tags, and backlinks, to align with search engine algorithms.

On the Google SE paper work Brin and Page (1998), the authors remark that the goals of the advertising business model do not always correspond to providing quality search to users. For this type of reason authors expect that advertising funded search engines will be inherently biased towards the advertisers and away from the needs of the users.

To illustrate these SE discrepancies, the table 4.2 presents the top 3 results from various SE for the keyword 'SP500' limited to news content on June 25, 2024[3]. As shown, all websites in the table are different, with no shared URLs across the search engines, indicating a high variety in the search results.

SE	Site	URL	Headline
Google	FXStreet	fxstreet.es/news/el-sp-500-opera-en-terreno-negativo	The S&P 500 operates in negative territory for the third consecutive session
	MarketScreener	marketscreener.com/	S&P 500
	20Minutos	20minutos.es/	The bears surrender to the power of the S&P 500 and the bet rises to 6,000 points
Bing	Business Insider	businessinsider.com/	An overload of warning signals mark the 'last straw' that could send the S&P 500 plunging 70%, famed permabear says
	Investopedia	investopedia.com/	S&P 500 Gains and Losses Today: Oil Stocks Surge Amid Global Tensions, Summer Demand
	FXEmpire	fxempire.com/	S&P 500 Price Forecast - S&P 500 Continues to See Support
DuckDuckGo	Yahoo Finance	finance.yahoo.com/news/	Why I Can't Seem to Get Enough of This Ultra-High-Yielding Dividend S&P 500 ETF

Continued on next page

[3]The searches were conducted on a machine located in Spain, titles and headlines were translated to English. We described how result may vary by location

Table 4.2 – *Continued from previous page*

SE	Site	URL	Headline
	MSN Money	msn.com/	Tech Hits Stocks as Nvidia Extends Selloff to 13%: Markets Wrap
	MSN Money	msn.com/	Are stocks melting up? Two S&P 500 sectors swell to pre-'Tech Wreck' levels.
Yahoo!	Bolsamania	bolsamania.com/	Levels of the S&P 500 to consider if worrying graphic patterns in Nvidia and other technologies mean a crash is approaching
	MSN Money	msn.com/	The bears surrender to the power of the S&P 500 and the bet rises to 6,000 points
	MSN Money	msn.com/	The S&P 500 presents a clearly positive trend and can reach 5,624 points

Table 4.2: Top 3 search results for 'SP500' on different SE. *Source:* own made.

4.3 Related work

Previous studies have shown that SE can model and predict real-world events. However, this predictive power is highly limited by the keyword spectrum searched. Current researches haven't yet formed a complete methodology on the Internet search data preprocessing. For example, by what standards or principles are keywords selected? How to measure the time-difference relationships between keywords and target indicator? How to composite a leading keywords index to reflect the target indicator trend? These questions have not yet answered completely by current literatures.

The search data in former literatures almost come from Google trends or Google Insights. Vosen and Schmidt (2011) developed a private consumption indicator using Google Trends data, showing it outperforms most survey-based indicators in prediction accuracy. Other studies have enhanced forecast accuracy for automobile purchases and cinema admissions Hand and Judge (2012) using online search data. Recently, tourism researchers have adopted search query data to predict tourist

arrivals, with Höpken et al. (2018) demonstrating that Google Trends can improve tourism demand forecasting accuracy for both long- and short-term predictions.

For improving search keywords for prediction purposes, three key tasks are commonly found in the literature. First, researchers select domain-specific keyword candidates using domain knowledge, web scraping, text mining, or keyword recommendations from search engine providers Liu et al. (2012). Second, they calculate temporal relationships between candidate queries and dependent variables to identify significant time differences. Third, they construct a data set with input variables that have significant predictive power.

Chapter 5

Dataset

The initial step in a data science project involves acquiring and pre-processing data. In this chapter we will explain the processes used to obtain a dataset on financial news using search engines, and how to obtain other necessary toolkits to perform the sentiment analysis. Before the dataset, we will explain the toolkits.

5.1 Standard & Poors 500

The S&P 500 index, a benchmark of the U.S. equity market, comprises 500 of the largest publicly traded companies and serves as a crucial indicator of the financial health of the economy. Analyzing its historical and real-time data can help in forecasting economic trends, constructing investment portfolios, and conducting financial research on investor sentiment. Our dataset is based on news articles using S&P 500 keywords. We chose S&P 500 for this initial approach because it generates a large volume of news content daily, making it easier to collect, track and analyze the data. Consequently, the dataset content is conditioned by the S&P-related keywords searched. We expect that this broad approach allows us to collect news on various topics, including specific companies, macroeconomics, central banking, and politics. We discussed the advantages and disadvantages of different strategies in subsection 4.1.3.

To import the index into the data model, we use libraries already built in Python, in this case, the library. The library provides different index prices: `Open`, `Close`, `Maximum`, `Minimum`. It is standard to use the `Closing` price, so from now on we refer the index as this price. The data is on a daily basis, as is the news that we will collect. In the 5.1 we can see a sample of the index obtained from the library.

```
1 import yfinance as yf
2 from datetime import datetime
```

```
3
4  # Define the ticker symbol and date range
5  ticker_symbol = "^GSPC"
6  start_date = "2019-01-01"
7  end_date = "2024-06-28"
8
9  # Download stock data
10 stock_data = yf.download(ticker_symbol, start=start_date, end=end_date)
11 stock_data.index.name = 'Date'
12 stock_data.reset_index(inplace=True)
```

Table 5.1: S&P 500 index

Date	Open	High	Low	Close	Adj Close	Volume
2019-01-02	2476.96	2519.49	2467.47	2510.03	2510.03	3733160000
2019-01-03	2491.92	2493.14	2443.96	2447.89	2447.89	3858830000
2019-01-04	2474.33	2538.07	2474.33	2531.94	2531.94	4234140000
2019-01-07	2535.61	2566.16	2524.56	2549.69	2549.69	4133120000

Between January 1, 2019, and May 20, 2024, the S&P 500 Index experienced notable fluctuations due to various economic events. Starting at approximately 2,507 points, the index closed at about 4,758 points by the end of the period. This period saw a total yield of around 89.8%, translating to an annual yield of approximately 13.6%. The highest closing price was around 5,021.84 points in early 2024, and the lowest was near 2,237 points in March 2020, during the peak of the COVID-19, with a maximum drawdown of about 34.1%. The performance was influenced by several key events: the significant drop during the COVID-19 pandemic, the subsequent recovery fueled by fiscal stimulus and low interest rates, and the market adjustments to inflation and rising interest rates in the later years. This period showcased the resilience of the market despite significant global disruptions.

5.2 Valence Aware Dictionary and Sentiment Reasoner

VADER (Valence Aware Dictionary and Sentiment Reasoner) is a lexicon and rule-based sentiment analysis tool tailored for sentiments expressed in social media. It utilizes a sentiment lexicon, which is a list of lexical features (e.g., words) labeled according to their semantic orientation as either positive or negative Hutto and Gilbert (2014).

VADER was developed using a combination of qualitative and quantitative methods to create a sentiment lexicon fine-tuned for microblog-like contexts. It incor-

porates over 7,500 gold-standard[1] lexical features, from well-established sentiment word-banks (LIWC, ANEW, and GI), Hutto and Gilbert (2014), including emoticons, acronyms, and slang. They were benchmarked against ten human raters and includes both sentiment polarity notations (positive - negative - neutral classes) and its intensity on a scale from -1 to +1.

We chose VADER for our analysis due to several key advantages, considering that nor evaluating literature models nor creating one was under the study scope. VADER is efficient, analyzing text in seconds compared to hours for complex models like SVM and its rules are transparent, unlike machine-access-only black-box models. Additionally, VADER's performance matches or exceeds eleven other top sentiment analysis tools. On Table 5.2, we can see results on various models tested on NY York Times news[2].

Table 5.2: VADER results. *Source:* Hutto and Gilbert (2014)

Correlation to benchmark (20 human raters)	**Precision**	**Recall**	**F1 score**
Ind Humans	0.745	0.87	0.55
VADER	0.492	0.69	0.49
Hu-Liu04	0.487	0.70	0.45
SCN	0.252	0.62	0.47
GI	0.362	0.65	0.44
SWN	0.262	0.57	0.49
LIWC	0.220	0.66	0.17
ANEW	0.202	0.59	0.32
WSD	0.218	0.55	0.45

VADER installation and examples (examples from the Github-Page)

```
    # --- Install -------
from vaderSentiment import SentimentIntensityAnalyzer

    # --- examples -------
sentences = [
  "VADER is smart, handsome, and funny.",  # positive sentence example
  "VADER is smart, handsome, and funny!",  # punctuation emphasis
   handled correctly (sentiment intensity adjusted)
```

[1]The authors consider gold-standard lexicons that have a non-zero mean rating, and whose standard deviation was less than 2.5 as determined by the aggregate of those ten independent raters.

[2]In general, news are the most complicated text-corpus to analyze. VADER F1 score, representing overall accuracy, from 0 to 1, on other sources social media (0.96), product marketplace reviews (0,85) and movies reviews (0,92)

```
  "VADER is not smart, handsome, nor funny.",  # negation sentence
   example
  "The book was good.",  # positive sentence
  "At least it isn't a horrible book.",  # negated negative sentence
   with contraction
  "The plot was good, but the characters are uncompelling and the dialog
    is not great.", # mixed negation sentence
  "Make sure you :D today!",  # emoticons handled
  ]

analyzer = SentimentIntensityAnalyzer()
for sentence in sentences:
  vs = analyzer.polarity_scores(sentence)
  print("{:-<65} {}".format(sentence, str(vs)))
```

Output:

```
VADER is smart, handsome, and funny.  -->
      {'pos': 0.746, 'compound': 0.8316, 'neu': 0.254, 'neg': 0.0}

VADER is smart, handsome, and funny!  -->
      {'pos': 0.752, 'compound': 0.8439, 'neu': 0.248, 'neg': 0.0}

VADER is not smart, handsome, nor funny.  -->
      {'pos': 0.0, 'compound': -0.7424, 'neu': 0.354, 'neg': 0.646}

The book was good.  -->
      {'pos': 0.492, 'compound': 0.4404, 'neu': 0.508, 'neg': 0.0}

At least it isn't a horrible book.  -->
      {'pos': 0.363, 'compound': 0.431, 'neu': 0.637, 'neg': 0.0}

The plot was good, but the characters are uncompelling and the dialog is
    not great.  -->
      {'pos': 0.094, 'compound': -0.7042, 'neu': 0.579, 'neg': 0.327}

Make sure you :D today!  -->
      {'pos': 0.706, 'compound': 0.8633, 'neu': 0.294, 'neg': 0.0}
```

5.3 Financial News dataset using search engines

Textual analysis, particularly sentiment analysis, has grown recently, but there is still a lack of literature on dataset to use. As reviewed on Chapter 2, the current landscape of proposed solutions aim to use Web-Crawling bots for mining news on concrete financial newspapers. Still, in this study we propose scrapping search engine entry results.

In deciding which search engine to use, we aimed to leverage the most significant players in the market. Obtaining information is becoming increasingly difficult and costly, as many information services are closing to the public, forcing

us to rely on third parties data providers or develop a Web Crawler. For the academic context of this study, we planned to develop a Web Crawler on the Google search engine, leaving aside external dependencies. However, its development is extremely complex and is far from the scope of the project resources since, despite workarounds, Google's continuous maintenance makes it challenging to develop a reliable long-term solution.

We then decided to explore the available market for search engine APIs. The Google News Search API was officially deprecated in 2011 and is unavailable as of June 2024[3]. Not is the case with the Microsoft Bing news API.

5.3.1 Bing News API

Microsoft Bing is a web search engine launched on May 28, 2009, by former Microsoft CEO Steve Ballmer, aimed at enhancing Yahoo! Search capabilities before Microsoft and Yahoo! partnership. As of September 2022, Bing holds the position of the second-largest search engine globally, commanding a query volume of 3.72%, trailing behind Google (90.8%) and surpassing Yandex (1.58%) as seen in Table 4.1.

Furthermore, Microsoft facilitates access to Bing Search functionalities through an API, which is made available via the Microsoft Azure. Azure is the cloud computing platform developed by Microsoft and offers management, access and development of applications and services to individuals, companies, and governments through a global infrastructure. Microsoft Azure supports many programming languages, tools, and frameworks, including Microsoft-specific and third-party software and systems. On our use case, to set up the necessary services in Azure, follow these steps:

1. Create an account in Azure. In this project, the Microsoft Scholar account provided by Universidad de Barcelona has been used.

2. Add a subscription to the account. Multiple subscription scopes can be managed for the same account. The University of Barcelona has provided us with an Azure for Students subscription.

3. Create a Resource Group in Azure and use the subscription. A resource group allows you to organize projects, services, and subscriptions.

[3]Refer to the Google Support conversations

4. Add the service Bing Web Search v7[4] In the Resource Group. This service is a container for multiple Bing Web Search APIs services. In our case, we are interested in the news service, Bing News Search API.

Upon registration, the resource is deployed and ready to use. In our case, the F1 subscription tier offers 10,000 monthly API calls at no cost, providing an economical solution for our project's requirements while ensuring compliance with the service's ToS acceptance. However, compared to web crawlers, 10,000 calls per month is relatively low, considering the vast amount of data crawlers can collect. Microsoft designed the Bing News API to complement existing applications rather than to serve as a primary resource for creating a database, making it less suitable for our use case. Additionally, the information we can obtain is conditioned by the API's server provider (as explained in subsection 3.1.3), which may cause in difference between the original search engine. Factors such as the user's context or the lack of advertiser intervention could affect the results, potentially making them different from a direct search through the search engine. This limitation is further explained in 4.2.

5.3.1.1 Bing news API documentation

To interact with APIs, we need to adhere to the API contract, which outlines the requirements for making calls and the type of information that can be retrieved. From now on, we will focus exclusively on the news aspect of the Bing Search API, without delving into the full service contract. The API contract, along with examples in Python, can be found on GitHub/Bing API.

The Bing News API has three main endpoints for obtaining news. An API endpoint is a specific URL where an API can access the resources and filter it to retrieve specific data: Trending topic news 5.1, Search news based on a news category 5.2 and Search for general news 5.3, being general new endpoint the one we are interested to use:

$$\text{https://api.bing.microsoft.com/v7.0/news/trendingtopics} \tag{5.1}$$

$$\text{https://api.bing.microsoft.com/v7.0/news?mkt=en-us} \tag{5.2}$$

$$\text{https://api.bing.microsoft.com/v7.0/news/search} \tag{5.3}$$

[4]Notice that this project claims the use of version 7.0 of the API, its latest version, as recommended by the Microsoft guide.

To get data from the API we send GET requests using the HTTP protocol, as reviewed in subsection 3.1.1. Typically, we must specify API headers and parameters to form a request structure. Headers are key-value pairs that provide additional information to the server, like authentications. To do so, we use the `Client ID` and the `API Key` generated for our Azure resource. Common headers include:

- **Authorization**: Contains credentials for authenticating the request.
- **Content-Type**: Indicates the media type of the resource, such as `application/json`.

Parameters specify or filter the data being requested and are defined using the strings '`?`', denoting the beginning of the query string, and '`&`', which separates multiple parameter specifications. Bing News API parameters are defined in its documentation and are summarized in the Table 5.3.

Table 5.3: Bing API key parameters.

Name	Value	Type	Required
count	The number of news articles to return in the response. The actual number delivered may be less than requested. The default is 10 and the maximum is 100	Unsigned Short	No
mkt	The market where the results come from	String	No
offset	The zero-based offset that indicates the number of news articles to skip before returning results. The default is 0. The offset should be less than (`totalEstimatedMatches` - count)	Unsigned Short	No
q	The user's search query term	String	Yes
setLang	The language to use for user interface strings	String	No
since	The UNIX epoch time (Unix timestamp) that Bing uses to select the trending topics	Integer	No

Continued on next page

Table 5.3 – continued from previous page

Name	Value	Type	Required
sortBy	The order to return news topics in. The following are the possible case-insensitive values: Date, Returns news topics sorted by date from the most recent to the oldest. Relevance, Returns news topics sorted by relevance	String	No
textFormat	The type of markers to use for text decorations	String	No

Upon a query request, the API will return either an `ErrorResponse` object[5] in case an error occurs, for example if we don't have authorization, or a `NewsAnswer` object. The `NewsAnswer` object's structure is divided into two: first we will find a preamble - `metadata` - with information about our request and second, we will find an attribute '`value`' that will contain a list of `NewsArticle` objects. Each `NewsArticle` includes the article's `name`, `description`, `image`, and `URL` to the article on the host's website. Be sure to use provider to attribute the article. If Bing can determine the `category` of news article, the article will include the category field.

Let's make an example using Python querying the keyword: `S&P 500`. First is we need to import necessary libraries, then we define the variables, the request headers and parameters. After deep research, we selected the parameters that better fit the study needs: we defined only America region, and the API will set as default to sort results by date and return HTML makers. Found the real query function in C.1 Last we perform the request and convert the binary data into readable JSON data:

```
import json
import request

subscription_key = "your subscription key"
search_term = "S&P 500"
search_url = "https://api.bing.microsoft.com/v7.0/news/search"
headers = {
"Ocp-Apim-Subscription-Key": subscription_key
}
params = {
```

[5]Note that this object does is related to the errors defined by the HTTP protocol, that is, if it is a 4XX error, the API returns an 'error' object instead of returning the simple error code.

```
"q": "SP 500",
"mkt": "en-US",
"count": 3,
"offset": 0
}

response = requests.get(url=search_url, headers= headers, params=
 params)
response = json.loads(response.content.decode("utf-8"))
```

The variable `response` has the content for the query. Let's inspect the metadata:

```
{
  {'_type': 'News',
  'readLink': 'https://api.bing.microsoft.com/api/v7/news/search?q=S
 %26P+500',
  'queryContext': {'originalQuery': 'S\&P 500', 'adultIntent': False},
  'totalEstimatedMatches': 63900,
  'sort': [{'name': 'Best match',
    'id': 'relevance',
    'isSelected': True,
    'url': 'https://api.bing.microsoft.com/api/v7/news/search?q=S%26P
 +500'},
   {'name': 'Most recent',
    'id': 'date',
    'isSelected': False,
    'url': 'https://api.bing.microsoft.com/api/v7/news/search?q=S%26P
 +500&sortby=date'}],
  'value': [ { ... } ]}
}
```

The API, by default, add some sorting context if we do not specify it in the parameter. The final URL was: `https://api.bing.microsoft.com/api/v7/news/search?q=S%26P+500&sortby=date`. Notice how the API added sorting based on Date and the `&` is treated with UTF-8 encode. The query remark us that we can reach almost `totalEstimatedMatches = 63900 NewsArticles`, although we declare to get the first 3 from the first page (`count = 3, offset = 0`). The first `NewsArticle` object is stored on 'value'- key:

```
{
  "name": "S&P 500 gain in first half of 2024 blows historical average
  'out of the water'",
  "url": "https://www.msn.com/en-us/money/markets/s-p-500-gain-in-
 first-half-of-2024-blows-historical-average-out-of-the-water/ar-
 BB1oX5vB",
  "image": {
    "thumbnail": {
      "contentUrl": "https://www.bing.com/th?id=OVFT.0
 dyfDhbA_oZPQ5vvZCyF-S&pid=News",
      "width": 700,
      "height": 350
    }
```

```
    },
    "description": "The S&P 500 is on course to post big gains for the
    first half of 2024, with its runup in the first six months of this
    year so far crushing the",
    "about": [
      {"readLink": "https://api.bing.microsoft.com/api/v7/entities/4
    ef0946d-31b7-304e-410a-4bef17f9fd05",
        "name": "S&P 500"}
    ],
    "mentions": [
      {"name": "MarketWatch"},
      {"name": "S&P 500"}
    ],
    "provider": [
      {"_type": "Organization",
      "name": "MarketWatch on MSN.com",
      "image": {
        "thumbnail": {
          "contentUrl": "https://www.bing.com/th?id=ODF.g28MuDD8Wj91-
    VMHwbl7qQ&pid=news"}
        }
      }
    ],
    "datePublished": "2024-06-26T20:39:00.0000000Z"
  }
```


This `NewsArticle` is titled "S&P 500 gain in first half of 2024 blows historical average 'out of the water'" from MarketWatch on MSN.com. It includes tags like "MarketWatch," and "S&P 500.", a thumbnail image and the publication date is formatted as "2024-06-26T20:39:00.0000000Z," which is in ISO 8601 format. The description talks about the gains on S&P 500 for the first half of 2024, clearly a positive new. Let's review the sentiment punctuation using VADER:

```
analyzer.polarity_scores(response['value'][0]['description'])
Result =
  {
  'neg': 0.081, 'neu': 0.848, 'pos': 0.07, 'compound': -0.0972
  }
```

The news was incorrectly classified as Neutral. As reviewed in Table 5.2, lexicons often struggle with classifying news accurately. VADER lexicons are designed for general use and not specifically for financial terminology. Although there are some fine-tuned VADER lexicons, they lack rigorous scientific validation. How did VADER classify this news? The VADER lexicon repository only registers two words from the news description: "Gain" and "Crushing." "Gain" has a mean sentiment rating of $+1.4$ points and "Crushing" has a mean rating of -1.5 points. The total compound score is $1.4 - 1.5 = -0.1$, which is classified as neutral. However, we, as reviewers, can see that "Crushing" refers to a comparison between

the historical average performance of the index and its performance now, not the performance itself. Clearly, the context is significantly affecting the sentiment evaluation.

As we can see in the `metadata` there are available registers for the query "" in the API. But we got only the first 10 `NewsArticles` since, as the `Count` parameter indicates, if it is not specified in the query, we will receive the first 10. To download all the data for a query we must paginate the API using the `Count` and `Offset` parameters: Here's a step-by-step example:

First Page: Set `Offset` to 0 and `Count` to 100. This fetches items 0 to 99. Second Page: Set `Offset` to 100 and `Count` to 100. This fetches items 100 to 199. Third Page: Set `Offset` to 200 and `Count` to 100. This fetches items 200 to 299.

By incrementing the `Offset` for each subsequent request, we can systematically retrieve and process all items in the dataset, page by page. First, it is crucial to make a dummy request with the selected parameters to each query. This allows us to get an approximate number of results via the `totalEstimatedMatches` parameter. Manually, we define when to end the process based on this parameter.

5.3.1.2 Search Engine keywords

In Chapter 4.1.3, we outline strategies for creating keyword lists to search for specific or general topics, or a combination of both, to build a sentiment index from the collected data. It is important to note that this study is not aimed at developing a prediction or investment strategy for the S&P 500. Instead, the goal is to describe the tools and set a standard on analysis procedures.

Thus, we test the sensitivity of SE results within and between groups of keywords regarding the sentiment of financial news. We will define three groups of keywords that express different sentiments within the financial context: those with a positive bias, those with a negative bias, and those with no bias. The neutral group is set to be a benchmark to compare groups. For each keyword, we will obtain news articles and evaluate their sentiment. Additionally, we will build a sentiment index and compare it to the S&P 500 index, both on overall and across different groups. The hypotheses to be tested can be summarized in:

- **Hypothesis 1:** There is a significant difference in sentiment between groups of keywords.
- **Hypothesis 2:** There isn't a significant difference in sentiment within groups of keywords.

- **Hypothesis 3:** The group defined as positive will show a relatively more positive sentiment compared to the benchmark.
- **Hypothesis 4:** The group defined as negative will show a relatively less positive (or more negative) sentiment compared to the benchmark.
- **Hypothesis 5:** The sentiment index built from the keyword groups will correlate with the S&P 500 index.

In the finance slang, positiveness is associated with the term 'Bullish' and negativeness to the term 'Bearish'. So, we defined the keywords adding 'Bullish', 'Bearish' and 'Neutral' adjectives to the base keyword 'S&P 500'[6]. The chosen words are:

Table 5.4: Keywords

Neutral Adjectives	**Bullish Adjectives**	**Bearish Adjectives**
S&P Index, S&P 500 Stable, S&P 500 Average, S&P 500 Moderate, S&P 500 Performance, S&P 500 Trends, S&P 500 Outlook	S&P 500 Up, S&P 500 Bullish, S&P 500 Undervalued	S&P 500 Down, S&P 500 Bearish, S&P 500 Overvalued

Notice that the keys are not balanced per group. This is due to the limitations that have been had in finding data for positive and negative keywords. As we did not want to leave aside the comparison of the sentiment index with the S&P, it was decided to artificially increase the registrations using neutral keywords.

First, it is crucial to make a dummy request with the selected parameters to each query. This allows us to get an approximate number of results via the `totalEstimatedMatche` parameter. Manually, we define when to end the process based on this parameter. Total Estimated Matches for each search term: S&P Index, (1.500.000), S&P 500 Stable (10.000), S&P 500 Average (53.800), S&P 500 Moderate (16.200), S&P 500 Up (82000), S&P 500 Bullish (42.100), S&P 500 Undervalued (27400), S&P 500 Bearish (68.200), S&P 500 Volatile (10.700), S&P 500 Crisis (7.080), S&P 500 Performance (58.700), S&P 500 Trends (21.700), S&P 500 Outlook (34.000). The grand total number of estimated matches is 1,931,880.

[6]In the background research, we studied several base keywords such as 'SP 500', 'Standard and Poor's 500', 'Standard and Poors', and its ticker '$SPX'. However, we found that 'S&P 500' served as the best baseline for the Bing API case.

5.3.1.3 Building the dataset

All in all, we are ready to obtain the dataset. From the `NewsArticle` we gahter the varialbes: **URL**, **description** and **datePublished** to form the register ith, we keep track of the keyword searched using a key, and we form a Pandas Dataframe with 4 columns. Due to API subscription tier limits, we couldn't scrape all estimated matches, so we randomized the selection for each keyword. To store the keywords, we assigned keys instead

```
searched_keys = {
  'S&P 500 Stable': 10000000,'S&P 500 Average': 10000001,'S&P 500 Moderate': 10000002,'S&P 500 Up': 10000003,'S&P 500 Bullish': 10000004,
  'S&P 500 Undervalued': 10000005,'S&P 500 Bearish': 10000006,'S&P 500 Volatile': 10000007,'S&P 500 Crisis': 10000008,'S&P 500 Performance': 10000009,
  'S&P 500 Trends': 10000010,'S&P 500 Outlook': 10000011
  }

  for i, term in enumerate(search_terms):
    download(term=term)
    print('Term: ', term, 'ready!')
  print('Finnished')
```

Ultimately, we gathered 109.700 `NewsArticles`, totaling over 4.6 million words. The term "S&P 500" (and its variations) appears nearly 100,000 times, stored in over 200 JSON files with a total size of 4.9 megabytes. The collection per term is as follows: S&P Index (18.287), S&P 500 Stable (3.962), S&P 500 Average (4.977), S&P 500 Moderate (3.020), S&P 500 Up (4.519), S&P 500 Bullish (11.572), S&P 500 Undervalued (5.266), S&P 500 Bearish (26.997), S&P 500 Volatile (4.670), S&P 500 Crisis (4.134), S&P 500 Performance (4.134), S&P 500 Trends (3.618), S&P 500 Outlook (4.334), or in terms of sentiment groups: Neutral (37.998), Positive (21.357) and Negative (35.801).

As we did in the MSN new example, finally we will calculate for every single row the 'compound' Vader sentiment, and we store the values in another column called 'Polarity'. In Table 5.5 we described the financial news dataset proposed.

Column Name	Column Type	Example
search key	int	10000012
url	string	https://seekingalpha.com/article/4683941-d-r-horton-housing-supply-shortage-u
description	string	D.R. Horton: Housing Supply Shortage Unresolved, Leaving Ample Room For Alpha. Since then, DHI has gained by 105.9%, far exceeding the 36.1% gain on the S&P 500 Index and the 79.0% gain on the SPDR S&P Homebuilders ETF. The underlying factors driving the acute housing
datePublished	datetime	2024-04-16
polarity	float	-0.12

Table 5.5: Financial news Dataset using Search Engines

Chapter 6

Exploratory Data Analysis

One of the worrying things about scraping in the Bing search engine through the API is the potential for skewed or unbalanced data with a concentration of data in recent periods. This bias can distort the analysis, making it difficult to draw meaningful conclusions or observe long-term trends.

The following bar plot, which shows the frequency distribution of collected dates, highlights this issue starkly. The distribution of dates for the non-duplicated data reveals a clear and significant concentration in the year 2024, with more than 65% of the data falling in this year alone. Instead of a gradual, consistent data collection over multiple years, we are left with a data set that disproportionately represents the most recent information, ultimately impacting the validity of our analysis and the conclusions we draw from it. It seems that search engines may be especially useful, in sentiment analysis techniques on streaming, only for the latest trends and trend topics. As we have seen during the parameterization of the API in subsection 5.3.1, Bing in fact allows making requests exclusively in this field.

Following with the Analysis, we may want to explore duplicate values, since if we aggregate sentiment for building the index this distortion the overall result. Continuing with the bad news, in the table 6.1 we see how many duplicate per groups and within groups. The number of duplicates is more than 95% in all groups and between groups, the number is even higher. This fact is completely remarkable and will deeply hamper future analyses. As we have commented in theory, we are completely dependent on the data that the server provides when we scrape. Consuming an API there is an even greater dependency, since we are bound by a contract. Of the 109,700 initial registrations, only 6807 remain usable.

Taking into account previous the date bar plot, we will consider only 2022 onwards. Also, there is an important consideration, we can have different values pf unique rows in the dataset: total unique rows, unique rows by keyword, unique

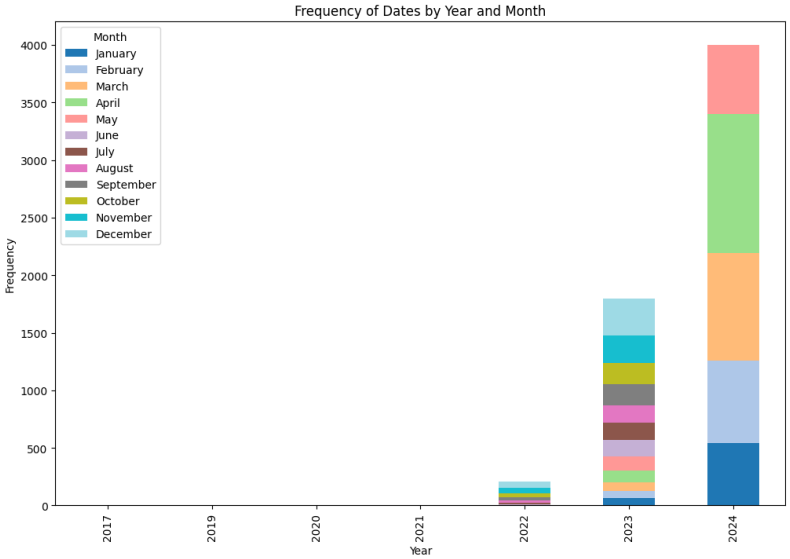

Figure 6.1: Dates Distribution

Table 6.1: Percentage of duplicates per group

	Neutral	**Positive**	**Negative**
Neutral	0.979974	NaN	NaN
Positive	0.978780	0.975934	NaN
Negative	0.983094	0.983197	0.987501

rows by categories. From now on it will be understood that if we are studying keywords, we are not interested in losing information and therefore let us apply unique rows stratifying by keywords, as in the case of categories or the total dataset (in this case, without grouping). Therefore, the total number of unique rows may vary.

Inspired by the concepts reviewed in subsection 1.1.3 concerning topics about news dataset formulations, we represented the "Top 5 Most Common Websites". Despite the apparent concentration of data on a few sources, we observe a more democratic and reliable collection process from highly trusted sources.

The largest portion, 62.4%, of the data comes from Reuters, followed by Nasdaq, contributing 33.4% of the data and Seeking Alpha with 3.3% representation. Overall we scraped 19 different websites, demonstrating a robust and democratic approach to data collection, and still this dataset leverage the trustworthiness and reliability of major financial news providers such as Reuters, Nasdaq, and Seeking Alpha.

Finally, an exploration on the remaining column, Description. The word cloud

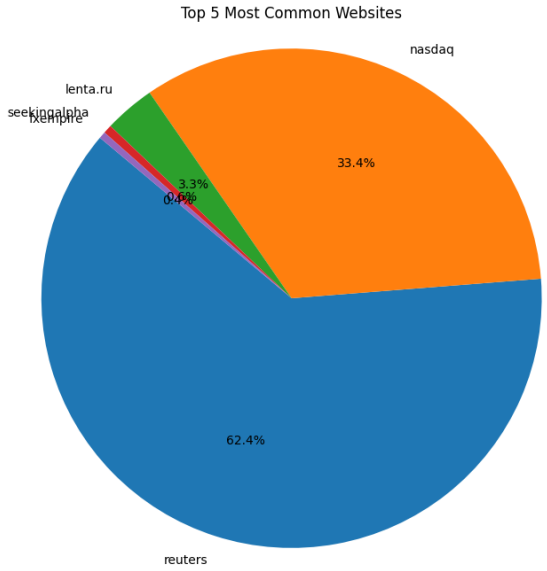

Figure 6.2: Most common website

visualization displays the most common words extracted from our dataset, and it is evident that the terminology aligns well with financial slang with words such as ”stock,” ”etf,” ”report,” ”trading,” ”shares,” and ”market”.

However, it also reveals areas where text cleaning could be improved. Some words like ”click,” ”free,” and ”zacks” seem out of context or carry no significant meaning in financial analysis. These extraneous terms suggest that the text preprocessing step could be refined to remove irrelevant or less meaningful words.

The importance of words in this context cannot be overstated, especially when it comes to determining polarity in sentiment analysis, as commented in 1.1.3 accuracy on data science project remains most of the time on processed data than in the model itself. Therefore, improving text cleaning will enhance the precision of our sentiment analysis, leading to more reliable and insightful conclusions.

Figure 6.3: Most common Words on News

Chapter 7

Results and discussion

7.1 Result

On this chapter we will discuss results, once we proceed with the preprocessing and the sentiment assignation. We can start assessing summary polarity measures per keyword.

Table 7.1: Summary statistics by Keywords

Search Key	Mean	Median	Std	Count
S&P 500 Stable	0.14	0.00	0.22	518
S&P 500 Average	0.15	0.00	0.23	464
S&P 500 Moderate	0.15	0.00	0.22	516
S&P 500 Up	0.73	0.78	0.20	545
S&P 500 Bullish	0.72	0.78	0.21	570
S&P 500 Undervalued	0.72	0.77	0.21	548
S&P 500 Bearish	-0.50	-0.46	0.19	190
S&P 500 Volatile	-0.49	-0.48	0.19	224
S&P 500 Crisis	-0.47	-0.46	0.26	246
S&P 500 Performance	0.17	0.00	0.23	484
S&P 500 Trends	0.16	0.00	0.23	533
S&P 500 Outlook	0.15	0.00	0.29	577
INDEX SP500	0.16	0.00	0.27	604

From the table we can infer that tags are important when searching for SP500 news data: Terms like ”S&P 500 Up,” ”S&P 500 Bullish,” and ”S&P 500 Undervalued” exhibit high mean polarity scores, indicating strong positive sentiments. On the other hand, ”S&P 500 Bearish,” ”S&P 500 Volatile,” and ”S&P 500 Crisis” show negative mean polarity scores, suggesting significant concern and pessimism during market downturns or instability. The ”S&P 500 Stable,” ”S&P 500 Average,”

"S&P 500 Moderate," "S&P 500 Performance," "S&P 500 Trends," and "S&P 500 Outlook" terms have mean scores that hover around neutral, indicating a balanced sentiment without a strong inclination towards either positivity or negativity.

The positive sentiment category has an average mean polarity of 0.657, a median close to 0.748, and a standard deviation of about 0.305. The neutral sentiment category shows an average mean polarity of about 0.163, with median values at 0 and a standard deviation around 0.280, indicating a slightly positive sentiment with low to moderate variability. The negative sentiment category exhibits an average mean polarity of approximately -0.382, median values around -0.427, and a standard deviation of about 0.359, reflecting significant pessimism with higher variability.

The analysis reveals that negative terms such as "S&P 500 Bearish," "S&P 500 Volatile," and "S&P 500 Crisis" have lower counts compared to other tags. This lower frequency of negative terms suggests a generally positive market environment. Moreover, neutral tags tend to show slightly positive mean values, indicating that even neutral sentiments lean towards positivity. Overall, we can infer that the market sentiment for the S&P 500 data collected is positive. This conclusion aligns with the distribution of the collected data dates and the market's tendency during the period: we could primarily scrap data within 2023 and 2024 (with some data from mid 2022) and SP500 tendency was bullish during that time.

For a better overall polarity sentiment between groups, the plot 7.1 shows a kernel density estimate (KDE) plot that displays the distribution of polarity values for three categories: Positive, Negative, and Neutral. The Positive category, shown in blue, has a mean polarity of 0.66, the Negative category, represented in red, has a mean polarity of -0.38, and the Neutral category, depicted in green, has a mean polarity of 0.16. There is a remarkable pattern in Neutral distribution. There are concentrated values around polarity 0 and the rest values have a positive bias. This is caused mainly because, VADER cannot identify lexicons in neutral news and then the polarity sentiment is 0 (since we have non-positive/non-negative lexicons to add). For example for keyword "SP 500 index", we have a new with the text: "US CPI Rose 3.2% Annually in July, Less than Expected 24 Apr 2024 20:19:05 GMT NASDAQ Index, SP500, Dow Jones Forecasts - SP500 Is Flat Despite Tesla's RallyWed, 24 Apr 2024 19:51:02 GMT Shelter drove 90% of inflation, while excluding food and ...". Vader can not identify any word as a lexicon, due to its limitation in the financial domain, so it assigns a polarity value of 0.

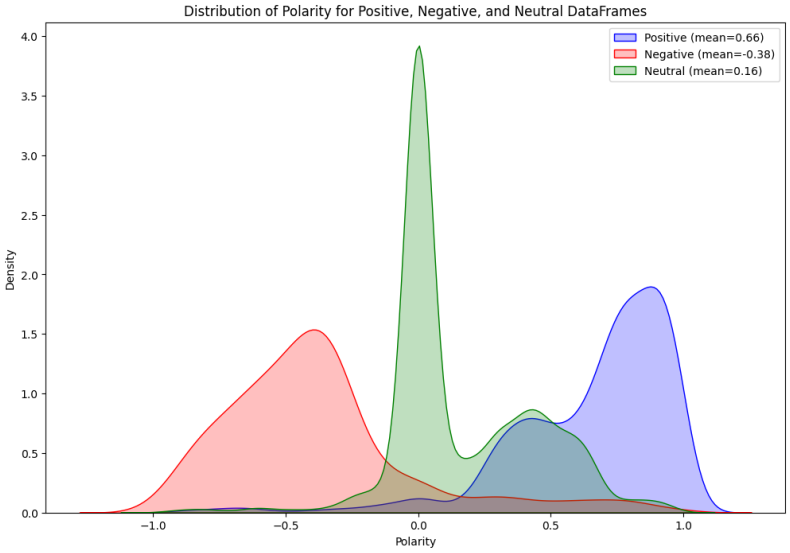

Figure 7.1: Polarity distribution by category

To test if categories are statistically different we may performa an ANOVA test. The ANOVA (Analysis of Variance) test is a statistical method used to compare the means of three or more groups to determine if at least one of the group means is significantly different from the others. The null hypothesis (H_0) states that all group means are equal:

$$H_0 : \mu_1 = \mu_2 = \cdots = \mu_k$$

The alternative hypothesis (H_a) states that at least one group mean is different:

$$H_a : \text{At least one } \mu_i \text{ is different}$$

To calculate the F score, we use the ratio of the variance between the groups to the variance within the groups. The formula for the F score is:

$$F = \frac{\text{MS}_{\text{between}}}{\text{MS}_{\text{within}}}$$

where $\text{MS}_{\text{between}}$ (mean square between) is calculated as:

$$\text{MS}_{\text{between}} = \frac{\text{SS}_{\text{between}}}{k - 1}$$

and $\text{MS}_{\text{within}}$ (mean square within) is calculated as:

$$\text{MS}_{\text{within}} = \frac{\text{SS}_{\text{within}}}{N - k}$$

Here, $\text{SS}_{\text{between}}$ is the sum of squares between the groups, $\text{SS}_{\text{within}}$ is the sum of squares within the groups, k is the number of groups, and N is the total number of observations.

The ANOVA F-value = 3887.10, with a P-value = 0.0 so we can assure that the differences between the means are statistically significant.

7.1.1 Description and Commentary on the SP500 Chart

The provided chart displays the S&P 500 performance from January 2022 to mid-2024, marked with six red dashed lines indicating the end of downtrends and five green lines marking local maximums or the ends of uptrend.

Figure 7.2: SP500-Time Lapses

From January 2022 to October 2022, the S&P 500 experienced a clear downtrend, significantly influenced by the Federal Reserve's monetary policy on combating rising inflation. For instance, the Fed raised the federal funds rate from near zero at the beginning of 2022 to a range of 3.00% to 3.25% by September 2022, marking the most aggressive pace of rate increases since the early 1980s .

[1] From that period, we see a strong recovery of the index, raising to historical maximums on previous months. Overall, this period shows a strong recovery with a bullish trend before late 2022 in the S&P 500.

This bullish trend can be attributed to various macroeconomic conditions, including a switch in monetary policies after slowing inflation rates, a stable commodity

[1] S&P 500 in 2024: Here's what Wall Street predicts, `https://finbold.com/`

market, and strong employment rates. The Federal Reserve's indication of potential rate cuts contributed to this optimism, leading to a substantial 25& advance in the S&P 500 during 2023, pushing the index near its historical highs (Morgan Stanley).

In addition to favorable monetary policies, the U.S. labor market remained robust, with unemployment rates hovering around historically low levels [2]. Moreover commodity prices remain low relative to slumpy volatile prices seeing during Covid. For instance, the Brent crude oil prices are expected to remain stable, averaging around $83 per barrel in 2024, which helps keep inflation in check and supports economic stability.

Despite occasional downtrends, as indicated by the red dashed lines, the market recovered and reached new local maximums, signifying overall investor optimism. Analysts predict that the S&P 500 will continue to benefit from these factors, with forecasts suggesting targets as high as 5,200 by the end of 2024, assuming continued economic stability and moderate inflation.

Period	Period Identifier	Freq_Positive	Freq_MuchPositive
2022-01-04 - 2022-03-14	Positive	0	0
2022-01-04 - 2022-05-22	Negative	2	1
2022-03-30 - 2022-06-20	Positive	6	2
2022-05-31 - 2022-10-16	Negative	34	10
2022-08-16 - 2023-10-30	Positive	269	73
2023-08-01 - 2024-04-22	Negative	250	48
2024-03-31 - 2024-06-01	Positive	63	3

Freq_Neg	Freq_MuchNeg	Prop_Pos	Prop_Neg	Mean	Std	Min	Max
4	2	0	1	-0.62	0.29	-0.92	-0.34
13	9	0.13	0.87	-0.44	0.48	-0.92	0.87
11	6	0.35	0.65	-0.16	0.51	-0.88	0.87
16	5	0.68	0.32	0.17	0.41	-0.84	0.83
77	20	0.78	0.22	0.23	0.36	-0.90	0.90
15	1	0.94	0.06	0.35	0.20	-0.52	0.85
0	0	1	0	0.33	0.12	0.05	0.79

Table 7.2: Time-Lapse polarity summarization

At the beginning of 2022, when the Federal Reserve commenced its rate hikes to combat rising inflation, the S&P 500 started to collapse, and the polarity index

[2] 2024 outlook: Navigating the last mile of the cycle, Edward Jones, https://www.edwardjones.com/us-en/market-news-insights

reflected significant pessimism in the markets. During the period from January 4, 2022, to March 14, 2022, the polarity index had a mean of -0.6199, indicating a clear negative sentiment. The frequency of positive news was zero, with a proportion of negative sentiment at 1, highlighting the market's adverse reaction to the Fed's policy changes. Still, remark the scarcity of data before 2023.

Parallel to S&P 500 recovery, the polarity index also showed a gradual increase in optimism. For instance, from August 16, 2022, to October 30, 2023, the polarity index had a mean of 0.2261, with a significant increase in the proportion of positive sentiment to 0.7775. This shift suggests that as the market adapted to the new monetary policy, investor sentiment became more positive.

Despite these trends, the polarity index demonstrated low sensitivity to rapid market changes. For example, during the period from May 31, 2022, to October 16, 2022, marked as 'Negative,' the polarity index mean was 0.1694, which still indicated a somewhat positive sentiment despite the market downturn. The frequency of negative news increased to 16 from 11 in the previous period, but the proportion of positive sentiment remained at 0.68, suggesting that the index took time to fully reflect the negative market conditions.

Additionally, as we move to more recent dates, the standard deviation of the polarity index decreases. For example, during the period from March 31, 2024, to June 1, 2024, the standard deviation was 0.1168, compared to 0.4785 earlier in the timeline. This reduction in variability is likely due to the increased availability of data. The increase in data points over time helps to smooth out short-term fluctuations, resulting in a more reliable polarity index.

7.1.2 Line Plot

The three line plots offer different perspectives on the relationship between the Index Polarity and the S&P 500 over the same date periods, enhancing our understanding of market sentiment and its impact on stock performance.

The first plot presents a raw comparison between the Index Polarity (left axis) and the S&P 500 (right axis) from early 2021 to mid-2024. Throughout this period, the polarity index shows consistent positive values, reflecting stable optimism among newsreaders. This optimism persists despite fluctuations in the S&P 500, suggesting that market sentiment is resilient and not overly sensitive to short-term market changes.

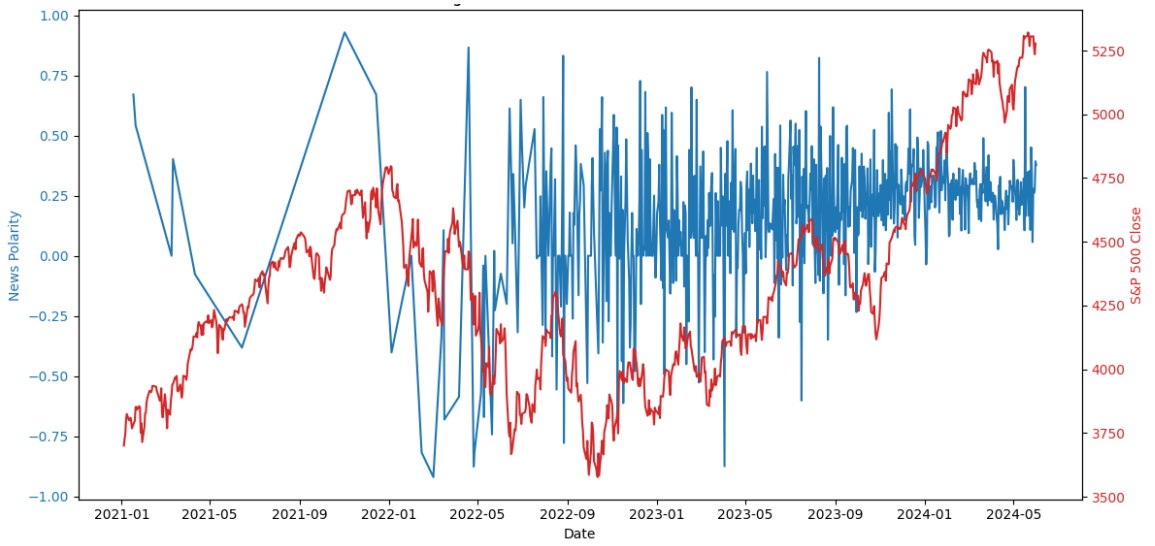

Figure 7.3: SP500 vs Polarity Index

There are noticeable periods where the S&P 500 dips, such as during the early 2022 downturn. Despite these downturns, the polarity index does not show drastic negative values, indicating that while market prices fell, overall sentiment remained cautiously optimistic. Data Density Over Time:

The long-term upward trend in the S&P 500 aligns with the generally positive polarity index, reinforcing the relationship between positive market sentiment and rising stock prices. Still the intraday volatility of Polarity index prevent us from taking remarkable observations. This could be caused by nor identifying entity tendency [reference]

The second plot presents a 30-day moving average for the Index Polarity and the S&P 500, which smooths out short-term fluctuations and provides a clearer view of the underlying trends over time.

This plot reveals a more pronounced correlation between the Index Polarity and the S&P 500. As the Index Polarity trends upward, the S&P 500 generally follows suit, and vice versa. This enhanced correlation is due to the smoothing effect of the moving average, which highlights the overall direction and momentum of sentiment and market performance.

Lead-Lag Relationship:

A notable observation is that local maxima in the Index Polarity often precede those in the S&P 500. For example, periods where the Polarity Index peaks can be seen slightly earlier than the corresponding peaks in the S&P 500. This suggests that changes in sentiment might serve as a leading indicator for market move-

Figure 7.4: SP500 vs 30-day Moving Average Polarity Index

ments. As commented, during the recovery phase from mid-2022 onwards, the polarity index shows a gradual and consistent rise in parallelism with SP500.

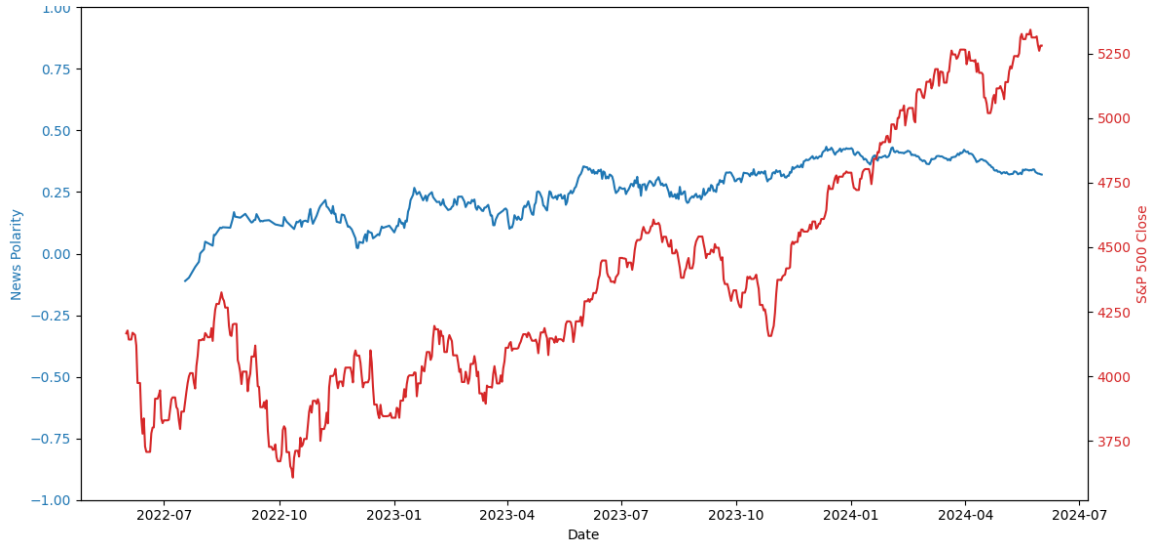

Figure 7.5: SP500 vs 30-day Moving Average Polarity Index full scale

The third plot, similar to the second, uses a 30-day moving average for the Index Polarity and the S&P 500 but with an expanded axis for the Polarity Index to its maximum and minimum values. With the expanded axis, the Polarity Index shows a slightly positive tendency over the long term, as seen in 7.3. Recent Decrease in Polarity:

Notably, in the latest periods of 2024, the Polarity Index starts to enter on a cool-

ing/lateral tendency phase. This downward trend in sentiment is concurrent with observations in the S&P 500, where the most recent peak is lower compared to the previous peak, indicating a potential weakening of market momentum and could indicate an imminent correction. This is supported by the cooling effect in the Polarity Index, reflecting growing investor caution and potentially predicting a

7.1.3 Moving Average Polarity Stratified by category

The plot showcases the moving average polarity of news sentiment stratified by positive, neutral, and negative tags, alongside the S&P 500 index. This stratification provides a detailed view of how different sentiment tags correlate with market performance over time.

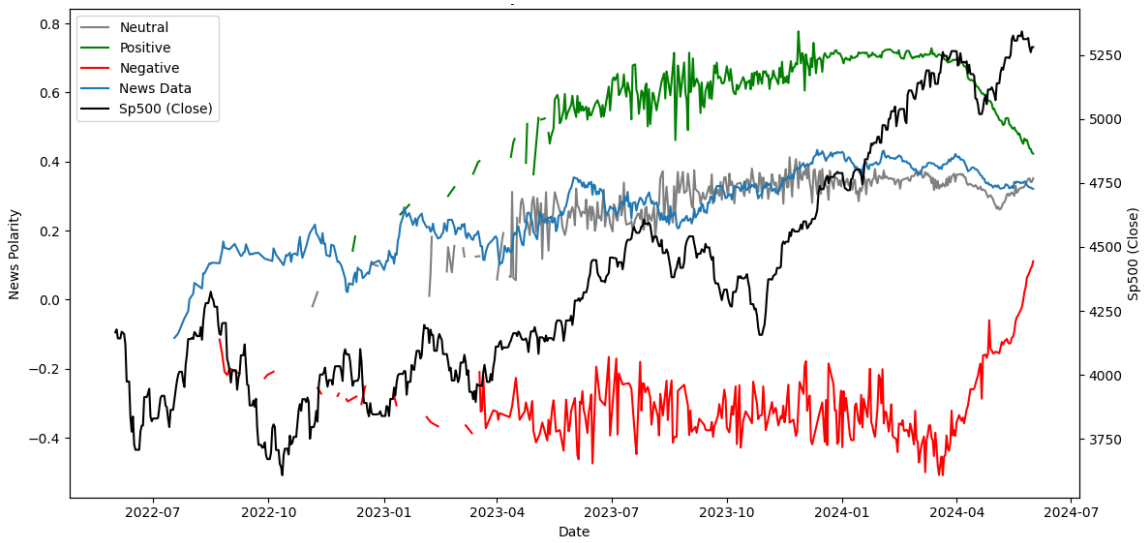

Figure 7.6: SP500 vs Polarity Index by Category

Positive Sentiment: Represented by the green line, positive news tags show a generally bullish sentiment. Shown in gray, neutral tags exhibit slightly positive sentiment, reflecting the overall tendency of the S&P 500 to increase during this period. This confirms that even news categorized as neutral tends to lean positive, consistent with the overall market trend. Illustrated by the red line, negative news tags correlate with bearish sentiment. This is evident in the plot where downturns in the S&P 500 align with spikes in negative polarity, demonstrating the search engine's capability to identify market pessimism accurately. Impact of Data Volume:

The stratified polarity indices are less smooth compared to the aggregated polarity index due to the lower volume of data for each sentiment type. This lower data

density can cause more significant fluctuations and less stable trends within each category, particularly in the older periods where data availability is reduced.

The performance of the search engine in capturing sentiment through specific tags is noteworthy. Positive tags consistently yield bullish news, negative tags correspond to bearish news, and neutral tags result in slightly positive sentiment. This effective categorization indicates that the tagging system is robust in reflecting true market sentiment.

However, in the latest periods of 2024, the sentiment trends become less clear, with the polarity indices switching roles and showing erratic behavior. This strambolic sentiment may reflect the market cooling down, as indicated by the S&P 500's relatively lower maximums and increased volatility.For instance, Morningstar forecasts GDP growth to slow significantly, partly due to higher interest rates impacting consumer spending and commercial real estate projects (Morningstar).

7.1.4 Logarithmic Returns

On the way of comparing sentiment and market indices, we would also like to review its behavior on a portfolio management view, using logarithmic return. Logarithmic returns are a fundamental concept in financial analysis, used to measure the rate of return of an asset over a period of time. Unlike simple returns, logarithmic returns have the advantage of being additive over multiple periods, which simplifies the analysis of time series data. This property makes them particularly useful for analyzing stock prices and understanding market trends.

$$r_t = \ln\left(\frac{P_t}{P_{t-1}}\right)$$

When we first obtained the polarity graph, it seemed to remind us of a graph of logarithmic returns. In the graph 7.7 we found, on a double scale, the polarity indicator and the logarithmic returns for the year 2022 onwards. Both polarity and returns suffer from high volatility in early 2022, although this may be due to a lack of data in the case of polarity. As we obtain more data, both indicators become decoupled and while polarity has an upward trend, returns remain positive after October 2023 but without observing that trend

We are interested in the study of in the study between the similarities of returns and polarity. The plot 7.8 presented shows the moving correlation between the S&P 500 daily returns and the sentiment index over time. Initially the correlation was negative until October 2022. This negative correlation suggests that during

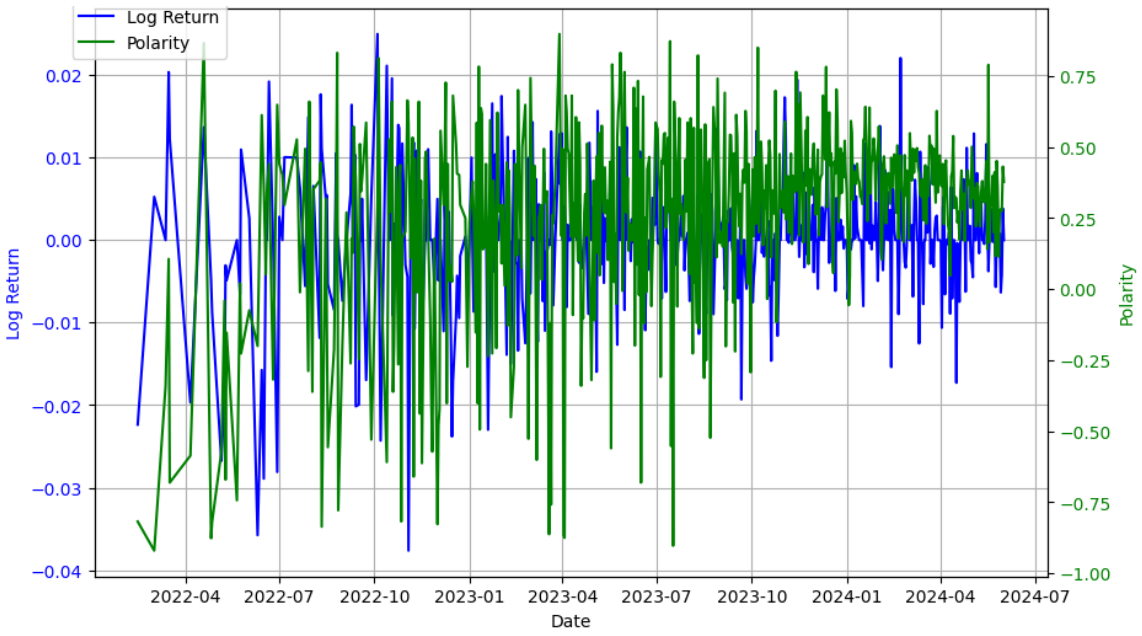

Figure 7.7: SP500-Time Lapses

the S&P 500's significant downturn in 2022, investor sentiment was inversely related to market performance. After that date, the correlation reverses and becomes positive indicating that both the S&P 500 and the sentiment index started moving in the same direction.

The correlation plot exhibits considerable volatility, indicating that the relationship between the S&P 500 returns and the sentiment index is not stable over time.

The overall correlation between the S&P 500 daily returns and the sentiment index is 0.22. This relatively low correlation suggests that while there is some relationship between market returns and sentiment, it is not strong or significant enough to be considered a reliable predictor on its own, primarily due to the lack of data, particularly in the earlier periods. As more data becomes available over time, the correlation becomes more stable and shifts towards a positive trend.

Finally, a plot illustrates the moving standard deviation between the Polarity Index and the S&P 500 daily returns over time. This plot helps us understand the variability and stability of both the sentiment index and market returns.

At the beginning of the observed period (mid-2022), both the Polarity Index and the S&P 500 returns exhibit high standard deviation values. From mid-2022 onwards suggests that as the market and sentiment stabilize, the variability in returns and sentiment decreases, indicating a more stable market environment. In the latest periods shown in the plot (2024), the standard deviations for both indices

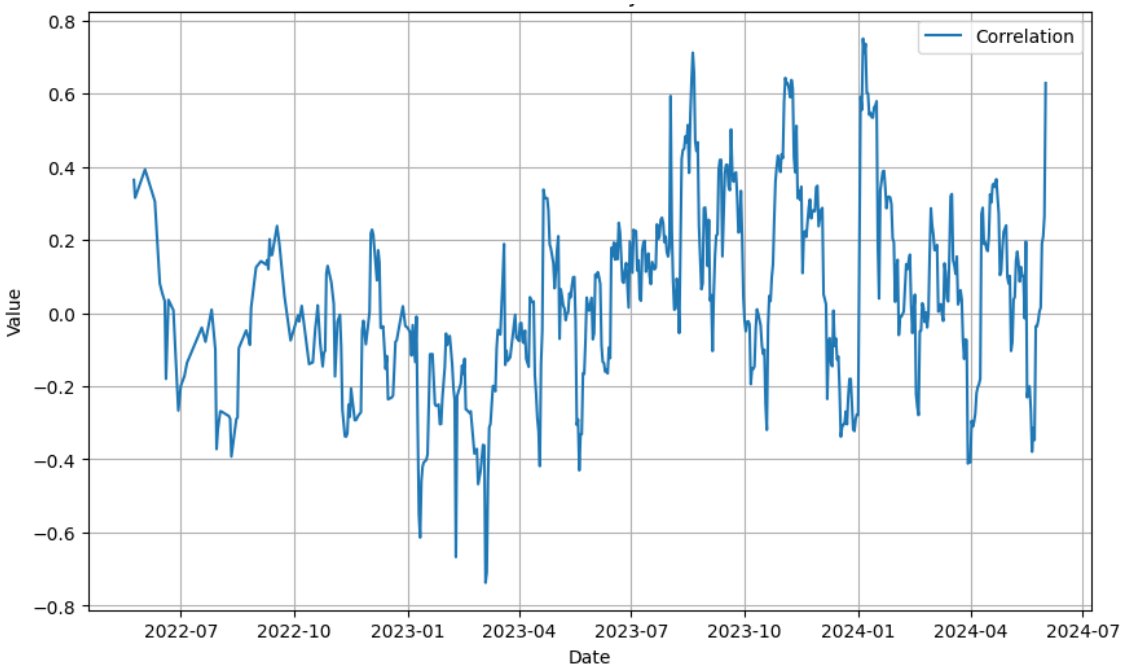

Figure 7.8: Logarithmic returns and Polarity correlation, Moving Average

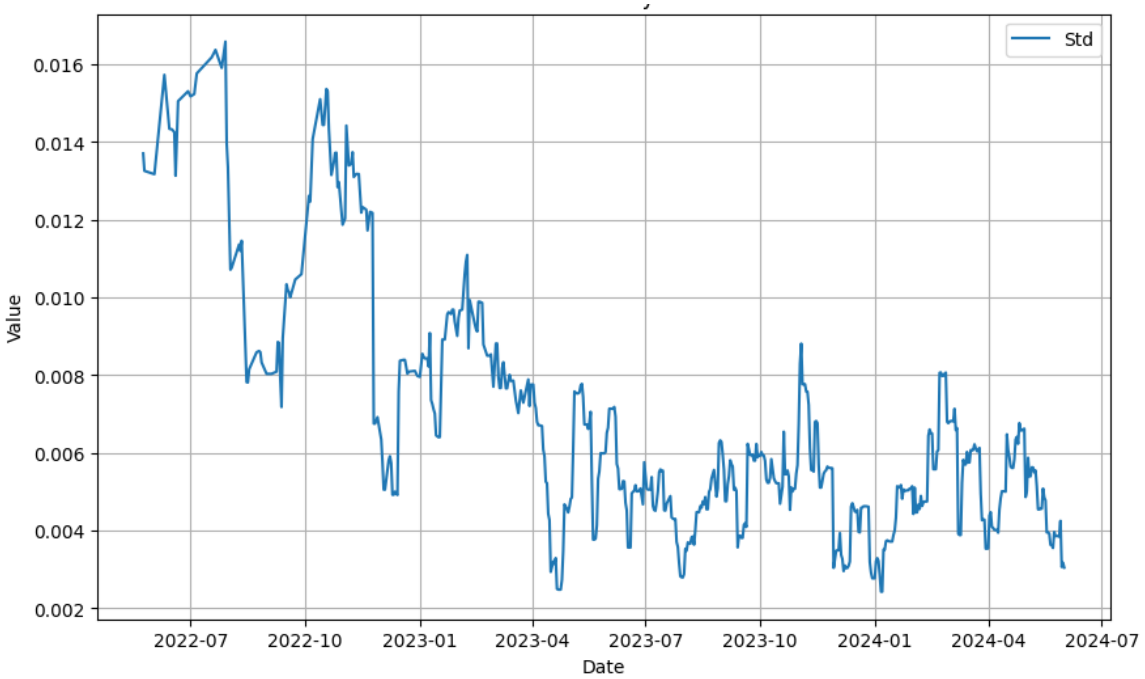

Figure 7.9: Logarithmic returns and Polarity deviation, Moving Average

appear to be at their lowest levels.

Chapter 8

Conclusions

The modern world is increasingly driven by data, with consumers, companies, public institutions, and other entities continuously generating vast amounts of information. While Big Data presents a complex challenge, when properly managed, it can yield valuable insights that inform competitive decision-making and create value across various industries. This study specifically focuses on the realm of textual financial data.

The exponential growth in available and continuously generated information necessitates an evolution of traditional data science methodologies. To thrive in this highly dynamic environment, more flexible approaches are required. A new wave of statistical analysis is emerging, leveraging innovative machine learning techniques that utilize diverse data types beyond traditional quantitative analysis. These models can learn from various inputs, including images, audio, and extensive text, yielding promising results that are garnering significant attention. It is imperative for the financial sector to adapt and modernize its quantitative methods to stay abreast of these advancements and not fall behind.

Data science in finance, particularly quantitative finance, has always been fraught with significant challenges. Successful backtesting requires precise data with no leakage to prevent overfitting when assessing past returns. In the current literature, building news datasets often involves scraping specific domains, resulting in complex and difficult-to-maintain web crawlers or outdated datasets. Given that the Internet has become the quintessential data source, with a vast universe of information being continuously shared awaiting to be retrieved, we proposed search engines for collecting and storing financial news. Search engines provide constant, real-time updates and have demonstrated high predictive capacity for economic events by aggregating information from various sources instead of relying on individual newspapers. A key differentiator of our proposed method is its flexibility

in defining keywords for specific data collection. This allows us to employ various strategies to gather information from diverse sources, providing a comprehensive view of the current market status. For instance, we can collect data on macroeconomics, microeconomics, specific markets, and politics, at the same time which can then be aggregated to build an indicator. Since this study focuses on describing this technique rather than prediction, we defined three groups of sentiment-based keywords (positive, negative, and neutral) to analyze their differences and particularities with the aim of understanding how the Search Engines behaves as data sources.

We used the Bing API, a well-known Microsoft search engine, to scrape over 100,000 news entries from more than 90 different websites domains based on 12 defined keywords. During the data exploration process, we found that the API performs poorly for collecting historical data but works well for trending topics news. Notably, over 95% of the entries were duplicate values. Therefore, we conclude that the benefits of using an API, such as speed and simplicity, are not worthwhile for building a robust financial news dataset if the goal is to obtain valuable insights from the data. Thus, with the 6,000 remaining unique data entries, we decided to analyze it using sentiment analysis.

The landscape of sentiment analysis models has advanced significantly, with the rise of sophisticated tools like large language models (LLMs) and chatbots. However, for our study, we opted to use lexicon-based models rather than machine learning models. Lexicon models are straightforward and transparent, unlike the ”black box” nature of machine learning models. This transparency is crucial when exploring new data-collection techniques, as it allows us to understand the data features contributing to the results, prioritizing clarity over predictive accuracy. The Valence Aware Dictionary and Sentiment Reasoner (VADER) is the best open-source lexicon for sentiment analysis, often outperforming typical machine learning models like SVM or Naive Bayes. However, applying VADER in finance presents challenges, as it is designed for generic use and lacks a rigorously tested financial fine-tuning. This mirrors the broader difficulty in finding effective sentiment analysis tools tailored specifically for the financial sector, where specialized alternatives are consistently harder to identify.

We build a sentiment indicator, *”Polarity”*, to evaluate the dataset with two types of analysis. First, we can assess the diversity and richness of information provided by the keywords by comparing sentiment both between and within the defined keyword categories. Second, we can contribute to studies on collecting investor

sentiment and comparing it with market trends; in this case we compared the formed index against the S&P 500. In this study, we have focused on descriptive analysis rather than a predictive modeling. Our hypothesis is that search engines can provide differentiated information based on various keywords, making this a promising field for further research. However, defining predictive keywords within each group is complex, and the content often overlaps. Ultimately, we believe that alternative data sources, such as financial news, can offer valuable insights into the market.

Analyzing the results, consider that by choosing to study the dataset using sentiment analysis, we address two significant biases simultaneously. First, we assume that the news content is reliable, a factor that is extremely complex to evaluate using scraping techniques (since we often do not know the exact nature of the collected content). Second, we assess that the sentiment is accurately extracted from the news content, which is not always true, especially in news sentiment analysis. Still, we concluded that the diversity of the content provided by the search engine is highly valuable. Despite only having news data from the last two years (2022-2024), an extremely profitable period for the stock market, we observed a clear differentiation in sentiment, with negative news distinctly separated from positive and neutral news. Although we may not have chosen the optimal keywords for each group, there were no substantial differences within keywords of the same group. Considering the data quality issues and the lack of correlation between the general index and the S&P 500, we conclude that sentiment based on news captures, at least, the broader market trends.

This study is a timely opportunity to reevaluate and incorporate new techniques in dataset building and modeling within the financial sector. Despite the challenges, we found the concept of defining keyword strategies to extract "custom" data from the vast Internet extremely useful. We encourage researchers to explore this approach further, using richer web scrapers and more complex models to enhance the quality and utility of financial data.

Appendix A

Understanding transformers

Transformers utilize the self-attention mechanism for modeling, enabling them to encode long text sequences, unlike LSTM networks which have memory and computational limitations. By replacing LSTM networks with a complete Attention structure, Transformers address the sequence-to-sequence problem more effectively, achieving superior results and reducing computational complexity.

Example:

Let's take the sentence: "The movie was incredibly good."

Feature Extraction:

- **Tokenization:**
  - Tokens: ['The', 'movie', 'was', 'incredibly', 'good', '.']
- **Self-Attention Mechanism:**
  - Self-attention allows each token to focus on other tokens in the sentence. For example, 'good' would consider the context of 'The', 'movie', 'was', and 'incredibly' to understand its full meaning.
  - For 'good', the self-attention scores might be higher for 'incredibly' and 'movie' as they provide relevant context.

Explanation:

In this example, the self-attention mechanism enables the Transformer to understand the relationship between 'good' and 'incredibly' as well as 'movie', ensuring that the sentiment associated with 'good' is accurately captured.

A special Transformer model named **BERT** (Bidirectional Encoder Representations from Transformers) was proposed by Google Research in 2018. BERT consists of Transformer Encoder layers, supplemented by word and positional encoding, to

encode the syntax and semantics of text comments and uses contextual information to produce token-level representations.

Example:

Consider the sentence: "The movie was incredibly good."

Feature Extraction:

- **Tokenization:** Tokens: ['The', 'movie', 'was', 'incredibly', 'good', '.']
- **Positional Encoding:** Each token is assigned a position in the sequence to maintain the order:
  - 'The' $\rightarrow$ position 1
  - 'movie' $\rightarrow$ position 2
  - 'was' $\rightarrow$ position 3
  - 'incredibly' $\rightarrow$ position 4
  - 'good' $\rightarrow$ position 5
- **Bidirectional Context:** BERT considers the context from both directions (left-to-right and right-to-left). For 'good', it understands that 'The movie was incredibly' influences its meaning, and it also checks the preceding context for comprehensive understanding.

Explanation:

BERT's bidirectional approach ensures that the token 'good' is interpreted with the full context of the sentence, both preceding and following tokens, providing a richer representation of meaning.

Moreover, BERT infuses auxiliary sentiment knowledge by incorporating sentiment contextual information into language representation models. The contextual word embedding of this language model includes inter-sentence relationships and understanding the context of the entire comment, preserving the semantic meaning of words across various domains.

Example:

Consider two sentences:

- "The movie was incredibly good."
- "I would recommend it to everyone."

Feature Extraction:

- **Inter-sentence Relationships:**
  - BERT models the relationship between these sentences. The sentiment of the first sentence affects the understanding of the second.
  - 'Recommend' in the second sentence is influenced by 'good' in the first, reinforcing positive sentiment.

Explanation:

BERT's ability to incorporate inter-sentence relationships means it understands that the positive sentiment in the first sentence enhances the recommendation in the second, capturing a comprehensive sentiment representation across the entire comment.

According to Google Research, one of the standout features of BERT is its pre-training on large text corpora like Wikipedia, using a **masked language model** to predict missing words and a **next-sentence prediction task** to understand the relationship between sentences. This pre-training allows BERT to be fine-tuned on smaller datasets for specific tasks like sentiment analysis, achieving state-of-the-art results by leveraging its deep, bidirectional understanding of language contexts.

Appendix B

Rendering HTML

When a browser starts to render a page, it first transforms the HTML code into a DOM tree. This process includes two main activities:

- **HTML tokenization**: Transforming input text characters into HTML "tokens".
- **DOM tree building**: Transforming HTML tokens from the previous step into a DOM tree.

When the browser receives an HTML document from the server, everything is transfered as raw bytes. Thus, to decode those bytes into readable text characters, the browser will first run the encoding sniffing algorithm to detect the document's encoding. UTF-8 is the most commonly used character encoding on the web because it supports a wide range of characters from various languages.

The parsing process can be subdivided into lexical analysis and syntax analysis. Lexical analysis involves breaking the input into tokens, representing the language's vocabulary or valid building blocks, akin to words in a human language dictionary. Syntax analysis, on the other hand, applies the language's syntax rules.

For example, parsing the expression 2 + 3 - 1 could return this tree:

After the stream of bytes is decoded into a stream of characters, it's then fed into an HTML tokenizer. The tokenizer is responsible for transforming input text characters into HTML tokens. There are fives types HTML tokens:

- **DOCTYPE**: Represent and contain information about the document doctype. Yes, that useless `<!DOCTYPE html>` isn't as useless as you think.
- **Tag**: Represent both start tag (e.g. `<html>`) and end tag (e.g. `</html>`).

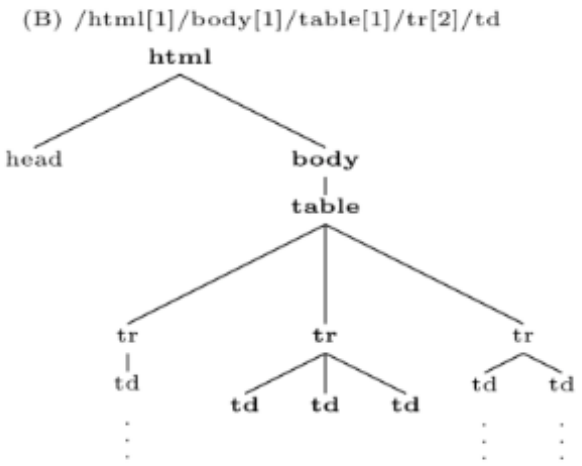

Figure B.1: DOM schema *Source:* Ferrara et al. (2014)

- **Comment**: Represent a comment in the HTML document.
- **Character**: Represent a character that is not part of any other tokens.
- **EOF**: Represent the end of the HTML document.

Appendix C

Python Code

This appendix contains only referenced Python Code. Found all Python utilities in GitHub-RamonCoronado

C.1 Query @Function()

```python
import json
import bs4
import requests


def _request_with_cooloff(url: str, api_usage: bool, num_attempts: int, **kwargs):
    """
    Call the url using requests. If the endpoint returns an error wait a cooloff
    period and try again, doubling the period each attempt up to a max num_attempts.

    url: the URL to call
    usage: Define if we need a request for a Web URL or an API
    num_attempts: The number of attemps before canceling conexion
    **kwargs: arguments for API autentication -> headers and params
    """
    cooloff = 1
    response = None
    call_count = 1
    while call_count <= num_attempts:
        try:
            response = requests.get(url, **kwargs, timeout=360)
            response.raise_for_status()
            call_count = num_attempts + 1
        except requests.exceptions.ConnectionError as e:
            if call_count != (num_attempts - 1):
                time.sleep(cooloff)
                cooloff *= 2
```

```
28                  call_count += 1
29                  continue
30              else:
31                  if response is not None:
32                      return f"ERROR!: {response.status_code}"
33                  else:
34                      return f"ERROR!: 444"  # Max retries exceeded with
    url
35          except requests.exceptions.HTTPError as e:
36              if response.status_code == 404:
37                  return "404 error!:"
38
39              if call_count != (num_attempts - 1):
40                  time.sleep(cooloff)
41                  cooloff *= 2
42                  call_count += 1
43                  continue
44              else:
45                  if api_usage:
46                      return response
47                  else:
48                      return f"ERROR!: {response.status_code}"  # Return
    an error message if not using JSON
49          if api_usage:
50              return response
51          else:
52              soup = bs4.BeautifulSoup(response.text, "html.parser")
53              return soup
54
55  def request_with_cooloff(url: str, api_usage: bool, num_attempts: int =
    3, **kwargs):
56      """
57      Call the url using requests. If the endpoint returns an error wait a
        cooloff
58      period and try again, doubling the period each attempt up to a max
    num_attempts.
59
60      url: the URL to call
61      usage: Define if we need a request for a Web URL or an API
62      num_attempts: The number of attemps before canceling conexion
63      **kwargs: arguments for API autentication -> headers and params
64      """
65      result = _request_with_cooloff(url, api_usage, num_attempts, **
    kwargs)
66      return json.loads(result.content.decode("utf-8")) if api_usage else
    result
67
68  def bing_query(query: str, offset:int) -> dict:
69      """
70      Provides necessary Bing API parameters and headers to perform a
    request_with_cooloff() request.
71
72      query: The word to search on the engines.
73      numresults: The number of links to display in the query.
```

```
    """
    BING_KEYS = "YOUR SUBSCRIPTION KEY"
    SEARCH_URL = "https://api.bing.microsoft.com/v7.0/news/search"
    try:
        api_usage = True
        params = {
            "q": query,
            "mkt":"en-US",
            "count": 100,
            "offset": offset,
            }
        api_key = BING_KEYS
        headers = {"Ocp-Apim-Subscription-Key": api_key}
        response = request_with_cooloff(
            url=SEARCH_URL, api_usage=api_usage, headers=headers, params
=params
        )
        return response
    except Exception as e:
        print(f"Error in bing for {query}: {e}")
        return f"ERROR!: {response.status_code}"
```

C.2 Download @Function()

```
import json
import bs4
import requests

def download(term, total):
    for offset in range(0, total, 100):
        data = bing_query(term, offset)
        save_json(data, term, offset)
        time.sleep(0.55)
```

Bibliography

D. H. Bailey, J. M. Borwein, M. L. de Prado, and Q. J. Zhu. Pseudomathematics and financial charlatanism: The effects of backtest over fitting on out-of-sample performance. *Notices of the AMS*, 61(5):458–471, 2014.

D. H. Bailey, S. Ger, M. Lopez de Prado, and A. Sim. 20 - statistical overfitting and backtest performance. In E. Jurczenko, editor, *Risk-Based and Factor Investing*, pages 449–461. Elsevier, 2015. ISBN 978-1-78548-008-9. doi: https://doi.org/10.1016/B978-1-78548-008-9.50020-4. URL https://www.sciencedirect.com/science/article/pii/B9781785480089500204.

S. Behdenna, F. Barigou, and G. Belalem. Document level sentiment analysis: a survey. *EAI endorsed transactions on context-aware systems and applications*, 4(13): e2–e2, 2018.

T. Berners-Lee, J. A. Hendler, and O. Lassila. The semantic web: A new form of web content that is meaningful to computers will unleash a revolution of new possibilities. *Linking the World's Information*, 2023. URL https://api.semanticscholar.org/CorpusID:265709614.

M. Birjali, M. Kasri, and A. Beni-Hssane. A comprehensive survey on sentiment analysis: Approaches, challenges and trends. *Knowledge-Based Systems*, 226: 107134, 2021.

S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, 30(1-7):107–117, 1998.

Z. Dong, X. Fan, and Z. Peng. Fnspid: A comprehensive financial news dataset in time series, 2024.

J. Edwards, K. McCurley, and J. Tomlin. An adaptive model for optimizing performance of an incremental web crawler. In *Proceedings of the 10th international conference on World Wide Web*, pages 106–113, 2001.

E. Ferrara, P. De Meo, G. Fiumara, and R. Baumgartner. Web data extraction, applications and techniques: A survey. *Knowledge-based systems*, 70:301–323, 2014.

C. Hand and G. Judge. Searching for the picture: forecasting uk cinema admissions using google trends data. *Applied Economics Letters*, 19(11):1051–1055, 2012.

W. Höpken, T. Eberle, M. Fuchs, and M. Lexhagen. Search engine traffic as input for predicting tourist arrivals. In *Information and Communication Technologies in Tourism 2018: Proceedings of the International Conference in Jönköping, Sweden, January 24-26, 2018*, pages 381–393. Springer, 2018.

C. Hutto and E. Gilbert. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the international AAAI conference on web and social media*, pages 216–225, 2014.

M. N. M. Ibrahim and M. Z. M. Yusoff. The impact of different training data set on the accuracy of sentiment classification of naïve bayes technique. In *2017 IEEE Conference on Open Systems (ICOS)*, pages 17–20. IEEE, 2017.

G. Inc. Web search engine with graphic snapshots. `https://patents.google.com/patent/US6643641B1/en`, 2003. Classified under G06F16/951 Indexing; Web crawling techniques.

P. K. Jain, R. Pamula, and G. Srivastava. A systematic literature review on machine learning applications for consumer sentiment analysis using online reviews. *Computer science review*, 41:100413, 2021.

S. Kazemian, S. Zhao, and G. Penn. Evaluating sentiment analysis in the context of securities trading. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2094–2103, 2016.

M. A. Khder. Web scraping or web crawling: State of art, techniques, approaches and application. *International Journal of Advances in Soft Computing & Its Applications*, 13(3), 2021.

Y. Liu, B. Lv, G. Peng, and Q. Yuan. A preprocessing method of internet search data for prediction improvement: application to chinese stock market. In *Proceedings of the Data Mining and Intelligent Knowledge Management Workshop*, DM-IKM '12, New York, NY, USA, 2012. Association for Computing Machinery. ISBN 9781450315517. doi: 10.1145/2462130.2462133. URL `https://doi.org/10.1145/2462130.2462133`.

M. López de Prado. Beyond econometrics: A roadmap towards financial machine learning. *Available at SSRN 3365282*, 2019.

Y. Mao, Q. Liu, and Y. Zhang. Sentiment analysis methods, applications, and challenges: A systematic literature review. *Journal of King Saud University-Computer and Information Sciences*, page 102048, 2024.

S. O'Reilly. Nominative fair use and internet aggregators: Copyright and trademark challenges posed by bots, web crawlers and screen-scraping technologies. *Loy. Consumer L. Rev.*, 19:273, 2006.

B. Pang, L. Lee, et al. Opinion mining and sentiment analysis. *Foundations and Trends® in information retrieval*, 2(1–2):1–135, 2008.

G. Peng, Y. Liu, J. Wang, and J. Gu. Analysis of the prediction capability of web search data based on the he-tdc method–prediction of the volume of daily tourism visitors. *Journal of Systems Science and Systems Engineering*, 26:163–182, 2017.

S. Sadiq and P. Papotti. Big data quality - whose problem is it? In *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*, pages 1446–1447, 2016. doi: 10.1109/ICDE.2016.7498367.

A. Sinha, S. Kedas, R. Kumar, and P. Malo. Sentfin 1.0: Entity-aware sentiment analysis for financial news. *Journal of the Association for Information Science and Technology*, 73(9):1314–1335, 2022.

B. Staff. Tech jobs in danger of becoming extinct. *Brainggainmag*, 2015.

M. W. Uhl, M. Pedersen, and O. Malitius. What's in the news? using news sentiment momentum for tactical asset allocation. *The Journal of Portfolio Management*, 41(2):100–112, 2015.

S. Vosen and T. Schmidt. Forecasting private consumption: survey-based indicators vs. google trends. *Journal of forecasting*, 30(6):565–578, 2011.

J. Wen, G. Zhang, H. Zhang, W. Yin, and J. Ma. Speculative text mining for document-level sentiment classification. *Neurocomputing*, 412:52–62, 2020.

S. Yang, J. Rosenfeld, and J. Makutonin. Financial aspect-based sentiment analysis using deep representations. *arXiv preprint arXiv:1808.07931*, 2018.

B. Zhao. Web scraping. *Encyclopedia of big data*, 1, 2017.