

# **SOCIAL MEDIA BASED STOCK MARKET ANALYSIS USING BIG-DATA INFRASTRUCTURE**

**A PROJECT REPORT**

*Submitted By*

|                               |                    |
|-------------------------------|--------------------|
| <b>SHASHANKA VENKATESH</b>    | <b>18 5001 145</b> |
| <b>VENKATARAMAN NAGARAJAN</b> | <b>18 5001 192</b> |
| <b>VISHAKAN SUBRAMANIAN</b>   | <b>18 5001 196</b> |

*in partial fulfillment for the award of the degree*

*of*

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**



**Department of Computer Science and Engineering**

**Sri Sivasubramaniya Nadar College of Engineering**

**(An Autonomous Institution, Affiliated to Anna University)**

**Rajiv Gandhi Salai (OMR), Kalavakkam - 603110**

**May 2022**

# **Sri Sivasubramaniya Nadar College of Engineering**

**(An Autonomous Institution, Affiliated to Anna University)**

## **BONAFIDE CERTIFICATE**

Certified that this project report titled “**SOCIAL MEDIA BASED STOCK MARKET ANALYSIS USING BIG-DATA INFRASTRUCTURE**” is the *bonafide* work of “**SHASHANKA VENKATESH (185001145), VENKATARAMAN NAGARAJAN (185001192), and VISHAKAN SUBRAMANIAN (185001196)**” who carried out the project work under my supervision.

Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

**DR. T.T. MIRNALINEE**  
**HEAD OF THE DEPARTMENT**  
Professor,  
Department of CSE,  
SSN College of Engineering,  
Kalavakkam - 603 110

**DR. N. SUJAUDEEN**  
**SUPERVISOR**  
Assistant Professor,  
Department of CSE,  
SSN College of Engineering,  
Kalavakkam - 603 110

Place:

Date:

Submitted for the examination held on.....

**Internal Examiner**

**External Examiner**

## **ACKNOWLEDGEMENTS**

I thank GOD, the almighty for giving me strength and knowledge to do this project.

I would like to thank and deep sense of gratitude to my guide **Dr. N. SUJAUDEEN**, Assistant Professor, Department of Computer Science and Engineering, for his valuable advice and suggestions as well as his continued guidance, patience and support that helped me to shape and refine my work.

My sincere thanks to **Dr. T.T. MIRNALINEE**, Professor and Head of the Department of Computer Science and Engineering, for her words of advice and encouragement.

I express my deep respect to the founder **Dr. SHIV NADAR**, Founder, SSN Trust. I also express my appreciation to our **Dr. V. E. ANNAMALAI**, Principal, for all the help he has rendered during this course of study.

I would like to extend my sincere thanks to all the teaching and non-teaching staffs of our department who have contributed directly and indirectly during the course of my project work. Finally, I would like to thank my parents and friends for their patience, cooperation and moral support throughout my life.

**SHASHANKA  
VENKATESH**

**VENKATARAMAN  
NAGARAJAN**

**VISHAKAN  
SUBRAMANIAN**

## **ABSTRACT**

Several factors influence the value of a stock apart from the typical quantitative and qualitative parameters seen in the fundamental analysis of stocks like balance sheets, income statements, cash flow statements etc. In recent years, one such factor that has gained prominence is social media trends. It has been observed that industry-relevant social media posts that gain heavy traction from the public has the effect of swinging the respective market's stock price suddenly. The aim of this research is to analyze and understand the effect of such articles over the value of a stock at any given time. Towards the realization of this objective, we consider Twitter, a popularly used micro-blogging platform as our social media source. We perform sentiment analysis on collected Tweets, and measure their correlation to stocks of companies within the relevant sector the Tweet appeals to. Since Tweet & stock market data are voluminous in nature, we propose to build an architecture using tools like Apache Spark & Apache Kafka, which provide us with efficient large-scale data processing capabilities.

# TABLE OF CONTENTS

|   |             |
|---|-------------|
| <b>ABSTRACT</b>                               | <b>iii</b>  |
| <b>LIST OF TABLES</b>                         | <b>vii</b>  |
| <b>LIST OF FIGURES</b>                        | <b>viii</b> |
| <b>1 INTRODUCTION</b>                         | <b>1</b>    |
| 1.1 INTRODUCTION . . . . .                    | 1           |
| 1.2 MOTIVATION . . . . .                      | 2           |
| 1.3 PROBLEM STATEMENT . . . . .               | 3           |
| 1.4 ORGANIZATION OF THE REPORT . . . . .      | 4           |
| <b>2 LITERATURE SURVEY</b>                    | <b>5</b>    |
| 2.1 A PRACTICAL PERSPECTIVE . . . . .         | 5           |
| 2.2 PREVIOUSLY EXPLORED IDEAS . . . . .       | 6           |
| 2.3 EXPLORING TECHNOLOGIES INVOLVED . . . . . | 9           |
| <b>3 DESIGN OF PROPOSED SYSTEM</b>            | <b>11</b>   |
| 3.1 DESIGN OVERVIEW . . . . .                 | 11          |
| 3.2 DATASETS USED . . . . .                   | 13          |
| 3.2.1 Tweet Data . . . . .                    | 13          |
| 3.2.2 Stock Data . . . . .                    | 14          |
| 3.3 STREAM DATA GENERATION . . . . .          | 15          |
| 3.3.1 Tweet data stream . . . . .             | 16          |

|          |  |           |
|----------|--|-----------|
| 3.3.2    | Stock data stream . . . . .            | 16        |
| 3.4      | DATA TRANSFORMATION . . . . .          | 16        |
| 3.4.1    | Design Considerations . . . . .        | 17        |
| 3.5      | DATA STORAGE . . . . .                 | 18        |
| 3.5.1    | Design Consideration . . . . .         | 19        |
| 3.6      | SENTIMENT ANALYSIS . . . . .           | 20        |
| 3.7      | CORRELATION ANALYSIS . . . . .         | 22        |
| 3.8      | VISUALIZATION . . . . .                | 24        |
| <b>4</b> | <b>IMPLEMENTATION</b>                  | <b>26</b> |
| 4.1      | INSTALLATION . . . . .                 | 26        |
| 4.1.1    | Apache Spark . . . . .                 | 26        |
| 4.1.2    | Apache Kafka . . . . .                 | 29        |
| 4.1.3    | Kafka Manager (GUI) . . . . .          | 30        |
| 4.2      | DATA COLLECTION . . . . .              | 31        |
| 4.3      | DATA PRE-PROCESSING . . . . .          | 32        |
| 4.3.1    | Tweet Cleaning . . . . .               | 32        |
| 4.3.2    | Stock Approximation . . . . .          | 34        |
| 4.4      | STARTING APACHE KAFKA & CMAK . . . . . | 35        |
| 4.4.1    | Apache Kafka . . . . .                 | 35        |
| 4.4.2    | Creating Clusters & Topics . . . . .   | 35        |
| 4.5      | DATA PROCESSING . . . . .              | 36        |
| 4.5.1    | Producer . . . . .                     | 36        |
| 4.5.2    | Consumer . . . . .                     | 36        |
| 4.6      | DATA ANALYSIS AND PREDICTION . . . . . | 39        |
| 4.6.1    | Understanding The Data . . . . .       | 39        |

|          |                                       |           |
|----------|---------------------------------------|-----------|
| 4.6.2    | Correlation Analysis . . . . .        | 44        |
| 4.6.3    | Predicting the Stock Market . . . . . | 49        |
| <b>5</b> | <b>CONCLUSIONS AND FUTURE WORK</b>    | <b>56</b> |
|          | <b>REFERENCES</b>                     | <b>58</b> |

## LIST OF TABLES

|      |   |    |
|------|---|----|
| 3.1  | Twitter API Schema . . . . .  | 13 |
| 3.2  | Stock API Schema . . . . .  | 15 |
| 4.1  | Sample Tweet data collected from Twitter-API . . . . .              | 31 |
| 4.2  | Sample stock data collected from Polygon-API . . . . .              | 32 |
| 4.3  | SQLite Schema for Tweet Data & Stock Data . . . . .                 | 32 |
| 4.4  | SQLite Schema for Aggregated Data after stream processing . . . . . | 38 |
| 4.5  | Aggregated Tweet Schema . . . . .                                   | 38 |
| 4.6  | Data Points Count . . . . .   | 40 |
| 4.7  | Correlation & p-value Table . . . . .                               | 47 |
| (a)  | (NASDAQ: MSFT) . . . . .  | 47 |
| (b)  | (NYSE: XOM) . . . . .   | 47 |
| (c)  | (NASDAQ: TSLA) . . . . .  | 47 |
| (d)  | (NASDAQ: MRNA) . . . . .  | 47 |
| 4.8  | XGBoost - Optimal Hyper Parameters . . . . .                        | 51 |
| 4.9  | XGBoost Regressor Metrics . . . . .                                 | 52 |
| (a)  | (NASDAQ: MSFT) . . . . .  | 52 |
| (b)  | (NYSE: XOM) . . . . .   | 52 |
| (c)  | (NASDAQ: TSLA) . . . . .  | 52 |
| (d)  | (NASDAQ: MRNA) . . . . .  | 52 |
| 4.10 | CatBoost - Optimal Hyper Parameters . . . . .                       | 53 |
| 4.11 | CatBoost Regressor Metrics . . . . .                                | 54 |
| (a)  | (NASDAQ: MSFT) . . . . .  | 54 |
| (b)  | (NYSE: XOM) . . . . .   | 54 |
| (c)  | (NASDAQ: TSLA) . . . . .  | 54 |
| (d)  | (NASDAQ: MRNA) . . . . .  | 54 |

## LIST OF FIGURES

|     |   |    |
|-----|---|----|
| 3.1 | Proposed Architecture . . . . .                                 | 12 |
| 3.2 | Sample Tweet JSON Object . . . . .                              | 14 |
| 3.3 | Sample Stock JSON Object . . . . .                              | 15 |
| 3.4 | Stream Data Generation . . . . .                                | 15 |
| 3.5 | Tweet Data Transformation . . . . .                             | 17 |
| 3.6 | Sentiment Analysis Model Flowchart . . . . .                    | 21 |
| 4.1 | Raw EV data distribution . . . . .                              | 33 |
| 4.2 | Cleaned EV data distribution . . . . .                          | 34 |
| 4.3 | Merged EV Dataset . . . . .                                     | 41 |
| 4.4 | Pharma Data - Distribution & Probability Plots . . . . .        | 42 |
|     | (a) Distribution - Ind_Pos Attribute (Unscaled) . . . . .       | 42 |
|     | (b) Probability Plot - Ind_Pos Attribute (Unscaled) . . . . .   | 42 |
|     | (c) Distribution - Ind_Pos Attribute (Log Scaled) . . . . .     | 42 |
|     | (d) Probability Plot - Ind_Pos Attribute (Log Scaled) . . . . . | 42 |
| 4.5 | Company Closing values . . . . .                                | 43 |
|     | (a) (NASDAQ: MSFT) . . . . .                                    | 43 |
|     | (b) (NYSE: XOM) . . . . .                                       | 43 |
|     | (c) (NASDAQ: TSLA) . . . . .                                    | 43 |
|     | (d) (NASDAQ: MRNA) . . . . .                                    | 43 |
| 4.6 | Correlation Heatmap . . . . .                                   | 44 |
|     | (a) Tech - (NASDAQ: MSFT) . . . . .                             | 44 |
|     | (b) Oil - (NYSE: XOM) . . . . .                                 | 45 |
|     | (c) EVs - (NASDAQ: TSLA) . . . . .                              | 45 |
|     | (d) Pharma - (NASDAQ: MRNA) . . . . .                           | 46 |
| 4.7 | XGBoost Prediction Results . . . . .                            | 51 |
|     | (a) (NASDAQ: MSFT) . . . . .                                    | 51 |
|     | (b) (NYSE: XOM) . . . . .                                       | 51 |
|     | (c) (NASDAQ: TSLA) . . . . .                                    | 51 |
|     | (d) (NASDAQ: MRNA) . . . . .                                    | 51 |
| 4.8 | CatBoost Prediction Results . . . . .                           | 53 |
|     | (a) (NASDAQ: MSFT) . . . . .                                    | 53 |
|     | (b) (NYSE: XOM) . . . . .                                       | 53 |
|     | (c) (NASDAQ: TSLA) . . . . .                                    | 54 |
|     | (d) (NASDAQ: MRNA) . . . . .                                    | 54 |

# CHAPTER 1

## INTRODUCTION

### 1.1 INTRODUCTION

Stock market analysis has been a topic of great interest ever since public stock exchanges came into existence. It is an avenue that offers great profit, which by itself is a stimulus for most researchers who work in this realm. The task is also considerably demanding due to these returns as well as high randomness and various external influences affecting the current valuation of a stock[6].

In recent years, social media has transformed in a revolutionary manner from a niche utility to an ubiquitous means of information exchange throughout the world. People have started utilizing social media as a tool not only for the purpose of socialising, but also as a means to consume news, share their opinions, create awareness, follow the ideas of popular personalities and luminaries, among many others.

The architecture of social media provides the avenue for interesting ideas to disseminate among society in near-instant time. In this age, such sudden outbursts of information about a company or its product can make or break the valuation of that product or company - since information travels fast and people tend to react towards sensational articles.

To highlight the severity of social media trends towards the stock market, we can consider the example of the Tweet posted by Elon Musk, the CEO of Tesla Motors,

on 27 Jan 2021 towards a declining video-game retail company, GameStop. His Tweet about the company immediately raised the company's stock value by more than 60% within hours. Such is the influence of social media on an already volatile stock market. Hence, it is worthwhile to consider understanding the relationship between social media trends and the stock market.

## 1.2 MOTIVATION

Understanding the different factors that influence the stock market is essential towards making informed investing decisions in it. The stock market, as a whole, is an intricately large and complex entity that is hard to predict. But, it is possible to understand how certain factors influence the market. Our area of research delves into understanding how social media affects the stock market.

The sentiments expressed by such social media posts & news articles cause a lot of volatility in the share market, leading to unexpected fluctuations of the stock value.

Since social media is inherently voluminous with a high degree of velocity, i.e. millions of people posting their ideas and opinions simultaneously on the same platform, it is difficult to wrangle and analyse meaningful patterns from it with the use of traditional methods.

Another significant reason for exploring this area is the fact that the internet is a rapidly growing community. Even though a good portion of the world's population was online in 2017, there has been an increase of nearly 70 million users on Twitter between 2017 and 2021. This gives yet another reason to explore the possibility

that there could be statistically significant correlation between social media and stock prices.

To process data at scale and perform exploratory data analysis upon it, we need a powerful engine, which can operate on high volumes of data with the help of parallel computing.

### 1.3 PROBLEM STATEMENT

In a general sense, stock market analysis aims to determine the future movement of the stock value of a financial exchange. An accurate prediction of such movement leads to more profits for an investor. Predicting how the stock market moves is a challenging issue due to many factors involved in it, like interest rates, economic growth, current trends, politics, etc.

Stock market experts use various sources of information to determine whether or not to buy/sell stocks of any company. Two such sources that are widely used (especially in recent years) are news events & social media trends.

We wish to understand and analyse the statistical correlation between social media trends and its effect on the relevant industry's stock price valuations with the use of a big-data architecture consisting of Apache Kafka and Apache Spark, since stock data and social media articles tend to be voluminous, and because we require a large sample space to accurately evaluate the correlation, the amount of data that needs to be processed becomes very high.

## 1.4 ORGANIZATION OF THE REPORT

The remainder of this report is organized as follows:

- Chapter 2 discusses the literature survey performed towards stream processing and correlation analysis.
- Chapter 3 details the architecture of the system and the design considerations that have been considered for each component in the architecture.
- Chapter 4 discusses the implementation, experimental setup and the results obtained.
- Chapter 5 concludes and provides future directions.

## CHAPTER 2

# LITERATURE SURVEY

## 2.1 A PRACTICAL PERSPECTIVE

The stock market hasn't been left out in the wave of data science and machine learning that has swept the entire world. It has become a prime focus of various ML and AI researchers, given the volatile and unpredictable nature of the stock market.

Stock market prices are basically influenced by supply and demand from the market. The rate at which stocks rise or fall in value depends on how the stock's demand changes over time. The stock market, although partially regulated, completely relies on the investor's hands on properly understanding the market behavior and thus buying/selling shares consequently.

Some factors that affect stock prices are:

- Random Speculation
- Government Policies
- News
  - Price Hike
  - Fraud detection
- Trends
- Seasonal(s)
- Similar industry growth/decline

Our main idea is to analyse the correlation between stock valuation and one of its factors that exert a significant degree of influence, especially in recent years: social media trends.

## 2.2 PREVIOUSLY EXPLORED IDEAS

The research undertaken by Lee et. al. [7] delves into analyzing Twitter data, by classifying it into relevant industrial sector categories and performing sentiment analysis to predict the trend of stock prices, and comparing it to reality, to consequently find the correlation between the two. Their paper states that they used a sample of 1000 news articles to create a classifier. They were able to obtain a 77% accuracy on the classification of Tweets into different sectors. They have used 100 Tweets overall, with 20 Tweets per category in order to test out their hypothesis.

Bollen et al. [1] indicate that the accuracy of DJIA predictions (Dow Jones Industrial Average) can be significantly improved by including specific mood dimensions. Their research work analyzes mood in terms of 6 dimensions: Calm, Alert, Sure, Vital, Kind & Happy.

Following the results of the above paper, another research study was performed by Mittal et al. [11] that uses Twitter to predict public mood, and use the predicted mood and previous days' DJIA values to predict stock market movements. They have proposed an algorithm to approximate stock market data on days in which the market was closed due to public holidays and weekends. For prediction of stock

market movements, they highlight several algorithms like Regression, SVM and SOFNNs (Self-Organizing Fuzzy Neural Networks).

Nayak et al. [12] have done significant work on sentiment analysis of Twitter data for stock price prediction using trend analysis of stock data obtained from Yahoo Finance for three sectors: Banking, Mining & Oil and sentiment data extracted from relevantly collected Tweets. The authors have also put forth algorithms to calculate closed price trends for each day, to check continuous days up/down for a stock, and an algorithm to combine sentiment with historical data. Their research work also discusses data pre-processing methods using the NLTK package available in Python to process Tweet data. They predict stock market movement based on available historical stock data and extracted social media sentiment data using 3 different supervised machine learning models, and obtain the best results using the Boosted Decision Tree model.

An analysis between the stock market and public sentiment was done by Kalyanaraman et al. [5] purely using news articles. They considered 11 companies under the National Stock Exchange (NSE), and considered 100 news articles for each company. They manually pre-processed the articles to remove irrelevant ones. This further reduced their dataset to 50-60 articles per company. They then manually labelled the sentiment of the articles as being positive or negative. They used a custom sentiment dictionary and vectorized the news articles to build a classifier using Linear Regression and Gradient Descent. Their model predicted the sentiment of articles with around 60% accuracy and the direction of stock price movement (positive or negative) with an accuracy of 81.92%.

The Stanford Core NLP library has been used by Kanavos et al. [6] to perform text pre-processing along with Naive Bayes classifier to perform sentiment analysis (using a traditional n-grams approach) on Twitter data for stock price predictions.

Pagolu et al. [14] performed correlation analysis between Tweets related to Microsoft and its stock price movements during the time period of August 2015 - August 2016. They transformed correlation analysis into a classification problems with the input features being the total negative, neutral and positive emotions in Tweets observed in a 3 day period. They utilized the methodologies of Word2Vec for sentiment analysis and Logistic Regression & SVM to perform the classification task, which yielded an accuracy of around 70% to predict the stock movement as an upward or downward trend.

Mehtab and others [10] further built upon the work done by Mittal et al. and developed eight regression and eight classification models for predicting the stock price movement of NIFTY 50, which is augmented using public mood data which was analysed and aggregated from relevant Tweets.

The paper published by Selvamuthu, Kumar and Mishra [17] presents several variations of ANNs (Artificial Neural Networks) with different learning algorithms to predict the Indian stock market using tick data. The authors suggest using a more extensive dataset to capture seasonal trends, and consider the avenue of sentiment analysis, since statements expressed by renowned personalities are known to affect the stock market, to gain an extra edge in stock market prediction.

## 2.3 EXPLORING TECHNOLOGIES INVOLVED

Earlier research methodologies utilized various NLP techniques like n-grams, Word2Vec, TF-IDF etc. Several of these research publications have mentioned the use of deep learning models as a future work that can be undertaken to better improve the sentiment analysis task. In our methodology, we thus use a popular and state-of-the art NLP transformer model, BERT, to perform the Tweet sentiment analysis. More specifically, we utilize a pre-trained RoBERTa-base model, which was trained on 124M English Tweets collected during the period of Jan 2018 to Dec 2021 [18].

The above discussed model was bench marked against several other systems on the TweetEval task [9]. The model competes well and performs better than most other systems, giving an overall test result accuracy of 66.2%, second only to the BERTweet model, which performed at an accuracy of 67.9%. The decision to use this specific model stems from the fact that it has been trained on a newer Tweet dataset, and also due to the fact that the BERTweet model has a lower token limit than the RoBERTa based model.

Another piece of technology involved in this experiment is the big-data architecture. We require a scalable & robust architecture to handle big-data collected from the various information sources. Gurcan et. al [4] explains the lifecycle associated with real-time big-data processing and highlights the use-cases of different big-data tools. From their research, we chose to go with Apache Kafka for our data ingestion module & Apache Spark to handle batch and stream processing of data. The authors also highlight the different challenges

associated with unstructured data, which should be kept in mind while constructing the architecture.

Peng's research [15] gives an overview of machine learning with Apache Spark and HDFS for stock market analysis. He also proposes a generic pipeline for big-data architectures. His paper also explains on how to pre-process data using PySpark, which is an interface for Apache Spark written in Python.

# CHAPTER 3

## DESIGN OF PROPOSED SYSTEM

This chapter covers in detail, the design of the proposed system along with the design considerations and alternatives considered wherever applicable. The architecture of the proposed system is based on streaming architecture. The following modules are explained: Streaming Data Generation, Data Transformation, Data Storage, Sentiment Analysis, Correlation Analysis and Visualization.

### 3.1 DESIGN OVERVIEW

Tweets based on a few industrial sectors are collected based on appropriately chosen keyword(s) and stock market data is collected for the corresponding companies present in a few chosen industrial sectors. This data is then cleaned, stored and then published into our stream handling framework.

Subsequently, with the use of Apache Spark and Apache Kafka integrated architecture, the published data is read from the Kafka topics to which the data is published relevantly. The data pulled is then processed using PySpark methods (MapReduce) and the intermediate results are stored in a database for further analysis. A periodic check-pointing is also done on the data pulled from the topics to avoid redundant processing.

Using the intermediate results, correlation between the Tweets and stock data are analysed, apart from other analysis. For a better understanding of the results obtained, visualizations from the analysis are made.

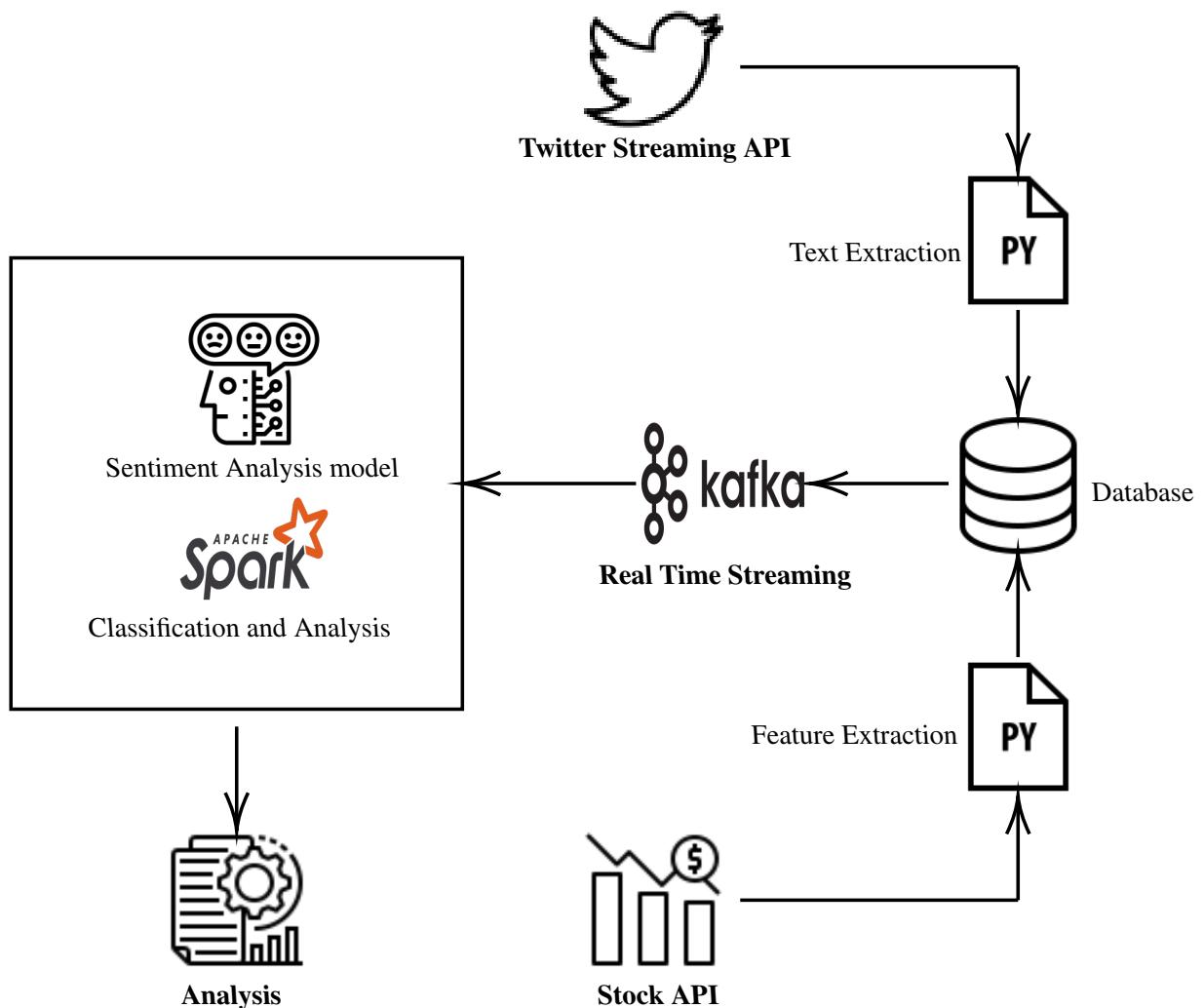


FIGURE 3.1: Proposed Architecture

An overview of the entire architecture is shown in Figure 3.1. Each module in the diagram is explained in detail in Sections 3.2 through 3.8.

## 3.2 DATASETS USED

### 3.2.1 Tweet Data

The official Twitter API is used to collect Tweet data. It has three different tiers of data access. With the free Essential Access plan, historical data upto a period of 2 months can only be obtained. We require a larger sample space to properly study our objective.

The Academic Research Access plan provided by the Twitter API gives us access to the entire archive of Tweet data, with a rate limit of upto 50 requests per 15 minutes. We use this API level plan to collect Tweet data for our dataset. The attributes that constitute the schema of the collected Tweets can be inferred from Table 3.1. Figure 3.2 is an example of the output obtained from the API request.

| Attribute |                | Description  |
|-----------|----------------|--|
| DATA      | source         | Utility used to post the Tweet                                 |
|           | lang           | Machine-detected language of the Tweet text                    |
|           | created_at     | UTC time when this Tweet was created                           |
|           | text           | Body of the Tweet  |
|           | id             | The unique ID of this Tweet                                    |
|           | public metrics | Number of times this Tweet has been retweeted                  |
|           | retweet_count  | Number of times this Tweet has been replied to                 |
|           | reply_count    | Number of times this Tweet has been liked by Twitter users     |
|           | like_count     | Number of times this Tweet has been quoted by Twitter users    |
|           | quote_count    |  |
| META      | newest_id      | The Tweet ID of the most recent Tweet returned in the response |
|           | oldest_id      | The Tweet ID of the oldest Tweet returned in the response      |
|           | next_token     | A pointer used to paginate the next set of results             |
|           | results_count  | The number of tweets fetched by the API call                   |

TABLE 3.1: Twitter API Schema

```

1  {
2      'data' : [ {
3          'source' : 'Twitter for Android',
4          'lang' : 'en',
5          'created_at' : '2021-12-24T15:22:51.000Z',
6          'text' : 'Low risk #stocks for decades to come #msft & #goog',
7          'id' : '1474400183454601219',
8          'public_metrics' : {
9              'retweet_count' : 0,
10             'reply_count' : 0,
11             'like_count' : 0,
12             'quote_count' : 0
13         }
14     } ],
15     'meta' : {
16         'newest_id' : '1474400183454601219',
17         'oldest_id' : '1474400183454601219',
18         'next_token' : '235467890098',
19         'result_count' : 1
20     }
21 }
```

FIGURE 3.2: Sample Tweet JSON Object

### 3.2.2 Stock Data

We considered several third-party stock data API providers like Zerodha, Moneycontrol etc. The drawback of the above API providers is that they charge a significant price for accessing historical stock data.

Thus, we ended up choosing Polygon.io, a popular API service that provides us with access to 2 year historical data for free, with a rate limit of 5 requests per minute. The attributes that constitute the schema of the collected stock data can be inferred from Table 3.2. Figure 3.3 is an example of the response JSON object given by the Polygon API.

| Attribute  | Description  |
|------------|--|
| status     | A response from API indicating success (OK) or failure               |
| symbol     | The ticker symbol whose stock data is requested                      |
| from       | Requested date of the stock  |
| open       | The opening price of the stock on the requested date in USD (\$)     |
| close      | The closing price of the stock on the requested date in USD (\$)     |
| high       | The highest price of the stock on the requested date in USD (\$)     |
| low        | The lowest price of the stock on the requested date in USD (\$)      |
| preMarket  | The pre-market price of the stock on the requested date in USD (\$)  |
| afterHours | The after hours price of the stock on the requested date in USD (\$) |
| volume     | The amount of shares or contracts traded over the day                |

TABLE 3.2: Stock API Schema

```

1 {
2   'status' : 'OK',
3   'from' : '2021-05-27',
4   'symbol' : 'TSLA',
5   'open' : 620.24,
6   'high' : 631.13,
7   'low' : 616.21,
8   'close' : 630.85,
9   'volume' : 26370593,
10  'afterHours' : 629.23,
11  'preMarket' : 616.63
12 }
```

FIGURE 3.3: Sample Stock JSON Object

### 3.3 STREAM DATA GENERATION

Initially, two different streams - Tweet data stream and stock data stream - are generated as shown in Figure 3.4.

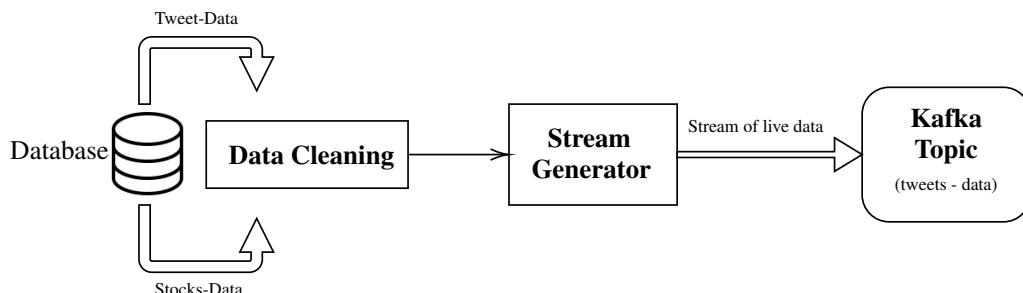


FIGURE 3.4: Stream Data Generation

Next, the dataset is cleaned - i.e irrelevant columns are removed, and missing values are imputed, using a Python script. An Apache Kafka producer script is implemented to simulate a uniform stream from the dataset. To do this, the program maintains a timer and accordingly publishes records from the dataset at periodic intervals of time.

### **3.3.1 Tweet data stream**

The Tweet dataset (with columns: Category, Date, Tweet & Retweet Count) is initially grouped based upon the category that they belong to, and then produced to the relevant Kafka topic through the producer script mentioned above.

### **3.3.2 Stock data stream**

Similarly, the stock dataset (with columns: Ticker, Date, Open, Close, High & Low) is grouped based upon the ticker that they belong to, and then subsequently produced to the relevant Kafka topic through the producer script.

## **3.4 DATA TRANSFORMATION**

Data transformation is predominantly done with the use of Apache Spark, through the PySpark API. The entire process is diagrammatically represented in Figure 3.5.

The Tweet data published to various Kafka topics are consumed through Spark's Structured Streaming - a stream processing engine built upon the SparkSQL

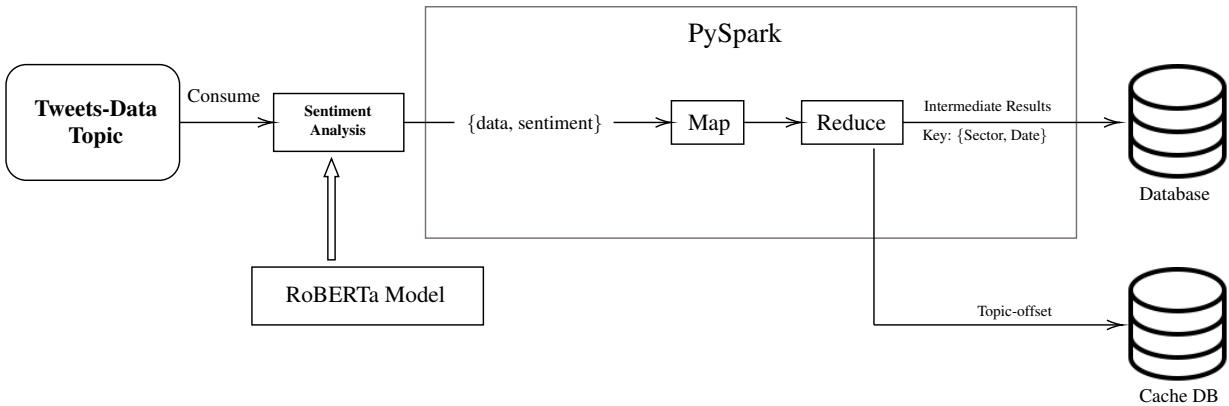


FIGURE 3.5: Tweet Data Transformation

engine - it takes care of processing data incrementally and updating the results. Upon being invoked, they can perform a variety of actions such as processing logic on the incoming data (in micro-batches), and writing the output to a permanent storage. PySpark provides us with a DataFrame API level of abstraction to express streaming aggregations, event-time windows, stream-to-batch joins etc.

In PySpark, aggregate methods are written to transform individual Tweet data can be applied efficiently (via SparkSQL Engine) on the incoming data stream, to obtain intermediate, sector-wise results, which can then be promptly stored in a database for further analysis.

### 3.4.1 Design Considerations

Gurcan et. al's [4] research identifies the different challenges and tasks involved in big-data processing and lists out several tools that can be utilized to construct an efficient big-data architecture.

For the data ingestion phase - a process in which big data is ingested from heterogeneous sources, the most appropriate tools are identified as Apache Kafka, Apache Flume and Apache NiFi. Of the three choices, we went ahead with Apache Kafka considering its better performance in terms of scalability, latency and throughput. Kafka is ubiquitously used as a distributed streaming framework and thus has huge community support - ensuring ease of development.

For the data processing phase - a process in which big-data streams are processed and structured for analytics, the most popular choices are Apache Spark, Apache Storm, Apache Flink and Apache Samza. Of these four alternatives, we proceeded with Apache Spark, since Spark Streaming allows us to build scalable, fault-tolerant applications; it also incorporates the Stream as well as Batch Processing paradigms. Due to the wide community support, it has APIs available in multiple languages and integrates with Apache Kafka seamlessly through dependencies.

## 3.5 DATA STORAGE

To provide a source for the stream data generation and to record processed intermediate results obtained from the Spark interface, we require a data storage solution.

More specifically, a database solution is required for the following reasons: We require that the processed data be maintained in a consistent state since updates are done frequently to the intermediate results by the PySpark interface. In case of any failures, we also require the ability to revert back to a previous state.

There are two tables to be maintained for Tweet data: one that contains raw Tweet data collected from the API, and another that contains sector-wise aggregated Tweet sentiment data that is to be used for analysis.

Similarly, there are two tables to be maintained for stock data: one for storing raw stock data, and another to store stock data with approximated values calculated for days during which the stock market was inoperative.

### 3.5.1 Design Consideration

While considering a candidate application for handling our data storage needs, we were able to construct well-defined schemas for our different datasets (consisting of Tweet data and stock market data) and intermediate results. Thus, we inclined more towards SQL-based solutions rather than NoSQL databases like MongoDB, Apache Cassandra etc.

Among the many available SQL database offerings, we opted for the SQLite database: a lightweight, server-less, self-contained RDBMS (Relational Database Management System), for its ease of use and cross-platform portability. It also utilises significantly fewer resources as compared to alternatives like MySQL and PostgreSQL.

SQLite is also a prudent choice considering the fact that our entire environment is set up and run locally. We can limit disk, cache & RAM usage by SQLite to optimally use our environment's available computing resources.

The server-less and embedded nature of SQLite also fares well compared to client-server based solutions like MySQL and PostgreSQL, both of which require a server

to be set up before usage. In case of any system failure, the SQLite database maintains its consistent state and does not require any manual server restart as needed in other client-server based SQL database applications.

## 3.6 SENTIMENT ANALYSIS

A crucial component in the pursuit of our study to understand the relation between social-media (Tweets) and the stock market is analysing the sentiment (or mood) reflected by social media at any given date.

To perform sentiment analysis on Twitter data, various NLP methods have been explored, as discussed in Section 2.3.

In the field of Natural Language, one of the more recent developments has been the introduction of the Encoder-Decoder (Transformer) Model - more specifically, the BERT (Bidirectional Encoder Representations from Transformers) Model [3]. BERT differs from conventional NLP language models in the fact that it uses bidirectional training and attention mechanism in order to have a deeper sense of language context compared to other unidirectional models. Since its introduction, the BERT model has been extensively used for popular NLP tasks like Text Summarization, Question-Answering, Text Classification, Sentiment Analysis etc.

In our methodology, we chose to use Cardiff NLP research group's publicly available TimeLMs based sentiment analysis model [18]. This model builds upon the existing RoBERTa-base model, which in turn is a BERT-base model that has been pre-trained on a large corpus of English data using the training approach as

discussed in its research paper [8]. This sentiment analysis model has been pre-trained on a dataset of approximately 124M Tweets collected between Jan 2018 - Dec 2021. To add further detail, their research aims at building language models (LMs) that specializes in understanding diachronic Twitter data [9].

Their model has been benchmarked against the TweetEval evaluation - a unified Twitter benchmark composed of seven heterogeneous Tweet classification tasks. The results obtained are promising, and perform competitively well - only outperformed by the BERTweet [13] model on the overall averaged results, by a small margin. Yet, we still expect this model to perform the best for our use-case, since it has been pre-trained on Tweet data that is similar to our period of study (Mar 2020 - May 2022). The better performance can be attributed to having a better language model that captures the context of the Twitter English language used during that time.

The model classifies a Tweet into three sentiments - Negative, Neutral and Positive, and returns an array of three confidence values which correspond to the probabilities of the Tweet being classified into the corresponding sentiments. Figure 3.6 denotes the sentiment analysis process diagrammatically:

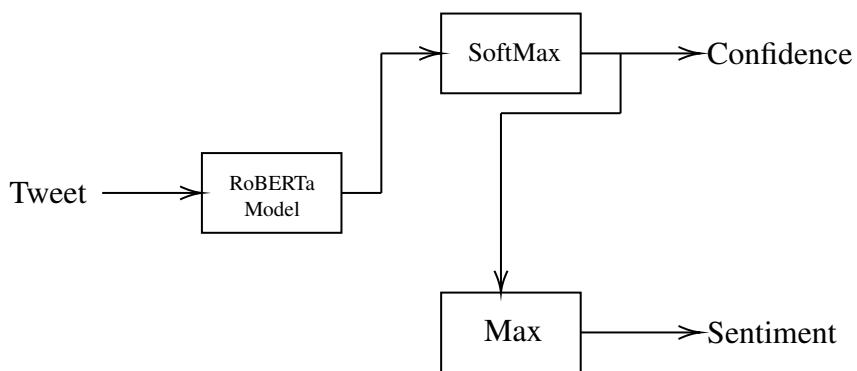


FIGURE 3.6: Sentiment Analysis Model Flowchart

In this manner, all Tweets available for a given day and a given sector are scored and aggregated by the Spark Processing Engine using this sentiment analysis model. The negative, neutral and positive mood observed on Twitter on a given day for a specific market sector is thus captured and quantified.

### 3.7 CORRELATION ANALYSIS

After processing the entire stream of Tweet data and obtaining intermediate results based on the sentiment scores, we move to the correlation analysis part.

Firstly, we group the intermediate results sector-wise and merge it with its relevant stock market ticker data, corresponding to a date-wise order. For example, the aggregated Tweet results obtained under the EVs (Electric Vehicles) category are merged with the stock market data of Tesla Motors (NASDAQ: TSLA). In essence, we now have a combined dataset that describes the stock market performance of a given stock and the social media mood captured on Twitter on that day for the stock's relevant market sector.

Following this, feature scaling is done on the numerical columns of the dataset using min-max normalization (shown in Equations 3.1 & 3.2). This is done since different features present in the dataset have different ranges, and it is necessary to bring them all to the same scale (Min-Max Normalization rescales all values between 0 and 1), for further analysis and easier visualization.

$$y' = \frac{y - x_{min}}{x_{max} - x_{min}}(x'_{max} - x'_{min}) + x'_{min} \quad (3.1)$$

For Min-Max normalization, typically  $\langle x'_{min}, x'_{max} \rangle = \langle 0, 1 \rangle$ .

$$y' = \frac{y - x_{min}}{x_{max} - x_{min}} \quad (3.2)$$

where:

$\langle x_{min}, x_{max} \rangle$  is the old range.

$\langle x'_{min}, x'_{max} \rangle$  is the new range.

$y \in \langle x_{min}, x_{max} \rangle$  is the value to be normalized.

$y' \in \langle x'_{min}, x'_{max} \rangle$  is the min-max normalized value.

The correlation analysis is performed with the Spearman's Correlation Coefficient (Equation 3.4), which is a non-parametric measure of rank correlation between two variables. It is equal to the Pearson's Correlation Coefficient between the rank values of those two variables (Equation 3.3). While Pearson's correlation assesses linear relationships, Spearman's correlation assesses monotonic relationships (irrespective of linearity). The confidence levels of each correlation score are also calculated, to accept/reject the Null Hypothesis (whether the correlation is significant or not). It is a better correlation metric than the Pearson's correlation metric for this specific scenario since the stock market may not be linearly related to the sentiments observed on social media.

$$\rho = \frac{\text{cov}(R(X), R(Y))}{\sigma_{R(X)} \sigma_{R(Y)}} \quad (3.3)$$

Where:

$\rho$  is Spearman's coefficient,  $\rho \in \langle -1, 1 \rangle$

$cov(R(X), R(Y))$  is the covariance of the rank variables.

$\sigma_{R(X)}$  and  $\sigma_{R(Y)}$  are the standard deviations of the rank variables.

If all  $n$  ranks are all distinct, it can be computed using the formula:

$$\rho = 1 - \frac{6 \sum_{i=1}^n d_i^2}{n(n^2 - 1)} \quad (3.4)$$

Where:

$\rho$  is Spearman's coefficient,  $\rho \in \langle -1, 1 \rangle$

$n$  is sample size.

$d_i$  is the difference between the ranks of the data being analyzed.

To visually understand the correlation score, scatter plots are also drawn between the two features to understand the distribution of the data points.

This procedure is carried out for various categories separately to observe and tabulate the correlation between that sector's stock market and its' respective social media mood.

## 3.8 VISUALIZATION

To perform various graphical visualizations on the collected and processed data in order to better understand the relationship between the different data attributes present and between Tweets and Stocks - we require the use of a graphing tool.

Python has a plethora of graphing packages that generate colourful and informative charts and diagrams. Thus we leverage packages like Matplotlib, Seaborn and Plotly to augment our study with meaningful and interactive graphs, and to perform exploratory data analysis (EDA). The entire workflow pertaining to data analysis, correlation study and visualization can thus be presented in a visually appealing and easy to understand manner with the use of Interactive Python Notebooks (IPYNBs). The IPYNBs act as an interactive front-end that enables us to view, manipulate and understand the processed data, supported by graphing utilities.

# CHAPTER 4

## IMPLEMENTATION

This chapter discusses Apache Kafka and Apache Spark setup, installation of components and implementation of modules and the results obtained. The applications are executed in a machine running on Ubuntu 20.04. The environment is installed with OpenJDK 11.0.14, Scala 2.11.12 and Python 3.8.10.

Apache Spark and Apache Kafka are also configured in the system, and the details on their installation are mentioned below.

### 4.1 INSTALLATION

#### 4.1.1 Apache Spark

Batch and stream processing of large-scale data can be done efficiently with the usage of Apache Spark. It uses the power of parallelization to wrangle data faster than traditional methods.

Apache Spark [21] was downloaded and kept in the home folder of the system. The version of Apache Spark that is used here is version 3.2.1.

#### 4.1.1.1 PySpark Dependencies

Since we will be interacting with Spark through from a Python interface, we need to install the PySpark library. PySpark's dependencies are installed using the following commands.

```
sudo apt update  
sudo apt install python3-pip  
pip3 install jupyter  
sudo apt install jupyter-core  
sudo apt update  
sudo apt install default-jre  
sudo apt install default-jdk  
sudo apt install scala  
pip3 install py4j
```

#### 4.1.1.2 Configuration

The downloaded tarball of Apache Spark is moved to home directory and extracted using the command mentioned below:

```
sudo tar -zxvg spark-3.2.1-bin-hadoop3.2.tgz
```

The following environment variables are to be added to the system's *.bashrc* file to run Apache Spark:

```

export SPARK_HOME="home/<name>/spark-3.2.1-bin-hadoop3.2"
export PATH=$SPARK_HOME:$PATH
export PYTHONPATH=$SPARK_HOME/python:$PYTHONPATH
export PYSPARK_DRIVER_PYTHON="jupyter"
export PYSPARK_DRIVER_PYTHON_OPTS="notebook"
export PYSPARK_PYTHON=python3

```

The access modifiers of these folders are to be changed as follows:

```

sudo chmod 777 spark-3.2.1-bin-hadoop3.2
cd spark-3.2.1-bin-hadoop3.2
sudo chmod 777 python
cd python
sudo chmod 777 pyspark

```

#### 4.1.1.3 Installing FindSpark

PySpark is not present on *sys.path* by default. To use the PySpark library outside of its home directory, it needs to be either sym-linked onto your site-packages, or adding it to the *sys.path* during run-time. The *findspark* package does the latter.

FindSpark package is installed using the following command:

```
pip3 install findspark
```

The following Python snippet can be used to make sure *findspark* is installed correctly:

---

```
import findspark
findspark.init('/home/<name>/spark-3.2.1-bin-hadoop3.2')
import pyspark
```

---

## 4.1.2 Apache Kafka

Apache Kafka is a robust, fault-tolerant, high-performance data streaming pipeline that can ingest and stream thousands of Tweets which can then be processed and mined to observe interesting results.

Apache Kafka [22] was downloaded and kept in the home folder of the system. The version of Apache Kafka that is used here is version 2.13-3.1.0.

### 4.1.2.1 Configuration

The downloaded tarball of Apache Kafka is moved to home directory and extracted using the command mentioned below:

```
sudo tar -xzf kafka_2.13-3.1.0.tgz
```

Apache Kafka's configuration needs to be edited before executing it in our environment. The configuration file *server.properties* is present inside the *config/* folder of Apache Kafka.

The following changes are made to the *server.properties* file:

```
advertised.listeners=PLAINTEXT://localhost:9092
zookeeper.connect=localhost:2181
```

### 4.1.3 Kafka Manager (GUI)

Yahoo-CMAK [23] is an Apache Kafka manager that provides a GUI service to create clusters, topics and maintain them through a web-interface. CMAK is installed onto our system by cloning its GitHub repository into our home directory, using the following commands:

```
sudo apt update
git clone https://github.com/yahoo/CMAK
```

#### 4.1.3.1 Configuration

The command below will create a zip file which can be used to deploy the CMAK application:

```
cd CMAK/
./sbt clean dist
```

Then, inside the created *target/universal* folder, we unzip the CMAK folder.

```
cd target/universal
unzip cmak-3.0.0.6.zip
```

Following this, we configure the application as follows:

```
ls cmak-3.0.0.6/
cd conf
```

The following changes are made in *application.conf*

- cmak.zkhosts="kafka-manager-zookeeper:2181"
- + cmak.zkhosts="localhost:2181"

## 4.2 DATA COLLECTION

Python scripts were written to collect the Tweet data and stock data from Twitter-Academic-API [19] and Polygon-API [20] respectively. Sample data collected from the respective APIs are shown in Tables 4.1 and 4.2, whose column schemas are explained in Tables 3.1 and 3.2 respectively.

| CATEGORY | DATE       | COUNT | TWEET  |
|----------|------------|-------|--|
| Tech     | 2020-04-07 | 7     | So we have a powerful Follow Through Day, and some stocks are showing positive action: \$VRTX \$AMZN ... \$MSFT ... \$SEGN |
| Oil      | 2020-04-07 | 1     | I think some \$XOM and \$KO puts should be purchased   |
| EVs      | 2020-04-07 | 1     | If \$TSLA gets to \$520, i will be happy   |

TABLE 4.1: Sample Tweet data collected from Twitter-API

| CATEGORY | TICKER | DATE       | OPEN (\$) | CLOSE (\$) |
|----------|--------|------------|-----------|------------|
| Tech     | MSFT   | 2020-10-01 | 213.49    | 212.46     |
| EVs      | TSLA   | 2020-10-01 | 440.76    | 448.16     |
| Oil      | XOM    | 2020-10-01 | 33.79     | 33.13      |
| Pharma   | MRNA   | 2020-10-01 | 69.57     | 70.03      |

TABLE 4.2: Sample stock data collected from Polygon-API

## 4.3 DATA PRE-PROCESSING

### 4.3.1 Tweet Cleaning

The collected JSON data from the Twitter API is cleaned to remove emoticons and extraneous characters. Links and other irrelevant data are removed from the Tweet, if any. Duplicate Tweets (which may be collected accidentally due to keyword-based search) are also removed. Following this, the Tweets are stored inside an SQLite database for later processing. The schemas of the Tweet and stock data as stored in our database are shown in Table 4.3.

| Attribute | Type    |
|-----------|---------|
| Category  | Text    |
| Date      | Text    |
| Count     | Integer |
| Tweet     | Text    |

| Attribute | Type |
|-----------|------|
| Category  | Text |
| Ticker    | Text |
| Date      | Text |
| Open      | Real |
| Close     | Real |

TABLE 4.3: SQLite Schema for Tweet Data &amp; Stock Data

Before using the raw Tweet data directly for social media mood analysis, it is imperative to check the distribution of the Tweet data.

Upon loading the Tweet data and analysing it with the usage of a boxplot, it can be clearly seen that the boxes are very compressed and almost indiscernible due

to the outliers in terms of retweet count. Using such data directly would lead to incorrect results. Thus, by method of trial and error, a good maximum limit is found, and Tweets with a retweet count above the chosen limit is capped to the maximum limit. This ensures that we do not miss valuable Tweets (Tweets with a high retweet score imply that the Tweet was popular and reflected the mood of a sizeable portion of social media users), and at the same time avoid the issue of it being an outlier. A boxplot of the EV Tweet data is shown in Figure 4.1 for reference.

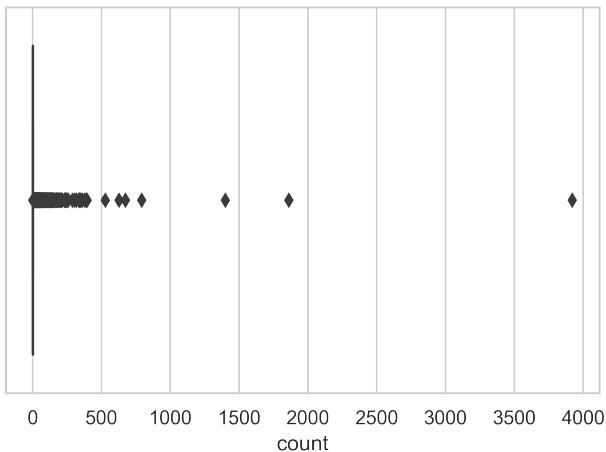


FIGURE 4.1: Raw EV data distribution

Similarly, on the other end of the spectrum, Tweets with a low retweet score are not reliable sources of information, and for the most part, constitute noise in terms of the overall sentiment observed on social media. Thus, it would be wise to eliminate Tweets with a low retweet score in order to obtain a closer to true analysis of social media and the stock market.

Thus, after removing noisy Tweets and capping the retweet counts of highly retweeted ones, we end up with a boxplot with the box clearly visible and small whiskers. This distribution is much more appropriate for usage as compared to

the raw Tweet data distribution. The modified boxplot of the EV Tweet data is shown in Figure 4.2 for reference.

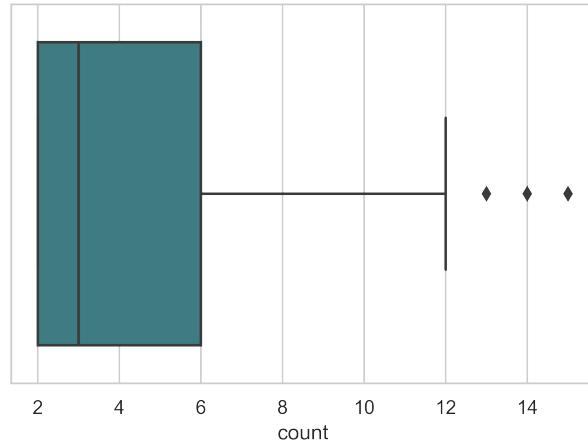


FIGURE 4.2: Cleaned EV data distribution

### 4.3.2 Stock Approximation

Stock market data is approximated on days when the market was closed to have a continuous period of data, without any missing values. . . The approximation is done using the formula given in Equation 4.1, as proposed by Mittal's paper [11].

$$z = \left( \frac{x+y}{2} \right) \quad (4.1)$$

Where:

$z$  is the missing stock value.

$x$  is the previously available stock value data.

$y$  is the next available stock value data.

## 4.4 STARTING APACHE KAFKA & CMAK

### 4.4.1 Apache Kafka

Before starting Apache Kafka, Apache Zookeeper must already be running in the background. To start Apache Zookeeper, the following command is executed:

```
cd ~/kafka_2.13-3.1.0/
bin/zookeeper-server-start.sh config/zookeeper.properties
```

Now, start Apache Kafka using the following command:

```
JMX_PORT=8004 bin/kafka-server-start.sh config/server.properties
```

### 4.4.2 Creating Clusters & Topics

The CMAK GUI is used to create and micromanage Kafka clusters and topics. It also provides status reports on the configured clusters.

The CMAK web-interface is started using the following command:

```
bin/cmak -Dconfig.file=conf/application.conf -Dhttp.port=8081
```

Now, open the CMAK interface via a web-browser:

```
http://localhost:8081
```

Then, we create a cluster using the UI, with the Zookeeper host value set to *localhost : 2181*. Additionally, JMX Polling & Poll Consumer Information features are enabled.

Inside the newly created cluster, we can now create topics. For our purposes, we can create a topic with a replication factor of 1 with a single partition allocated to it.

Dedicated topics can be created to hold specific data - for instance: a topic named *tech – stocks* can be created to transmit technology sector related stock data.

## 4.5 DATA PROCESSING

### 4.5.1 Producer

The producer logic is written in a Python script using the *kafka – python* library. Data is ingested into relevant Kafka topics from the SQLite database created earlier. Pre-processing is done to convert the data to an appropriate JSON format, which is then subsequently UTF-8 encoded before being produced into the topics as messages.

### 4.5.2 Consumer

The produced data can then be consumed by subscribing to the Kafka topics. In our scenario, we subscribe to the data directly via a Spark Session in PySpark.

The data is collected as a Stream of messages from the Kafka topics. To avoid redundant processing, we update an *offset* value to keep track of already processed messages. The collected data is then stored in an in-memory Spark SQL table, and processed periodically.

#### 4.5.2.1 Tweet Streaming

The Tweets stored in the in-memory Spark SQL table, are fetched and converted back into the JSON format.

The Twitter-RoBERTa sentiment analysis model [18] is used to perform sentiment analysis on the consumed Tweet data. The Twitter-RoBERTa model outputs an array of size 3 (Equation 4.2), which denote confidence values. (as discussed in Section 3.6.)

$$\begin{aligned}
 \text{model(tweet)} &\xrightarrow{\text{returns}} [ \\
 &\quad \text{negative\_confidence}, \\
 &\quad \text{neutral\_confidence}, \\
 &\quad \text{positive\_confidence} \\
 ] \tag{4.2}
 \end{aligned}$$

$$\text{sentiment(tweet)} = \text{index}(\max(\text{model(tweet)})) \tag{4.3}$$

The most appropriate sentiment is obtained for each Tweet w.r.t to equation 4.3 and then stored with the corresponding Tweet's JSON data.

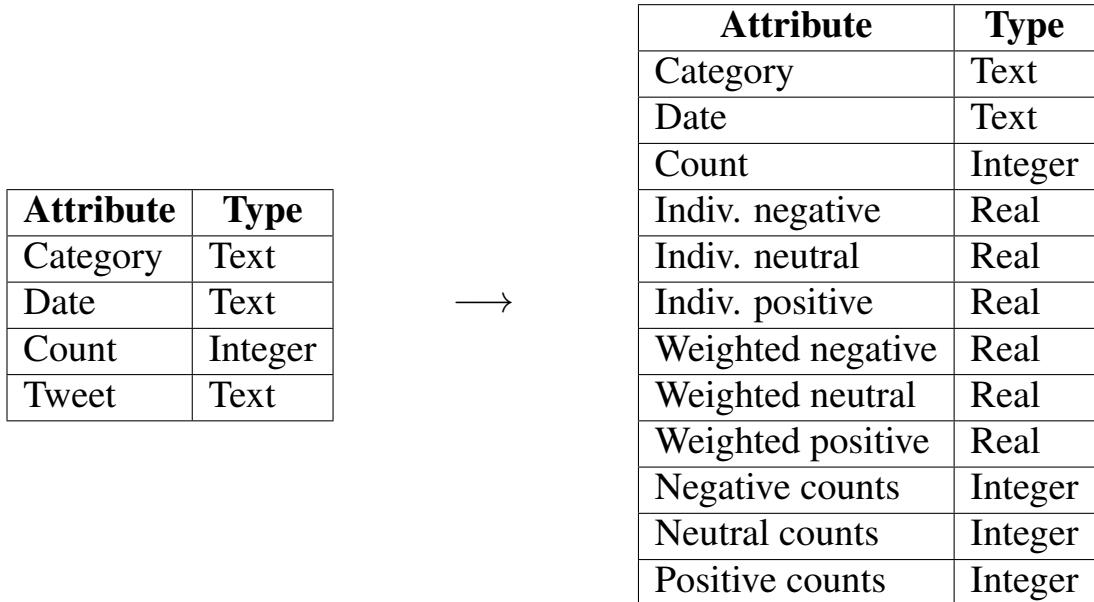


TABLE 4.4: SQLite Schema for Aggregated Data after stream processing

| Attribute           | Description  |
|---------------------|--|
| Category            | The industry sector name   |
| Date                | UTC date that the Tweet belongs to   |
| Count               | Total number of Tweets for a given category and date                       |
| Individual Negative | The aggregated negative confidence of the scored Tweets                    |
| Individual Neutral  | The aggregated neutral confidence of the scored Tweets                     |
| Individual Positive | The aggregated positive confidence of the scored Tweets                    |
| Weighted Negative   | The aggregated negative confidence of the Tweets weighted on retweet count |
| Weighted Neutral    | The aggregated neutral confidence of the Tweets weighted on retweet count  |
| Weighted Positive   | The aggregated positive confidence of the Tweets weighted on retweet count |
| Negative counts     | The number of negative Tweets for a given category and date                |
| Neutral counts      | The number of neutral Tweets for a given category and date                 |
| Positive counts     | The number of positive Tweets for a given category and date                |

TABLE 4.5: Description of the Aggregated Tweet schema

For efficient and quicker processing of such voluminous amounts of Tweets, we take advantage of the parallel processing capabilities provided by PySpark with the use of Resilient Distributed Datasets (RDDs). RDDs are immutable, fault-tolerant data structures that run and operate on multiple nodes to do parallel processing

on a cluster. Multiple operations can be applied on RDDs to achieve the desired result.

Thus, the consolidated Tweet in JSON format is converted into an RDD. The data is then mapped with the Tweet's respective sector as the key and then reduced to obtain an intermediate aggregate result. The aggregated results are then stored in an SQLite database.

The aggregation process of raw Tweet data is shown in Table 4.4. The schema of the intermediate aggregated results is described in Table 4.5.

## 4.6 DATA ANALYSIS AND PREDICTION

### 4.6.1 Understanding The Data

Before performing correlation analysis on the obtained intermediate results from the PySpark module, there is a need to understand the details of the data that we are working with.

We have chosen four industrial sectors (categories) for which Tweets were collected based on relevant hashtags and keywords: Electric Vehicles (EVs), Oil & Gas, Technology and Pharmaceuticals. The following considerations were kept in mind while choosing candidate entities for our study.

1. Company Location - The United States of America (USA). This consideration is backed by the fact that the majority of Twitter users are

located in the USA - nearly 77 million (as of Jan 2022). Thus, market discussion would be more oriented towards American companies.

2. Market Capitalization by Segment - A large market cap indicates that it is a significant performer in that market segment. Hypothetically, social media discussion about relevant market topics would affect the performance of these stocks much more than other companies with lower market capitalization.

Table 4.6 indicates the amount of Tweets collected and processed for each sector during the two-year time period:

| <b>Sector</b>            | <b>Data Points</b> |
|--------------------------|--------------------|
| Electric Vehicles (EVs)  | 116,355            |
| Oil                      | 53,253             |
| Technology (Tech)        | 632,181            |
| Pharmaceuticals (Pharma) | 313,407            |

TABLE 4.6: Data Points Count

Initially, the stock market data and the aggregated Tweet sentiment data are merged based on the date column. As a result, we have a total of 16 columns in our dataset:

| <i>Category</i> | <i>Date</i>     | <i>Open</i>     | <i>Close</i>      |
|-----------------|-----------------|-----------------|-------------------|
| <i>High</i>     | <i>Low</i>      | <i>Count</i>    | <i>Neg_Counts</i> |
| <i>Ind_Neg</i>  | <i>Ind_Neu</i>  | <i>Ind_Pos</i>  | <i>Neu_Counts</i> |
| <i>Wted_Neg</i> | <i>Wted_Neu</i> | <i>Wted_Pos</i> | <i>Pos_Counts</i> |

The schema of these columns are mentioned in Table 4.4.

The entire dataset is now split and grouped according by the category column, so that each sector's data can be analysed separately.

Exploratory data analysis is done to understand the distribution and nature of the different columns.

It has been identified that there are no missing values in any of the columns, and that different columns have different ranges.

|   | date       | open     | close    | high     | low     | count | ind_neg   | ind_neu   | ind_pos   | wted_neg   | wted_neu   | wted_pos   |
|---|------------|----------|----------|----------|---------|-------|-----------|-----------|-----------|------------|------------|------------|
| 0 | 2020-04-07 | 109.0000 | 109.0900 | 113.0000 | 106.468 | 405   | 29.760626 | 42.419716 | 33.819664 | 61.686187  | 105.406782 | 237.907026 |
| 1 | 2020-04-08 | 110.8400 | 109.7680 | 111.4416 | 106.666 | 243   | 29.776691 | 47.333942 | 25.889368 | 64.262435  | 109.520843 | 69.216720  |
| 2 | 2020-04-09 | 112.4180 | 114.6000 | 115.0364 | 111.422 | 209   | 29.311668 | 44.887577 | 28.800758 | 57.566045  | 100.556852 | 50.877107  |
| 3 | 2020-04-10 | 115.2250 | 122.3950 | 122.7182 | 113.764 | 272   | 24.159784 | 47.111198 | 29.729010 | 39.036394  | 120.448533 | 112.515063 |
| 4 | 2020-04-11 | 116.6285 | 126.2925 | 126.5591 | 114.935 | 514   | 29.267159 | 42.111918 | 28.620923 | 289.783706 | 153.905570 | 70.310696  |

FIGURE 4.3: Merged EV Dataset

For a given category, all the features in our merged dataset, shown in Figure 4.3, are numerical values, apart from the date column. Thus, understanding the inherent distribution of each column is necessary.

It can be observed from the different distributions using a distribution and probability plot that few columns have a skewed distribution, and noticeable positive/negative skew in the probability plot. To fix the skew, we employ a log transformation of the skewed columns.

For reference, Figures 4.4(a) and 4.4(b) are shown to understand the original distribution of the Ind\_Pos column of the Pharma dataset. Figures 4.4(c) and 4.4(d) highlight how the dataset distribution changes for the same column upon log-transformation.

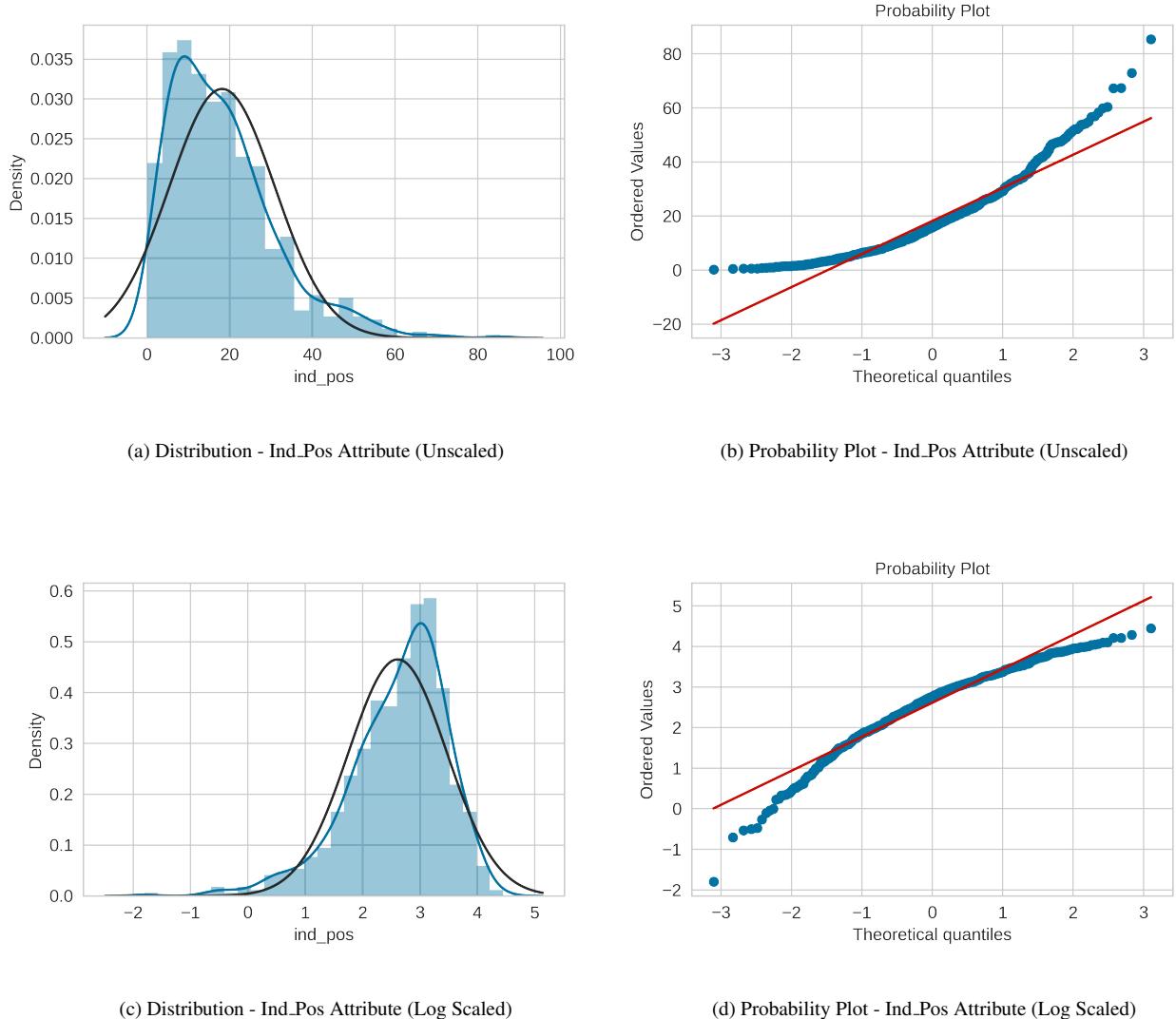


FIGURE 4.4: Pharma Data - Distribution & Probability Plots

To understand the market trends, a time-series plot is drawn on the close value for each company under study. The y-axis corresponds to the daily closing price of the company, while the x-axis corresponds to the date. Figures 4.5(a), 4.5(b), 4.5(c) & 4.5(d) show the variation of closing price for Microsoft Inc., Exxon Mobil Corp., Tesla Inc., and Moderna Inc., respectively between 2020 and 2022. It is observed that there are different close price trends for each company.



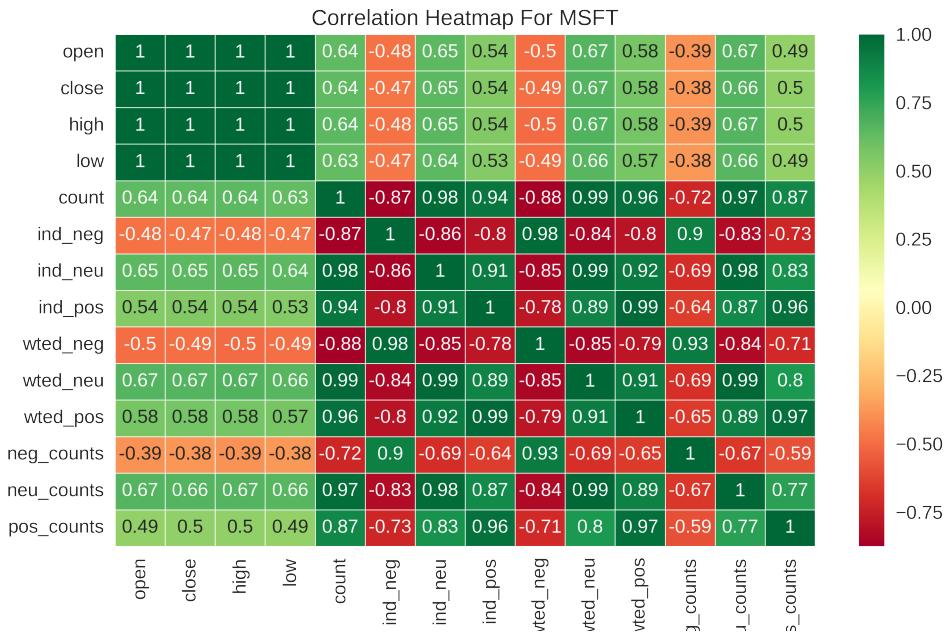
FIGURE 4.5: Company Closing values

The next step is to perform min-max normalization (Equation 3.1) on the datasets. After normalizing, all numerical columns in the dataset now have the same range of  $[0 - 1]$ .

## 4.6.2 Correlation Analysis

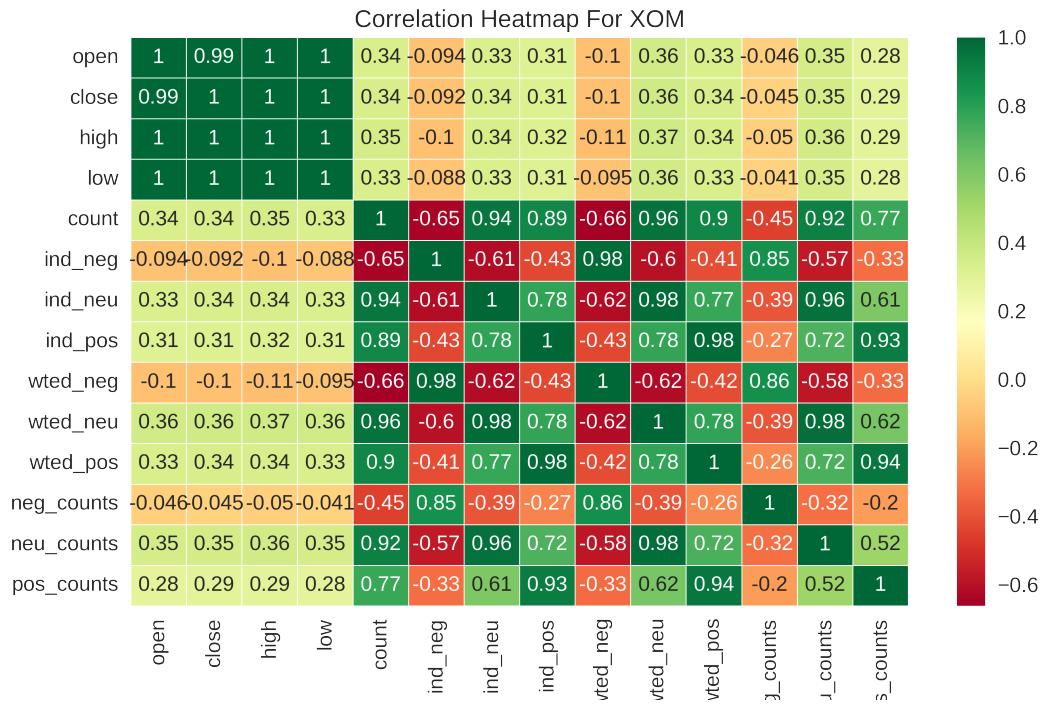
The crux of our research is to perform a correlation analysis between social media trends and stock market data. For the same, now we perform the Spearman's Correlation Coefficient calculation between every two columns of the dataset separately for each industrial sector.

A heatmap is plotted to capture the correlation observed in a visually appealing manner. The plotted heatmaps are shown in Figure 4.6.

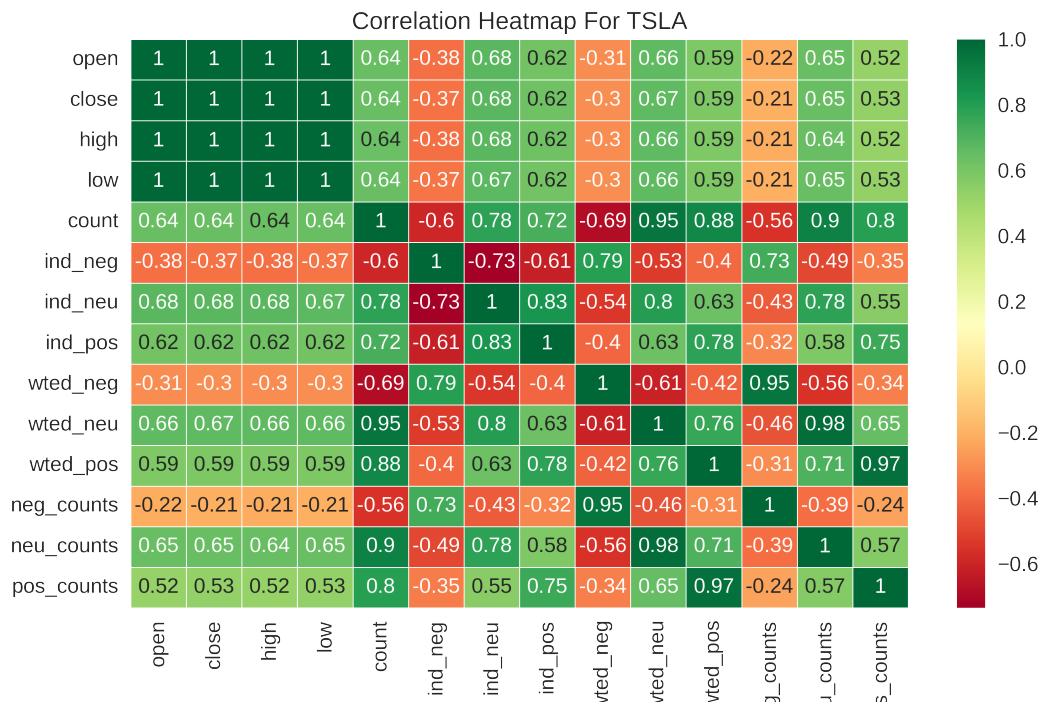


(a) Tech - (NASDAQ: MSFT)

FIGURE 4.6: Correlation Heatmap between Tweets and Stock Market

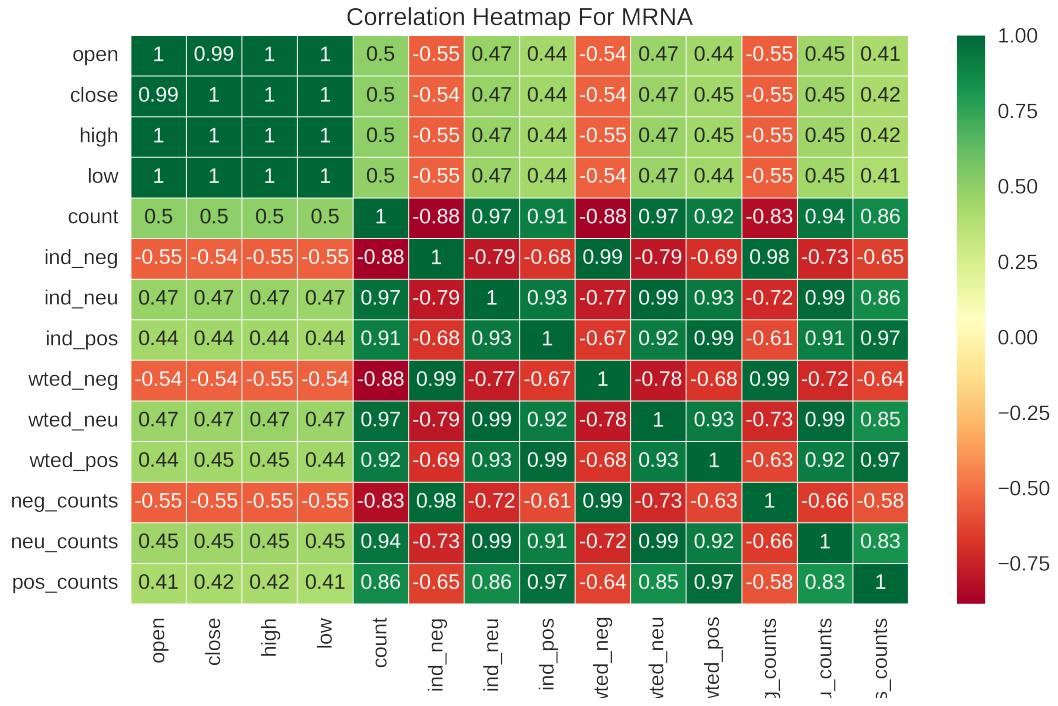


(b) Oil - (NYSE: XOM)



(c) EVs - (NASDAQ: TSLA)

### Correlation Heatmap between Tweets and Stock Market



Correlation Heatmap between Tweets and Stock Market

A statistical test of significance is also performed for the observed Spearman's correlation coefficient values (Table 4.7). Python's Scipy library implementation of the Spearman's correlation coefficient returns a 2-tailed p-value. This number signifies the probability of the following Null Hypothesis being true.

Null Hypothesis: The two variables under study are uncorrelated.

Alternate Hypothesis: The two variables under study are correlated.

The correlation value is considered statistically significant (i.e. the alternate hypothesis is accepted) if the p-value is lower than 0.01 (for a 99% confidence interval). Otherwise, the correlation is statistically insignificant, and can be ignored for all practical purposes.

| <b>Feature</b> | $\rho$ | $p$           |
|----------------|--------|---------------|
| Wted_Neg       | -0.493 | $5.418e - 46$ |
| Wted_Neu       | 0.667  | $3.368e - 95$ |
| Wted_Pos       | 0.576  | $5.669e - 66$ |

(a) (NASDAQ: MSFT)

| <b>Feature</b> | $\rho$ | $p$           |
|----------------|--------|---------------|
| Wted_Neg       | -0.100 | $7.235e - 3$  |
| Wted_Neu       | 0.364  | $5.945e - 24$ |
| Wted_Pos       | 0.339  | $6.987e - 21$ |

(b) (NYSE: XOM)

| <b>Feature</b> | $\rho$ | $p$           |
|----------------|--------|---------------|
| Wted_Neg       | -0.299 | $2.657e - 16$ |
| Wted_Neu       | 0.665  | $9.465e - 93$ |
| Wted_Pos       | 0.593  | $2.030e - 69$ |

(c) (NASDAQ: TSLA)

| <b>Feature</b> | $\rho$ | $p$           |
|----------------|--------|---------------|
| Wted_Neg       | -0.542 | $4.398e - 57$ |
| Wted_Neu       | 0.469  | $2.514e - 41$ |
| Wted_Pos       | 0.446  | $4.001e - 37$ |

(d) (NASDAQ: MRNA)

TABLE 4.7: p-values for Spearman correlation between Weighted Sentiments and Close

For our analysis, we consider only the daily close value to understand the correlation between a sector’s market performance and the corresponding day’s social media trend.

Thus, from Figure 4.6(a), we can see a strong positive correlation between the Ind\_Neu (0.68) and Ind\_Pos (0.62) columns to the Close column of Tesla (EVs) respectively. The neutral sentiment’s result can be interpreted as the market performing well with a rise in the amount of discussion about electric vehicles in general. The positive sentiment’s result can be interpreted in the following manner: A positive sentiment about the Electric Vehicle market being expressed in social media correlates to the appreciation of Tesla Motor’s stock market value. There is a moderate negative correlation between Wted\_Neg and the Close column (-0.37), indicating that the social media’s negative mood does not translate as much to depreciation of the market value.

On observation of the results in Figure 4.6(b) for the Exxon Mobil. Corp (Oil &

Gas Market) and the social media sentiments, with a moderate positive correlation between the Wted\_Neu (0.36) and Wted\_Pos (0.34) to the closing price. The weak correlation between the Wted\_Neg and the closing price (-0.1) indicates that the oil market does not fluctuate if there are negative sentiments being expressed about it on Twitter - making it a low-risk stock in that aspect. But the market price shows considerable appreciation with positive/neutral discussion about it on the social media platform, although not being very susceptible to social media trends.

For the Technology sector, and its reference stock market entity - Microsoft Corp., Figure 4.6(c) shows a strong positive correlation was obtained between the closing price and Wted\_Neu (0.67) and Wted\_Pos (0.58), respectively. A medium negative correlation was obtained between the closing price and Wted\_Neg (-0.49) attribute. The results indicate that there is a remarkable correlation between general social media discussion of the Tech sector and the stock market in both ways, indicating that the technology market is volatile to the mood of social media.

In the Pharmaceutical sector, with the chosen stock market representative as Moderna Inc., it was observed from Figure 4.6(d) that there is a medium positive correlation between the three weighted attributes - Wted\_Neg (-0.54), Wted\_Neu (0.47) & Wted\_Pos (0.45). This indicates that the Pharma sector's market behaviour is also volatile to the sentiments expressed on social media, especially on the negative side - a negative mood correlates more to the pharmaceutical sector's market price going down.

It is also worth noting that the correlations observed in Figure 4.6 are almost always significantly higher for the weighted/individual sentiment scores, as compared to the volume-based scores (Neg\_Counts, Neu\_Counts & Pos\_Counts).

This pattern clearly indicates that performing a sentiment-based analysis of social media proves to be a much more useful and promising methodology of understanding stock market patterns, as compared to a purely volume (Tweet count) based analysis.

### **4.6.3 Predicting the Stock Market**

We decided to additionally perform stock market prediction using machine learning algorithms since we observed a good correlation between the weighted Tweet sentiment scores and the close price of the market. From Nayak et al.'s [12] research on stock market prediction augmented by sentiment analysis of Twitter and news, it can be seen that Boosted Decision Trees perform the best in this scenario when compared to other algorithms like Logistic Regression and Support Vector Machines.

Thus, we decided to compare and contrast two widely-used decision tree algorithms that employ the concept of boosting to predict the close value of the above used stock listings, augmented by the weighted Tweet sentiment scores.

The first boosting algorithm under consideration is DMLC's XGBoost [2] (eXtreme Gradient Boosting). It was first presented in 2014, and is touted as an optimized, distributed gradient boosting library that is designed to be highly efficient, flexible and portable. It is a step further than boosted decision trees in the sense that it uses a parallelized approach to building and pruning trees, has in-built cross-validation and does regularization to prevent overfitting of data.

Another boosting algorithm that has gained great prominence in the recent years for its performance is Yandex's CatBoost [16], which was introduced in 2017. It is also a gradient boosting algorithm that natively handles categorical features, uses the GPU for faster training times, employs ordered boosting to prevent overfitting and has dedicated visualization tools for model analysis. It also claims to provide great quality without the need for fine-tuning, a common procedure done manually to find the best hyper parameters for the machine learning model to optimally perform the specified task. It is often compared with other boosting models like XGBoost and LightGBM.

#### 4.6.3.1 XGBoost

We use the Python implementation of XGBoost to construct our model. An initial model was constructed using the default parameters present in the XGBoost Regressor to visualize the results. It was observed that fine-tuning was required, and thus we employed the GridSearchCV() method, bundled as part of the scikit-learn library to find out the best hyper parameters for our use-case.

The hyper-parameters returned by the grid search algorithm are shown in Table 4.8.

Four separate regressor models were created and executed to predict the close value of the four listings used: TSLA, XOM, MSFT & MRNA. The dataset was split into 85% for training and 15% for testing, without shuffle.

The time series plots obtained for the test data of the four sectors are shown in Figure 4.7. The metrics obtained for the same test data are highlighted in Table 4.9.

| Hyper Parameter         | Value                     |
|-------------------------|---------------------------|
| Maximum Tree Depth      | 3                         |
| Number of Estimators    | 500                       |
| Objective Function      | Squared Error             |
| Booster                 | GB Tree                   |
| Learning Rate           | 0.1                       |
| Column Sampling by Tree | 1                         |
| Evaluation Metric       | Mean Absolute Error (MAE) |

TABLE 4.8: XGBoost - Optimal Hyper Parameters



FIGURE 4.7: XGBoost Prediction Results

|                                 |        |
|---------------------------------|--------|
| <b>RMSE</b>                     | 0.0471 |
| <b>MSE</b>                      | 0.0022 |
| <b>MAE</b>                      | 0.0377 |
| <b>R2 Score</b>                 | 0.8381 |
| <b>Explained Variance Score</b> | 0.8427 |
| <b>Max Error</b>                | 0.1403 |

(a) (NASDAQ: MSFT)

|                                 |         |
|---------------------------------|---------|
| <b>RMSE</b>                     | 0.1914  |
| <b>MSE</b>                      | 0.0366  |
| <b>MAE</b>                      | 0.1420  |
| <b>R2 Score</b>                 | -0.6285 |
| <b>Explained Variance Score</b> | 0.2131  |
| <b>Max Error</b>                | 0.4236  |

(b) (NYSE: XOM)

|                                 |        |
|---------------------------------|--------|
| <b>RMSE</b>                     | 0.0562 |
| <b>MSE</b>                      | 0.0031 |
| <b>MAE</b>                      | 0.0463 |
| <b>R2 Score</b>                 | 0.7086 |
| <b>Explained Variance Score</b> | 0.7555 |
| <b>Max Error</b>                | 0.1630 |

(c) (NASDAQ: TSLA)

|                                 |        |
|---------------------------------|--------|
| <b>RMSE</b>                     | 0.0366 |
| <b>MSE</b>                      | 0.0013 |
| <b>MAE</b>                      | 0.0255 |
| <b>R2 Score</b>                 | 0.8185 |
| <b>Explained Variance Score</b> | 0.8370 |
| <b>Max Error</b>                | 0.1885 |

(d) (NASDAQ: MRNA)

TABLE 4.9: XGBoost Regressor Metrics

The XGBoost Regressor models were able to capture the inherent patterns in the dataset and predict the next day closing price of the stock market based on the mood reflected by people on social media with significant accuracy. It performs the best (83.81% accurate) on the Tech-MSFT dataset, which follows from the observed high correlation between its weighted sentiment scores and its closing price. Performance was high on the Pharma-MRNA dataset as well (81.85% accurate), and good on the EVs-TSLA dataset (70.86% accuracy). The R2 score was negative for the Oil-XOM dataset due to lower inherent correlation (-0.62).

#### 4.6.3.2 CatBoost

Similar to XGBoost, we use the Python implementation of CatBoost to construct our model. An initial model was constructed with the CatBoost Regressor to

visualize the results. It had some fluctuations in the test data, thus we decided fine-tuning was necessary. We used CatBoost's in-built `grid_search()` method to find the optimal hyper parameters for our datasets.

The hyper-parameters given obtained from grid search are shown in Table 4.10.

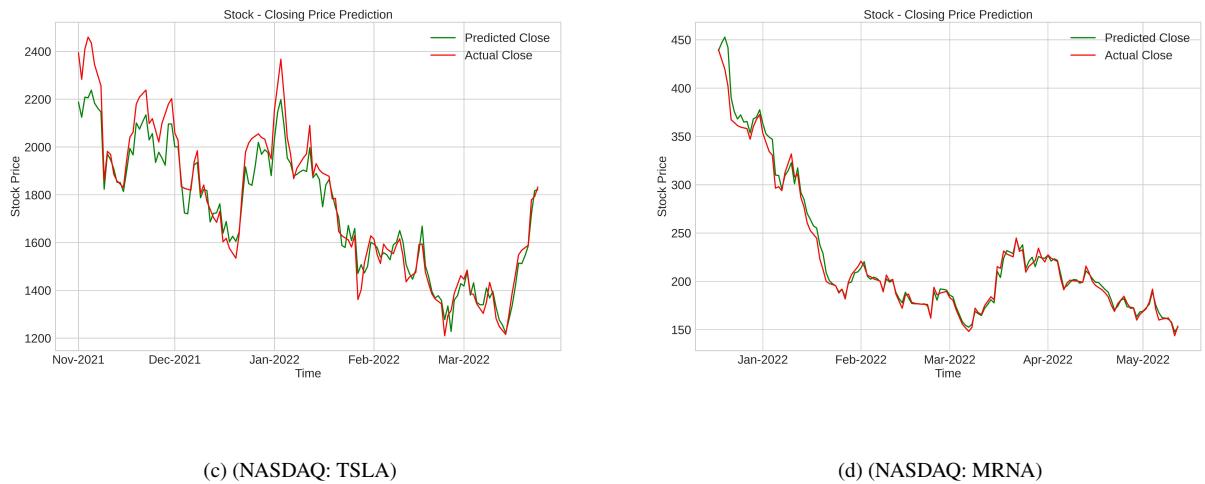
| Hyper Parameter        | Value                     |
|------------------------|---------------------------|
| Tree Depth             | 4                         |
| Iterations             | 200                       |
| Learning Rate          | 0.05                      |
| L2 Leaf Regularization | 10                        |
| Loss Function          | Mean Absolute Error (MAE) |

TABLE 4.10: CatBoost - Optimal Hyper Parameters

Separate regressor models were created and executed to predict the close value of the four listings. The dataset was split into 85% for training and 15% for testing, without shuffle. The time series plots obtained for the test data of the four sectors are shown in Figure 4.8. The metrics obtained for the same test data are highlighted in Table 4.11.



FIGURE 4.8: CatBoost Prediction Results



### CatBoost Prediction Results

|                                 |        |
|---------------------------------|--------|
| <b>RMSE</b>                     | 0.0480 |
| <b>MSE</b>                      | 0.0023 |
| <b>MAE</b>                      | 0.0383 |
| <b>R2 Score</b>                 | 0.8319 |
| <b>Explained Variance Score</b> | 0.8427 |
| <b>Max Error</b>                | 0.1288 |

(a) (NASDAQ: MSFT)

|                                 |         |
|---------------------------------|---------|
| <b>RMSE</b>                     | 0.1962  |
| <b>MSE</b>                      | 0.0385  |
| <b>MAE</b>                      | 0.1457  |
| <b>R2 Score</b>                 | -0.7114 |
| <b>Explained Variance Score</b> | 0.1810  |
| <b>Max Error</b>                | 0.4340  |

(b) (NYSE: XOM)

|                                 |        |
|---------------------------------|--------|
| <b>RMSE</b>                     | 0.0711 |
| <b>MSE</b>                      | 0.0050 |
| <b>MAE</b>                      | 0.0562 |
| <b>R2 Score</b>                 | 0.5328 |
| <b>Explained Variance Score</b> | 0.6047 |
| <b>Max Error</b>                | 0.2063 |

(c) (NASDAQ: TSLA)

|                                 |        |
|---------------------------------|--------|
| <b>RMSE</b>                     | 0.0337 |
| <b>MSE</b>                      | 0.0011 |
| <b>MAE</b>                      | 0.0251 |
| <b>R2 Score</b>                 | 0.8461 |
| <b>Explained Variance Score</b> | 0.8642 |
| <b>Max Error</b>                | 0.1491 |

(d) (NASDAQ: MRNA)

TABLE 4.11: CatBoost Regressor Metrics

It is observed that the CatBoost Regressor models were able to capture the inherent patterns in the dataset and come up with a fairly accurate model to predict the next day closing price of the stock market, with its best performance observed on the Pharma-MRNA dataset (84.61% accurate). Its performance was also high on the Tech-MSFT dataset with 84.27% accuracy, and 60.47% accuracy on the EVs-TSLA dataset. Like XGBoost, the R2 score was negative ( $-0.71$ ) on the Oil-XOM dataset due to the inherently lower correlation between the stock market and Twitter sentiment.

#### **4.6.3.3 Comparison of XGBoost and CatBoost**

It is observed that the XGBoost algorithm works better for the prediction of the closing prices of our four chosen stock listings. The XGBoost Regressor is able to match the observed (real) stock data more closely as compared to CatBoost. Both models are able to capture the trend of the market movement (increase/decrease). From Tables 4.9 and 4.11, we can conclude that the metrics are slightly better for the XGBoost model, except for the Pharma-MRNA dataset, in which CatBoost performed better with an accuracy of (84.61%), as compared to the XGBoost model which had an accuracy of (81.85%).

Thus, we believe that using either CatBoost or XGBoost for this task would be prudent, although there is a slight advantage to using XGBoost in terms of performance, as evidenced by Tables 4.9 and 4.11.

## CHAPTER 5

# CONCLUSIONS AND FUTURE WORK

In this report, a detailed discussion was done on the design and related implementation of stock market analysis using social media trends, using a big-data architecture. Apache Kafka and Apache Spark can be effectively used to construct a high performance, scalable pipeline that can use the power of parallelization and clustering to mine useful information from huge amounts of data. We were able to construct an end-to-end pipeline entirely using the Python language, straight from the first step of data collection, till the last step of visualization.

In our project, we have undertaken four stock market segments - Electric Vehicles, Oil & Gas, Technology and Pharmaceuticals - and four representative stock market entities - Tesla Inc., Exxon Mobil Corp., Microsoft Corp., Moderna Inc. to analyse the correlation between the sentiments observed in social media to the respective entity's stock market valuation.

From our observed results, we have seen conclusive proof that there is a correlation between the mood expressed on social media to stock market behaviour, although the extent of the correlation varies for each sector. We specifically chose these four sectors as they had a lot of discussion during the period of study (Mar 2020 - May 2022), and we chose market leaders as representative companies. The correlation might not be as heavily seen for other companies with a lower market share or low social media discussion. However, it is worth noting that for those companies which have a strong online community discussion, it is worth taking into account

those sentiments expressed by the community to predict how the stock market may change over time.

We were also able to identify two candidate machine learning algorithms - CatBoost and XGBoost, that can be utilized for efficient stock market prediction using the sentiment values.

We also took into account the conclusions of earlier research papers published in this area and introduced novel elements as per their specified line of future work. One example is the use of a transformer model (BERT) for the NLP task, and another is to use a larger dataset to broaden our sample space. Hence, we believe that our work has caused noteworthy progress in achieving the goals of this line of research.

With regards to future work, the inclusion of data from other social media websites like Facebook, Reddit etc. can present an even broader picture of the sentiment expressed by users online. The inclusion of the sentiments reflected in news articles could also prove to be more effective in trend analysis of the stock market. Thus, data from multiple sources can be pooled together and aggregated with a big-data architecture. Another consideration is the fact that the social media data that we work with are primarily produced by the English-speaking audience. Similar NLP techniques that work with other languages would be required to perform similar analyses in non-English speaking countries. With the inclusion of non-English Tweets (and other social media posts), it is possible to obtain a higher correlation score. The stock market data prediction task can further be improved and made reliable with the addition of other features that affect the stock market, apart from social media. These considerations remain as future areas of research to be explored.

## REFERENCES

1. Johan Bollen, Huina Mao, and Xiaojun Zeng. Twitter mood predicts the stock market. *Journal of Computational Science*, Vol. 2, No. 1, pages 1–8, Mar 2011.
2. Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 785–794. ACM, 2016.
3. Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Vol. 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019.
4. Fatih Gürcan and Muhammet Berigel. Real-time processing of big data streams: Lifecycle, tools, tasks, and challenges. *2018 2nd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*, pages 1–6, 2018.
5. Vaanchitha Kalyanaraman, Sarah Kazi, Rohan Tondulkar, and Sangeeta Oswal. Sentiment analysis on news articles for stocks. In *Proceedings of the 2014 8th Asia Modelling Symposium, AMS ’14*, page 10–15, USA, 2014. IEEE Computer Society.
6. Andreas Kanavos, Gerasimos Vonitsanos, Alaa Mohasseb, and Phivos Mylonas. An entropy-based evaluation for sentiment analysis of stock market

- prices using twitter data. In 15th International Workshop on Semantic and Social Media Adaptation and Personalization, SMAP 2020, Zakynthos, Greece, October 29-30, 2020, pages 1–7. IEEE, 2020.
7. Chungho Lee and Incheon Paik. Stock market analysis from twitter and news based on streaming big data infrastructure. In IEEE 8th International Conference on Awareness Science and Technology, iCAST 2017, Taichung, Taiwan, November 8-10, 2017, pages 312–317. IEEE, 2017.
  8. Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. CoRR, abs/1907.11692, 2019.
  9. Daniel Loureiro, Francesco Barbieri, Leonardo Neves, Luis Espinosa Anke, and José Camacho-Collados. Timelms: Diachronic language models from twitter. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics, ACL 2022 - System Demonstrations, Dublin, Ireland, May 22-27, 2022, pages 251–260. Association for Computational Linguistics, 2022.
  10. Sidra Mehtab and Jaydip Sen. A robust predictive model for stock price prediction using deep learning and natural language processing. SSRN Electronic Journal, Jan 2019.
  11. Anshul Mittal and Arpit Goel. Stock prediction using twitter sentiment analysis. Standford University, CS229 (2011 http://cs229. stanford.edu/proj2011/GoelMittal-StockMarketPredictionUsingTwitterSentimentAnalysis.pdf), 15:2352, 2012.
  12. Aparna Nayak, M. M. Manohara Pai, and Radhika M. Pai. Prediction models for indian stock market. Procedia Computer Science, Vol. 89, pages 441–449, 2016.

13. Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. BERTweet: A pre-trained language model for English tweets. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pages 9–14, Online, October 2020. Association for Computational Linguistics.
14. Venkata Sasank Pagolu, Kamala Challa, Ganapati Panda, and Babita Majhi. Sentiment analysis of twitter data for predicting stock market movements. 2016 International Conference on Signal Processing, Communication, Power and Embedded System (SCOPES), pages 1345–1350, 2016.
15. Zhihao Peng. Stocks analysis and prediction using big data analytics. 2019 International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS), pages 309–312, 2019.
16. Liudmila Ostroumova Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features. In Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada, pages 6639–6649, 2018.
17. Dharmaraja Selvamuthu, Vineet Kumar, and Abhishek Mishra. Indian stock market prediction using artificial neural networks on tick data. Financial Innovation, Vol. 5, pages 1–12, 2019.
18. Twitter-Roberta. <https://huggingface.co/cardiffnlp/twitter-roberta-base-sentiment-latest>. Accessed: 2022-01-07.
19. Twitter-API. <https://developer.twitter.com/en/products/twitter-api>. Accessed: 2022-02-10.
20. Polygon-API. <https://api.polygon.io/v1/open-close/>. Accessed: 2022-02-10.

21. Apache Spark. <https://www.apache.org/dyn/closer.lua/spark/spark-3.2.1>. Accessed: 2022-03-02.
22. Apache Kafka. <https://kafka.apache.org/downloads>. Accessed: 2022-03-09.
23. Yahoo-CMAK. <https://github.com/yahoo/CMAK>. Accessed: 2022-03-09.