CORRELATION ANALYSIS AND PREDICTION OF STOCK MARKET USING SOCIAL MEDIA TRENDS

A PROJECT REPORT

Submitted By

NUZAIR MOHAMED S. 195001073

SANYOG KAVE P. 195001099

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)
Kalavakkam - 603110

May 2023

Sri Sivasubramaniya Nadar College of Engineering

(An Autonomous Institution, Affiliated to Anna University)

BONAFIDE CERTIFICATE

Certified that this project report titled **"CORRELATION ANALYSIS AND PREDICTION OF STOCK MARKET USING SOCIAL MEDIA TRENDS"** is the *bonafide* work of "**NUZAIR MOHAMED S. (195001073)** and **SANYOG KAVE P. (195001099)**" 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**, Chairman, SSN Institutions. 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.

NUZAIR MOHAMED S. **SANYOG KAVE P.**

ABSTRACT

Stock market has always been a minefield of computational analysis and inference. The value of a stock is influenced by various factors besides the traditional quantitative and qualitative analysis of stocks, such as financial statements. In recent years, social media trends have become a major factor affecting the value of a stock. The sentiment expressed on such social media platforms about a particular stock tend to influence the investor's impression and acts as a catalyst for any stock and investment related decisions. This research aims to investigate the impact of these trends on stock value and to iteratively predict the price at which a stock is valued through a real-time simulation of data flow.. The study focuses on Twitter and Reddit as the data sources and performs sentiment analysis on extracted text data to determine their correlation with stock values within the relevant industry sector and use batch and stream processing methods to develop, simulate and test an effective active-learning based prediction model. To efficiently handle the large amount of data involved, the study proposes using popular tools for real-time large-scale data analysis, such as Apache Kafka and Apache Spark.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Computer science as a domain always has exhibited high scope for invention and discovery. Exploiting existing repositories of data from multiple domains and experimenting with new techniques and methodologies is the way forward as it always has been. Going ahead with such approach leads to widening of horizons and identification of newer domains of interest.

One such venue for exploration is the process of unearthing the relationship between different sectors that exist in the society. Naturally some sectors in their existence and operation interact with some other sectors and influence each other in a causal manner.

Performing a correlation analysis between stock market and social media trends as the related domains is more relevant now than ever due to the fact that almost everyone is on social media these days and huge volume of information is consumed. Opinions about topics are formed based on what is expressed in social media platforms like Twitter and Reddit. This leads to the truth behind the assumption that sentiments expressed on social media like Twitter and Reddit regarding a particular company and/or its stock have an almost direct effect on the price of a stock.

Stock market in itself is also a highly volatile space that involves numerous factors that influence it, huge amounts of monetary dealing and plays a very vital role in the economy of a country. Additionally, it can be considered a minefield of research as huge amounts of profit can be availed with statistical analysis and

prediction models. Thus the correlation analysis becomes relevant and to add to its importance, companies have started collaborating with stock market agencies to identify usage of spam and bots to attempt at price manipulations.

Such is the scale of interrelation between these domains and it is fruitful to decode the relationship and predict the influence on a numerical basis.

1.1 MOTIVATION

The value of a stock is influenced by various factors besides the traditional quantitative and qualitative analysis of stocks, such as financial statements. In recent years, social media trends have become a major factor affecting the value of a stock.

This research aims to investigate the impact of these trends on stock value and to iteratively predict the price at which a stock is valued through a real time simulation of data flow. The study focuses on Twitter and Reddit as the data sources and performs sentiment analysis on extracted text data to determine their correlation with stock values within the relevant industry sector and use batch and stream processing methods to develop and test an effective active-learning based prediction model.

To efficiently handle the large amount of data involved, the study proposes using popular tools for real-time large-scale data analysis, such as Apache Kafka and Apache Spark.

1.2 PROBLEM STATEMENT

Understanding and predicting the movement and the subsequent states of the stock market is difficult due to numerous factors, including economic climate, current trends, politics, and interest rates. Investors rely on multiple sources of information to make decisions about buying or selling stocks, including news events and social media trends.

Social media trends, in particular, have become increasingly influential for amateur stock market investors as they tend to be regarded by them as highly credible sources of mass information. Immediate reactions to the stock market can be seen when incidents are reported or propagated using extreme sentiments on social media platforms.

Hence it becomes essential to assess and quantify the effect of social media on the economics of different sectors and hence its influence on the stock market to effectively employ a prediction model. To analyze and infer the correlation, a large sample space with credible information is required.

Since the considered data sources are of high volume with a high degree of velocity, deriving interesting patterns and inferences facilitate the need for a big-data architecture. Specific sectors like Tech, EVs and Finance are used as the sample space for the analysis and prediction as there exists a growing innate correlation between these topics and the stock market.

Hence predicting the stock values in these domains becomes the need of the hour with growing engagement and influence on the economic strength.

CHAPTER 2

LITERATURE SURVEY

Since the stock market has proved to be a minefield for research and investigation, it has come under scrutiny with the advent of artificial intelligence, data science and machine learning. AI and ML researchers have taken interest in it as there is a high nature of risk and uncertainty involved.

The prices of a stock are primely impacted by the supply and demand from the market. The rate of change of the price value is determined by how the demand and supply change over time. In general, there is protection from manipulations by regulating it, but the decisions regarding the handling i.e buying or selling of stocks is solely at the discretion of the investor and subject to their analysis and grasp.

There are several factors by which stock prices are affected like Government Policies, Economic outlook, Political upheaval, Interest rates, Current events, Behavior of the stock historically, Industry value rise/fall, Seasonals, Exchange rate fluctuations, Natural calamities, etc.

Our interest is to explore the correlation that exists between stock values and one significant factor that has a insurmountable degree of influence in the recent years: social media trends. Twitter has been considered as the source of data for multiple research work. Here we intend to incorporate the platforms Twitter and Reddit as data sources.

2.1 EXISTING IDEAS

The research work by Lee et. al.[1] focuses on analyzing Twitter data and establishing a streaming pipeline. Initially twitter and news data are collected and preprocessed, using which a classification model is constructed to classify real time tweet data into various categories. This classification is superimposed on to stock data to find correlation between stock market and events. The results are visualized for the real time trend and the stock market.

Bollen et al. [2] analyze whether emotions can profoundly affect individual behavior and decision-making. This research identified that by focusing on specific mood dimensions, the accuracy of Dow Jones Industrial Average (DJIA) can be increased.

A subsequent research study was conducted by Mittal et al. [3] built upon the findings of the aforementioned paper. This paper utilized Twitter data to predict public mood and combined it with past DJIA (Dow Jones Industrial Average) values to forecast stock market movements. They also proposed an algorithm to estimate stock market data for days when the market was closed due to public holidays or weekends. Various algorithms, including Regression, SVM (Support Vector Machine), and SOFNNs (Self-Organizing Fuzzy Neural Networks), were employed for stock market prediction in their study.

Nayak et al. [4] have conducted significant research on sentiment analysis of Twitter data for stock price prediction. They analyzed trend data from Yahoo Finance for three sectors: Banking, Mining & Oil, and combined it with sentiment data extracted from relevant Tweets. The authors also proposed algorithms to calculate closed price trends for each day, identify continuous up or

down days for a stock, and combine sentiment with historical data. They used three supervised machine learning models to predict stock market movement based on historical stock data and social media sentiment data, with the Boosted Decision Tree model yielding the best results.

Kalyanaraman et al. [5] conducted an analysis of the relationship between the stock market and public sentiment using news articles. They focused on 11 companies listed on the National Stock Exchange (NSE) and collected around 60 news articles for each company after pre-processing. The sentiment of the articles was manually labeled as positive or negative, using a custom sentiment dictionary. The news articles were then vectorized and used to build a classifier using Linear Regression and Gradient Descent. Their model achieved a sentiment prediction accuracy of around 60% and a stock price movement direction (positive or negative) accuracy of 81.92%.

This study by Kanavos et al. [6] presents a methodology that forecasts the movement of the initial price (opening price) of the shares for a specific company. The twitter data was collected using Apache Flume streaming engine and classified for preprocessing using Naive Bayes classifier. They performed sentiment analysis using Stanford Core NLP library.The estimated stock prices for the test data is compared to the actual prices derived from the dataset. Employs the usage of n grams.(n - number of sentiments). High value of entropy indicated that the appearance distribution of an n-gram in different datasets when considering sentiment analysis tends to become uniform.

Pagolu et al.[7] conducted a correlation analysis between Tweets related to Microsoft and its stock price movements over a time period from August 2015 to August 2016. They converted the correlation analysis into a classification

problem, using the total negative, neutral, and positive emotions observed in Tweets within a 3-day period as input features. They employed Word2Vec for sentiment analysis and utilized Logistic Regression and SVM for classification. Their model achieved an accuracy of approximately 70% in predicting the stock movement trend.

The work of Mehtab et al.[8] builds upon the research conducted by Mittal et al. by expanding the scope of their prediction models to include eight regression and eight classification models for NIFTY 50 stock price movement. Additionally, they incorporate public mood data extracted from relevant Tweets to enhance the accuracy of their predictions.

In their published paper, Selvamuthu, Kumar, and Mishra [9] discuss the utilization of various Artificial Neural Networks (ANNs) with different learning algorithms to predict the stock market in India using tick data. The authors propose the incorporation of a larger dataset to capture seasonal trends and emphasize the potential of sentiment analysis, as statements made by influential individuals can impact the stock market. This approach could provide an additional advantage in predicting stock market movements.

Attigeri et. al.[10] in their work attempted prediction of stock market using a big data approach. Technical analysis is done using historic stock data by applying several machine learning algorithms. Fundamental analysis is done by considering stock data along with the factors that affect it, taking into account both historic and present data with the objective of making financial forecasts. It was shown that this method is able to predict the sentiment and the stock performance and its recent news and social data are also closely correlated.

Observing the work of Pei et al., FinNLP 2022,[11] a dataset that was specific for stock sentiment was created as the research revealed that traditional sentiment analysis models provide sentiment labels and values based on feelings and emotions of good and bad but this definition of sentiment is not an accurate indicator of public opinion about specific stocks.

From the work of Yadav et. al.[12], methodology to predict stock market live was explored. Deep learning models for live prediction task, one based on FastRNN and the other, a hybrid model utilising the best features of FastRNNs, Convolutional Neural Networks, and Bi-Directional LSTM models were developed and the performance was compared to existing state-of-the-art models. Data from 4 companies on a minute to minute basis was taken for 7 hours 10 minutes as training data and live prediction for every minute in the next 50 mins was carried out.

Rouf et. al.[13] explore a decade long survey of methodologies, recent developments and future directions in the field of stock market predictions. This work aimed at deploying a generic framework for prediction as a result of the decade long survey. The uncovered information from the last decade was analyzed critically and a full fledged comparative analysis study was carried out to discover the direction of significance to proceed with. The paper concluded that SVM was the most popular technique used but more recently, ANN and DNN attracted more research and that the deep learning approaches provided better results.

The research carried out by Shashanka et. al.[14] has conclusive proof that social media trends particularly influence the stock market. Twitter and stock data were collected with respect to 4 specific domains and a numerical correlation analysis

was presented. The results provide an insight into the degree of influence that the twitter sentiments have on the stock market.

2.2 EXPLORING TECHNOLOGIES INVOLVED

In previous research, various natural language processing (NLP) techniques such as n-grams, Word2Vec, and TF-IDF were used for sentiment analysis. However, some research papers as stated above have mentioned that utilizing deep learning models could potentially improve the accuracy of sentiment analysis. In our methodology, we have incorporated the use of a popular and state-of-the-art NLP transformer model called BERT which stands for Bidirectional Encoder Representations from Transformers, specifically a pre-trained RoBERTa-base model. This model has been trained on a large dataset of 124 million English Tweets collected from January 2018 to December 2021, as mentioned in reference [15].

In the TweetEval task[16], several other models' performance was compared against the RoBERTa-base model. This model demonstrated competitive performance, outperforming most other systems, with a sentiment analysis test result accuracy of 71.3%. The models that performed slightly better were BERTweet, which achieved an accuracy of 73.4% and RoB-RT with an accuracy of 72.6%. We chose to use the discussed model because it has been trained on a more recent Tweet dataset and also because BERTweet has a lower token limit compared to the RoBERTa-based model. Also the results of the sentiment analysis from RoBERTa-base model can be obtained as 3 values which denote the

probability score assigned to the tweet in the 3 categories of negative, neutral and positive sentiment score which aided our analysis in the further course.

Coming to the architecture of our system, from the work of Gurcan et. al.[17], valuable insights into the understanding of the lifecycle, related tools and tasks, and challenges of real-time big data processing can be inferred. It describes existing tools, tasks, and frameworks by associating them with the phases of the lifecycle, which include data ingestion, data storage, stream processing, analytical data store and analysis and reporting. The paper also investigates the real-time big data processing tools consisting of Storm, Nifi, Flume, Kafka, Spark Streaming, Splunk, Flink, S4, Samza, Hbase, Cassandra, Hive Sap Hana and also the limitations and challenges of real time big data processing. Further the work by Shashanka et. al. was of high importance for adapting the architectural design of the system using Apache Kafka and Spark for data ingestion, consumption, streaming and processsing. The pipeline that they created was efficient and adapted in this project.

Peng's research[18] 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.

Apache Kafka documentation was of high use to incorporate data ingestion and streaming. The collected data was stored and produced to a kafka topic using producer code that was customized for this use case. The creation and management of kafka clusters and topics can be carried out via command line or using a GUI tool called Cluster Manager for Apache Kafka (CMAK).

Apache Spark documentation was explored for the process of consuming the streamed data from the kafka topic. Spark Structured Streaming concepts were used to process the data from the pipeline based on our needs. The concept of RDDs(Resilient Distributed Dataset) and their efficient usage for parallel processing were studied.

After the task of correlation analysis, to carry out a prediction task based on the observed correlation, several algorithms were analyzed and it was identified from Nayak et al.'s [4] research, that Boosted Decision Trees perform the best in this scenario.

Thus we decided to compare and contrast very commonly used gradient boosting frameworks that employ the concept of boosting for the prediction task.

A boosting algorithm that showed great relevance in recent years for high performance in prediction tasks is Yandex's CatBoost[19] which was first introduced in 2017. It is a machine learning algorithm that belongs to the family of boosted decision tree algorithms. CatBoost is a gradient boosting framework that is designed to handle categorical features efficiently and effectively. What sets CatBoost apart is its ability to handle categorical features directly without the need for explicit feature engineering, which is often required in other gradient boosting algorithms. CatBoost employs a unique algorithm for handling categorical features called "ordered boosting", which automatically handles the encoding of categorical features during the training process. This makes CatBoost particularly well-suited for datasets with a mix of numerical and categorical features. CatBoost also provides various advanced features such as built-in handling of missing values, support for GPU acceleration, and automatic handling of overfitting through a novel technique called "best-fit" grid search.

These features, along with its ability to handle categorical features efficiently, make CatBoost a popular choice for machine learning tasks that involve datasets with categorical features.

Another algorithm that was considered is LightGBM[20] (Light Gradient Boosting Machine), a gradient boosting framework that is based on decision tree algorithms which was introduced by Microsoft Research in 2016. It is a machine learning algorithm that belongs to the family of boosted decision tree algorithms. Gradient boosting is an ensemble technique that combines the predictions of multiple weak learners, such as decision trees, to create a stronger overall prediction model. In gradient boosting, the weak learners are sequentially trained to correct the errors made by the previous learners. LightGBM is one such implementation of gradient boosting that uses decision trees as its base learners. LightGBM is known for its efficiency and scalability, as it is designed to handle large datasets with millions or billions of instances and features. LightGBM supports various advanced features such as handling missing values, categorical features, and early stopping to further improve the performance of the model.

CHAPTER 3

DESIGN OF PROPOSED SYSTEM

This section provides a comprehensive description of the proposed system's design, including the factors and alternatives that were taken into account during the design process to fit the use case best. The proposed system architecture is majorly based on data streaming architecture as that is the major part that holds high importance. This section involves various modules outlined, such as

- Data Collection
- Data Storage
- Streaming Data Generation
- Sentiment Analysis
- Data Transformation
- Correlation Analysis
- Visualization
- Prediction

Each module is discussed in detail.

3.1 SYSTEM ARCHITECTURE

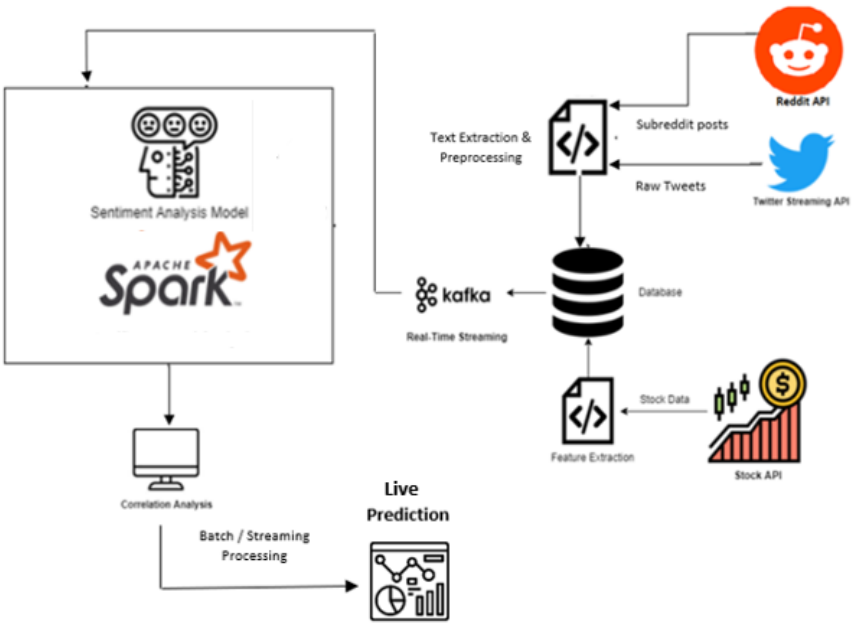

FIGURE 3.1: Proposed Architecture

3.2 OVERVIEW OF THE PROPOSED SYSTEM

Data from several APIs are collected separately and then preprocessed appropriately to preserve cleanliness of the data. Data are now integrated to be of a consistent schema and loaded into a database so that subsequent processing steps can uniformly be applied. Stock data are gathered using Stock API and required features are extracted and also loaded into the database. Then streaming of data is carried out using Kafka. The input stream of data is now fed to a pretrained sentiment analysis model.

Sentiment analysis is carried out and the resulting sentiments are aggregated per category per day. Stock data is also aggregated in the similar manner by calculating

the percentage of change of the value based on close and open values of the stock on that day.

Now these data are correlated to derive inferences about the correlation. The extracted features are all correlated and the resulting correlation values are observed. The correlation results can be now used to decide whether prediction of stock using the sentiments got can be performed.

The data can be batch or stream processed based on the requirement and effectively used to predict the price of a stock such as close, high, low value.

An overview of the entire architecture is shown in figure 5.1. Each module in the proposed system architecture is explained in detail in the subsections below with sufficient examples of usage.

3.3 DATASETS USED

3.3.1 Stock Data

Many APIs provided stock data but eventually charged a lot to get data for a longer span of time or to provide it in a consistent format that was required for this use case. We then decided to use Polygon.io Stock Api[22] to get the required data but on usage we found that their database had a lot of missing values in terms of days and also found a lot of inconsistency in their data.

We then ended up using the free Yahoo Finance API[26] to get the required stock ticker values. The API allows us to query them based on a company's ticker and the time period for which we need the data. The API returned the

1. Opening
2. Closing
3. High
4. Low

values of the stock on each day in the requested time frame.

3.3.2 Tweet Data

To collect Tweet data, we make use of the official Twitter API[21], which offers three different levels of data access. However, the free Essential Access plan only provides historical data for a limited period of up to 2 months, which is insufficient for our objectives. Therefore, we require a larger sample size to conduct our research effectively.

Fortunately, the Academic Research Access plan offered by the Twitter API allows access to the entire archive of Tweet data, with a rate limit of up to 50 requests per 15 minutes. The API hits limit was sufficiently high and gets reset on day to day basis. Hence, we employ this higher-tier API plan to collect the required Tweet data for our dataset.

The API returns various tweets from its database based on the keyword we give as a query parameter. The API allows us to query with options like date range in which tweets are to fetched, sort by filter, and also options whether to include links and retweet data. The API returns an array of tweet data objects having the following keys :

1. Tweet Title
2. Owner
3. Date and time of the tweet
4. Retweet count
5. Other meta data related to the tweet

3.3.3 Reddit Data

Finding the correct API for reddit data was the hardest part of data collection. The official API restricts us access to only recent reddit data and didn't allow us to query based on a particular time frame. Since all existing high level APIs are paid, owing to feasibility reasons, we had to search extensively to identify a credible and free source that can be exploited to extract data as per the needs.

We then explored to find a currently under-development API made by pushshift.io.[27] It allows us to query the reddits and subreddits based on keyword and also in the particular time frame required. This API is under development and hence allowed us to get reddit data from the month of August 2022. The API returns an array of reddit data objects having the following keys :

1. Reddit Post Title
2. Owner
3. Date and time of the Reddit post
4. Upvotes count
5. Other meta data

3.4 DATA STREAM GENERATION

The collected data from sources i.e Twitter, Reddit and Stock data are generated as a stream to provide as input for further processing. Data cleaning is performed here since the sources are social media platforms, noisy and unrelated data are bound to be present and are inevitable.

Data are manually analyzed to infer the presence of commonality and general patterns so that they can be collectively removed. The data collection part is tuned in such a way that all the data post collecting and cleaning can be organized into a single schema thus effectively producing an integrated data source and achieving data integration.

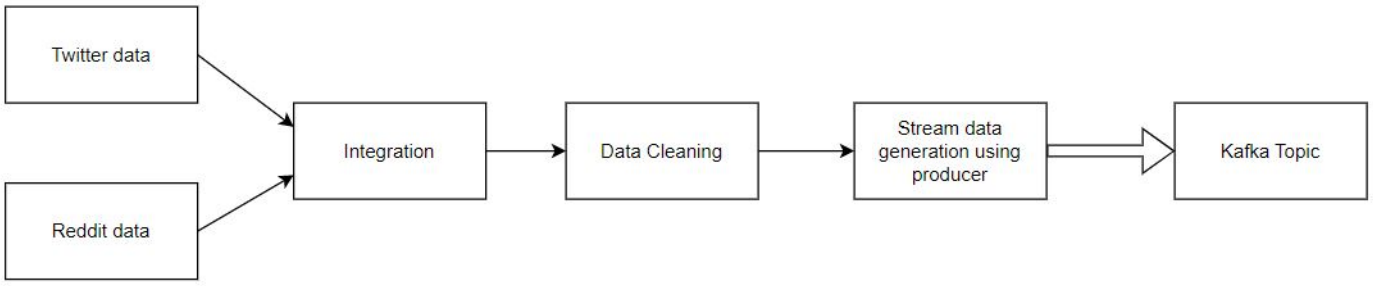

FIGURE 3.2: Data Stream Generation

Regarding the stock data, the presence of missing values is identified and a python script is written to approximate the values based on the corresponding data fields and this achieves the state where the data provides a view of continuous availability.

Since the amount of data being dealt with is huge, the concept of stream processing was considered to be incorporated in the design process. Now that data is ready after preprocessing, a producer script is written to produce the data and stream it uniformly. The script is designed in such a way that the data are streamed in batches of predefined size and accompanied by a specific timeout interval. Categories were decided for the research and relevant topics were created in Kafka and the producer script can be tweaked such that data corresponding to that particular topic can be generated and streamed into the relevant topics.

3.5 SENTIMENT ANALYSIS

Once the data stream part is ready, the data lines have to be analyzed for the sentiment expressed in it as sentiment analysis is an integral component in the research for comprehending the existence of an impact of social media trends(emotions) on stock market.

Natural Language Processing domain has had many developments in recent times of which the deployment of Encoder - Decoder model based on Transformers, specifically Bidirectional Encoder Representations from Transformers (BERT)[21] has grasped the attention due to its high efficiency and performance. BERT is a standout model as it employs the training and attention mechanism in a Bidirectional manner leading to a more profound sense of understanding and grasp of the context than unidirectional attention models. BERT has taken the NLP domain by storm and is being used frequently in several NLP tasks like Question-Answering, Text Classification, Text Summarization, Sentiment Analysis, etc.

Existing sentiment analysis models provide a single output definitively stating the label of the sentiment expressed. The sentiments expressed are classified generally into the following three labels:

- Positive
- Neutral
- Negative

But for the specifics of this use case, more than a definitive label stating the sentiment expressed, it is required that for each data point the probability that the data point belongs to each of the 3 sentiment labels needs to be quantized as a numeric value. For this reason, several sentiment analysis models were analyzed, and as per recent research the models [16] were shortlisted and the factors stated in the literature survey were considered.

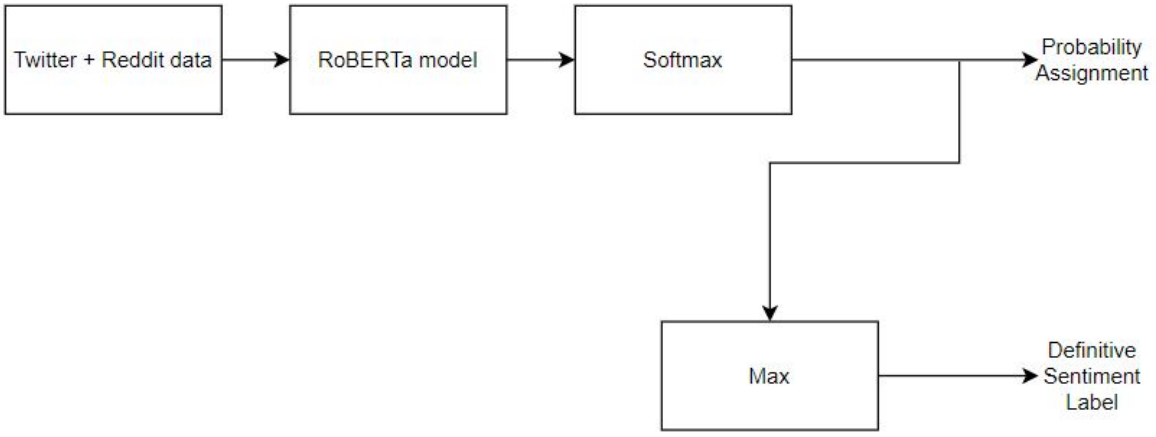

FIGURE 3.3: Sentiment Analysis Flowchart

Cardiff NLP research group's publicly available TimeLMs based sentiment analysis model[15] was chosen as the output was able to be obtained in the required format and its performance is comparatively competitive. This model was trained based on the already existing RoBERTa-base model, that then is a BERT-base model where a large corpus of English data was used to pre-train the model using the training approach as discussed in its research paper[22]. This sentiment analysis model was pre-trained on a dataset of approximately 124M Tweets collected between Jan 2018 - Dec 2021.

This model has been compared with other models in the TweetEval task - a unified Twitter benchmark composed of seven heterogeneous Tweet classification

tasks. The results obtained were satisfying, and perform comparatively well - only overtaken by the BERTweet[23] model on the final mean results, by only a relatively small margin. Yet, this model was chosen as it can perform the best for our use-case, as it has been pre-trained on Tweet data and can provide output in the required format. The performance being on a slightly upper side, can be credited to this as it is a competent language model that understands the context of language used in it.

3.6 DATA TRANSFORMATION

Data transformation part is achieved majorly by the use of Apache Spark, using the PySpark API. The complete transformation process can be visualized in the diagram shown. The combined Twitter and Reddit data that is stored in the database is taken and streamed into the corresponding Kafka topics based on the category. The data from these topics are now consumed through a stream processing engine built on the SparkSQL framework, Spark's Structured Streaming. This now performs all the incremental calculations and updation of results. Once the spark session is initiated and data is fed, the Spark engine can do several tasks like applying the processing logic on the data (whilst dividing them into micro batches and operating on them) and finally writing the output to external permanent storage. PySpark offers abstraction in the DataFrame API level that can execute variety of functions like stream to batch joins, event-time window management, streaming aggregations, etc.

FIGURE 3.4: Data Transformation

Aggregation methods and procedures are created to convert individual data points from the fed data stream, to get intermediate grouped data based on categories, and the result is stored for further computations and inference. For the two identified tasks in data ingestion and processing, several tools were analyzed like Apache Flume, Apache NiFi, Apache Kafka, Apache Samza, Apache Flink, Apache Spark. Of these, Apache Kafka and Apache Spark were chosen for the ingestion and processing tasks respectively as both these tools had wide community support, were scalable, lightweight and hence incorporation was really simple. Apache Spark could also provide options for batch and stream processing. Additionally, the pipeline created and established by Shashanka et. al.[14] was already proven to provide results.

3.7 CORRELATION ANALYSIS

Post processing the full stream of combined Twitter and Reddit data and having the intermediate results, the intended process of correlation analysis is carried out.

The sentiment data are now grouped based on category and date, such that finally the data are organized in such a way that, for a given category on a given date, the sum of individual sentiments i.e positive, negative and neutral, their weighted

average, multiplied by the retweet count for twitter data and upvote count for reddit data, and the individual counts of each sentiment are obtained as part of the aggregation process done using the Spark framework.

Stock data are also aggregated in a similar way, such that for a given category on a given date, the aggregate percentage is calculated based on the open and close values.

These aggregated data are now merged into a single dataframe based on the date and category so finally we have a table that can clearly indicate the overall sentiment expressed on a day about a particular category and its respective stock market performance on that particular day.

Next, using min-max normalization, feature scaling is done on the numerical columns in the data and this makes sure that different columns present in the data are mapped into the same range and it is crucial that they are all in the same range for effective comparison and performance analysis.

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

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

$$y' = \frac{y - x_{min}}{x_{max} - x_{min}}$$

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.

FIGURE 3.5: Min - Max normalization

The Correlation Analysis can be performed with Pearson's correlation coefficient which analyses the linear relationships in the data, Spearman's correlation coefficient which is a non parametric measure of rank correlation between two variables and analyses monotonic relationships without considering linearity and Kendall correlation coefficient which is used to measure the similarity of the orderings of the data when ranked by each of the quantities.

Spearman's can be accepted as the better correlation method as it discards the linearity aspect of the analysis since stock market can be influenced by multiple factors and the impact may not be linear.

$$\rho = 1 - \frac{6\Sigma\, d_i^2}{n(n^2 - 1)}$$

ρ 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:

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

Where:

ρ 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.

FIGURE 3.6: Spearman correlation coefficient

To visually analyze the correlation, heatmaps are generated for each category and the correlation between each pair of attributes from the sentiment values and stock values is displayed.

3.8 VISUALIZATION

To gain a better understanding of the relationship between different data attributes and the correlation between social media data and Stocks, it is essential to perform graphical visualizations on the collected and processed data.

Python offers a wide range of graphing packages that generate colorful and informative charts and diagrams. Hence, we utilize popular packages like Matplotlib, Seaborn, and Plotly to enhance our research with meaningful and

interactive graphs, facilitating exploratory data analysis (EDA). This enables us to present the entire workflow related to data analysis, correlation study, and visualization in an easy-to-understand and visually appealing manner using Interactive Python Notebooks (IPYNBs).

By leveraging IPYNBs as an interactive front-end, we can easily view, manipulate and understand the processed data, supported by graphing utilities. This helps us gain valuable insights into the data and identify patterns and trends that may have otherwise gone unnoticed. Overall, the use of graphing tools and IPYNBs significantly enhances the effectiveness and impact of our research.

3.9 PREDICTION

The correlation obtained between the social media sentiments and stock market values can be leveraged to predict the same. We use batch processing of the collected historic data and feed the the ML Model to train them. The trained model can be then used to predict the stock market values such as closing, high and low based on the live sentiments in social media collected for that particular day.

Nayak et al.[4] conducted a research on predicting stock market trends by utilizing sentiment analysis of Twitter and news. The findings suggest that Boosted Decision Trees outperformed other algorithms such as Logistic Regression and Support Vector Machines in this scenario and hence motivated us to use the same for our prediction.

CHAPTER 4

IMPLEMENTATION

This chapter gives a detailed explanation of our approach and implementation of the proposed system. It involves all steps right from the data collection scripts to streaming setup till the prediction module mentioned in the previous chapter. The chapter also gives in detailed explanation for the installation and setup of the Apache Kafka - Spark architecture.

The implementation is carried out in a system running on Windows 10 OS. The system environment incorporates the usage of OpenJDK 11.0.18, GraalVM CE 22.3.1, Scala 3.2.2, Python 3.9.13

The installation procedure and details of Apache Spark and Kafka are mentioned as adopted for Windows from the pipeline established by Shashanka et. al.[14] for Linux based systems.

4.1 DATA COLLECTION & PRE-PROCESSING

4.1.1 Stock Data collection

After experimenting with the Polygon API[23], it was decided that the Yahoo Finance API (GET https://query1.finance.yahoo.com/v7/finance/ download/)[27] can be used to collect stock ticker data as it was quicker and provided instantaneous data. The API accepted the type of ticker data required, and the

date to get the data from. A Python script was used to collect Stock data for the span of August 2022 to March 2023, and was stored in a dataset categorized according to the sector of the companies queried. We get the opening, closing, high , low data of each stocks considered from this API.

4.1.2 Tweet Data collection

The Twitter Streaming API (GET https://api.twitter.com/2/tweets/search/all) was used to collect the data from Twitter. Textual data was extracted from the raw data by removing emojis and the junk characters. The general developer API had limitations like dataset, time period and filter conditions due to which, the Twitter Academic Research access API had to be used. This API was used to query the data by providing keywords for each category of analysis over the span of August 2022 to March 2023. A Python script was written to collect the data, and generate a dataset of tweets labelled according to their category.

4.1.3 Reddit Data collection

The Reddit API (GET https://api.pushshift.io/reddit/search/submission/) was used to collect data from the social media platform Reddit. A python script was used to extract data and do small cleaning from the subreddits using the specific keywords chosen and saved in a CSV file according to their extracted category.

Sector	Data Points
Finance	1,77,548
Electric Vehicles (EVs)	57,621
Technology (Tech)	41,623

TABLE 4.1: Collected Data Points - Tweets & Reddits

4.1.4 Data cleaning

4.1.4.1 Tweet and Reddit data cleaning

Since the dataset involves raw data from the databases of the various API and is queried on keywords, there are lot of chances of noisy and irrelevant data. Various fine tuning in the query parameters of the API that were given as option were used to filter out only relevant data from the API. The JSON data obtained from both the API is processed to eliminate emoticons and unnecessary characters. Any irrelevant information, such as links, is also filtered out from the tweet. Additionally, duplicate tweets are removed to avoid repetition, which may occur due to keyword-based searching. The python script was written in order to collect data sorted on descending order of the retweet count and upvote count. This made it possible to limit the data response we get to certain range and get the social media sentiments with the most count and thereby eliminate irrelevant data with less count.

4.1.4.2 Stock Approximation

The stock data had missing values on certain days due to reasons like market holiday or simply data missing in database. The stock market data is approximated on these days to have a continuous period of data, without any missing values. The approximation is done using the formula given in 5.1, as proposed by Mittal's paper [3].

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

FIGURE 4.1: Stock Data approximation formula

where:

z is the missing stock value.

x is the previously available stock value data.

y is the next available stock value data.

4.2 DATA STORAGE

Collected datasets are of great volumes and needs to stored in order to reuse it for future purposes. Also, in order to generate a continuous stream of data and store the results obtained during processing via the Spark interface, we need to implement a suitable data storage system. Since we are collecting data from

APIs, there are limitations in their hits and also raw data involved a lot of repeated cleaning and tweaking process.

It is necessary to ensure that the processed data is kept consistent as the PySpark interface frequently updates the intermediate results. Additionally, we need to have a mechanism to revert to a previous state in the event of any failures. So a database solution is required for handling this.

We decided to use SQLite database for the same as it is very lightweight and very compatible with python scripts that our application is based on. We had created a single database in SQLite named 'fypdb-main'. Inside the database, we had separate tables for :

- Raw Stock Data
- Aggregated Stock Data
- 3 separate tables for Tweet Data of each Category
- 3 separate tables for Reddit Data of each Category
- 3 tables for Social Media Aggregated data

We decided to have separate tables for each category to ensure easy in writing common scripts for all future transformations and analysis. A common script was written to transform the collected CSV data to the SQLite tables in python.

4.3 KAFKA SETUP

4.3.1 Apache Kafka

Based on the work of Gurcan et. al.[4], for streaming of data, Apache Kafka pipeline was chosen for its robust, fault-resilient nature to stream data points at a quick rate for further processing.Apache Kafka [25] was downloaded and placed in the C drive of the system. The version of Kafka adopted is 2.13-2.4.0

4.3.1.1 Configuration

The zip file of Kafka is downloaded and extracted in the C drive using WinRaR.Before using Kafka, some environment changes need to be adopted. Inside the extracted Kafka folder, the configurations of Kafka and Zookeeper are present in *server.properties* and *zookeeper.properties* respectively.

In server.properties, the *log.dirs* value has to be changed from the default to any specific directory path where the log files need to be stored. The log files are super crucial for the continuous and healthy execution of Kafka server in the environment. These logs keep track of which clusters are created, what topics are created inside the cluster, and the partition information inside each topic.

The *advertised.listeners* value needs to be set to *PLAINTEXT://localhost:9092* to ensure that the kafka server runs locally in the machine on port number 9092.

The *zookeeper.connect* value needs to be set to *localhost:2181* to ensure that the zookeeper server runs on the local machine using the port 2181.

Next in the *zookeeper.properties* file the value of *dataDir* needs to be set to the directory where the zookeeper stores the logs. Ensure that the zookeeper port is set to 2181.

4.3.2 CMAK (GUI)

Cluster Manager for Kafka[26], initially knows as Kafka Manager is a GUI tool that offers efficient management of Kafka clusters, topics and helps to run and maintain as a service via a web-interface. CMAK can be installed into the environment by cloning its GitHub repository into the C drive of the system using the command *git clone https://github.com/yahoo/CMAK*

4.3.2.1 Configuration

After the repository is cloned, verify the installation of Java version as using CMAK on Windows requires the installation of GraalVM which is a virtual machine on which the CMAK is deployed. For GraalVM to be detected, the system needs to have OpenJDK 11.0+ installed. Then add the following to the *Path* variable in the system environment variables: *C:/Users/Arshath/graalvm-ce-java11-22.3.1/bin* .After installing and verifying the mentioned steps, use the command line and move into the directory and perform the following commands that can be used to setup the CMAK application:

- cd CMAK/

- sbt -Djdk.lang.Process.allowAmbiguousCommands=true
- clean dist

This creates a folder named *universal* inside the *target* folder inside CMAK. Inside this folder structure is present the CMAK folder named *cmak-3.0.0.7*. Unzip this folder using WinRaR. Now move into this directory and make the following changes to the *application.conf* file which manages all the configurations of the CMAK service:

- Remove the line *cmak.zkhosts="kafka-manager-zookeeper:2181"*
- Replace it with *cmak.zkhosts="localhost:2181"*

4.4 SPARK SETUP

For the efficiency of batch and stream processing, Apache Spark is used in this use case. The feature of parallelization that is offered by Spark can be leveraged to process data at a much better rate. Apache Spark[24] is downloaded and kept in the C drive of the system. The version used is spark-3.3.2

Utilization of the PySpark API is adapted from the work of Shashanka et al. and hence PySpark library needs to be installed. Pyspark application can be run using Jupyter notebook for which python3, pip, jupyter notebook, scala, py4j were required and installed on the command line.

4.4.0.1 Configuration

For Windows OS, a separate *winutils.exe* file is required and the version of the file depends on the Hadoop version that the installation of Spark accompanies. While downloading Spark zip file, choose Hadoop 2.7 version and its corresponding winutils.exe file can be downloaded from the repository *https://github.com/steveloughran/winutils*.

Create a directory structure as Hadoop inside which another bin folder and place this *winutils.exe* file inside this.

The downloaded zip file of Apache Spark is extracted using WinRaR and stored in the C drive. The following Environment variables need to be set to ensure the running of Apache Spark:

- *HADOOP_HOME = C:/Users/Username/Hadoop*
- *PYSPARK_HOME = Path of the python executable file in the system*
- *PYSPARK_PYTHON = python*
- *SPARK_HOME = C:/Users/Username/spark-3.3.2-bin-hadoop2*
- *PYTHONPATH=%SPARK_HOME%/python; %SPARK_HOME%/python/lib/py4j-0.10.9.5-src.zip; %PYTHONPATH%*
- *PATH = %HADOOP_HOME%/bin*
- *PATH = C:/Users/Username/spark-3.3.2-bin-hadoop2/bin*

PySpark is not found in the system path by default but the requirement is such that it must be used everywhere, even outside the location of its installation. Thus the findspark package must be installed so that pyspark can be used from any directory. The following code snippet can be used to make sure of the working of pyspark:

import findspark
findspark.init('C:/Users/Username/spark-3.3.2-bin-hadoop2')
import pyspark

4.5 RUNNING KAFKA, CMAK

Since the existing pipeline was suited to Linux based systems, the .sh files (shell script files) can be run. But now, as the environment is windows, the .bat files that are present inside bin-¿windows directory have to be run.

4.5.1 Apache Kafka

To start Kafka server, Apache Zookeeper must have been started before that in the background. To start Apache zookeeper, execute the following commands:

- cd C:/Users/Username/kafka_2.13-3.4.0
- ./bin/windows/zookeeper-server-start.bat ./config/zookeeper.properties

Now that zookeeper is up and running, start Apache Kafka server using the commands:

- cd C:/Users/Username/kafka_2.13-3.4.0
- set JMX_PORT=8004
- ./bin/windows/kafka-server-start.bat ./config/server.properties

4.5.2 Creation of Clusters & Topics

Clusters and Topics can be created on the command line using the following command:

- ./bin/kafka-topics.bat –create –zookeeper localhost:2181 –replication-factor 3 –partitions 50 –topic

To enable ease of use and better management, the CMAK tools provides a GUI that can be used for creation of clusters and topics. We can also see status reports about the created clusters and topics. There are options for creating, listing and providing summary view about the clusters and topics.

To start the CMAK web interface, use the following command:

- cd C:/Users/Username/CMAK/target/universal/cmak-3.0.0.7
- ./bin/cmak -Dconfig.file=conf/application.conf -Dhttp.port=8080

This opens the web interface on http://localhost:8080

Using the UI, cluster can be created and then a topic can be created. The zookeeper host value must be set to *localhost:2181*. Furthermore, the features of JMX polling and Poll Consumer Information must be set.

Using this method, separate topics can be created to stream data belonging to each sector separately.

4.6 DATA TRANSFORMATION

4.6.1 Producer

Producer code is written in python and it makes use of the kafka-python library. The general template for producer code is given in Kafka documentation. Using that template, the producer logic can be written for our case. Data is sent to corresponding kafka topic from the database. Data is sent in JSON format, and then produced into topics.

4.6.2 Consumer

The data that has been produced into the topics can be consumed by subscribing to the data topic straight through a spark session in PySpark. Data is collected as a stream from the topics and to prevent repeated processing, an offset mechanism is employed such that every time a batch is processed, the offset for that particular topic is updated as the sum of the size of the batches. The data thus collected is further processed.

4.6.3 Data Streaming

The Twitter and Reddit combined data are now obtained in JSON format and processed further.

Cardiff NLP research group's publicly available TimeLMs based sentiment analysis model is used to perform sentiment analysis. This model, provided a data point as input, returns an array of size 3 which contains in each index the probability of the data point belonging to each of the 3 sentiment labels i.e. negative, neutral and positive.

The maximum of this array is identified as the most suited sentiment for each data point and assigned as the MaxScore. Next, using the retweet/upvote count, the weighted average for each sentiment is computed as the product of the count and each sentiment score. Using the MaxScore value of each data point, the total count of data points for each sentiment are also calculated. The data are now aggregated based on the Category and Date as key using map reduce functionality.

The feature of Resilient Distributed Datasets (RDDs) offered by PySpark are utilized. The data points in JSON format are transformed into RDDs and then mapped based on the category of the data point for a particular date and reduced that produces an intermediate aggregated result. This is stored in database.

4.7 CORRELATION ANALYSIS

Now that the data are collected in the desired format, the next step is to apply analysis techniques to inspect the correlation.

The first step is to merge the stock market data and aggregated social media data based on the date attribute. This now enables to analyze each category's data separately as the data are split and grouped based on the category.

Several data exploratory functions are tried and the plot of close value of the stock for each category is printed and visualized. After this, the data are now normalized using min-max normalization so that all numerical columns are in the normalized range of [0-1].

Now the merged dataframe is used to perform correlation analysis between different attributes in the dataframe. Spearman correlation coefficient is primarily used as the correlation analysis method for reasons stated in Chapter 3. The Results and inferences made from them are presented in Chapter 5.

4.8 PREDICTION

We noticed a significant correlation between the market's close , high and low price and the weighted sentiment scores of Tweets and Reddits, which is consistent with Nayak et al.'s [4] research on stock market prediction. According to their study, Boosted Decision Trees outperform other algorithms such as Logistic Regression and Support Vector Machines in this scenario. Therefore, we aimed to compare and contrast two commonly used decision tree algorithms that utilize the concept of boosting to forecast the close value of the previously mentioned stock listings, incorporating the weighted sentiment scores of Tweets.

We considered algorithms like CatBoost, XGBoost and LightGBM for the prediction analysis but since XGBoost provided relatively poor feature importance, it was dropped.

Data collected from the SQLite Database was merged and normalised by using the MinMaxScaler() from sklearn package. To construct our model, we utilize the Python implementation of CatBoost and LightGBM. Initially, we used the default parameters present in the respective Regressor to visualize the results. We then found out that finetuning of the hyper parameters improved the results. To optimize our model's hyper parameters, we utilized the GridSearchCV() function from the scikit-learn library. The algorithm was employed to find the optimal hyper parameters for our specific use-case. The resulting hyper parameters are displayed in the tables.

Hyperparameter	**Value**
iterations	200
l2-leaf-reg	10
depth	4
learning-rate	0.05
loss-function	Mean Absolute Error (MAE)

TABLE 4.2: Optimal parameters for CatBoost Regressor

Hyperparameter	Value
num-leave	31
l2-leaf-reg	10
feature-fraction	1
learning-rate	0.1
max-depth	3
n-estimators	500
n-jobs	2
random-state	3
loss-function	Mean Absolute Error (MAE)

TABLE 4.3: Optimal parameters for LightGBM Regressor

As we mentioned above, we found significant correlation between the previous days sentiments from social media and the required close, high and low value of the stocks of the current day. So hence we decided to train the model with the weighted sentiment values of the previous day along with the opening value of the stocks in both of the days. 3 separate Regressor models were trained on the data with the mentioned features to predict the close value of the company ticker chosen. This was followed to predict the high and low value of the stock too. Both CatBoost and LightGBM regressor models were trained and tested with a split of 80:20 to compare and contrast the results.

Input Features	open , prev_open, prev_wted_neg, prev_wted_neu, prev_wted_pos
Output	close / high /low stock value

TABLE 4.4: Features used to train the model

4.8.1 Simulation of Real-time Prediction

We have also developed a single script of this application and implemented it such that it can perform the entirety of the project for a particular day and predict the close, high and low values of the stock market for the next day.The script extracts data from twitter, reddit and stock market; performs sentiment analysis and aggregation of the data; merges it with the stock data and uses the trained models to perform the prediction task.

CHAPTER 5

RESULTS

In this chapter, the results and inferences of the intended correlation analysis and prediction of stock values are shown. Correlation analysis results are displayed as heatmaps for each category wherein different attributes are correlated and their correlation values are analyzed. For prediction task, comparative performance analysis is carried out between the two chosen models CatBoost and LightGBM and their performance metrics are tabulated. The performance of the live script is also analyzed and inferences are made.

5.1 CORRELATION ANALYSIS

The main focus of our research is to conduct a correlation analysis between social media trends and stock market data. To accomplish this, we calculate the Spearman's Correlation Coefficient for every pair of columns in the dataset, separately for each industrial category. We represent the observed correlations visually by plotting heatmaps.The below heatmaps shows the correlation between the input features that we have chosen from experimenting and analysing various factors that affect the stock value.

5.1.1 Finance

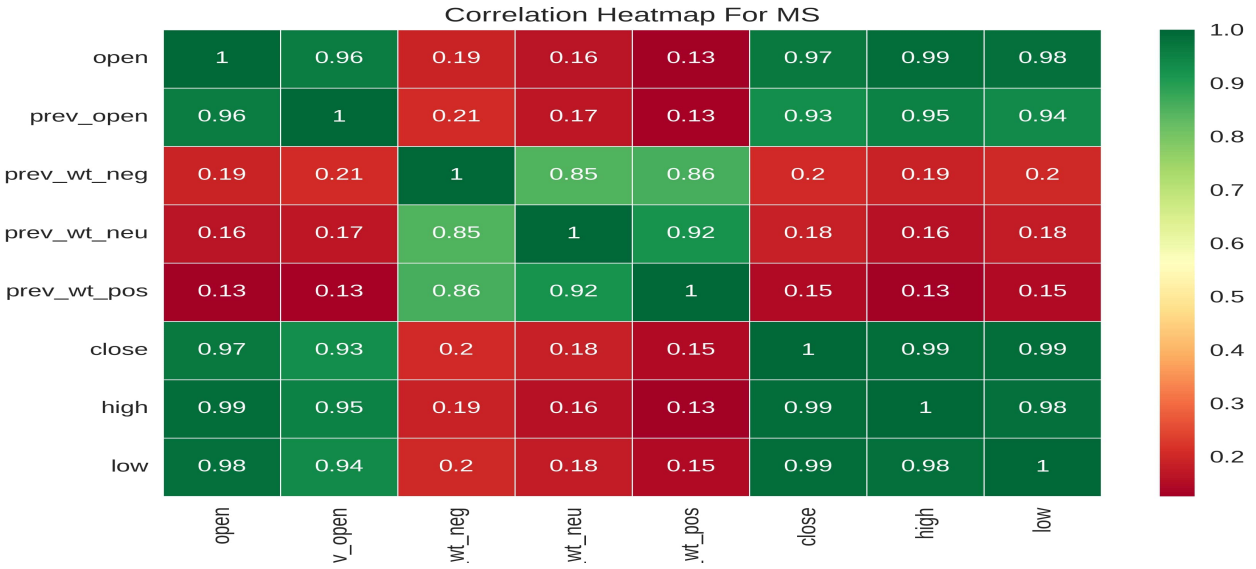

FIGURE 5.1: Correlation Heatmap - MS

5.1.2 EVs

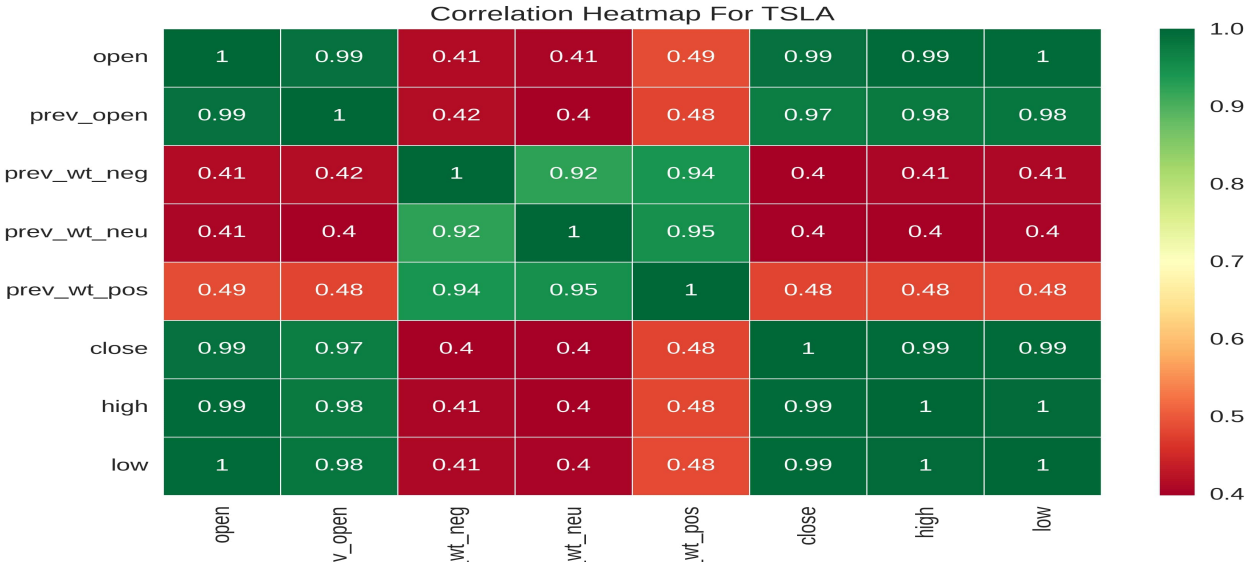

FIGURE 5.2: Correlation Heatmap - TSLA

5.1.3 Tech

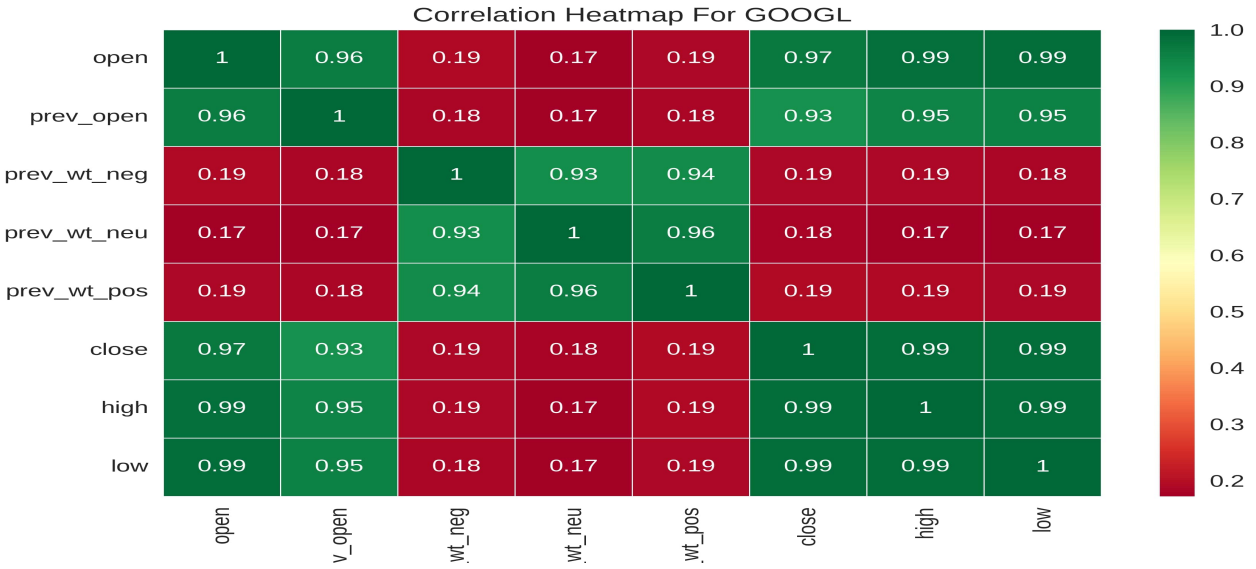

FIGURE 5.3: Correlation Heatmap - GOOGL

It was seen that the previous days' sentiments had good impact on the closing, high and low value of the current day. The current days' sentiments didn't have much correlation and it clearly shows market doesn't fluctuate on very immediate sentiments except on very rare occasions. Additionally, the opening value is required as it is the base factor on which increase and decrease of value can be predicted. Hence we decided to use the weighted sentiments of the previous day and the opening value as the input features.

The correlation varied for each category as seen. In the case of Tesla, we found the best correlation between the sentiments and the predicting value of about 0.45 to 0.5. In the case of Google and Morgan Stanley, we found decent correlation of about 0.2. This trend was similar for all 3 values that we intend to predict - Close, High and Low.

5.2 PREDICTION

The actual value against the predicted value for the test set of data is plotted in the graphs shown below. Important metrics for Regression like RMSE, R2 scores were noted and shown in tables below for each algorithm for comparison.

5.2.1 CatBoost

5.2.1.1 Finance

FIGURE 5.4: CatBoost Prediction (high) - MS

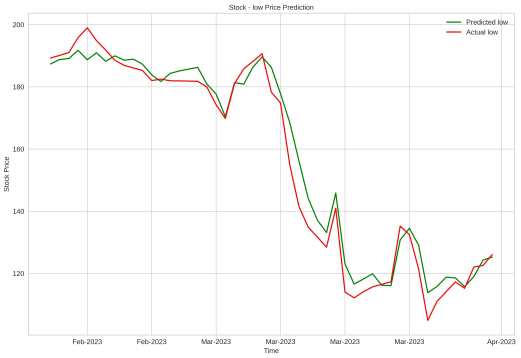

FIGURE 5.5: CatBoost Prediction (low) - MS

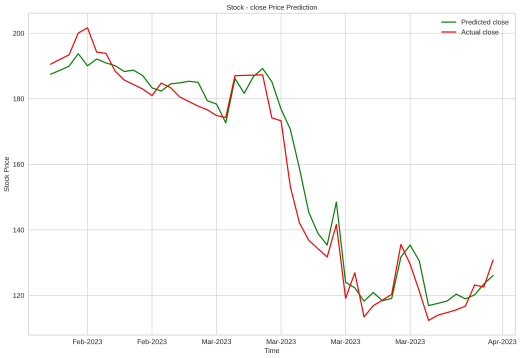

FIGURE 5.6: CatBoost Prediction (close) - MS

RMSE	0.037185
MSE	0.001383
R2 Score	0.976386
Explained Variance Score	0.976729
Max Error	0.102192

TABLE 5.1: Metrics - Catboost - MS - High

RMSE	0.056626
MSE	0.003207
R2 Score	0.945656
Explained Variance Score	0.955233
Max Error	0.164122

TABLE 5.2: Metrics - Catboost - MS - Low

RMSE	0.062865
MSE	0.003952
R2 Score	0.926154
Explained Variance Score	0.936513
Max Error	0.189636

TABLE 5.3: Metrics - Catboost - MS - Close

- The CatBoost algorithm performed the best for this category overall with maximum R2 score of 0.97 in the case of predicting the high value and also the better result than LighGBM
- The Root Mean Squared Error stayed on the lower side of range 0.03 to 0.06.

5.2.1.2 EVs

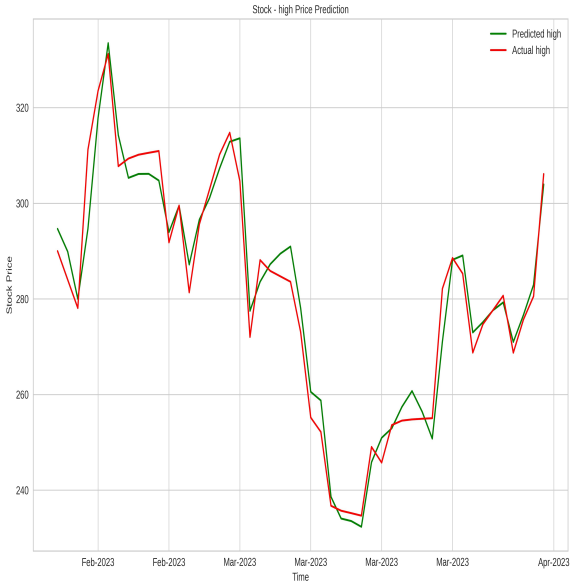

FIGURE 5.7: CatBoost Prediction (high) - TSLA

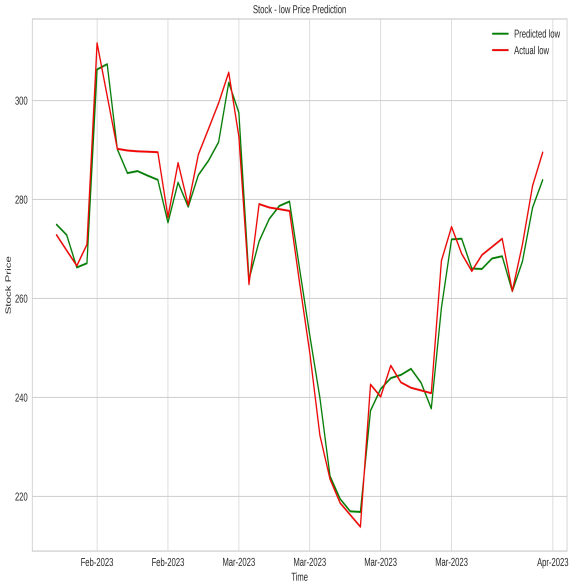

FIGURE 5.8: CatBoost Prediction (low) - TSLA

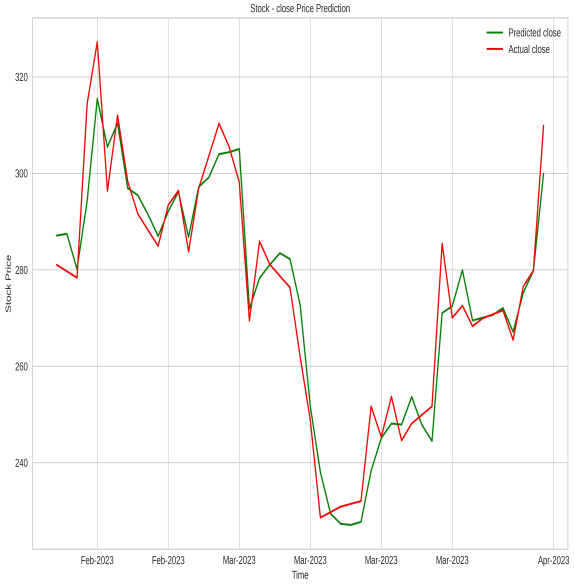

FIGURE 5.9: CatBoost Prediction (close) - TSLA

RMSE	0.024196
MSE	0.000585
R2 Score	0.776602
Explained Variance Score	0.779931
Max Error	0.078823

TABLE 5.4: Metrics - Catboost - TSLA - High

RMSE	0.020799
MSE	0.000433
R2 Score	0.829707
Explained Variance Score	0.843244
Max Error	0.050538

TABLE 5.5: Metrics - Catboost - TSLA - Low

RMSE	0.032153
MSE	0.001034
R2 Score	0.608224
Explained Variance Score	0.609552
Max Error	0.096511

TABLE 5.6: Metrics - Catboost - TSLA - Close

- The CatBoost algorithm performed decently for this category overall with maximum R2 score of 0.82 in the case of predicting the low value.
- The R2 score of 0.61 was achieved in the case of predicting close value which happened to be the least the model has performed overall in all cases.
- The RMSE score and Max Error stayed on the lower side with the max value of Max Error as 0.096.

5.2.1.3 Tech

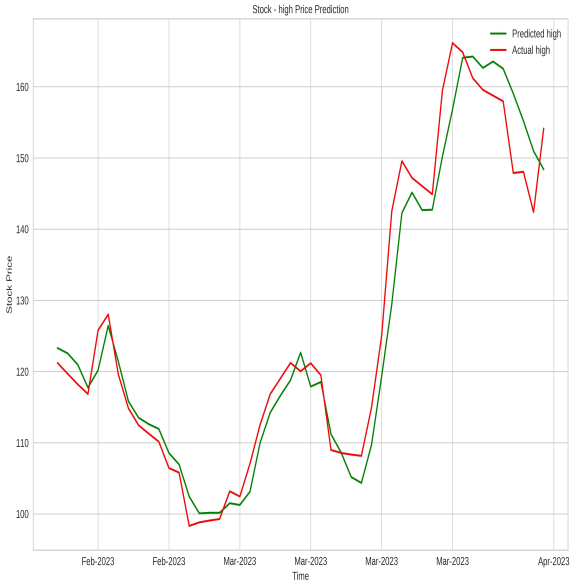

FIGURE 5.10: CatBoost Prediction (high) - GOOGL

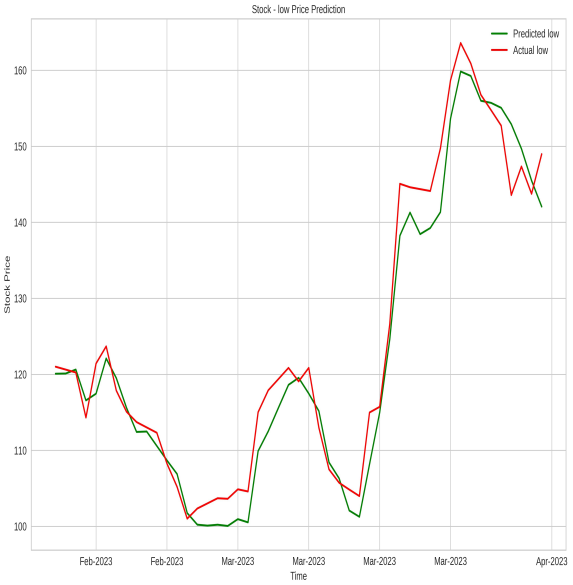

FIGURE 5.11: CatBoost Prediction (low) - GOOGL

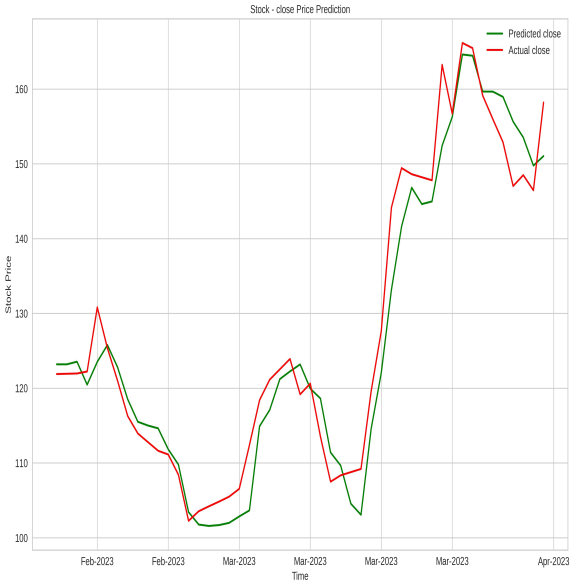

FIGURE 5.12: CatBoost Prediction (close) - GOOGL

RMSE	0.044982
MSE	0.002023
R2 Score	0.895595
Explained Variance Score	0.899675
Max Error	0.129150

TABLE 5.7: Metrics - Catboost - GOOGL - High

RMSE	0.037457
MSE	0.001403
R2 Score	0.915029
Explained Variance Score	0.934954
Max Error	0.093531

TABLE 5.8: Metrics - Catboost - GOOGL - Low

RMSE	0.045158
MSE	0.002039
R2 Score	0.880161
Explained Variance Score	0.887322
Max Error	0.109612

TABLE 5.9: Metrics - Catboost - GOOGL - Close

- The CatBoost algorithm predicted very closely to the actually values as seen in the graph.
- It achieved great R2 scores in the range 0.88 to 0.94 for the 3 values predicted.
- While the RMSE score remained on the same lower side, the Max Error increased by a slight margin in this case. This shows very good consistency in predicting to the actual value but a slight deviation on very few occasions.

5.2.2 LightGBM

5.2.2.1 Finance

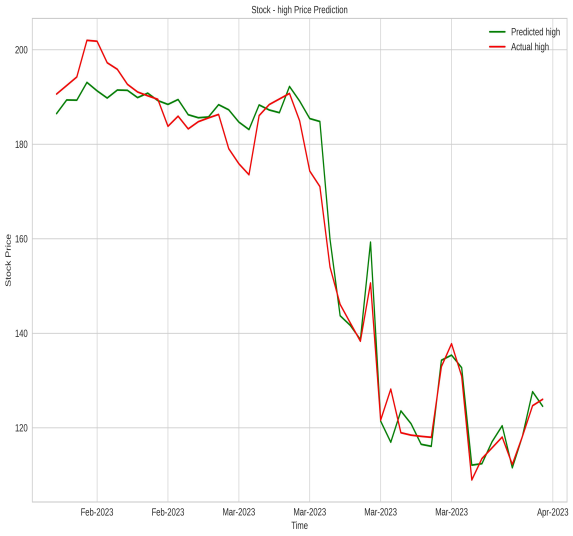

FIGURE 5.13: LightGBM Prediction (high) - MS

FIGURE 5.14: LightGBM Prediction (low) - MS

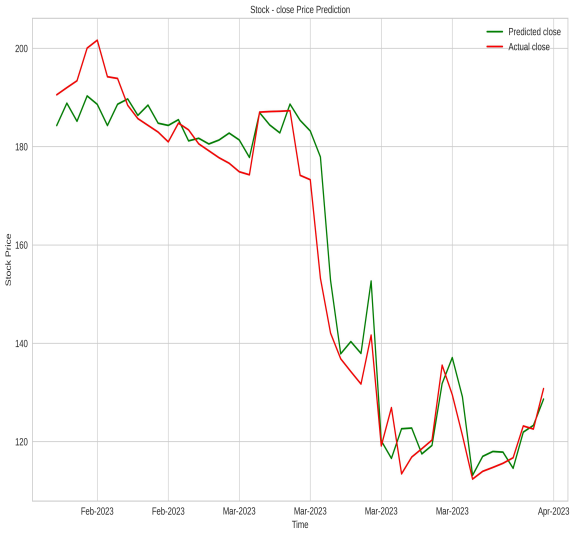

FIGURE 5.15: LightGBM Prediction (close) - MS

RMSE	0.053201
MSE	0.002830
R2 Score	0.951663
Explained Variance Score	0.952733
Max Error	0.143437

TABLE 5.10: Metrics - LightGBM - MS - High

RMSE	0.067375
MSE	0.004539
R2 Score	0.923066
Explained Variance Score	0.925475
Max Error	0.230706

TABLE 5.11: Metrics - LightGBM - MS - Low

RMSE	0.072564
MSE	0.005265
R2 Score	0.901612
Explained Variance Score	0.906985
Max Error	0.267623

TABLE 5.12: Metrics - LightGBM - MS - Close

- The LightGBM algorithm was slightly behind in terms of the R2 score and RMSE values in this category.
- However, the R2 score was really great in terms of its individual performance ranging from 0.90 to 0.95.
- The Max Error however was found to be a little big comparatively, meaning the model has performed a little less on few days. The max error among all was found in the case of predicting MS close value as 0.267.

5.2.2.2 EVs

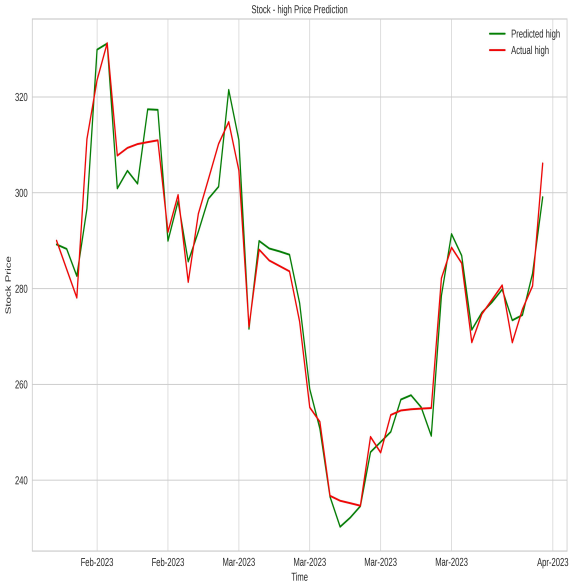

FIGURE 5.16: LightGBM Prediction (high) - TSLA

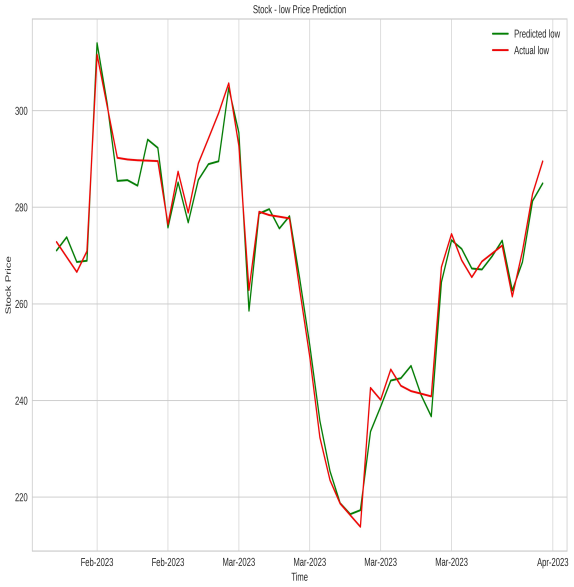

FIGURE 5.17: LightGBM Prediction (low) - TSLA

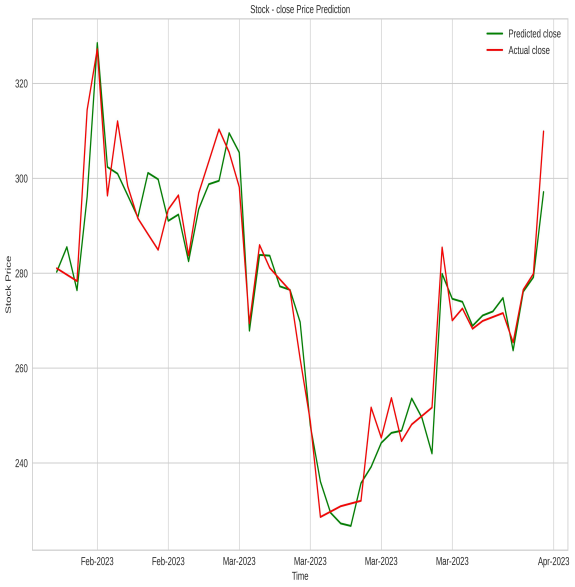

FIGURE 5.18: LightGBM Prediction (close) - TSLA

RMSE	0.022468
MSE	0.000505
R2 Score	0.807373
Explained Variance Score	0.807479
Max Error	0.069324

TABLE 5.13: Metrics - LightGBM - TSLA - High

RMSE	0.017583
MSE	0.000309
R2 Score	0.878302
Explained Variance Score	0.883090
Max Error	0.051044

TABLE 5.14: Metrics - LightGBM - TSLA - Low

RMSE	0.031836
MSE	0.001014
R2 Score	0.615913
Explained Variance Score	0.619992
Max Error	0.087492

TABLE 5.15: Metrics - LightGBM - TSLA - Close

- The LightGBM algorithm performed better than CatBoost in the EVs category by giving more feature importance to the weighted sentiments in this case.
- The R2 score was really good in predicting the high and low value (0.81,0.88) while dipped to 0.62 in predicting the close value.
- The RMSE value and Max Error stayed very low around 0.02 and 0.07 respectively, indicating a very consistent performance.

5.2.2.3 Tech

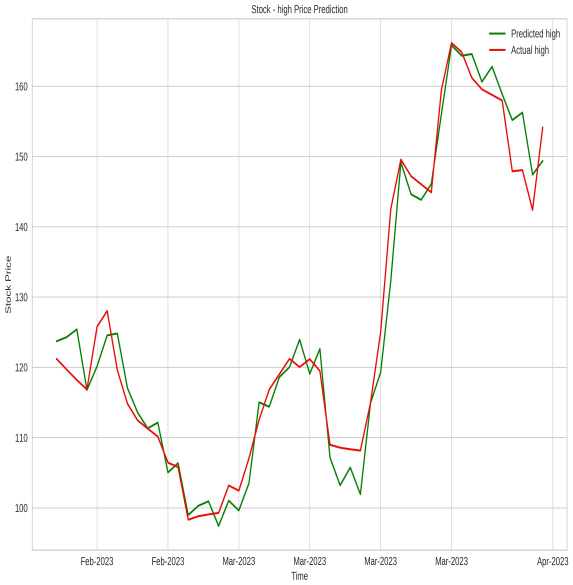

FIGURE 5.19: LightGBM Prediction (high) - GOOGL

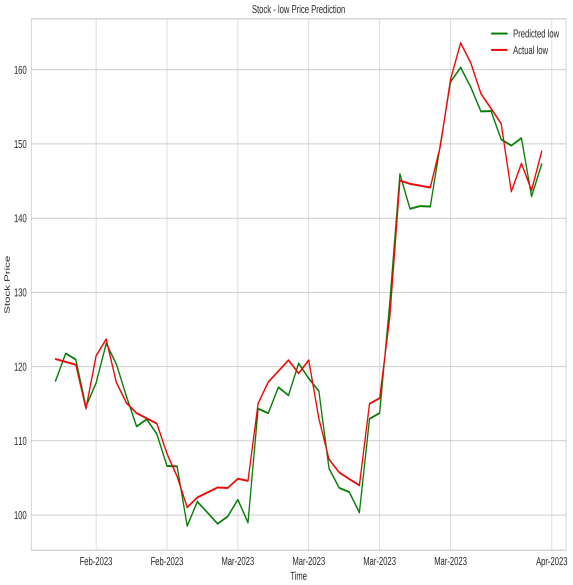

FIGURE 5.20: LightGBM Prediction (low) - GOOGL

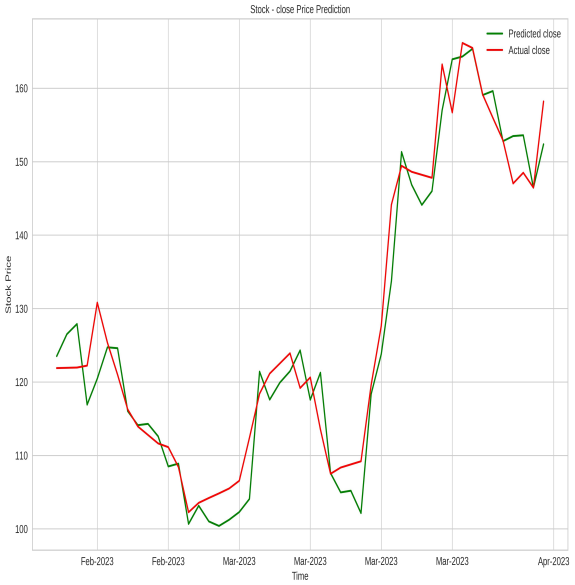

FIGURE 5.21: LightGBM Prediction (close) - GOOGL

RMSE	0.037921
MSE	0.001438
R2 Score	0.927733
Explained Variance Score	0.927773
Max Error	0.101128

TABLE 5.16: Metrics - LightGBM - GOOGL - High

RMSE	0.028247
MSE	0.000798
R2 Score	0.951678
Explained Variance Score	0.962385
Max Error	0.062849

TABLE 5.17: Metrics - LightGBM - GOOGL - Low

RMSE	0.045034
MSE	0.002028
R2 Score	0.880818
Explained Variance Score	0.887663
Max Error	0.106845

TABLE 5.18: Metrics - LightGBM - GOOGL - Close

- The LightGBM algorithm performed the best for this category among the three.
- The R2 score was really high in all 3 values predicted with the highest value of 0.95 in the case of prediction of low value..
- While the RMSE score remained on the same lower side, the Max Error increased by a slight margin in this case. This shows very good consistency in predicting to the actual value but a slight deviation on very few occasions.

5.3 COMPARISION OF CATBOOST & LIGHTGBM

CatBoost algorithm was mainly introduced to handle categorical data very efficiently. It can take in categorical, text as well as numerical directly as input and does the necessary steps to convert it for its processing. It is really good when it comes to learning from very limited number of data and performs well for the same and hence was really helpful in our case. It goes on to build the tree in a symmetric manner with same decision parameter at a level.

LightGBM on the other hand also handles categorically data directly in a very fast manner. It uses a very novel technique of GOSS - Gradient-based One Side Sampling Technique to make sure the boosting models are trained efficiently. Instead of building a symmetric tree, it deploys leaf wise growth, making it really faster.

On getting to know the feature importance considered by each model in our case, we found that CatBoost gave very big importance to highly correlated columns like open and prev_open, while medium importance were given to the weighted sentiments having relatively lower correlation with the output variable. However, LightGBM gave even distribution of importance to all features varying slightly based on the correlation factor.

Due to the above difference, CatBoost performed slightly better in terms of R2 score followed by LightGBM with a small margin. The CatBoost had above 0.95 R2 score in majority of cases except for TSLA for which it dipped to range of

0.65 to 0.8, wherease LightGBM had consistent values of above 0.9 in terms of R2 score.

The Root Mean Squared Error was also found to be similar for both the models ranging from 0.03 to 0.05 in case of CatBoost while from 0.02 to 0.07 in the case of LightGBM. The max error however remained very low and similar in all the cases of CatBoost with the max value found to be 0.129150 in the case of predicting GOOGL high value. The max error for LightGBM remained comparatively high with the max value of 0.267623 in the case of MS close value.

In terms of performance, LightGBM completed little faster than CatBoost but would surely outperform the later if the dataset is very large.

5.4 REALTIME SIMULATION OF PREDICTION

The main script that was written to simulate real time prediction for a day was run for the date 20th of April, 2023. The result of it is shown in the below figures.

```
Actual Close value of MS :     91.120003
Actual High value of MS  :     91.339996
Actual Low value  of MS  :     89.169998
Actual Close value of TSLA :     165.080002
Actual High value of TSLA  :     166.0  %%%%%
Actual Low value  of TSLA  :     161.320007
Actual Close value of GOOGL :     105.410004
Actual High value of GOOGL  :     106.0
Actual Low value  of GOOGL  :     104.779999
```

FIGURE 5.22: Actual Stock values on 20-04-2023

```
Predicted close value of MS :    90.08695758120955
Predicted high value of MS :    91.15777826183822
Predicted low value of MS :    89.44190214877037
Predicted close value of TSLA :    171.90466357428778
Predicted high value of TSLA :    181.7697235856433
Predicted low value of TSLA :    171.05838912614882
Predicted close value of GOOGL :    105.72699453361459
Predicted high value of GOOGL :    106.0656316730367
Predicted low value of GOOGL :    104.84814636981767
```

FIGURE 5.23: Predicted Stock values on 20-04-2023

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

To conclude, we have discussed, in this report, the detailed design and related frameworks and implementation for a project to extract data from several social media platforms and stock market data, and perform correlation analysis to identify and unearth any existing correlation between them. We also employed the usage of big data architecture that was already established but adopted for this use case with the help of Apache Kafka and Spark. These lightweight and robust frameworks can be used to construct scalable analytical solutions by exploiting the effective parallelization in offer. An effective end to end pipeline was constructed by using only python, right from data gathering till the inference of the experiment.

In this research, three categories of the society namely Finance, Electric Vehicles and Tech were chosen and representative stock market tickers - Morgan Stanley, Tesla, and Google were chosen to infer the correlation between social media sentiments with respect to the value of the stock market entity.

As the concluded results state, the correlation can be visualized and inferred, even though it is comparatively lesser when looking at the existing research. This can be attributed to the fact that multiple data sources were considered which were noisy and irrelevant data points had seeped in, given the nature of the sources. It is acceptable and varies with each category that is chosen. The timeframe of observation was chosen as August 2022 to March 2023 as the availability of Reddit data was restricted to this period because of financial restrictions to use free APIs. It is to be noted that the correlation exists only for sectors and representative companies that have a noticeable presence in the online forums.

The research also made use of two regressor tree based machine learning algorithms CatBoost and LightGBM, that have been used for efficient analysis and prediction of stock market values based on the sentiments expressed online.

Taking into account the existing research in this domain, this project introduced novel elements like integration of data from additional data sources and usage of a ground breaking algorithm called LightGBM for prediction task. A simulation of real time prediction of next day stock values based on sentiment expressed on the immediately preceding day was also experimented.

As a part of our future work, incorporation of data from other social media platforms can be carried out provided there are no financial constraints and credible data sources are available thus providing an even wider perspective of the actual impact between trends and stocks. Next consideration could be to try and include language sources other than English that can provide valuable insights. For other languages, NLP technique exploration is a major hurdle in the task as such efficient sentiment analysis models are yet to be implemented for them. The same pipeline could also be used to analyse the correlation between two other related sectors thus paving way for correlation analysis from all dimensions.

REFERENCES

1. 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.

2. 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.

3. 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.

4. 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

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. 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.

8. 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.

9. 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.

10. G. V. Attigeri, Manohara Pai M M, R. M. Pai and A. Nayak, ”Stock market prediction: A big data approach,” TENCON 2015 - 2015 IEEE Region 10 Conference, Macao, China, 2015, pp. 1-5, doi: 10.1109/TENCON.2015.7373006.

11. Yulong Pei, Amarachi Mbakwe, Akshat Gupta, Salwa Alamir, Hanxuan Lin, Xiaomo Liu, and Sameena Shah. 2022. TweetFinSent: A Dataset of Stock Sentiments on Twitter. In Proceedings of the Fourth Workshop on Financial Technology and Natural Language Processing (FinNLP), pages 37–47, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

12. Yadav, K., Yadav, M., and Saini, S.: Stock values predictions using deep learning based hybrid models. CAAI Transactions on Intelligence Technology 7(1), 107–116,(2022).

13. Rouf, N.; Malik, M.B.; Arif, T.; Sharma, S.; Singh, S.; Aich, S.; Kim, H.-C. Stock Market Prediction Using Machine Learning Techniques: A Decade Survey on Methodologies, Recent Developments, and Future Directions. Electronics 2021, 10, 2717.

14. Nannai John, S.,Subramanian, V.,Nagarajan, V., Venkatesh, S.: Social Media Based Stock Market Analysis using Big-Data Infrastructure, 2022

15. Twitter-Roberta. https://huggingface.co/cardiffnlp/ twitter-roberta-base-sentiment-latest.

16. Daniel Loureiro, Francesco Barbieri, Leonardo Neves, Luis Espinosa Anke, and Jose 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.

17. Fatih Gurcan 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

18. 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.

19. 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, Montreal, ´ Canada, pages 6639– 6649, 2018

20. K. Guolin, M. Qi, F. Thomas, W. Taifeng, C. Wei, M. Weidong, Y. Qiwei, L. Tie-Yan, ”LightGBM: A Highly Efficient Gradient Boosting Decision Tree,” Advances in Neural Information Processing Systems vol. 30, pp. 3149-3157, 2017.

21. 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.

22. 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.

23. 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.

23. Twitter-API. https://developer.twitter.com/en/products/twitter-api.

23. Polygon-API. https://api.polygon.io/v1/open-close/.

24. Apache Spark. https://www.apache.org/dyn/closer.lua/spark/spark-3.2.1.

25. Apache Kafka. https://kafka.apache.org/downloads.

26. Yahoo-CMAK. https://github.com/yahoo/CMAK.

27. Yahoo Finance API. https://query1.finance.yahoo.com/v7/

28. Reddit API. https://api.pushshift.io/reddit/search/submission/