**CCT College Dublin**

**Assessment Cover Page**

|  |  |
| --- | --- |
| **Module Title(s):** | Advanced Data Analytics  Big Data Storage and Processing |
| **Assessment Title:** | MSC\_DA\_BD\_ADAv5 |
| **Lecturer(s) Name:** | Muhammad Iqbal  David McQuaid |
| **Student Full Name:** | Ricardo Moran |
| **Student Number:** | 2022253 |
| **Assessment Due Date:** | 17 November 2023 |
| **Date of Submission:** | 17 November 2023 |

**Declaration**

|  |
| --- |
| By submitting this assessment, I confirm that I have read the CCT policy on Academic Misconduct and understand the implications of submitting work that is not my own or does not appropriately reference material taken from a third party or other source. I declare it to be my own work and that all material from third parties has been appropriately referenced. I further confirm that this work has not previously been submitted for assessment by myself or someone else in CCT College Dublin or any other higher education institution. |

# Abstract

*This report provides an overview of a sentiment analysis project that utilized a dataset containing 1.6 million tweets from the social media platform "X" (formerly known as Twitter). The main objective of the project was to predict sentiment trends, over time intervals one week, one month and three months. To achieve this goal various data analytics techniques were employed, including cleaning and storing the dataset in MongoDB and MySQL databases. Additionally, sentiment analysis was conducted using Spark, NLTK and Textblob. Comparative performance evaluations of the databases were carried out using benchmarks. The study yielded insights; however certain limitations such as the time span and imbalance of the dataset had an impact, on the accuracy of the models used. Furthermore, analysing social media sentiment proved to be complex and challenging. Based on these findings future research should focus on obtaining a dataset and implementing real time event monitoring.*

# Table of Contents

[Abstract 2](#_Toc151129568)

[Table of Contents 3](#_Toc151129569)

[Table of Figures 4](#_Toc151129570)

[Introduction 5](#_Toc151129571)

[Methodology 6](#_Toc151129572)

[Data Acquisition 6](#_Toc151129573)

[Data Exploration and Cleaning 6](#_Toc151129574)

[Data Storage 6](#_Toc151129575)

[Benchmarking Tests 10](#_Toc151129576)

[Data Processing with Spark 14](#_Toc151129577)

[Spark Set-up 14](#_Toc151129578)

[Database connection setup 14](#_Toc151129579)

[Data Analysis in Spark 15](#_Toc151129580)

[Sentiment Analysis and Forecasting 21](#_Toc151129581)

[Time-Series Analysis 22](#_Toc151129582)

[Model Selection and Forecasting 26](#_Toc151129583)

[Dashboard Implementation 27](#_Toc151129584)

[Weekly forecast results 27](#_Toc151129585)

[Monthly forecast results 28](#_Toc151129586)

[Quarterly forecast result 28](#_Toc151129587)

[Results and Discussion 28](#_Toc151129588)

[Results Overview 28](#_Toc151129589)

[Factors Affecting Model Accuracy 29](#_Toc151129590)

[Model Selection Rationale 29](#_Toc151129591)

[Analysis of Sentiment Over Time 29](#_Toc151129592)

[Conclusion and Future Work 29](#_Toc151129593)

[References 30](#_Toc151129594)

# Table of Figures

[Figure 1 – Duplicate values 5](#_Toc151129625)

[Figure 2 – ‘Flag’ values 5](#_Toc151129626)

[Figure 3 – MySQL credential set-up 6](#_Toc151129627)

[Figure 4 – Importing data in chunks to MySQL 7](#_Toc151129628)

[Figure 5 – Data visualization with MySQL shell interface 7](#_Toc151129629)

[Figure 6 – MongoDB command to create a database and import data. 8](#_Toc151129630)

[Figure 7 - MongoDB data import. 8](#_Toc151129631)

[Figure 8 - MongoDB data visualization. 9](#_Toc151129632)

[Figure 9 – Databases Throughput comparison 10](#_Toc151129633)

[Figure 10 – Databases Latency comparison 11](#_Toc151129634)

[Figure 11- Databases Runtime Comparison 12](#_Toc151129635)

[Figure 12 – Databases Garbage Collection comparison 13](#_Toc151129636)

[Figure 13 – PySpark libraries 14](#_Toc151129637)

[Figure 14 – Spark Session Parameters 14](#_Toc151129638)

[Figure 15 – MySQL Connection Setup 14](#_Toc151129639)

[Figure 16 – Visualization of Date column structure 15](#_Toc151129640)

[Figure 17 – Spark time parser policy 15](#_Toc151129641)

[Figure 18 – Data preview with standardized date format 15](#_Toc151129642)

[Figure 19 – dataset time range 15](#_Toc151129643)

[Figure 20 – Tweets per user table 16](#_Toc151129644)

[Figure 21 – Tweets per user comparison, distribution of tweets per user and Boxplot of tweets per user. 17](#_Toc151129645)

[Figure 22 – Top 10 Hashtags from dataset 18](#_Toc151129646)

[Figure 23 – Count of ‘FF’ tweets from the dataset 19](#_Toc151129647)

[Figure 24 – Top 10 Mentions from dataset 19](#_Toc151129648)

[Figure 25 – Distribution of Tweets over Time 20](#_Toc151129649)

[Figure 26 – Table of Spam Tweets from dataset 21](#_Toc151129650)

[Figure 27 – Table of sentiment calculation 22](#_Toc151129651)

[Figure 28 – Sentiment distribution by category 22](#_Toc151129652)

[Figure 29 – Sentiment distribution Over Time 23](#_Toc151129653)

[Figure 30 – Sentiment Score Aggregated Over Different Intervals 23](#_Toc151129654)

[Figure 31 – Weekly dataset decomposition 24](#_Toc151129655)

[Figure 32 – Initial Stationary testing 25](#_Toc151129656)

[Figure 33 – Stationary test after 3 differencing 25](#_Toc151129657)

[Figure 34 – Mean Sentiment Score Over Time After Differencing 25](#_Toc151129658)

[Figure 35 – ARIMA model forecast plot 26](#_Toc151129659)

[Figure 36 – ForecasterAutoreg forecast plot 26](#_Toc151129660)

[Figure 37 – Prophet forecast plot 27](#_Toc151129661)

[Figure 38 – Weekly Interactive Dashboard 27](#_Toc151129662)

[Figure 39 – Monthly Interactive Dashboard 28](#_Toc151129663)

[Figure 40 – Quarterly Interactive Dashboard 28](#_Toc151129664)

# Introduction

In today’s world the social media platform "X" formerly known as Twitter has gained popularity as a valuable source, for analysing public opinion. This project focuses on examining a dataset called "ProjectTweets.csv," which consists of 1.6 million tweets collected years ago using the Twitter API. The goal is to understand and predict trends in sentiment expressed in these tweets over different time intervals. One week, one month and three months.

This dataset contains tweet IDs, dates, query flags, user information and tweet content providing a foundation for conducting sentiment analysis. My intention is to analyse sentiment and forecast trends using data processing tools such as Hadoop MapReduce and Spark.

An important aspect of this study involves comparing databases using performance metrics to ensure an objective analysis. Additionally, the project will explore two time series forecasting methods to address the limited duration of the dataset.

Extracting sentiment from the tweets will be a part of this research. Will be carried out with careful consideration of various methodologies. The results will then be presented in a dashboard designed in accordance, with Tufts principles to represent the sentiment trends over time and allow users to interact with them.

This project is an investigation, into sentiment analysis and predictive analysis. Using technologies and analytical methods to gain insights, from a vast dataset collected from X.

# Methodology

For this project I took a systematic approach to analyse and predict sentiment using data analytics. The methodology included stages; gathering data preparing and refining the data storing it securely comparing it with benchmarks, processing and analysing the data and ultimately conducting sentiment analysis and making predictions. Each stage was essential, in maintaining the accuracy and dependability of the outcomes.

## Data Acquisition

The project commenced with the acquisition of a relevant dataset, which was critical for the subsequent analysis. The dataset was collected using the twitter API and is available for download on Moodle website as “ProjectTweets.csv”.

### Data Exploration and Cleaning

Once the dataset was obtained, it underwent a thorough preparation and cleaning process. This stage involved loading the dataset into a Python environment using Jupyter Notebook for preliminary exploration. Key tasks included the removal of empty rows, duplicates, and non-essential columns. This step was crucial for maintaining the quality of the dataset.

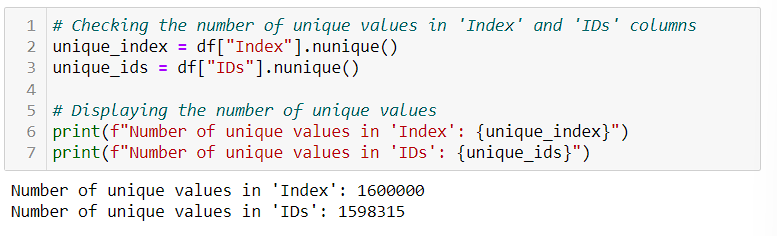
Identifying and removing duplicate records.  


Figure 1 – Duplicate values

Identifying unique values inside ‘Flag’ column.

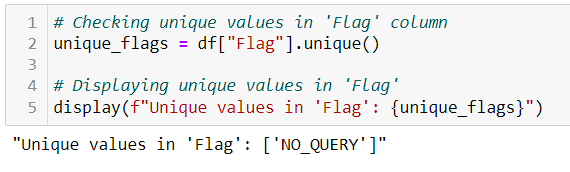


Figure 2 – ‘Flag’ values

The refined dataset was saved as 'ProjectTweets\_cleaned.csv' following the data cleaning process.

## Data Storage

The cleaned dataset was then stored in two distinct databases, MongoDB and MySQL, to leverage their unique storage capabilities and to facilitate a comparative analysis of their performance.

### MySQL

The choice of MySQL for this project was influenced by its strong suit in handling structured data. Given that the dataset of tweets has been cleaned and organized into structured columns (user, tweet, Date), MySQL's table-based system aligns perfectly with this format. My personal experience and proficiency with MySQL also played a pivotal role in this selection. This familiarity not only makes the database management process more efficient but also reduces the learning curve and potential setup complexities. The reliability and mature support ecosystem of MySQL further cement its suitability for this project, ensuring a stable and well-supported database environment.

**Setting up the database in MySQL can be approached in two ways:** using the MySQL shell or directly from Python. I opted for the latter, leveraging Python to gain greater control over data processing and storage. This approach also allowed for processing data in chunks, optimizing computational resource usage.

1. **Database Creation:** Initiated by executing a command in the MySQL shell, this step establishes the foundation for the database.



1. **Dataset Loading and Credential Setup:** As illustrated in Figure 3, this step involves preparing the dataset and configuring the database credentials.

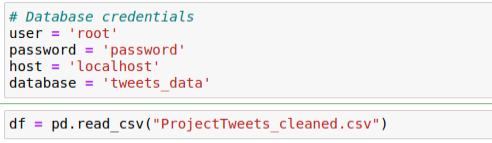


Figure 3 – MySQL credential set-up

1. **Data Import Using SQLAlchemy:** This phase, detailed in Figure 4, involves importing the data into MySQL. I incorporated additional complexity by processing the data in chunks, enhancing efficiency and resource management.

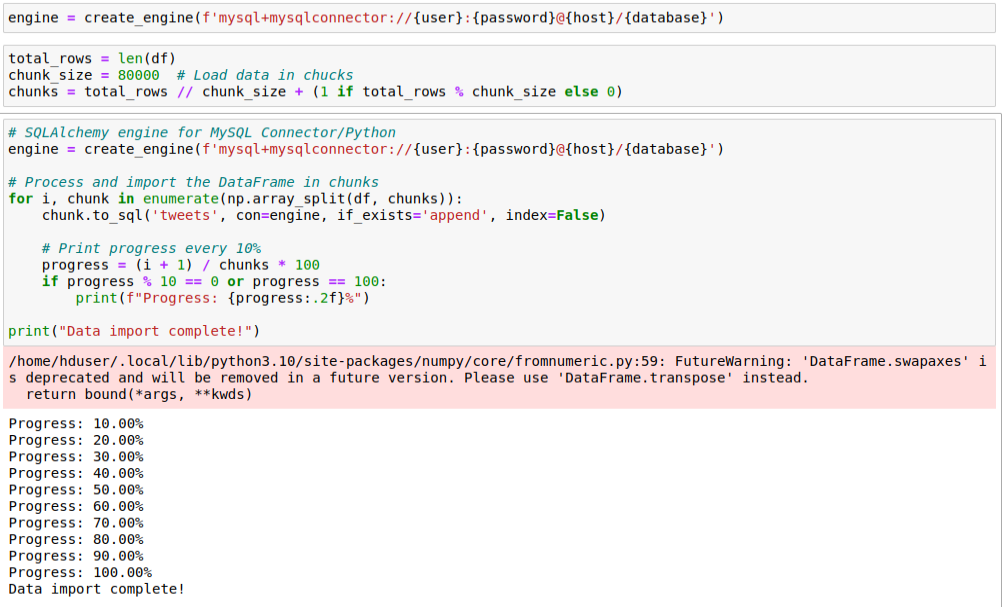


Figure 4 – Importing data in chunks to MySQL

1. **Visualization of the data using MySQL shell**

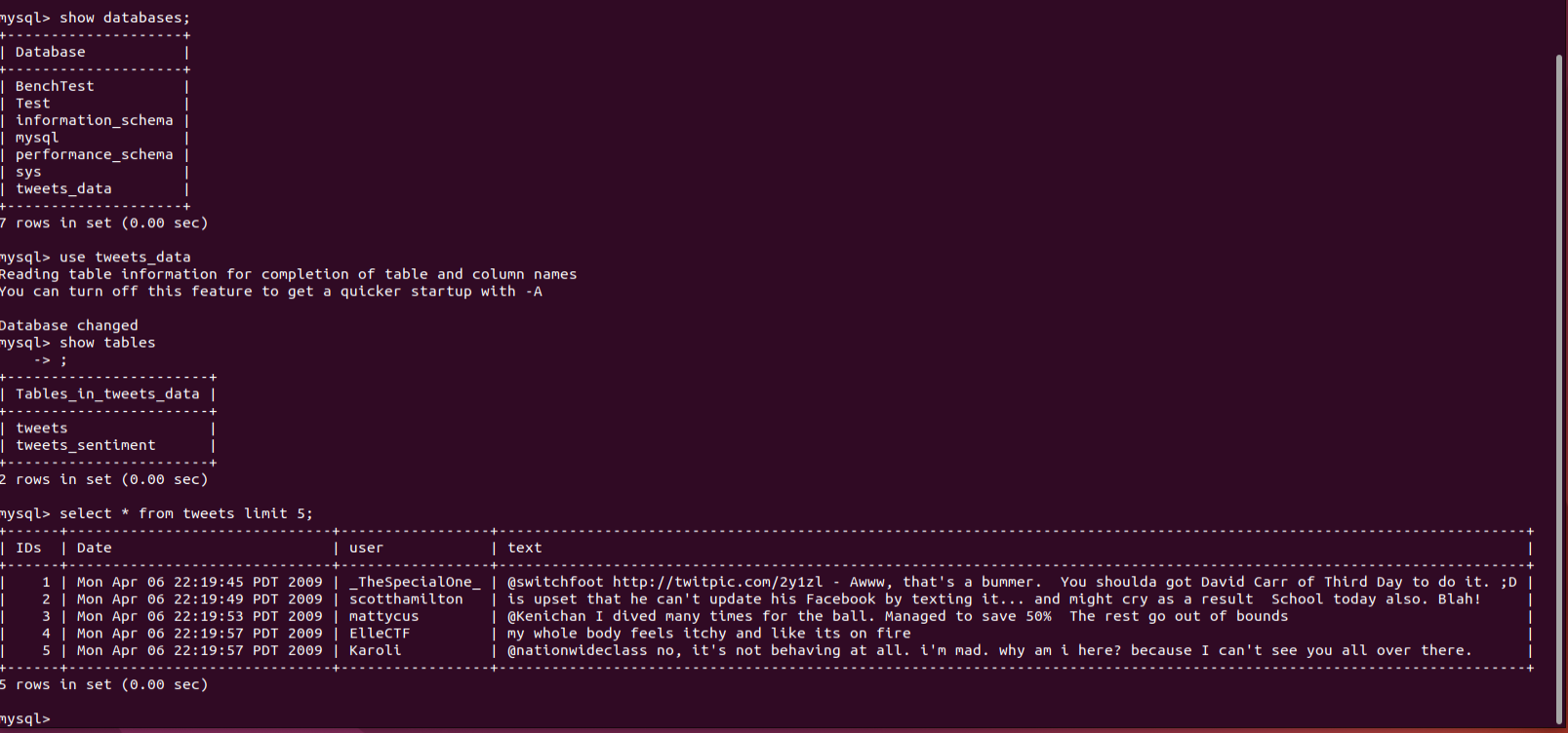


Figure 5 – Data visualization with MySQL shell interface

### MongoDB

MongoDB was chosen for its adeptness in managing large datasets and its schema-less nature, which offers the flexibility required for this project. Although the tweet dataset is now structured, the inherent flexibility of MongoDB remains advantageous, especially considering future scalability and potential alterations in data structure. My familiarity with JSON is another significant reason for opting for MongoDB. This proficiency aligns seamlessly with MongoDB’s BSON format, facilitating easier data manipulation and integration. The ease of setup and use of MongoDB, based on personal experience, further supports its selection for this project, ensuring an efficient and user-friendly database environment.

Setting up the database and uploading data into MongoDB was notably straightforward and efficient. This process was accomplished with a single line of code, significantly simplifying the setup procedure. The simplicity and effectiveness of this method are illustrated in Figure 6, showing the command used to create the database and import the data.



Figure 6 – MongoDB command to create a database and import data.

Time taken to upload the data into MongoDB largely depends on the file size. Remarkably, even for the substantial size of the file, the entire data upload process was completed in less than a minute.

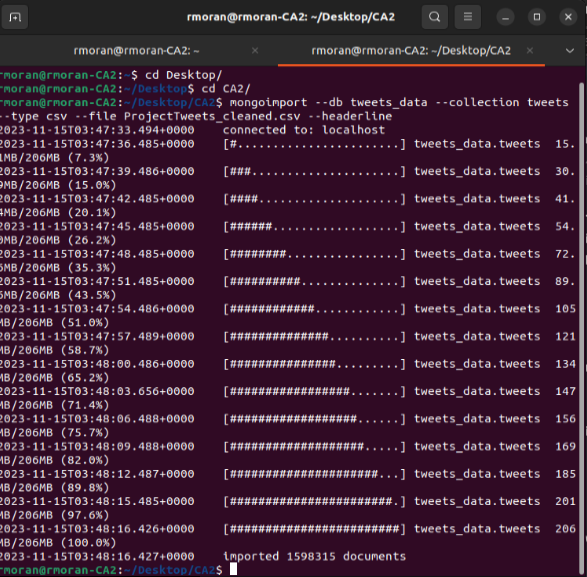


Figure 7 - MongoDB data import.

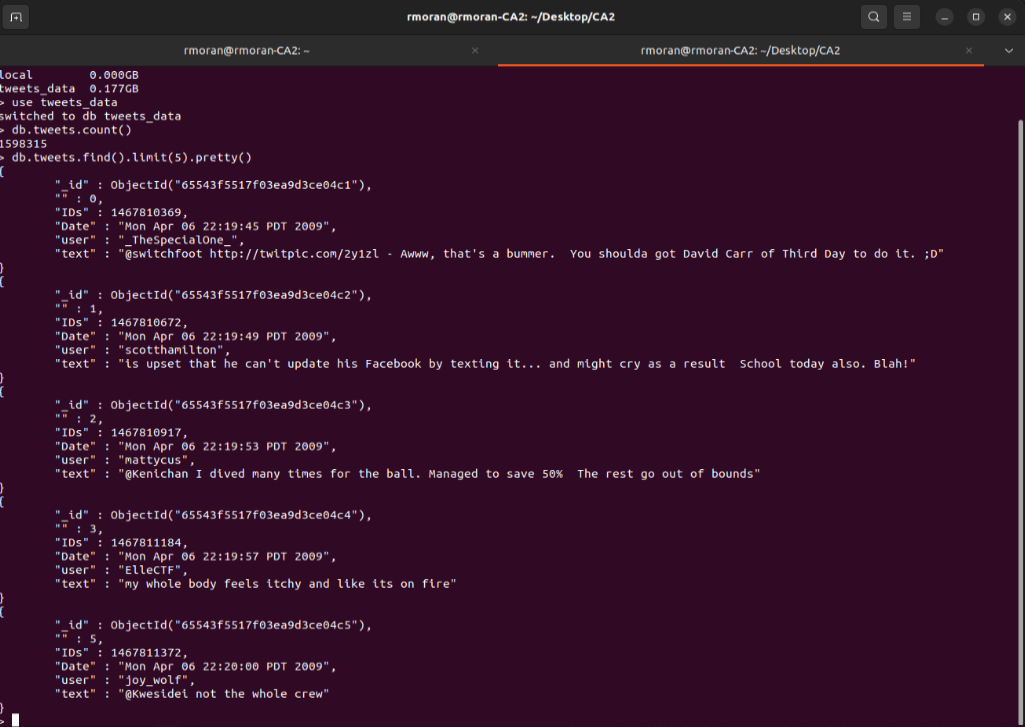


Figure 8 - MongoDB data visualization.

## Benchmarking Tests

The methodology adopted for the benchmark involved running a series of tests on both MongoDB and MySQL databases using YCSB. Three distinct workloads were selected to ensure a comprehensive evaluation covering various aspects of database performance. These workloads were designed to mimic realistic operations and queries that the databases might encounter in a real-world environment, thus providing practical insights into their performance.

### Workload execution

Three different workloads were executed on both MongoDB and MySQL databases. This approach allowed a direct comparison of how each database system handles different types of operations. During the execution of these workloads a range of performance metrics were collected and saved into different report. These reports served as the primary source of data for the subsequent comparative analysis.

### Comparative analysis of databases

1. **Throughput Analysis**

The throughput analysis is conducted to compare the number of operations per second that MongoDB and MySQL can handle under each workload. This metric is crucial as it directly reflects the databases' ability to handle load efficiently. Higher throughput indicates a database's superior performance in managing a higher volume of requests, which is critical in high-demand scenarios.

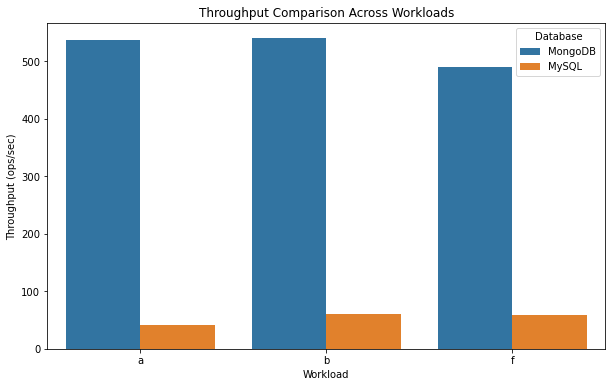


Figure 9 – Databases Throughput comparison

1. **Latency Analysis**

Latency analysis focuses on measuring the response time of each database. In this context, latency refers to the time taken to complete an operation or a query. Lower latency is desirable as it indicates a quicker response to user requests, contributing to better overall user experience and system efficiency.

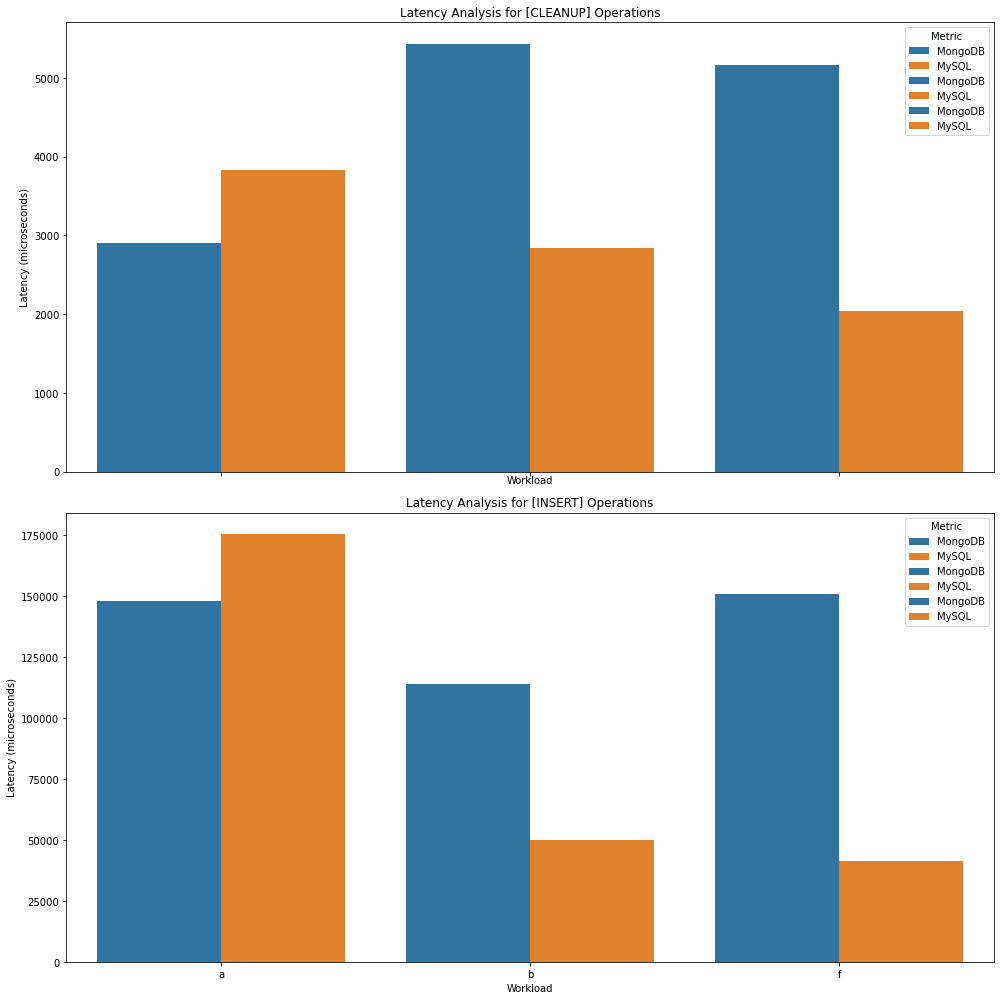


Figure 10 – Databases Latency comparison

1. **Runtime Comparison**

Evaluating the total time each database takes to complete the entire workload is essential in understanding their efficiency. A shorter runtime suggests a database's better capability to handle and process data swiftly, which is particularly important in time-sensitive applications.

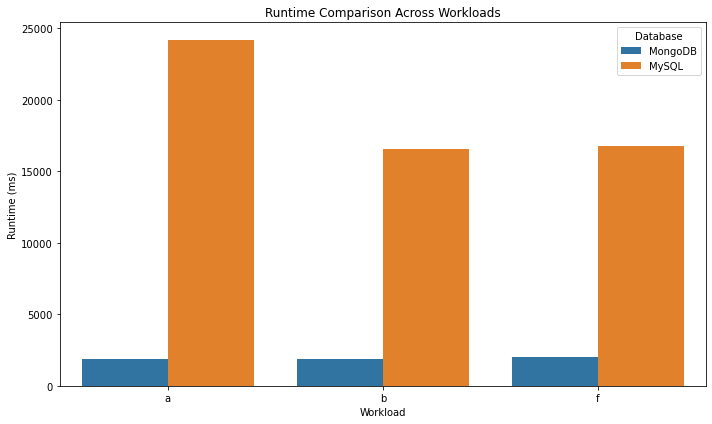


Figure 11- Databases Runtime Comparison

1. **Garbage Collection Metrics**

This section assesses the efficiency of memory management in MongoDB and MySQL. Garbage collection metrics provide insights into how each database manages memory allocation and cleanup, which can significantly impact performance, especially under heavy loads or long-running processes.

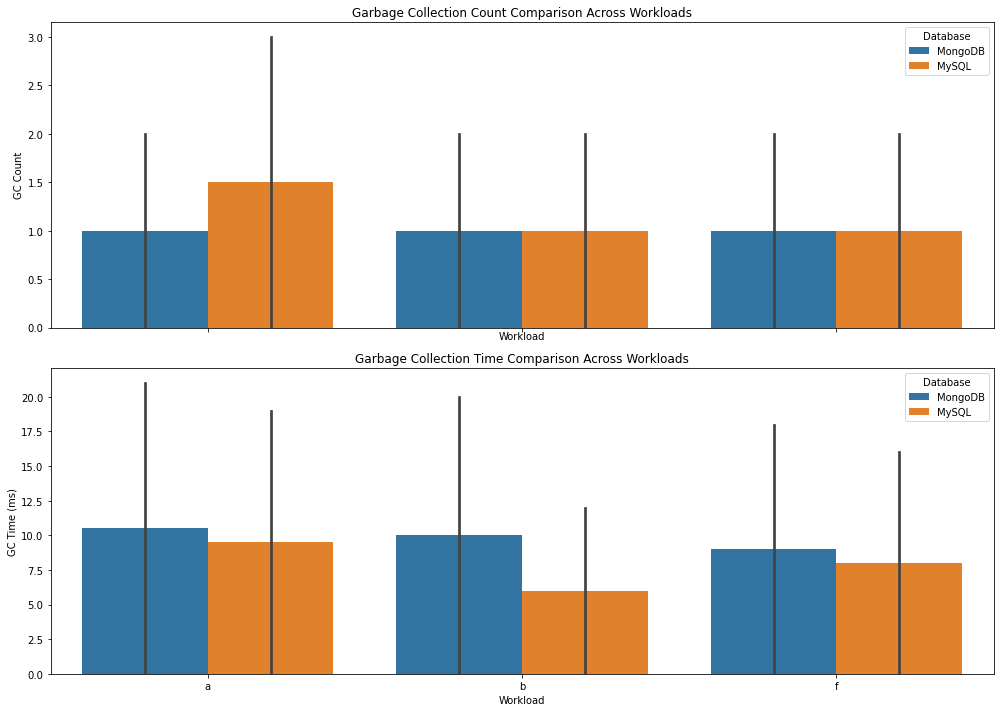
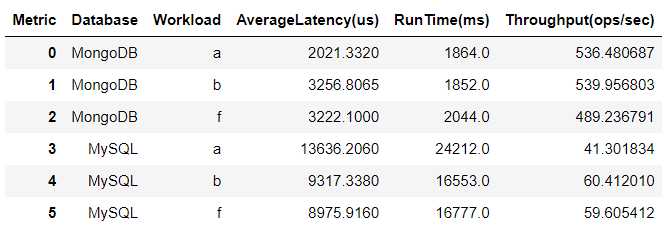


Figure 12 – Databases Garbage Collection comparison

1. **Summary**

These observations suggest that for the given workloads and test conditions, MongoDB outperforms MySQL in terms of throughput and latency. However, it's important to consider that these results are specific to the benchmarking environment and workloads tested.



# Data Processing with Spark

Spark was selected as the primary data processing environment due to its advanced capabilities in handling big data. The setup of Spark and the establishment of connections between Spark and the databases were crucial for efficient data processing.

## Spark Set-up

Spark is easy to setup and mainly consist of create your session for monitoring and import the necessary libraries to access all the features of spark

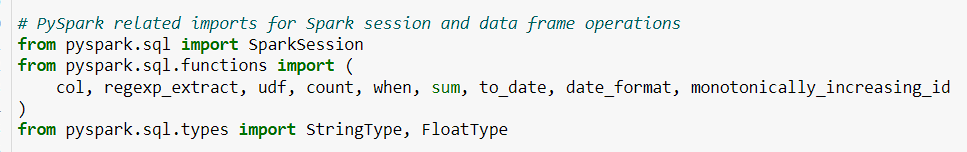


Figure 13 – PySpark libraries

A session can be initialized in spark by setting the parameters on Figure 14.



Figure 14 – Spark Session Parameters

## Database connection setup

The setup for MySQL database connection with spark are the ones shown at the Figure 15.

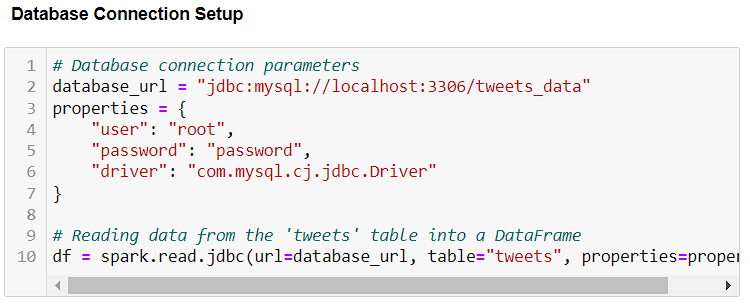


Figure 15 – MySQL Connection Setup

## Data Analysis in Spark

Upon importing the dataset into Spark. An initial inspection was made to confirm the correctness of the structure. Then the format of the Date column was inspected to later standardize the format to time-series analysis and temporal comparisons.

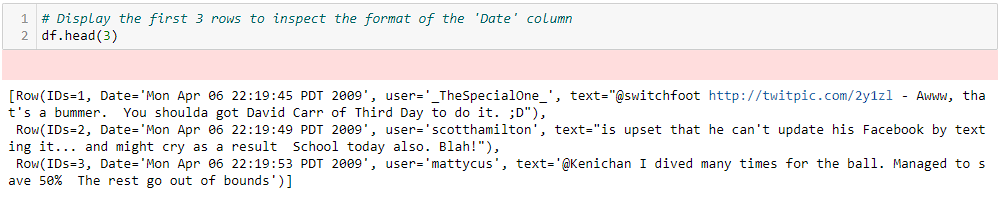


Figure 16 – Visualization of Date column structure

In order to transform the date to a standardized format was necessary to set the legacy time parser policy in Spark.



Figure 17 – Spark time parser policy

Next step was to convert the date to a datetime format more suitable for the analysis like “YYYY-MM-dd”. This conversion facilitates time-based analysis, as demonstrated in Figure 19, which outlines the dataset time range.

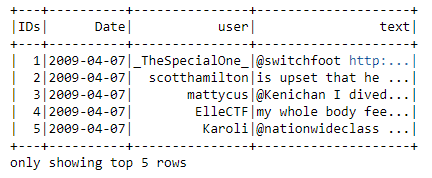


Figure 18 – Data preview with standardized date format

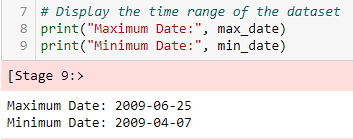


Figure 19 – dataset time range

### Temporal Analysis and User Engagement

In order to understand user behaviour and detect engagement patters data was grouped by user. A table was generated to count the number of tweets per user.

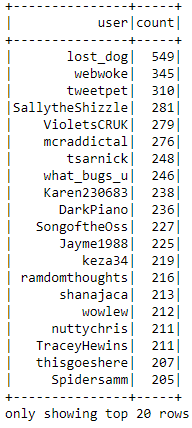


Figure 20 – Tweets per user table

Additionally in order to identify the relationship and distribution of the dataset in relation to the users the visualizations on Figure 21 where generated.

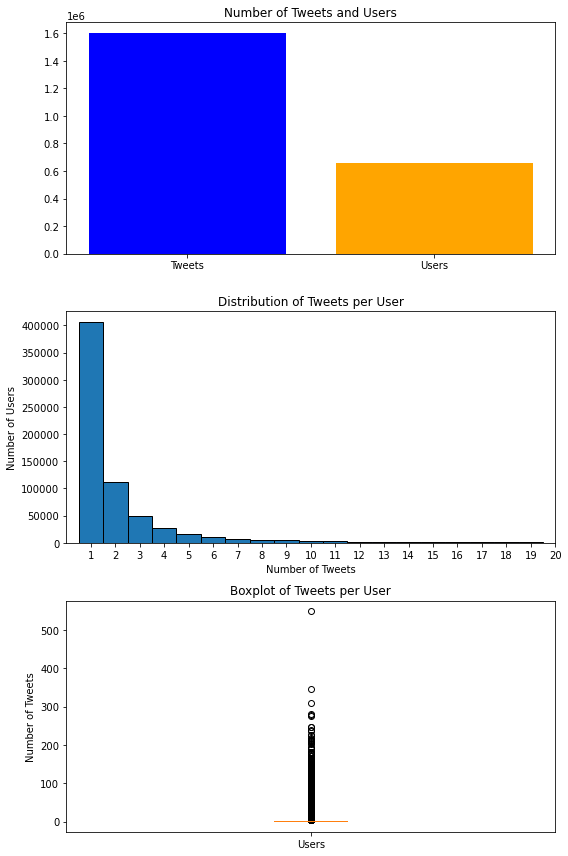


Figure 21 – Tweets per user comparison, distribution of tweets per user and Boxplot of tweets per user.

Key insights from graphs:

1. **Tweet-to-User Ratio:** With approximately 1.598 million tweets from around 659,775 unique users, the dataset suggests an average of just over 2 tweets per user. This ratio, indicating a balance of single and multiple tweets per user, is further illustrated in Figure 17, showcasing the top 20 users with the most tweets.
2. **Skewness in Tweet Distribution:** The data exhibits a right-skewed distribution, as most users tweeted only a few times, while a minority tweeted much more frequently. This skewness is evident in Figure 18, displaying the distribution of tweets per user.
3. **Boxplot Analysis:** The boxplot shows a concentration of median and interquartile ranges at lower tweet counts, with a long whisker and outliers indicating users with higher tweet counts.
4. **Identifying Outliers:** Among these outliers, the top 5 users, such as "lost\_dog" with 549 tweets, stand out for their high frequency of tweeting.
5. **Consideration of Potential Bias:** Despite some repetition among users, the dataset's broad user base helps minimize bias from frequent tweeters.
6. **Implications for Generalization:** The diversity of users in the dataset suggests its reliability for sentiment analysis, not unduly influenced by a few vocal users.
7. **Impact on Overall Analysis:** The substantial size of the dataset ensures that outliers have a limited effect on aggregate metrics, maintaining the integrity of median and percentile-based analyses.

### Analysing the Most Popular Hashtags

In this phase, I examined hashtags in tweets to ascertain their impact on sentiment analysis and forecasting. This involved identifying top hashtags to streamline the data by removing non-essential text, thereby improving sentiment scoring accuracy.

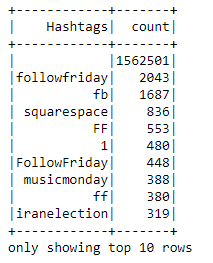


Figure 22 – Top 10 Hashtags from dataset

The analysis revealed trends such as “Follow Friday” and variations in hashtags referring to the same trend. I then quantified how many records were associated with these trends to understand their prevalence and impact on the dataset.

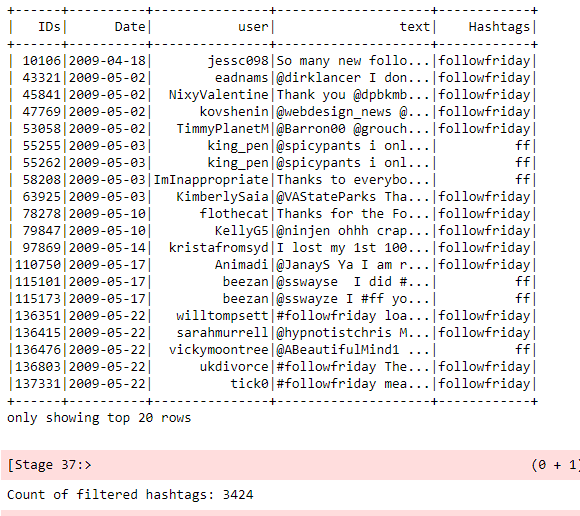


Figure 23 – Count of ‘FF’ tweets from the dataset

Based on the analysis, it is evident that the majority of tweets in the dataset do not contain hashtags. While hashtags like "Follow Friday" could potentially indicate a positive sentiment, such as users expressing happiness about gaining new followers, their infrequent use limits their significance as a variable for predicting future sentiment trends. This observation highlights the need to consider other factors or textual elements in the dataset for a more comprehensive sentiment analysis.

### Analysing the Most Popular Mentions

Same approach was used to identify the most common mentions from the dataset.

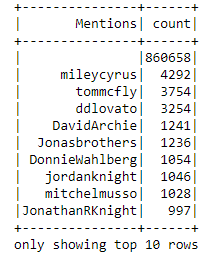


Figure 24 – Top 10 Mentions from dataset

The analysis reveals a predictable trend in social media behaviour, where public figures such as actors or singers are commonly mentioned or retweeted, especially in relation to newsworthy events like the release of a new song or movie. However, similar to the case with hashtags, the majority of tweets in the dataset do not include mentions. This observation suggests that while mentions of public figures are notable, they are not prevalent enough in the dataset to significantly influence overall sentiment analysis or trend predictions.

### Distribution of tweets over time

The distribution of the tweets was analysed in order to identify if the dataset was balanced or not, also this could help to identify periods with most active users.

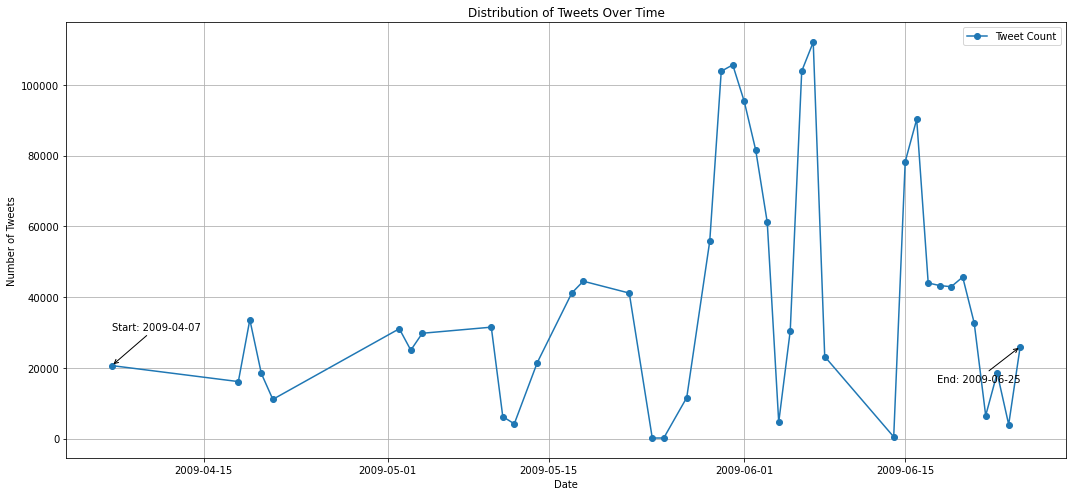


Figure 25 – Distribution of Tweets over Time

**Insights from Tweet Distribution Over Time**

* **Time Span:** The dataset covers April 7 to June 25, 2009, spanning nearly three months.
* **Tweet Volume Fluctuations:** User engagement shows significant variability, with notable fluctuations in tweet volume.
* **Initial Volume:** The period starts with lower tweet volumes around April 7, 2009.
* **Activity Peaks:** A substantial increase in tweets is observed mid-May, with a peak in early June reaching nearly 100,000 tweets.
* **Post-Peak Decline:** After early June's peak, tweet volume significantly drops towards June 25, 2009.
* **End-of-Period Surge:** A slight increase in tweet activity is noted just before the dataset ends, indicating a potential uptick in interest or relevance.

### Removing Spam Tweets

Finally Spam tweets were identified and removed. Spam tweets are identified using a specific repetitive text pattern, leading to the extraction of unique usernames associated with this pattern.

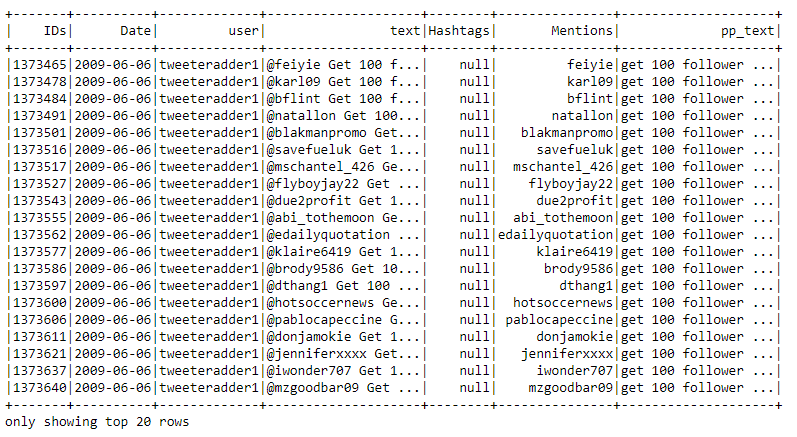


Figure 26 – Table of Spam Tweets from dataset

The removal of these spam tweets is a crucial step in refining the dataset. While the removal may not significantly alter the overall results due to the dataset's size, it aids in reducing potential biases and improves the reliability of sentiment analysis.

# Sentiment Analysis and Forecasting

Sentiment analysis is employed to evaluate the emotional tone conveyed in tweets. It will provide insights into the general sentiment expressed by users, categorizing it as positive, neutral, or negative based on the content of each tweet.

Using NLTK and Textblob libraries, I cleaned the tweet texts and calculated sentiment scores. This processed data was compiled into a new dataset, "tweets\_sentiment.csv," prepared for subsequent time-series forecasting.

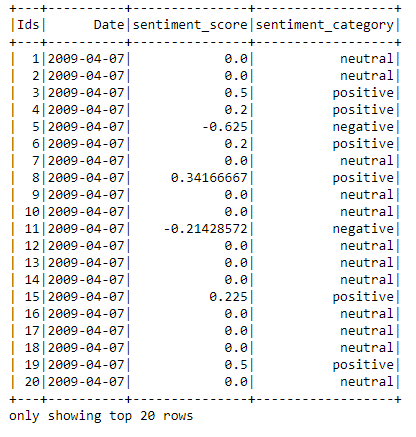


Figure 27 – Table of sentiment calculation

## Time-Series Analysis

First the distribution of the sentiments was analysed from the following plots.

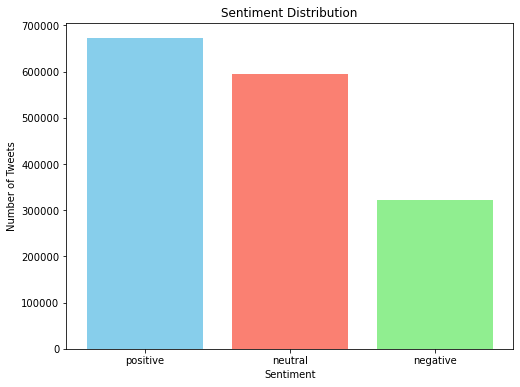


Figure 28 – Sentiment distribution by category

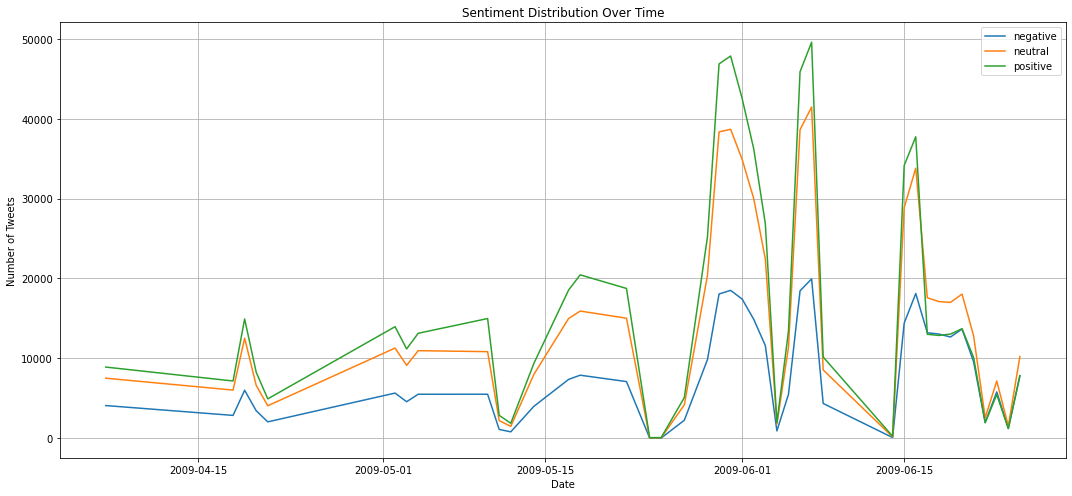


Figure 29 – Sentiment distribution Over Time

The dataset predominantly features positive sentiments, indicating an overall optimistic tone. A considerable number of neutral sentiments suggest the inclusion of informational or indifferent tweets. Negative sentiments, while less frequent, highlight the diversity of emotions and opinions present. This positive sentiment skew requires careful consideration to avoid bias in machine learning models. Furthermore, the consistency in sentiment proportions over time suggests stable sentiment distribution during the observed period.

### Forecast Feasibility Analysis from Resampled Sentiment Data

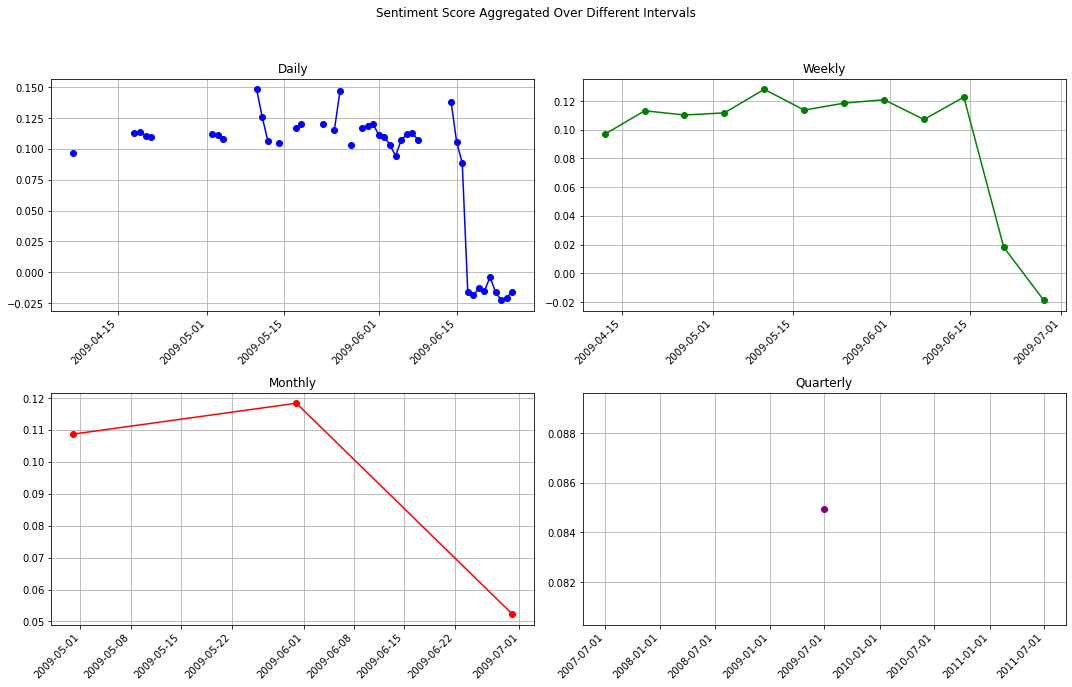


Figure 30 – Sentiment Score Aggregated Over Different Intervals

After visualizing the sentiment scores over daily, weekly, monthly, and quarterly intervals, the following observations are made:

* **Daily and Weekly Forecasting**: The data seems adequate for conducting daily forecasts and, to a lesser extent, weekly forecasts. The finer granularity of daily data allows for a more detailed trend analysis.
* **Limitations in Longer Intervals**: For monthly and quarterly forecasting, the dataset's short time span becomes a significant limitation. The lack of extensive historical data may result in less accurate predictions over these longer periods.
* **Concerns with Weekly Forecasts**: While weekly forecasting is possible, the brevity of the dataset raises concerns about the accuracy of such forecasts. The limited data points available for training could hinder the model's ability to learn robust patterns, potentially affecting forecast reliability.

While short-term forecasting (daily and weekly) is feasible with the current dataset, for longer-term predictions due to the dataset's limited time range pattern recognition could be not possible.

### Decomposition, trend, and seasonality analysis

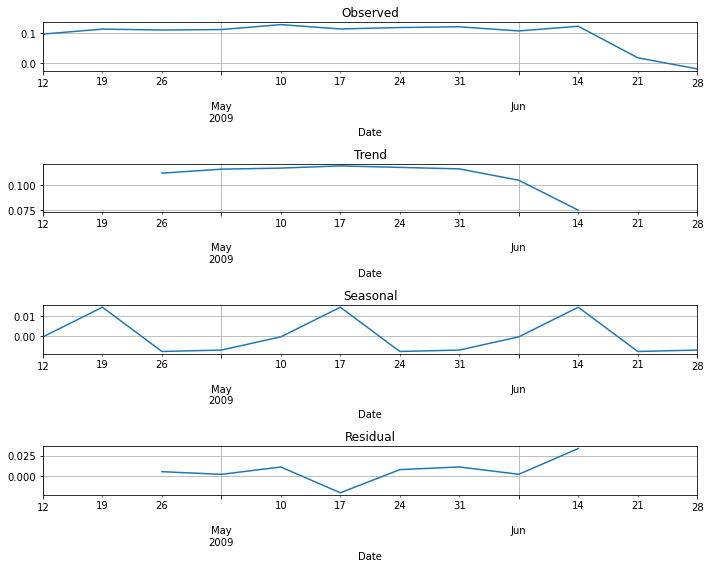


Figure 31 – Weekly dataset decomposition

### Stationarity testing and differencing

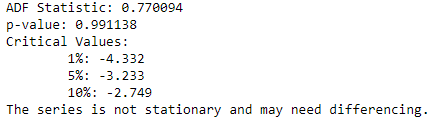


Figure 32 – Initial Stationary testing

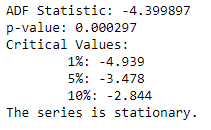


Figure 33 – Stationary test after 3 differencing

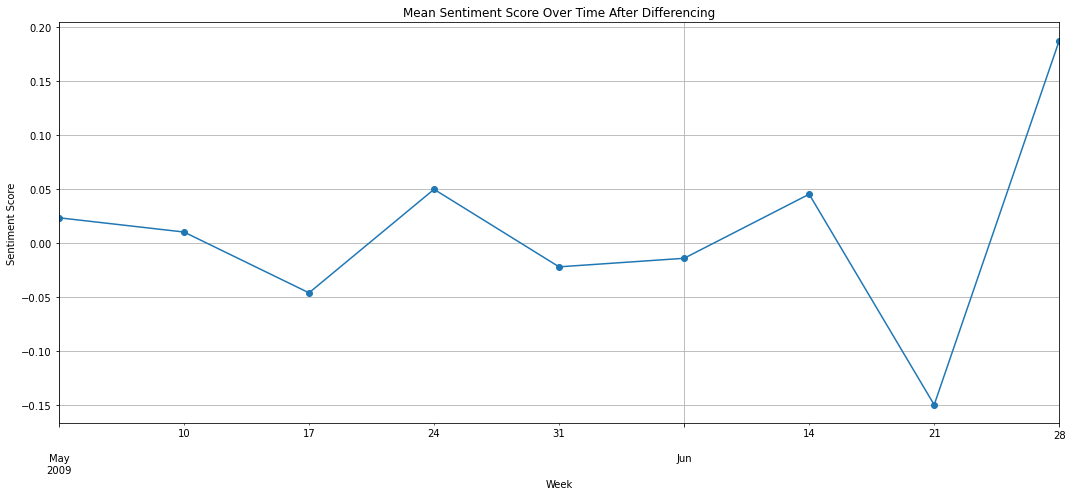


Figure 34 – Mean Sentiment Score Over Time After Differencing

### ARIMA, ForecastAutoreg, and Prophet models

## Model Selection and Forecasting

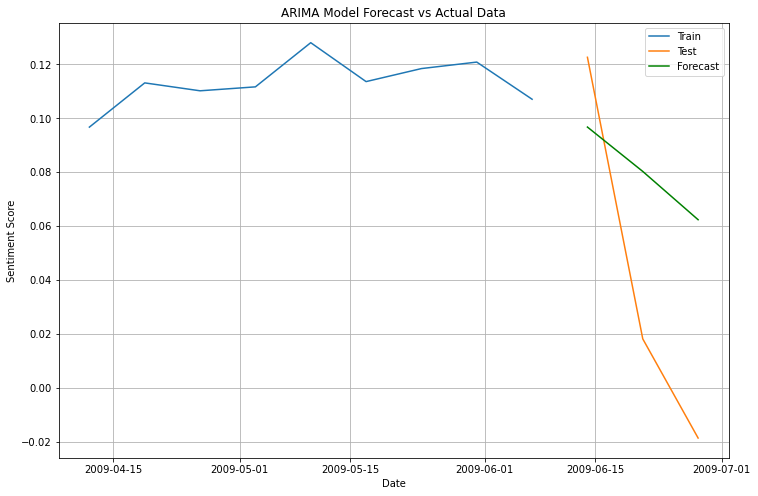


Figure 35 – ARIMA model forecast plot

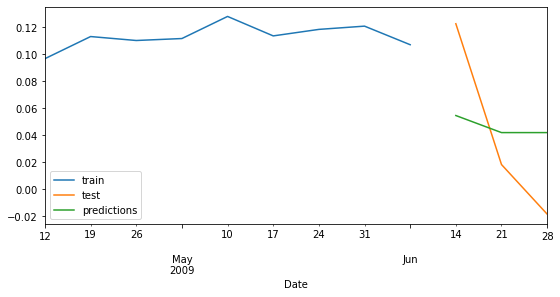


Figure 36 – ForecasterAutoreg forecast plot

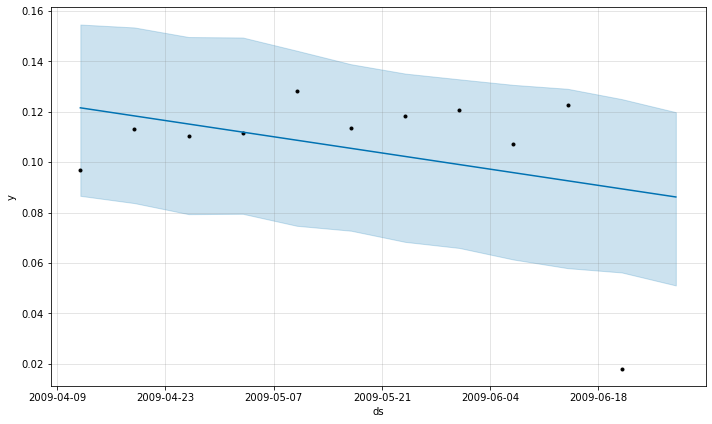


Figure 37 – Prophet forecast plot

# Dashboard Implementation

## Weekly forecast results

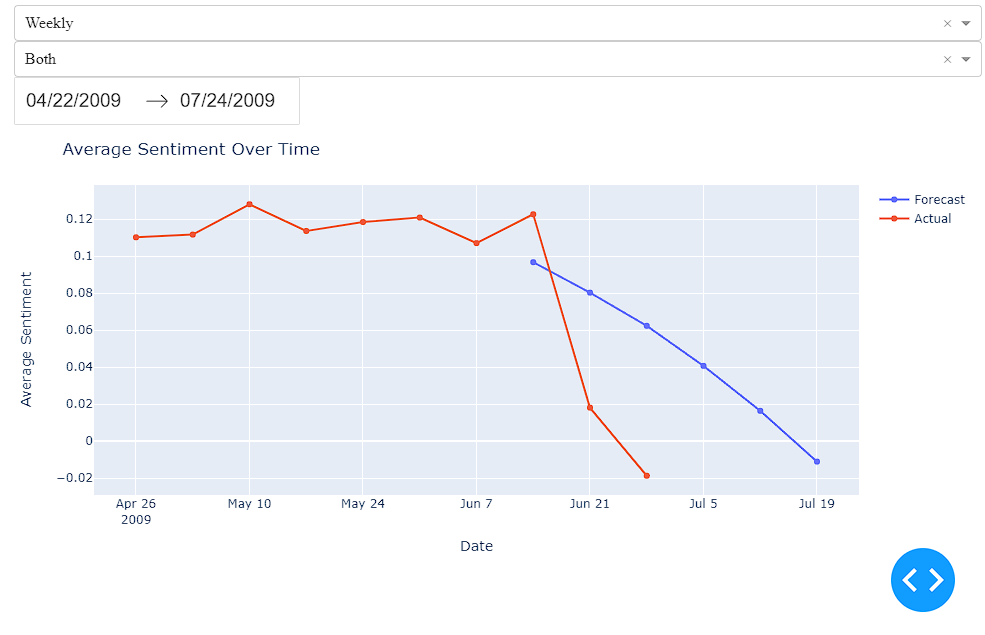


Figure 38 – Weekly Interactive Dashboard

## Monthly forecast results

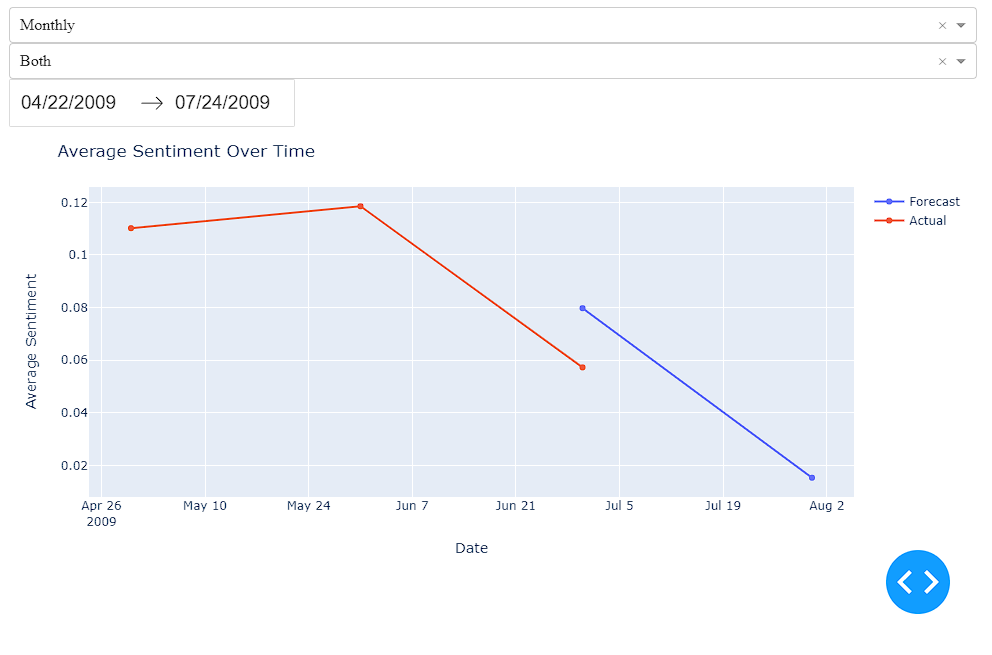


Figure 39 – Monthly Interactive Dashboard

## Quarterly forecast result

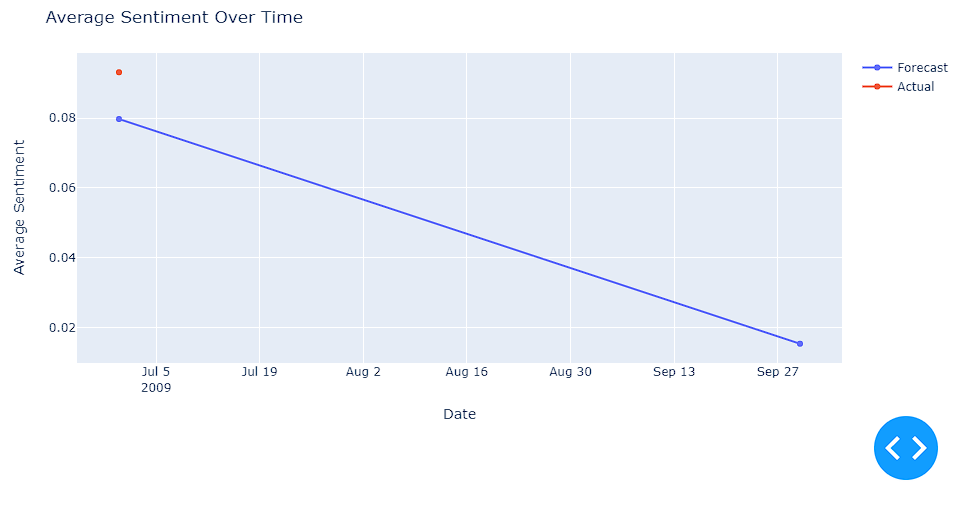


Figure 40 – Quarterly Interactive Dashboard

# Results and Discussion

## Results Overview

The purpose of this project was to use data analytics techniques to predict sentiment from media. Although I use methodologies such, as ARIMA, ForecastAutoreg and Prophet the results showed that these models were not completely accurate, in forecasting sentiment. There are factors that contributed to this outcome, which will be discussed below.

## Factors Affecting Model Accuracy

* **Limited Time Span of the Dataset**; The dataset only covers a period of time which posed a limitation. When it comes to sentiment prediction, in the realm of media having a broader historical data set is crucial for capturing and predicting trends. The narrow time range hindered the models ability to learn and adapt to long term fluctuations in sentiment.
* **Imbalance in the Dataset**; The dataset displayed a distribution, which likely affected how the models learned. When sentiment categories are not evenly represented it can introduce bias into predictions with a tendency to favor the occurring categories.
* **Complexity of Social Media Sentiment Analysis**; Sentiment expressed on media is heavily influenced by real world events, like concerts, sports matches, movie releases, product launches or political and war related incidents. This complexity introduces layers of unpredictability and variability to sentiment analysis making accurate forecasting quite challenging.

## Model Selection Rationale

* The ARIMA model (Autoregressive Integrated Moving Average) was chosen because of its effectiveness, in analysing time series data its ability to handle trends and seasonal patterns. Given that the dataset exhibits a trend or seasonal pattern ARIMA was considered as a choice for this analysis.
* ForecastAutoreg was selected due to its capability to automate the forecasting process while taking into account features. This model proves beneficial when historical data patterns are expected to persist into the future.
* Prophet, developed by Facebook is specifically designed for forecasting at scale. Considers holiday effects and multiple seasonalities. Its robustness in handling data with seasonality patterns made it a suitable option, for this study.

## Analysis of Sentiment Over Time

Upon examining the time period, it became evident that there was a decline, in sentiment towards the conclusion of the datasets timeframe. This finding implies a pattern in sentiment. However given the duration of the dataset it is challenging to establish this as a trend. The models seemed to have absorbed this decline, which could have influenced their forecasting bias. This limitation highlights the significance of having a sufficiently long dataset, for precise sentiment analysis and forecasting purposes.

# Conclusion and Future Work

In summary although the models we chose offered insights their predictive accuracy was restricted by the qualities of the dataset and the inherent complexities involved in analysing sentiment, on social media. Future research would benefit from a more balanced dataset enabling a holistic understanding of how sentiment trends evolve over time. Furthermore integrating real time event monitoring could improve the models capability to capture changes, in social media sentiment.