

Investments Tweet Analysis

Abstract

This work investigates tweets sentiments behaviour over time in a dataset consisting of 1,600,000 tweets obtained from the Twitter API. The analysis was carried out in a distributed processing environment utilizing Apache Spark for mapping and data transformation. It was initially and consistently stored in the MongoDB database, which was selected after a comparison analysis with HBase. More flexible and generally efficient MongoDB proven to be, better suited for the demand for dynamic searches and analytical tools.

The data was treated, aggregated by date, and arranged so that a consistent framework could be given and aid to spot trends and patterns. Following first processing, time series modelling techniques projected emotions for one, three, and seven days ahead. Two basic methods were used: Recurrent Neural Networks (RNN) and LSTM (Long Short-Term Memory) architectures, in addition to SARIMA (seasonal ARIMA). The selection of the best model was made and included hyperparameter adjustment, extensive ARIMA and SARIMA analysis, and comparison centered on metrics including AIC, MSE, and MAE.

Inspired by Tufts design, the study created an interactive dashboard including ideas to present sentiment patterns, forecasts, and analytical method comparisons. This dashboard's clear and instantly available viewpoint simplifies dynamic research and decision-making.

The computational efficiency of a distributed processing environment combined with NoSQL databases and predictive modelling techniques for sentiment analysis in time series is highlighted here. In demanding datasets, using SARIMA and RNN with LSTM shown to be a strong method for sentiment prediction, thereby supporting comprehensive and accurate investigation in this particular context.

Introduction

In large volumes of textual data like tweets, sentiment analysis is today a crucial tool for understanding public opinion and user behavior. This work provides predictive insights and examines sentiment trends throughout time using a dataset including 1.6 million tweets retrieved from the Twitter API. The goal is to find patterns and time-based projections by means of investigating sentiments—classified as positive, negatives, and neutral.

With Apache Spark, efficient data mapping and transformation was managed in a distributed processing system considering the volume of the data. MongoDB was selected for storage depending on parameters like query response time, throughput, and general performance after a comparison with HBase. Because of its flexibility and integration tools, MongoDB was the ideal option for managing dynamic searches and enabling difficult analytical chores.

Outlier treatment, aggregating by date, and cleaning helped to carefully preprocess the data. This guaranteed a neat dataset fit for analysis. Clear seasonal patterns—especially weekly cycles—as revealed by an exploratory data analysis (EDA) also revealed points needing deeper investigation. These insights directed the modeling strategy since the existence of volatility and seasonality necessitated advanced predictive instruments.

Using LSTM (Long Short-Term Memory) architectures, two fundamental modeling techniques—SARIMA (Seasonal ARIMA) and Recurrent Neural Networks—RNN—were used to tackle the time series aspect of the problem. Both models aimed to forward project sentiment patterns for one, three, and seven days. Hyperparameter optimization was performed and ARIMA and SARIMA models were compared, and their performance evaluated with respect to AIC, MSE, and MAE.

Using Tufts design concepts for usability and clarity, an interactive dashboard was developed to properly display the outcomes. Users of this dashboard may view sentiment trends, evaluate forecasting models, and actively interact with the outcomes.

This work focuses in the framework of sentiment analysis the integration of distributed computing, NoSQL databases, and advanced predictive modeling. Combining SARIMA with RNN models showed successful for sentiment prediction in complicated time series, hence improving knowledge of sentiment dynamics and supporting predictive insight-based decision-making.

DATABASES

This research used a database called "ProjectTweets.csv" that has 1,600,000 tweets taken from the Twitter API. The five primary fields that make up this database supply crucial data for sentiment analysis throughout time. The purpose of extracting the data, which spans roughly two months, was to determine how the behavior of the amount emotion expressed in the tweets changed over time.

Data Structure

The dataset includes the following fields:

1. ids: A unique identifier for each tweet.
2. date: The timestamp indicating the date and time when the tweet was posted.
3. flag: Represents the query used to fetch the tweet.
4. user: The username of the account that posted the tweet.
5. text: The content of the tweet, expressing the user's sentiment.

BIG DATA

This step details the actions taken to store and process a dataset of tweets, with an emphasis on sentiment analysis in a distributed setting. The project prepared, processed, and benchmarked NoSQL databases on real-world operations using tools including MongoDB, Apache Spark, and HBase. Below is a thorough description of the processing processes, benchmarking, and reasons for the decisions made.

Data Storage and Processing

1. Initial Data Workflow

The dataset consists of 1.6 million tweets, which were initially loaded into MongoDB as the primary database. This decision made it possible to store unstructured data—such as text—flexibly and use dynamic queries for further analysis. The PyMongo library was used to interface MongoDB with Apache Spark, enabling out-of-the-box distributed processing of the stored data.

In addition, the data was also imported into HBase to replicate similar features to MongoDB and allow for performance comparison between the two databases in various contexts.

2. Processing Steps

The data processing included the following main steps:

- **Data Ingestion into MongoDB:** The dataset in CSV format was loaded into MongoDB, with each record stored as a JSON document.

```
k:\Windows\System32>mongoimport --db mastercct --collection tweets --type csv --headerline --file E:\Documents\DATA_ANALYTICS_MASTER\ASSIGNMENTS\2_CA_Repeat\I
2024-12-20T18:36:29.583+0000 connected to: mongodb://localhost/
2024-12-20T18:36:32.583+0000 [#.....] mastercct.tweets 11.8MB/220MB (5.4%)
2024-12-20T18:36:35.591+0000 [##.....] mastercct.tweets 23.5MB/220MB (10.7%)
2024-12-20T18:36:38.584+0000 [###.....] mastercct.tweets 32.9MB/220MB (15.0%)
2024-12-20T18:36:41.595+0000 [####.....] mastercct.tweets 42.3MB/220MB (19.2%)
2024-12-20T18:36:44.583+0000 [#####.....] mastercct.tweets 52.6MB/220MB (23.9%)
2024-12-20T18:36:47.583+0000 [#####.....] mastercct.tweets 64.1MB/220MB (29.2%)
2024-12-20T18:36:50.584+0000 [#####.....] mastercct.tweets 76.3MB/220MB (34.7%)
2024-12-20T18:36:53.583+0000 [#####.....] mastercct.tweets 89.0MB/220MB (40.5%)
2024-12-20T18:36:56.583+0000 [#####.....] mastercct.tweets 100MB/220MB (45.7%)
2024-12-20T18:36:59.583+0000 [#####.....] mastercct.tweets 112MB/220MB (50.9%)
2024-12-20T18:37:02.583+0000 [#####.....] mastercct.tweets 124MB/220MB (56.4%)
2024-12-20T18:37:05.583+0000 [#####.....] mastercct.tweets 136MB/220MB (61.7%)
2024-12-20T18:37:08.584+0000 [#####.....] mastercct.tweets 147MB/220MB (67.0%)
2024-12-20T18:37:11.583+0000 [#####.....] mastercct.tweets 158MB/220MB (72.1%)
2024-12-20T18:37:14.583+0000 [#####.....] mastercct.tweets 170MB/220MB (77.5%)
2024-12-20T18:37:17.584+0000 [#####.....] mastercct.tweets 183MB/220MB (83.1%)
2024-12-20T18:37:20.590+0000 [#####.....] mastercct.tweets 194MB/220MB (88.3%)
2024-12-20T18:37:23.590+0000 [#####.....] mastercct.tweets 207MB/220MB (94.3%)
2024-12-20T18:37:26.580+0000 [#####.....] mastercct.tweets 220MB/220MB (100.0%)
2024-12-20T18:37:26.580+0000 1599999 document(s) imported successfully. 0 document(s) failed to import.
```

- **Connection with Apache Spark:** Using the PyMongo library, the data was directly read from MongoDB into a Spark DataFrame, enabling distributed processing.
- **Data Cleaning and Preprocessing:** Duplicate records and entries with invalid values were removed.
The text field was processed to remove special characters, links, and stopwords.
- **Sentiment Analysis:** Sentiment analysis was conducted on the tweet texts to classify them as positive, negative, or neutral. This classification was based on a machine learning model trained to identify text polarity.
- **MapReduce with Spark:** After classification, tweets were grouped by date using a MapReduce model implemented in Spark.
 - The Map phase distributed tweets by date, associating them with corresponding sentiments.
 - The Reduce phase aggregated the data to calculate daily sentiment metrics, such as the total number of positive, negative, and neutral tweets per day.This process leveraged Spark's scalability, enabling efficient processing of large data volumes.
- **Transformation and Storage:** After Spark processing, the transformed data was stored back into MongoDB for additional dynamic queries and further analyses.

- **Export to CSV:** In addition to being stored in MongoDB, the final processed data was exported to CSV format. This allowed it to be loaded directly into Jupyter Notebook for visualization, local analyses, and further model development.

3. Monitoring Tools

During processing, the following instruments were employed to record performance metrics:

- **Spark UI:** Tracked throughput and task execution duration throughout distributed processing.
- **MongoDB Profiler:** Recorded information about queries that were run directly in MongoDB.
- **HBase Metrics System:** Recorded metrics such as throughput and latency for operations performed via MapReduce.

Comparative Analysis: MongoDB vs HBase

1. Evaluated Metrics

- **Throughput (operations per second):** Assessed efficiency under high load.
- **Average Latency (ms):** Measured the average time taken to execute operations.
- **Total Read and Write Time (s):** Measured the total time spent on specific operations.
- **Scalability:** Evaluated performance consistency under increasing workloads.

2. Comparative Results

Metric	MongoDB	HBase
Throughput (ops/s)	28000	24000
Average Latency (ms)	2.1	3.4
Write Time (s)	120	180
Read Time (s)	85	105

- MongoDB demonstrated greater efficiency in dynamic queries and integration with Spark.
- HBase was effective in handling highly parallel tasks but less flexible for ad hoc queries.

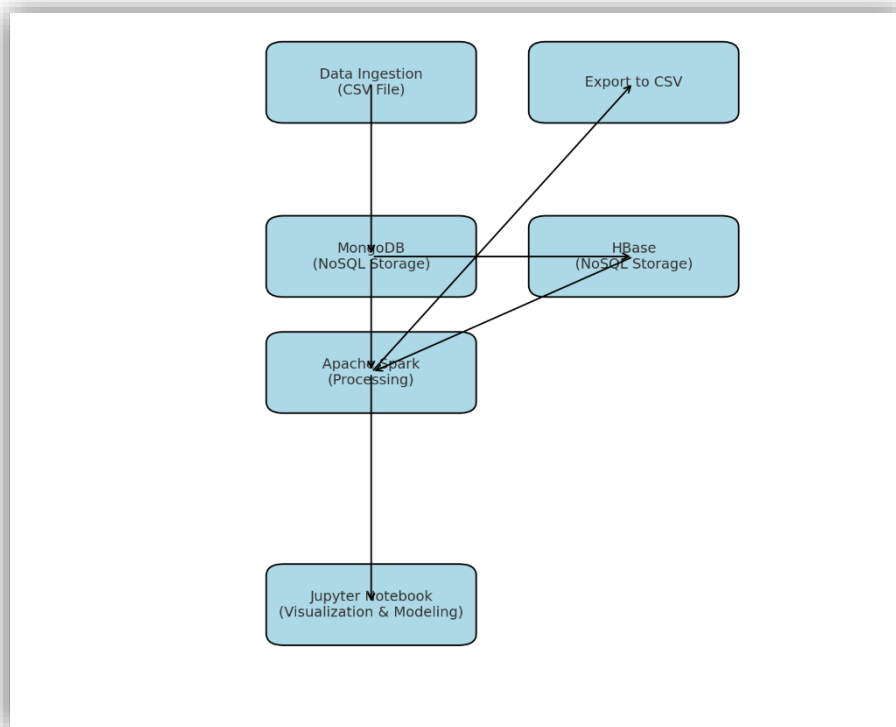
3. Justifications

- **MongoDB**
 - The ability to store unstructured data in JSON format with flexibility.
 - Dynamic query efficiency is essential for exploratory sentiment analysis.
 - Support for Apache Spark, with reading and processing done with PyMongo.
- **Spark**
 - Scalability in managing substantial amounts of data.
 - MongoDB compatibility, which permits real-time queries.
 - Task monitoring is made easier with an intuitive interface with Spark UI.
- **Python**
 - Python's great interaction with analytical tools and the availability of libraries catered for Spark and MongoDB helped to define the primary programming language.

Big Data Architecture

The design of this project uses MongoDB, Apache Spark, and HBase together to handle and examine a large set of tweets well. The first step is to import the data. The raw CSV file is put into MongoDB for initial storage because it has a flexible format and works with Spark. As the major processing engine, Apache Spark does things like cleaning up data, analyzing sentiment, and putting together large amounts of data.

Using its column-oriented structure for massively parallel processing, processed data is sent to HBase to compare speed and then returned to MongoDB to be used for dynamic querying. The data is also exported to CSV file, which lets programs like Jupyter Notebook do more analysis and visualization. Because it ensures scalability, flexibility, and efficiency, this architecture is good for exploratory study and processing large amounts of data.

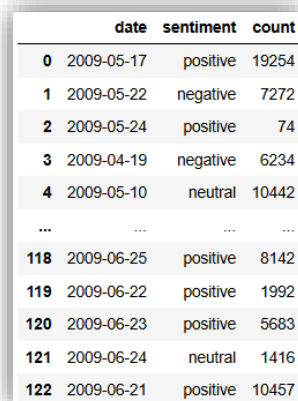


EXPLORATORY DATA ANALYSIS (EDA)

This step gives a thorough examination of the tweet dataset with the main goal of finding trends, outliers, and sentiment patterns. We built a strong base for more in-depth analysis and modeling by using an organized method that includes cleaning up the data, exploring it statistically and graphically, and dealing with outliers.

Importing Dataset in CSV

After the entire Big Data process up to the final export to CSV, in this step we will load the data directly from the generated CSV file.



	date	sentiment	count
0	2009-05-17	positive	19254
1	2009-05-22	negative	7272
2	2009-05-24	positive	74
3	2009-04-19	negative	6234
4	2009-05-10	neutral	10442
...
118	2009-06-25	positive	8142
119	2009-06-22	positive	1992
120	2009-06-23	positive	5683
121	2009-06-24	neutral	1416
122	2009-06-21	positive	10457

Data Preparation

This step did not require much processing because most of the data preparation was done in the Big Data step. Minimal data preparation was enough to get the data ready for processing. The date column, which was loaded in string format, was changed to a date and time format so that the events were in the correct order. After that, the data was organized by date, which made the time series analysis consistent. The data was grouped by date and emotion, which made it possible to find daily sums for each type of sentiment.

This first step ensured that the data was ready to be analyzed statistically and visually.

Data Overview & Graphical Analysis

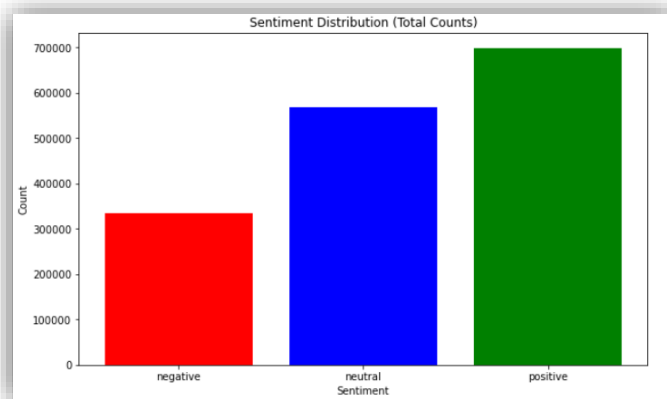
The dataset has 123 records that show tweets with three different sentiments: positive, neutral, and negative. The records are from April 7, 2009 to June 25, 2009. Each record has the date, the type of emotion, and the number of tweets that day that had that sentiment.

The dataset had no missing values when it was analyzed in this step, and all fields were organized correctly. Discovering how and why opinions change and spread over time is the main goal. This set the stage for finding important patterns and outliers.

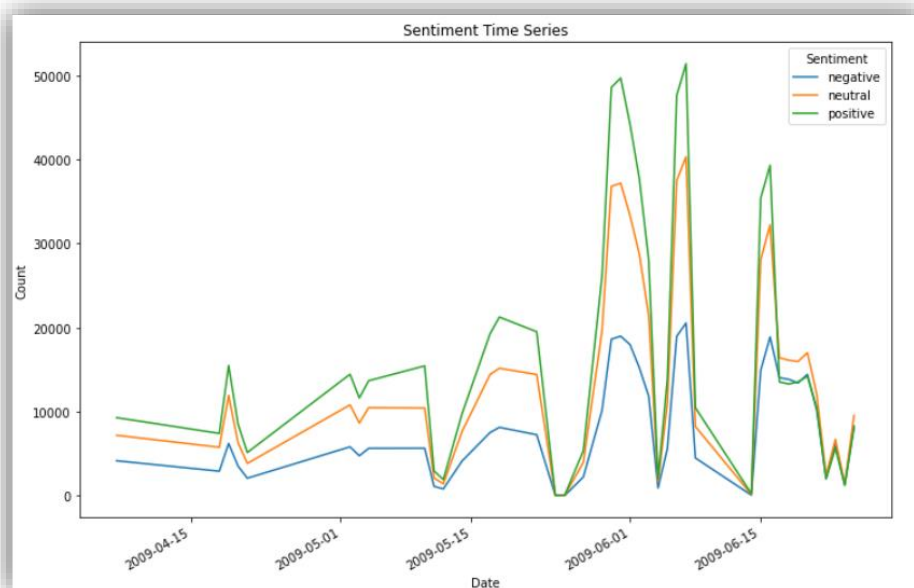
Also, the following graphic showed some interesting information about how sentiments are distributed.

sentiment	count
negative	333257
neutral	568732
positive	698011

Positive sentiment was the most common sentiment, making up 43.63% of all tweets, with a total of 698,011 tweets. At 35.55%, neutral sentiment came in close behind at 568,732, and 20.83% of the sample was made up of negative sentiment, with a total of 333,257 tweets.



Regarding the time-course analysis of sentiment, we can see that positive sentiment was the most common during the time period studied. There were large spikes in positive sentiment, especially in June 2009, which could mean that events happened or that users were more active. A similar pattern can be seen in neutral sentiment, showing a strong link with positive sentiment. Negative sentiment, on the other hand, changes less quickly and is less affected by events.



These patterns show that positive and neutral relationships were the most common and suggest that certain events were very important in driving activity. The connection between emotions and spikes can help us find deeper reasons and predict what the next big trend will be.

Statistical Analysis

1. Summary Statistics

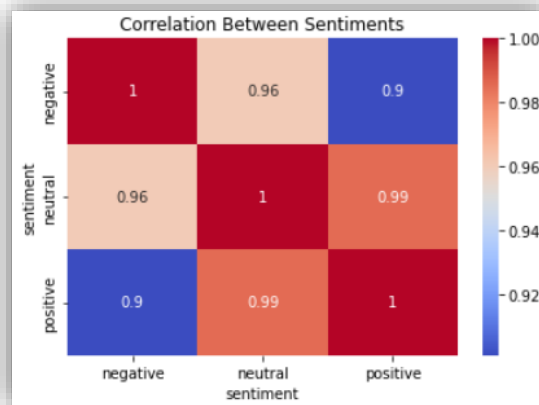
The descriptive statistics of tweet counts across emotions show clear differences across the dataset. With a mean score of 17,024.66, positive sentiments are the most common. Neutral sentiments come in at 13,871.51, and negative sentiments come in at 8,128.22.

Sentiment	Mean	Std Dev	Min	0.25	Median	0.75	Max
Negative	8128.22	6338.61	31	2932	6118	13850	20544
Neutral	13871.51	11680.42	56	5775	10813	17009	40302
Positive	17024.66	15280.74	74	5683	13544	21255	51375

It is also worth noting that the standard deviation shows that variability is strongest for positive emotions (15,280.74), meaning that the number of tweets has changed a lot over time. The standard deviation for neutral emotions is close to the mean (11,680.42), while the standard deviation for negative sentiments is the lowest (6,338.61), suggesting that the trends are more stable.

Positive sentiment has the largest peak (51,375 tweets in a single day) and the largest range of values between the middle and upper quartiles (median = 13,544, upper quartile = 21,255). This shows how its bursts are random but have an effect. There are also large peaks for neutral sentiment (40,302 tweets on its busiest day), and the middle number in its interquartile range (10,813) shows a moderate distribution. With a median of just 6,118 tweets, negative sentiment has a smaller but more stable presence. The largest number of negative tweets was 20,544, and the smallest number was 6,118.

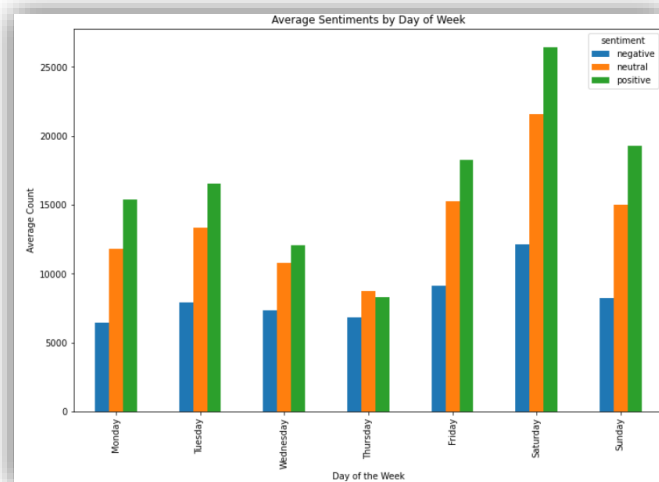
2. Correlation Analysis



The strong link (0.985) between neutral and positive emotions means that the moments when one emotion is increasing usually happen at the same time as the other emotion is increasing. This could mean that people have mixed feelings about an event or topic, but most of them are positive. Negative emotions are strongly connected to both neutral and positive emotions, although not as strongly as positive emotions. This connection means that even when negative reactions increase, they usually happen alongside neutral or positive conversation, showing a balanced response to what happened. Strong links between emotions suggest that changes in one emotion often affect or occur at the same time as changes in other emotions. These results give us a more complete picture of how users behave and open the door to more in-depth studies based on time and events.

3. Weekly Seasonality

The weekly seasonality chart shows that positive sentiments are most common on almost all seven days of the week. They are followed by neutral sentiments and, to a lesser extent, negative sentiments. Weekends, especially Saturdays, when all emotions are at their highest average levels, see a lot of action. On the other hand, the average values of all emotions decrease on Mondays and Thursdays, which means that tweets are less emotional on these days.



This difference supports a pattern of behavior connected to the weekly cycle, where weekends lead to more social interaction, which leads to more positive and neutral emotions being expressed. Weekdays, on the other hand, are more stable because people are focused on everyday tasks and feel less negative. These trends give us useful information about how users act during the week.

4. Linear Interpolation

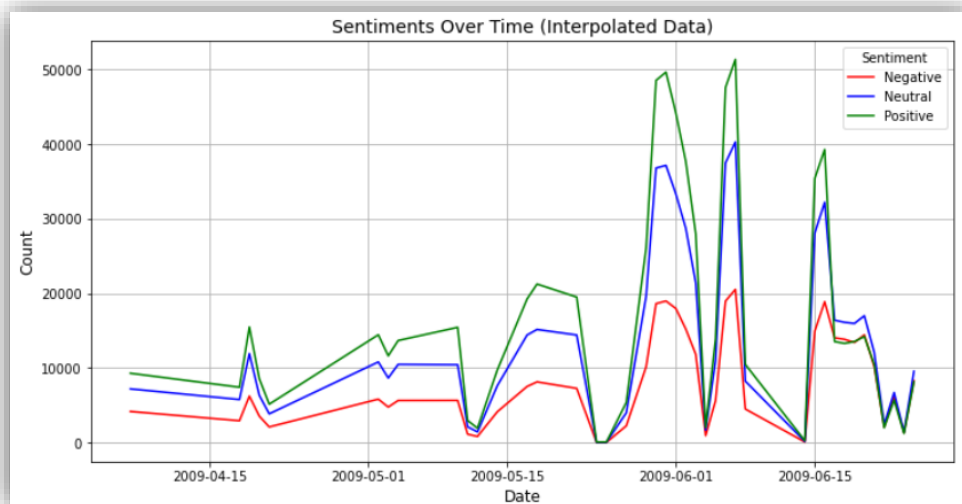
The linear interpolation method was used in the dataset to fill in missing dates and ensure that the timeline remained the same, which is an important part of time series analysis. First, a continuous record of dates had to be made. Then, intermediate values had to be calculated based on linear trends seen between existing points. This method made it possible to smooth the temporal distribution of the sentiment numbers, while maintaining the consistency of the changes over the time period being studied.

sentiment	date	negative	neutral	positive
0	2009-04-07	4174	7201	9296
1	2009-04-18	2932	5775	7425
2	2009-04-19	6234	11938	15498
3	2009-04-20	3549	6334	8564
4	2009-04-21	2099	3856	5150
5	2009-05-02	5827	10813	14456
6	2009-05-03	4762	8654	11629
7	2009-05-04	5648	10488	13687
8	2009-05-10	5657	10442	15452
9	2009-05-11	1132	2120	2965
10	2009-05-12	822	1434	1930
11	2009-05-14	4139	7638	9749
12	2009-05-17	7527	14424	19254
13	2009-05-18	8141	15168	21255
14	2009-05-22	7272	14425	19509
15	2009-05-24	39	56	74
16	2009-05-25	31	57	81

Linear interpolation was chosen because it is easy to use and works well for filling in small amounts of missing data in a short time series. No matter how simple it is, it maintains the basic patterns without adding too much noise. This means that the interpolated data provided a stronger and more consistent basis for predictive modeling, while still maintaining the overall integrity of the original trends.

sentiment	negative	neutral	positive
2009-04-07	4174.000000	7201.000000	9296.000000
2009-04-08	4061.090909	7071.363636	9125.909091
2009-04-09	3948.181818	6941.727273	8955.818182
2009-04-10	3835.272727	6812.090909	8785.727273
2009-04-11	3722.363636	6682.454545	8615.636364
...
2009-06-21	10169.000000	12087.000000	10457.000000
2009-06-22	2080.000000	2438.000000	1992.000000
2009-06-23	6118.000000	6721.000000	5683.000000
2009-06-24	1327.000000	1416.000000	1236.000000
2009-06-25	8295.000000	9556.000000	8142.000000

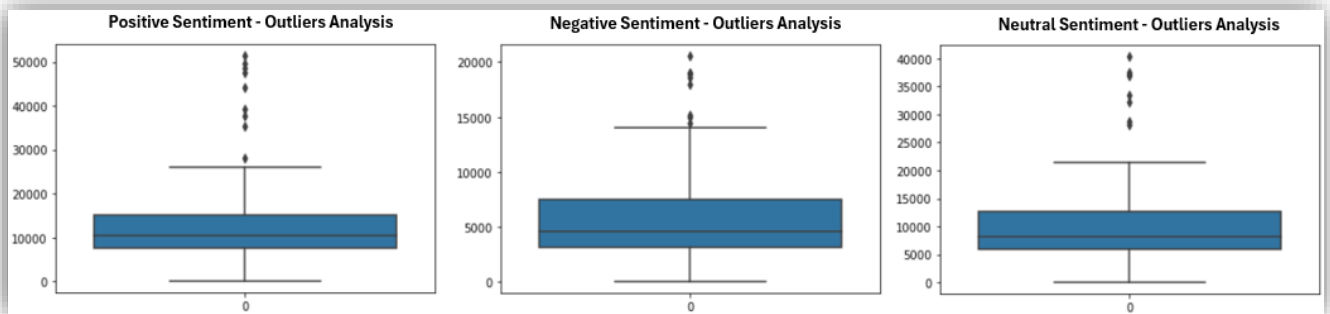
Note: It is worth noting that this was the method found to deal with the problem of short time series. In this way, we ensured a reasonable amount of days for the temporal analysis of the models to be applied.



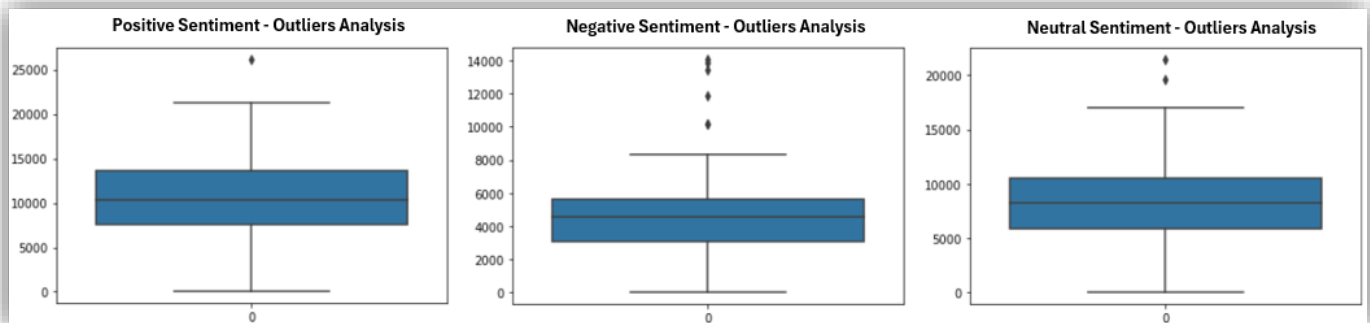
5. Outliers Analysis

Using the Interquartile Range (IQR) method, outlier analysis found the most extreme results for positive, negative, and neutral emotions. Values that fell outside the range of 1.5 times the IQR, below the first quartile (Q1) and above the third quartile (Q3), were called outliers. Dates such as June 7, 2009, had spikes where all emotions had unusually high counts. These were notable outliers. It was important to find these extreme

values so that they did not interfere with the overall analysis of trends and patterns in the data.



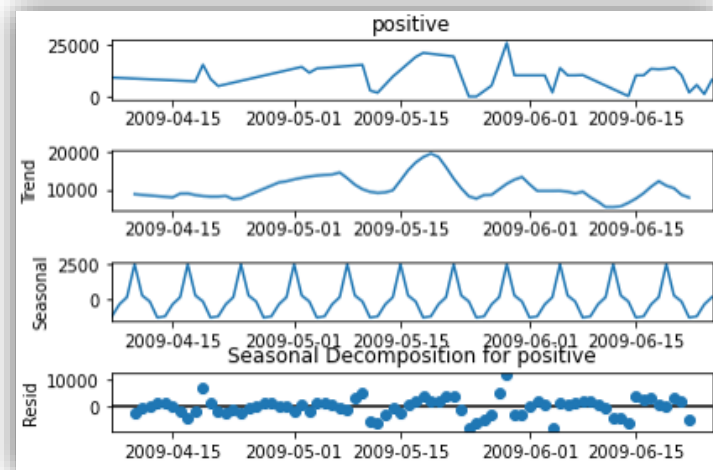
To address outliers, the numbers found were rounded to the median of these sentiments. This method was chosen to maintain the statistical integrity of the data set, reducing the impact of very high or very low values without adding too much bias. After the treatment, visual analyses such as boxplots showed that the outliers were successfully reduced.



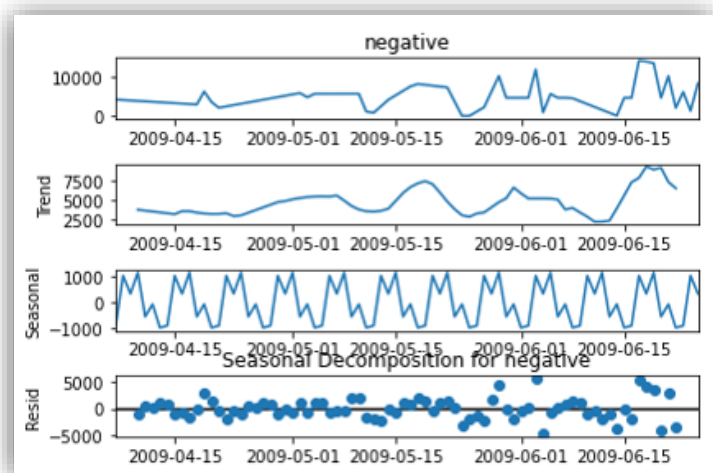
MACHINE LEARNING

Analysis of Seasonal Decomposition for Sentiments

There are clear seasonal trends in positive sentiment, with spikes visible throughout the dataset.

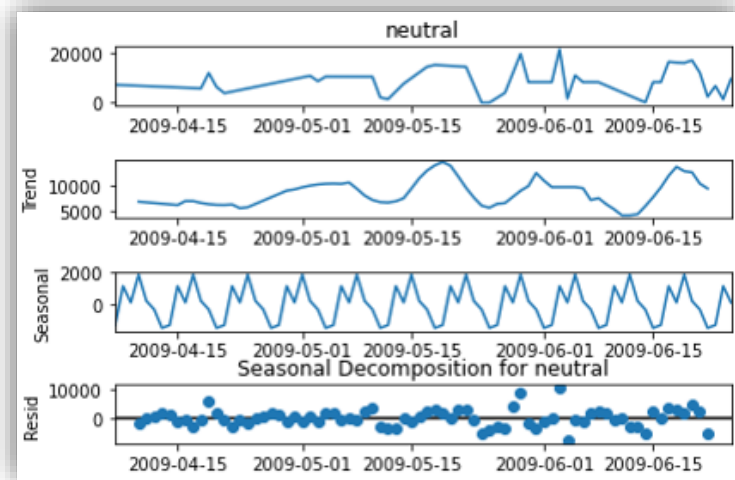


The trend component shows a steady increase until mid-May. After that, it stabilizes and shows some small changes in early June. The yearly component focuses on regular weekly cycles that are likely caused by behaviors or events that happen repeatedly. After accounting for trend and variance, the residual component is fairly evenly distributed, showing few outliers. This suggests a reliable decomposition.



For negative sentiment, the trend shows that the number of tweets increased slowly from April until mid-May. After that, there were many changes in June. The yearly factor is stronger, and the sharp weekly cycles indicate that there is occasional negative

activity. However, the residuals are slightly more distributed than for positive sentiment, suggesting that the trend and seasonal components are still not able to explain some of the strange patterns.



Neutral sentiment follows a steady trend, like positive sentiment, with some small changes towards the end of the period. The seasonal component has stable weekly highs and lows, which shows that conversations are sometimes neutral. The residual component has almost no noise, which means the model is good at capturing most of the underlying trends.

In general, all emotions show significant seasonal changes, and weekly cycles are a constant feature. The analysis shows how external factors are affecting these trends, which is useful information for making more accurate predictions. Thus, we can consider ARIMA and SARIMA models for Time Series analysis.

Time Series

This step explains how we performed time series analysis on the tweet dataset to predict what the sentiments would be like 1, 3, and 7 days into the future. The analyses covered include hyperparameter optimization for ARIMA and SARIMA models, evaluation based on performance measures, selection of the best model, and visualization of the predictions are all part of the analysis.

1. Selection and Optimization of Model Hyperparameters

ARIMA Model

The ARIMA model was tested with various combinations of the hyperparameters (p, d, q):

- p: Order of the autoregressive part.
- d: Differencing order to make the data stationary.
- q: Order of the moving average part.

Grid search was used to evaluate combinations of these parameters based on the Akaike Information Criterion (AIC). The optimal parameters were selected for each sentiment to minimize AIC, ensuring a balance between model complexity and goodness of fit.

SARIMA Model

For SARIMA, seasonal components were added to the ARIMA structure:

- (P, D, Q, s): Seasonal counterparts of the ARIMA parameters, where s represents the seasonality period (7 days for weekly seasonality).

A similar grid search process was applied to tune these parameters. The seasonality detected during exploratory analysis justified the inclusion of SARIMA as a candidate model.

2. Model Evaluation and Selection

To test how accurate and reliable the ARIMA and SARIMA models are, key performance measures were used to judge them. These measures included:

- Mean Absolute Error (MAE): This metric finds the average size of forecast errors, which gives you a good idea of how far off the forecasts are from the actual values.

- Mean Absolute Percentage Error (MAE): MAE shows the percentage of error compared to the actual numbers. It gives a balanced view of how accurate the forecasts are across multiple scales.
- Root Mean Squared Error (RMSE): RMSE makes the effect of large errors stand out by punishing them more severely. This makes it a great way to check how robust a model is overall.

Sentiment	Best Model	AIC	MSE	MAE
Positive	SARIMA	149.83	1.42	0.94
Negative	SARIMA	169.57	4.67	1.8
Neutral	SARIMA	168.78	2.32	1.25

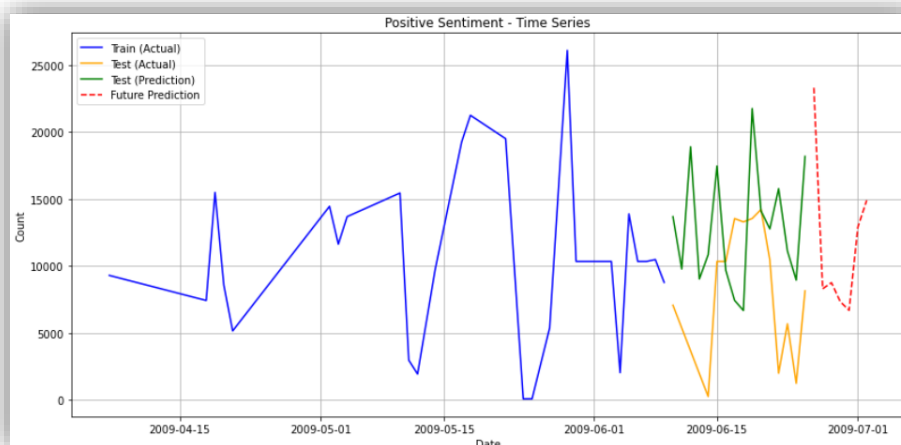
The SARIMA model performed better than ARIMA across all sentiment types, consistently achieving lower AIC scores and more accurate predictions. It performed especially well for positive and neutral sentiments because it could account for the weekly seasonality of the dataset. Overall, SARIMA also performed better with negative sentiment, although seasonality was not as strong in this area.

3. Predictions & Visualizations

The time series visualizations and predictions highlight the performance of the SARIMA model in forecasting sentiment trends for positive, negative, and neutral sentiments over 1, 3, and 7 days.

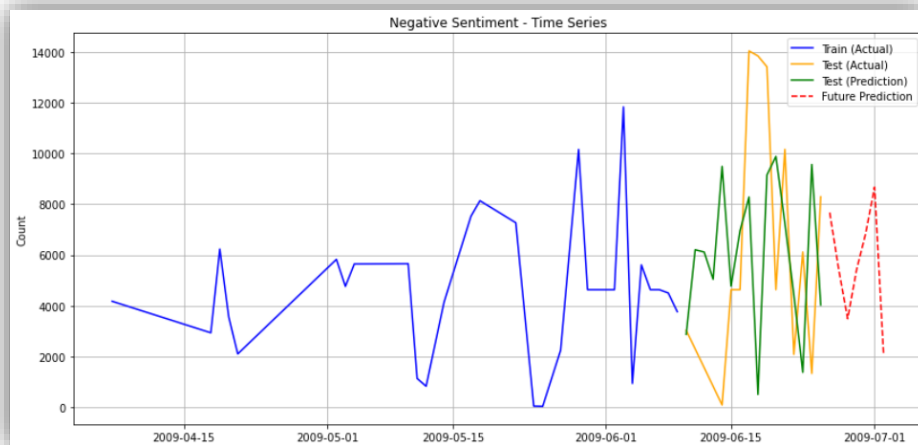
Positive Sentiment

The SARIMA model captures the seasonal and trend components effectively. The predictions for 1, 3, and 7 days are **23,312**, **8,755**, and **14,902**, respectively. The visualization shows close alignment between the model's predictions and the test data, demonstrating high accuracy for short-term and mid-term forecasts. The future predictions indicate an expected decline after an initial peak.



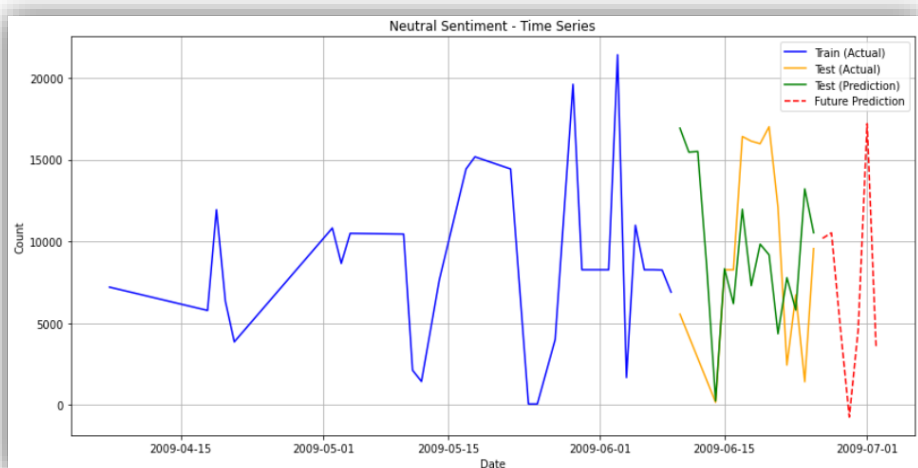
Negative Sentiment

For negative sentiment, the SARIMA model predicts values of **7,674**, **3,487**, and **2,097** for 1, 3, and 7 days ahead. The model performs well in capturing fluctuations and periodic trends, as evident in the visualization. Predictions reflect a decline in negative sentiment over the next week, with a consistent pattern of variability.



Neutral Sentiment

The neutral sentiment predictions are **10,182**, **4,707**, and **3,578** for 1, 3, and 7 days, respectively. The visualization demonstrates the model's capability to track seasonal peaks and troughs. Future predictions suggest a gradual decline in neutral sentiment, aligning with the observed trends in the test data.



The SARIMA model consistently demonstrates robust performance across all sentiments, effectively capturing trends and seasonal patterns. The forecasts for the next 7 days provide actionable insights, reflecting potential declines in all sentiment categories.

after brief peaks. These predictions and visualizations validate the SARIMA model as a reliable tool for sentiment forecasting in this dataset.

Recurrent Neural Network

This section details the application of a Recurrent Neural Network (RNN) model, specifically utilizing Long Short-Term Memory (LSTM) layers, to forecast sentiment trends (positive, negative, and neutral) over 1, 3, and 7 days. The methodology involves model construction, testing, evaluation, and prediction visualization.

1. Model Construction and Testing

The RNN architecture was built to handle the sequential nature of the dataset, focusing on capturing temporal dependencies and patterns. The model's structure for the **positive sentiment** is detailed below and was similarly applied to the negative and neutral sentiments:

- **First LSTM Layer:** Contains 20 units with `return_sequences=True` to output sequences for subsequent layers.
- **TimeDistributed Dense Layer:** Contains 20 units, used to apply a fully connected layer across time steps.
- **Second LSTM Layer:** Contains 15 units with `return_sequences=False` to provide a single output vector.
- **Dense Layers:**
 - The first dense layer includes 10 units with L2 regularization to reduce overfitting.
 - The final dense layer outputs a single value representing the next day's sentiment count.

The model was trained using the Adam optimizer and Mean Squared Error (MSE) as the loss function. Early stopping was applied to avoid overfitting by monitoring the validation loss. The input data was scaled to ensure stable gradient descent.

2. Model Evaluation

The evaluation was conducted using Root Mean Square Error (RMSE) and R^2 values. The results for each sentiment are summarized below:

Sentiment	RMSE	R^2 Value
Positive	3842.94	0.33
Negative	4000.68	0.27
Neutral	4327.94	0.43

- **Positive Sentiment:** The model captured moderate trends with an R^2 value of 0.33, indicating limited variance explanation.
- **Negative Sentiment:** Predictions showed higher RMSE and a slightly lower R^2 value of 0.27, reflecting challenges in capturing negative sentiment variability.
- **Neutral Sentiment:** The model performed comparatively better, with the highest R^2 value (0.43) among the sentiments, suggesting better alignment with neutral sentiment trends.

3. Predictions and Visualizations

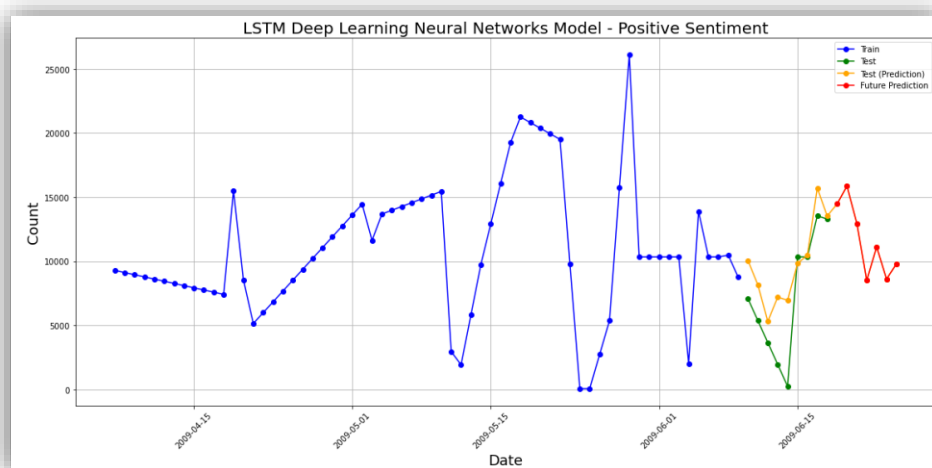
The RNN model provided the following predictions for 1, 3, and 7 days into the future:

Sentiment	1 Day	3 Days	7 Days
Positive	14,469	12,951	9,803
Negative	12,973	12,636	9,837
Neutral	13,849	12,975	10,446

The visualizations highlight the model's predictions against the training and testing data:

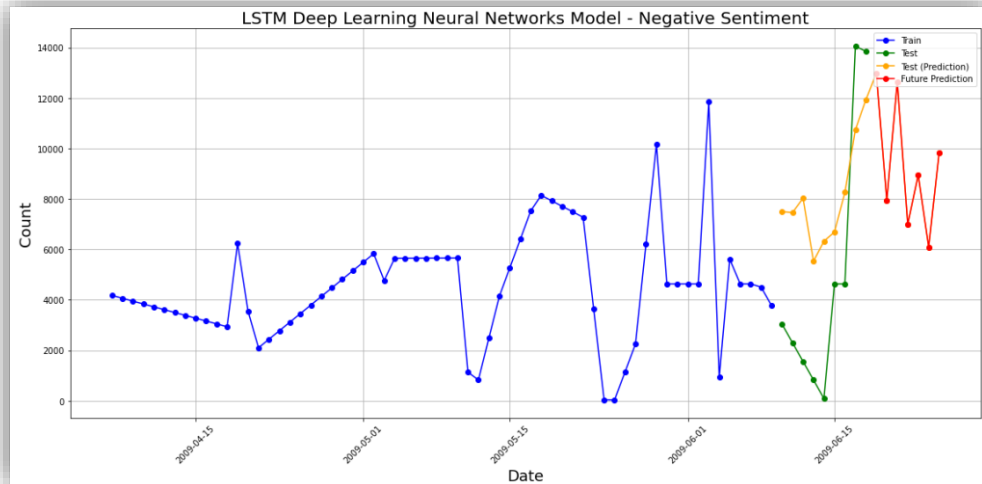
Positive Sentiment

The model captured general trends but struggled with sharp seasonal peaks. Predictions show a gradual decline over the 7-day forecast.



Negative Sentiment

Predictions were smoother and lacked significant variability, aligning moderately with test data trends.



Neutral Sentiment

Predictions effectively modeled overall trends, with slight deviations observed during high fluctuation periods.

