**CCT College Dublin Continuous Assessment**

**CCT College Dublin**

**Assessment Cover Page**

*To be provided separately as a word doc for students to include with every submission*

|  |  |
| --- | --- |
| **Module Title:** | BIG DATA PROCESSING AND STORAGE |
| **Assessment Title:** | LSTM AND AUTOREGRESSIVE TIME SERIES FORECASTING OF SENTIMENTS AT DAY 1,3 AND 7 GOING FOWARD |
| **Lecturer Name:** | DAVID MCQUAID  MOHAMMED IQBAL |
| **Student Full Name:** | DIANA FLORA NAMAEMBA |
| **Student Number:** | 2023385 |
| **Assessment Due Date:** | 19TH MAY 2024 |
| **Date of Submission:** | 18TH MAY 2024 |

**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. |

Contents

[INTRODUCTION 4](#_Toc167045573)

[METHOD 4](#_Toc167045574)

[DATA 4](#_Toc167045575)

[BIG DATA PROCESSING AND STORAGE 4](#_Toc167045576)

[PROJECT TWEETS DATA PROCESSING USING APACHE SPARK AND STORAGE USING MONGODB 4](#_Toc167045577)

[YCSB FOR COMPARATIVE ANALYSIS OF MYSQL AND MONGODB 6](#_Toc167045578)

[THE ARCHITECTURE FOR PROCESSING BIG DATA 11](#_Toc167045579)

[ADVANCED DATA ANALYTICS 12](#_Toc167045580)

[EXPLORATORY DATA ANALYSIS 12](#_Toc167045581)

[DATA PREPARATION AND CLEANING 13](#_Toc167045582)

[DATA VISUALIZATION OF THE DATE VARIABLE 16](#_Toc167045583)

[SENTIMENT EXTRACTION AND ANALYSIS 17](#_Toc167045584)

[HANDLING CLASS IMBALANCES 19](#_Toc167045585)

[TIME SERIES 19](#_Toc167045586)

[28](#_Toc167045587)

[References 29](#_Toc167045588)

**List of Figures**

[**Figure 1:The flow chart below shows the process of data Processing and storage** 5](#_Toc167042726)

[**Figure 2: Graph of RunTime comparison of MySQL vs MongoDB against other comparative parameters** 7](#_Toc167042727)

[**Figure 3: Graph of Throughput comparison of MySQL vs MongoDB against other comparative parameters** 8](#_Toc167042728)

[**Figure 4: Graph of Average Latency comparison of MySQL vs MongoDB against other comparative parameters** 8](#_Toc167042729)

[**Figure 5:Graph of RunTime comparison of MySQL vs MongoDB against other comparative parameters** 9](#_Toc167042730)

[**Figure 6:Graph of Throughput comparison of MySQL vs MongoDB against other comparative parameters** 9](#_Toc167042731)

[**Figure 7:Graph of Average Latency comparison of MySQL vs MongoDB against other comparative parameters** 10](#_Toc167042732)

[**Figure 8:Graph of RunTime comparison of MySQL vs MongoDB against other comparative parameters** 11](#_Toc167042733)

[**Figure 9:Graph of Throughput comparison of MySQL vs MongoDB against other comparative parameters** 11](#_Toc167042734)

[**Figure 10:Graph of Average Latency comparison of MySQL vs MongoDB against other comparative parameters** 12](#_Toc167042735)

[**Figure 11:The architecture for Processing Big Data** 12](#_Toc167042736)

[Figure 12:frequency distribution plot of tokenized text. 16](#_Toc167042737)

[Figure 13: Graph on the count of each unique day 17](#_Toc167042738)

[Figure 14:Histogram of the Date variable 17](#_Toc167042739)

[Figure 15: Distribution of the Different time components in the date variable 18](#_Toc167042740)

[Figure 16: Bar Graph of the sentiments extracted by text blob sentiment 19](#_Toc167042741)

[Figure 17: Plot of the sentiments after handling missing sentiments due to handling dates using interpolation 21](#_Toc167042742)

[Figure 18:Plot ACF for each sentiment category 21](#_Toc167042743)

[Figure 19:Decomposition plot for Positive sentiments 22](#_Toc167042744)

[Figure 20:Decomposition plot for Neutral sentiment 22](#_Toc167042745)

[Figure 21:Decomposition plot for Negative sentiments 23](#_Toc167042746)

[**Figure 22: Plot of the prediction vs baseline negative sentiments** 24](#_Toc167042747)

[**Figure 23: The prediction vs Baseline Neutral sentiments** 25](#_Toc167042748)

[**Figure 24: The prediction vs Positive sentiments** 25](#_Toc167042749)

[Figure 25:ARIMA Negative predictions vs baseline 26](#_Toc167042750)

[Figure 26:ARIMA Neutral sentiments prediction vs baseline 26](#_Toc167042751)

[Figure 27: ARIMA Positive sentiments prediction vs baseline 26](#_Toc167042752)

[Figure 28: Positive sentiments prediction vs baseline values 27](#_Toc167042753)

[Figure 29:Forecasting Day 1,3, and 7 Neutral sentiments 28](#_Toc167042754)

[Figure 30:Forecasting Day 1,3 and 7 Positive sentiments 28](#_Toc167042755)

[Figure 31:Forecasting Day 1,3 and 7 Negative sentiments 29](#_Toc167042756)

**List of Tables**

[Table 1: Table on the comparison parameters considered for YCSB workbench evaluation 6](#_Toc167042757)

[Table 2: Table of the ADF test findings 23](#_Toc167042758)

# INTRODUCTION

Social networking sites connect people in the world, allowing them to share pictures, content, videos and share their first-hand opinions on various issues. Big data analytics techniques are highly applied in the social networks because they are characterized by the 5V (Velocity, Volume, Veracity, Value AND Variety) s of big data.(Bazzaz Abkenar *et al.*, 2021). Some examples of social networking sites include Twitter, Facebook etc. Due to social media providing a big source of data, there has been an increase in application of machine learning, deep learning and time series techniques to better understand various problems. A lot of these analysis has been done mostly on twitter data. Twitter which has over 313 million monthly active users and which in a day over 500 million tweets are made, is one of the most utilised social media platforms when it comes to data.(Jianqiang and Xiaolin, 2017)

Some key techniques employed for understanding social media data include, sentiment analysis, times series analysis

# METHOD

## DATA

The dataset used for this project was project Tweets dataset, a csv dataset with 1,600,000 observations and five features extracted using the twitter api. It contains the following 5 fields:

* ids: The id of the tweet (eg. 4587)
* date: the date of the tweet (eg. Sat May 16 23:58:44 UTC 2009)
* flag: The query (eg. lyx). If there is no query, then this value is NO\_QUERY.
* user: the user that tweeted (eg. bobthebuilder)
* text: the text of the tweet (eg. Lyx is cool)

## BIG DATA PROCESSING AND STORAGE

### PROJECT TWEETS DATA PROCESSING USING APACHE SPARK AND STORAGE USING MONGODB

Big data processing is techniques utilised to access large scale data and extract meaningful information from them for decision making.(Mehdipour, Noori and Javadi, 2016), while big data storage are storage technologies that are not relational database systems that can be able to address the Volume, variety and velocity challenges of data.(Strohbach *et al.*, 2016). There are different big data storage and processing technologies available. Processing technologies include Hadoop Map-Reduce or Apache Spark, etc. While storage include either SQL or NoSQL databases such as HBase, HIVE, Spark SQL, Cassandra, MongoDB. Etc. MongoDB is a NoSQL database that is document oriented. It was preferred to other NoSQL databases because It is more useful for tweet analysis and other applications. It stores data in JSON format make it easy to analyse data.(Krishnan and Elayidom, 2016).

For this project Apache Spark was used for Preparing and Processing the Project Tweets Data, while MongoDB and spark SQL were used to Populate, Store and save Processed Data.Apache spark is a big data processing platform that has capabilities for both batch and stream processing. Spark was preferred over mapreduce because it outperfoms MapReduce in terms of performance.(Shaikh *et al.*, 2019). The Data was processed under batch processing, that is the its was a one time operation and after processing the output data was stored. Processing the data using Apache spark from MongoDB included: -

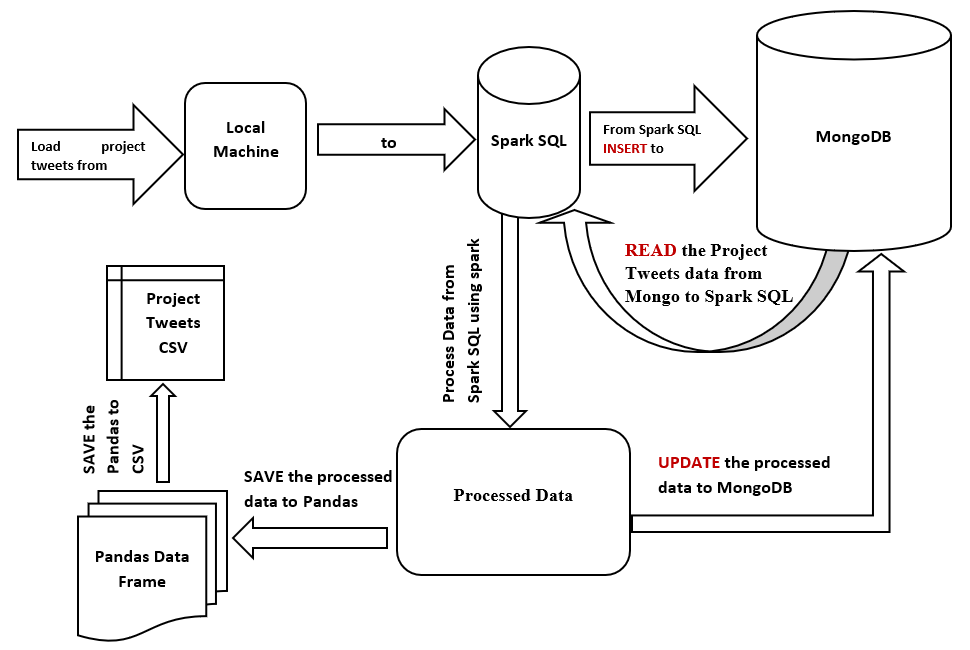
1. Performing CRUD (Create, Read, Update, Delete) operations that are possible for data stored in MongoDB. The CRUD operations are categorized into Read operations and Write Operations. The Project Tweets data was first loaded to Spark SQL from the Local machine. The CRUD operations were then performed as follows:
2. **Read operations** – These involved reading/loading the project tweets data stored in MongoDB into Spark SQL.
3. **Write Operations**- These operations involved creating/inserting, modifying and deleting data in MongoDB. Data was inserted from Spark SQL, deleting some parameters during processing and finally updating the processed data into MongoDB.
4. **Processing the data**

This involved Exploratory Data Analysis (EDA**).** EDA was performed to better understand the dataset, its patterns and characteristics. It involved checking for duplicates and missing data/dates in the dataset. The time components that are days and months were also extracted.

From EDA: -

* There were missing data points.
* There were duplicates

**Figure 1:The flow chart below shows the process of data Processing and storage**

****

The rationale behind choosing these two technologies was the need for efficient data processing and storage solutions that are capable of handling big data. Both Apache spark and MongoDB were chosen because of their performance capabilities, scalability, flexibility and suitability when it comes to handling tweets datasets, so as to ensure effective data manipulation, analysis and storage. Python was chosen as the programming language due to its capabilities and compatibility to effectively integrate with Apache spark.

### YCSB FOR COMPARATIVE ANALYSIS OF MYSQL AND MONGODB

There are various test strategies that can be implemented in order to perform comparative analysis of the capabilities of various big databases. One of the most common is Yahoo cloud service benchmark client (YCSB). YCSB is an open-source license tool used to benchmark new cloud database systems. Through YCSB one can be able to benchmark multiple systems and compare them by creating “workloads”. (Gaikwad and Goje, 2015). A YCSB Comparative analysis was conducted to compare Capabilities of MySQL and MongoDB.

The comparison involved comparing the two databases performance based on record counts 1,000, 10,000 and 100,000. Further the read and update proportions were also compared for different proportions and lastly the request distributions compared were Zipfian and uniform distribution for workload A, B and C, as shown in table below.

Table 1: Table on the comparison parameters considered for YCSB workbench evaluation

|  |  |  |
| --- | --- | --- |
| **Comparison Parameters** | **MYSQL** | **MONGODB** |
| Record Counts | 1,000, 10,000 and 100,000 | |
| Request distribution | Zipfian and Uniform | |
| Workload A – Update heavy | * Read: 50% and Update: 50% * Read: 70% and Update: 30% | |
| Workload B- Read heavy | * Read: 95% and Update: 5% * Read: 70% and Update: 30/5 | |
| Workload C- Read only | * Read: 100% and Update:0% * Read: 70% and Update: 30% | |

The metrics considered for quantitative analyses were:

1. ***Average Latency*** which is the average time that a database takes to read or write data. A lower value of the average latency indicated faster response time hence better performance by the database.
2. ***Throughput***is the rate at which a database processes a given number of operations within a given time period. A higher throughput value indicates the database has a better performance when handling workloads.
3. ***Runtime*** is the total time that is taken to execute a particular workload/ a set of operations. When the runtime is longer a database is able to perform a more comprehensive performance evaluation and stress testing of the database.

#### WORKLOAD A

##### Runtime

From the graph below, the runtime of MongoDB is lower than MySQL under all comparative parameters. In uniform distribution the runtime values are higher for both databases than for Zipfian distribution. The uniform distribution allows the databases to perform more comprehensive evaluations and stress testing. The runtime for the proportion 50/50 is higher than 70/30 for both distributions for both databases.

**Figure 2: Graph of RunTime comparison of MySQL vs MongoDB against other comparative parameters**

##### Throughput

Under Zipfian distribution MongoDB had higher values of throughput compared to MySQL. The values of throughput increase as the record count increases. While for MySQL the values are very low. The read/update value 50/50 has higher values of throughput compared to 70/30 proportion. Therefore, MongoDB had better performance when handling workload A compared to MySQL. The throughput is increasing linearly indicating good scalability for MongoDB.

**Figure 3: Graph of Throughput comparison of MySQL vs MongoDB against other comparative parameters**

##### Average Latency

MongoDB has lower average latency in both uniform and Zipfian distribution as compared to MySQL. The uniform distribution has lower average latency compared to Zipfian in MongoDB and the latency decreases with increase in read count. MongoDB has lower values of average latency, this indicated faster response time hence better performance by the database, hence a scalable system.

**Figure 4: Graph of Average Latency comparison of MySQL vs MongoDB against other comparative parameters**

#### WORKLOAD B

##### Runtime

From the graph below, the runtime of MongoDB is lower than MySQL under all comparative parameters. In uniform distribution the runtime values are higher for MySQL when the read count is 100,000 for both 95/5 and 70/30.

**Figure 5:Graph of RunTime comparison of MySQL vs MongoDB against other comparative parameters**

##### Throughput

Under Zipfian distribution MongoDB had higher values of throughput compared to MySQL. The values of throughput increase as the record count increases. While for MySQL the values are very low. The read/update value fluctuate between the two proportions. Therefore, MongoDB had better performance when handling workload B compared to MySQL. The throughput is increasing linearly indicating good scalability for MongoDB.

**Figure 6:Graph of Throughput comparison of MySQL vs MongoDB against other comparative parameters**

##### Average Latency

MongoDB has lower average latency in both uniform and Zipfian distribution as compared to MySQL. The uniform distribution has lower average latency compared to Zipfian in MongoDB and the latency decreases with increase in read count. MongoDB has lower values of average latency, this indicated faster response time hence better performance by the database, hence a scalable system

**Figure 7:Graph of Average Latency comparison of MySQL vs MongoDB against other comparative parameters**

#### WORKLOAD C

##### Runtime

From the graph below, the runtime of MongoDB is lower than MySQL under all comparative parameters. In uniform distribution the runtime values are higher for MySQL when the read count is 100,000 for both 100/0 and 70/30.

**Figure 8:Graph of RunTime comparison of MySQL vs MongoDB against other comparative parameters**

##### Throughput

Under Zipfian distribution MongoDB had higher values of throughput compared to MySQL. The values of throughput increase as the record count increases. While for MySQL the values were very low. The read/update value fluctuate between the two proportions. Therefore, MongoDB had better performance when handling workload C compared to MySQL. The throughput was increasing linearly indicating good scalability for MongoDB.

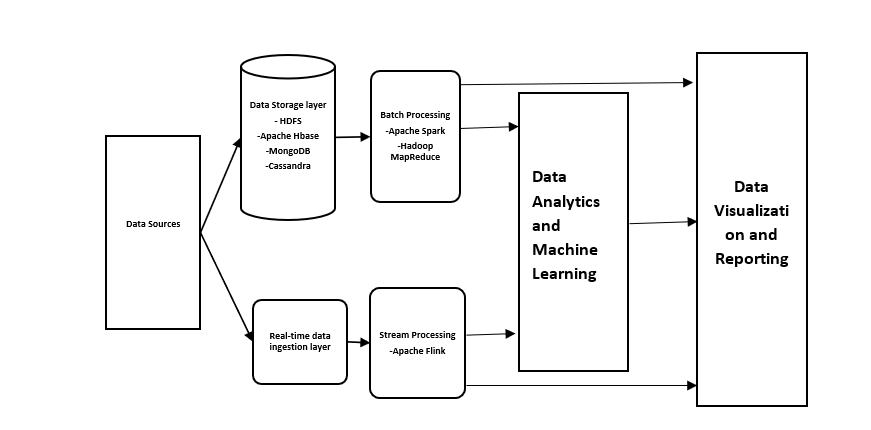
**Figure 9:Graph of Throughput comparison of MySQL vs MongoDB against other comparative parameters**

##### Average Latency

MongoDB had lower average latency in both uniform and Zipfian distribution as compared to MySQL. The uniform distribution had lower average latency compared to Zipfian in MongoDB and the latency decreased with increase in read count. MongoDB had lower values of average latency, this indicated faster response time hence better performance by the database, hence a scalable system

**Figure 10:Graph of Average Latency comparison of MySQL vs MongoDB against other comparative parameters**

### THE ARCHITECTURE FOR PROCESSING BIG DATA

****

**Figure 11:The architecture for Processing Big Data**

**Data sources**

The data sources could include: Real time data, Application generated data, Static (web server log files) and Application data (connection data). Social media data is a good source of data. The data could be structured, unstructured or both. The data can then be ingested either as batch or real-time.

**Data Storage Layer**

These is the layer where data is ingested. The data can be structured or unstructured, hence the reason the data storages are either NoSQL or Relational Databases. Structured data is stored in relational databases such as sparkSQL. While unstructured data is stored in NoSQL Databases such as HBase, MongoDB, Cassandra etc.

**Real Time data ingestion layer**

Allows for categorizing data so that the data can be smoothly transitioned into a deeper layer of the environment where the data is stored in real time for stream processing.

**Batch processing Layer**

This involves running long batch jobs to ensure the data is useful for analysis. The source files are read, processed and the output written to a new file. Hadoop is good for this.

**Data Analytics and Machine Learning store**

The processed data can now be utilised for analysis and machine learning. E.g Apache Hive

**Data Visualization and Reporting**

After analysis has been done, the findings can be visualized in dashboards etc or summarized in reports.

## ADVANCED DATA ANALYTICS

This involved performing EDA, Data Preparation, Sentiment analysis and timeseries forecasting

### EXPLORATORY DATA ANALYSIS

The primary aim of the EDA was to examine the data’s distribution, outliers, and any anomalies that would be used to generate specific hypotheses for testing and to assist in pattern recognition. (*Secondary Analysis of Electronic Health Records*, 2016). EDA was performed in three stages, exploring the entire data. Exploring the date variable and then exploring the text variable.

1. In the whole dataset wo main EDA processes that were beneficial for this data were: -

a) Checking for any missing data:Missing data have major effects on conclusions made from the data. Therefore, identifying them is crucial for handling problems they cause.(Dettori and Norvell, 2018). The data had 80 missing dates.

Checking for duplicates:Some impacts of duplicates include; the generation of erroneous observations, generation of more repeated observations, loss of observations, and incorrect statistics. (Cheng, no date). The duplicates were determined by checking three variables together, that is, user, date and text. A total of 3738 duplicates were found.

1. In the date variable EDA included, checking the start and last date, and missing dates. The start date was 7/4/2009 and the last date was 25/4/2009. The data had a total of 774,363 unique dates.
2. In the text variable EDA included: -

* Counting number words in the text. The largest text had 110 words.
* Counting the number of characters in the text. The maximum number of characters is 374.
* The average word count in the text
* The number of stop words. The maximum number of stop words is 25
* The number of hashtags. The maximum number is 24
* The number of @. The maximum number is 13.
* The number of upper cases. The maximum number is 40

This EDA guided the data cleaning and preprocessing of the text variable.

### DATA PREPARATION AND CLEANING

Data cleaning organizes data, making it ready for analysis. It helps identify and remove inconsistencies and errors in data, improving the data quality.(Ridzuan and Wan Zainon, 2019).

The Data cleaning steps included:

**Step 1: Handling missing data:** Handling missing data ensured the data was reliable, meaningful in analysis, and unbiased(Kang, 2013). The listwise deletion method was used to handle missing data in both datasets. Other techniques would alter the shape of the distribution.(Kang, 2013). The missing dates were handled using time series technique for handling missing dates called linear interpolation. interpolation was done by considering daily data. In linear interpolation method values between two known data points are estimated. The method was preferred over the other techniques because it assumes there is a relationship between a range of data points, which is the case in tweets.(*Preprocessing and Data Exploration for Time Series — Handling Missing Values | by Data Science Wizards | Medium*, 2023)

**Step 2: Removing features that were not used:** Removing irrelevant features helps overcome the curse of dimensionality and reduce overfitting problems.(Afshar and Usefi, 2022). Variable flag, user and ids were not used in the analysis.

**Step 3: Removing duplicate Observations:** Duplicate observations were dropped because they could result in incorrect statistics. These were 3738 observations.

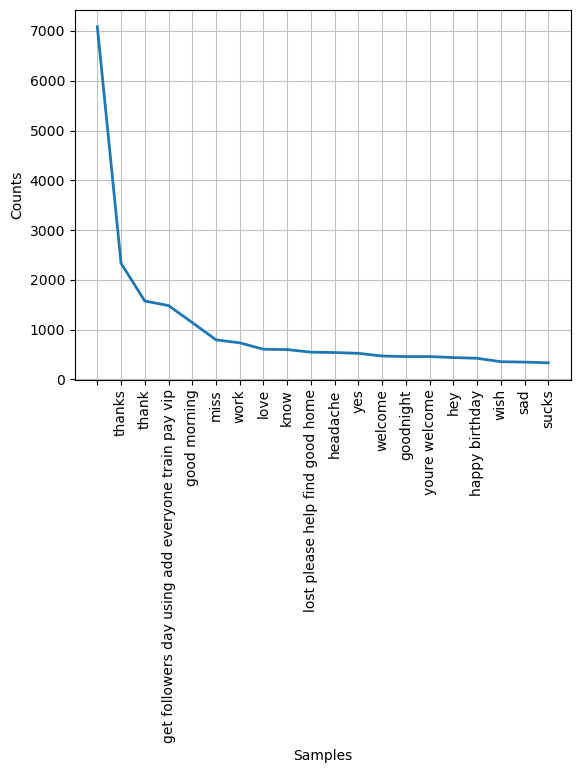
**Step 4: Tweets Processing**

After EDA, the identified problems were addressed during preprocessing. Preprocessing text is important since it helps to remove noise from text and reduce inconsistencies to ensure the data can be used for sentiment analysis of mining text.(Samuels and Mcgonical, 2019). Preprocessing involved this step: -

* **Text normalization** -which is the process of trying to reduce randomness in a text, by trying to make is closer to or even standard.(*Text Normalization for Natural Language Processing (NLP) | by Diego Lopez Yse | Towards Data Science*, 2021). It involved: -
* Remove special characters
* Change the upper cases to lower cases
* Remove numbers/integers
* Remove punctuations
* Remove white space
* Remove URLS/links
* Remove the username
* **Tokenization-** Tokenization is breaking the text into tokens. Tokens could be either words, symbols, phrases, or even the whole sentence Tokenization used was the word tokenization.
* **Remove stop words**- The number of stop words was counted for each text and then removed from the texts.
* **Lemmatization -** is the process of finding the root of a word rather than the stem. (S *et al.*, 2020) . Lemmatization was applied to the comment variable to obtain the root of the word. Stemming was done but the output did not convey any meaningful information, so lemmatization was done because the root of the words was more meaningful. Below Is the process of text processing: -

After text processing a frequency distribution plot was made for the tokenized text. Below is the graph of the frequency distribution.

Figure 12:frequency distribution plot of tokenized text.



### DATA VISUALIZATION OF THE DATE VARIABLE

Figure 13: Graph on the count of each unique day

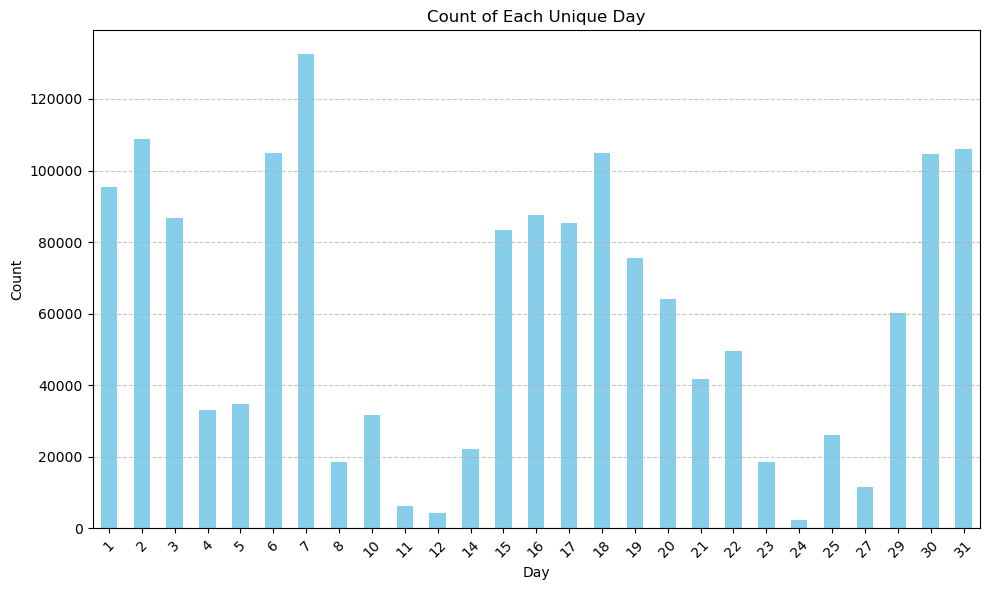


Figure 14:Histogram of the Date variable

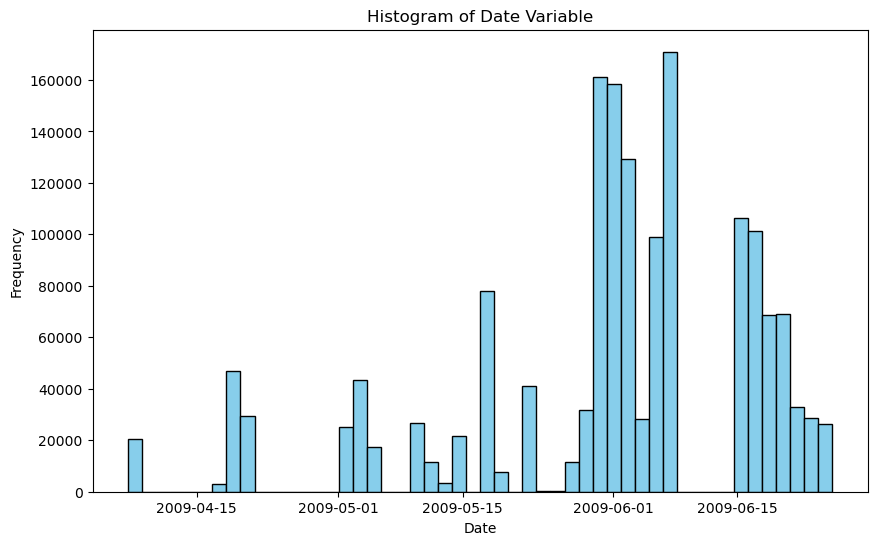
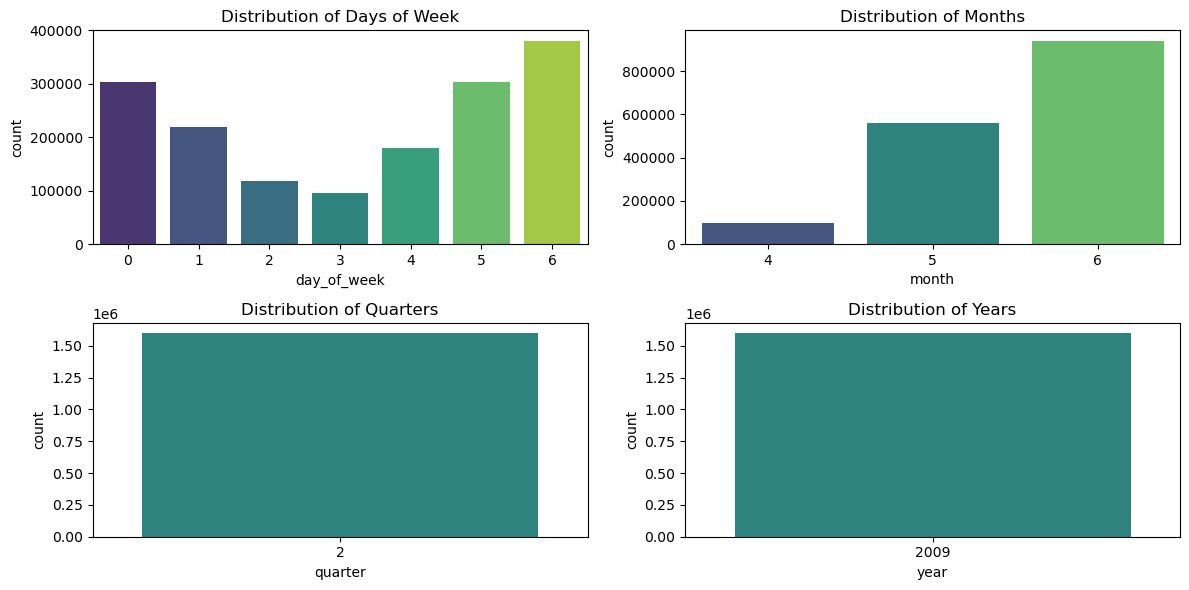


Figure 15: Distribution of the Different time components in the date variable



From the graphs above, the date variable has only 3 months, April, May and June.

### SENTIMENT EXTRACTION AND ANALYSIS

After text processing, the sentiments of the text were extracted. Two methods were explored, text blob and Vader sentiment.

**Text blob sentiment** is a lexicon-based sentiment analyser that has predefined rules which have scores that help to calculate sentence polarity. It returns two results polarity and subjectivity. Polarity score ranges from -1 to 1, where 1 is positive and -1 is negative score.(*Sentiment Analysis with Textblob and Vader in Python*, 2024).Text blob was used because it enables translation of tweets from one language to another.(Aljedaani *et al.*, 2022)

**VADER for Sentiment Analysis:** VADER is a rule and lexicon-based sentiment analysis tool that handles words, slang, emojis, and abbreviations that are normally found in social media. When compared to machine learning algorithms it is much faster and training of the data is not required. It was used for sentiment analysis because it can handle various characters that social media data has. (Pano and Kashef, 2020).

After the sentiments were extracted, the CountVectorizer and TfidfVectorizer were used to convert the text into matrices for training.

**TF-IDF vectorizer**: TF-IDF is a scheme that assigns weights to token frequencies in the form of matrices. (Dogra *et al.*, 2022). In TfidfVectorizer the tokens were converted into a numerical format using TfidfVectorizer. This is a vectorizer that uses the term frequency-inverse document frequency by calculating two matrices and representing the document as vectors for analysis.(Das Sarit Chakraborty Student Member and Member, 2018).

**Count Vectorizer:** was used because it used to convert text into numerical data

**Multinomial Naïve Bayes classification**

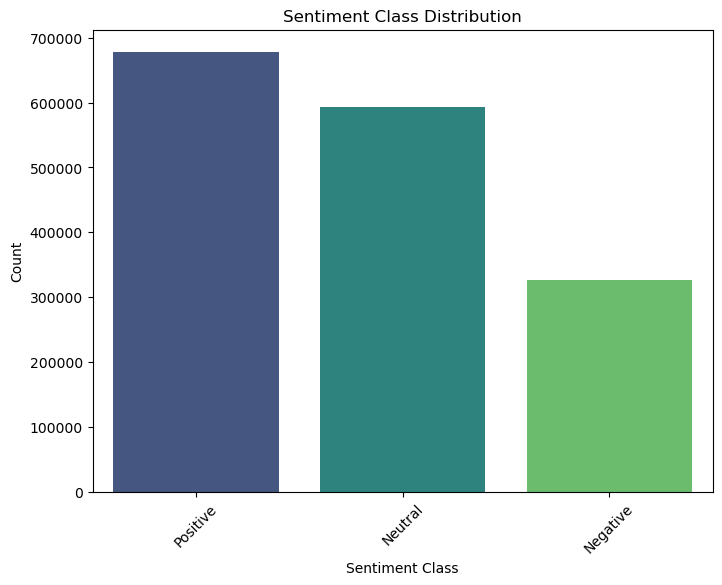
After preprocessing the resulting matrices from both CountVectorizer and TfidfVectorizer were split into X and Y, then split into training and test data. Multinomial Naïve Bayes (MNB) classification model was fit. The multinomial naïve Bayes works with the assumption that the document is a bag of words and takes into account the word frequency and information.(Abbas *et al.*, 2019).

##### Text Classification: The resulting matrices were classified using Multinomial Naïve Bayes classification and the results were evaluated using classification evaluation metric accuracy

|  |  |  |
| --- | --- | --- |
| **Sentiment analysis** | **TfidfVectorizer** | **CountVectorizer** |
| Text blob | 0.82 | 0.83 |
| VADER | 0.74 | 0.77 |

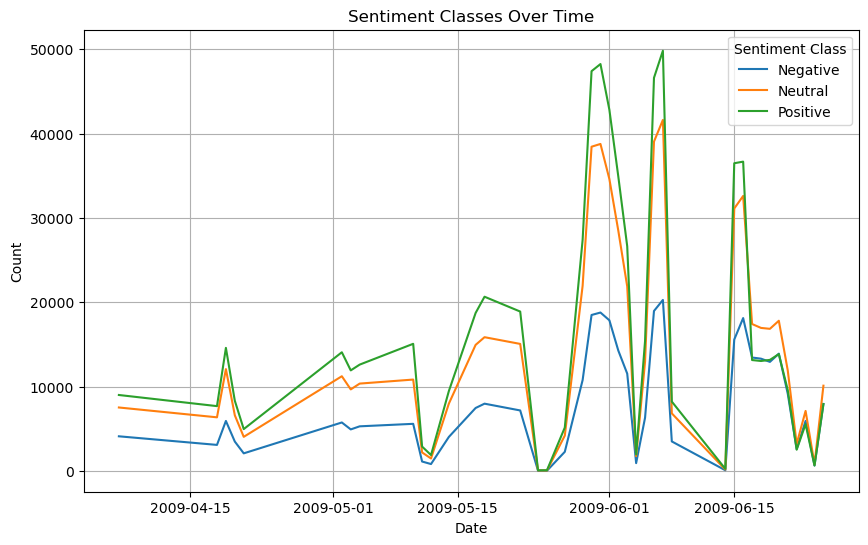
From the table above, text blob has a higher accuracy than Vader for both TfidVectorizer and CountVectorizer. Therefore, the sentiments used for time series forecasting were text blob sentiments.

Figure 16: Bar Graph of the sentiments extracted by text blob sentiment



From the graph above there is class imbalance between the three classes.

The sentiments were also plotted against the date to see the trend.



There was a lot of positive sentiments in June.

### HANDLING CLASS IMBALANCES

Class imbalance is an instance where some classes have more instances than other classes, causing the standard classifiers to be overwhelmed by the large class and ignoring the small class.(Javaheri, Sepehri and Teimourpour, 2013). After sentiment extraction the sentiment variable had class imbalance. The positive sentiments were way more than the negative sentiments. Therefore, the classes had to be balanced. There are various techniques used to balance classes. The balancing technique was selected based on the size of the data. The technique used here was the under-sampling technique, this is because it is appropriate for big datasets.(*What Is Undersampling? | Master’s in Data Science*, 2022).

### TIME SERIES

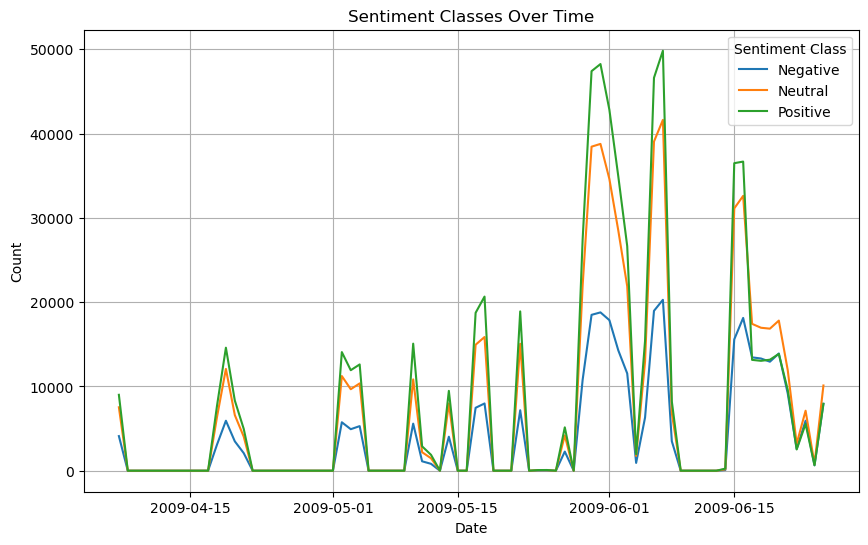
**The time series data analysis took the following steps: -**

* Handling missing dates
* Aggregate the data by date to get daily sentiment scores.
* Prepare the data for the forecasting models.
* Train and predict using LSTM.
* Train and predict using ARIMA and SARIMA.

**Handling missing dates**

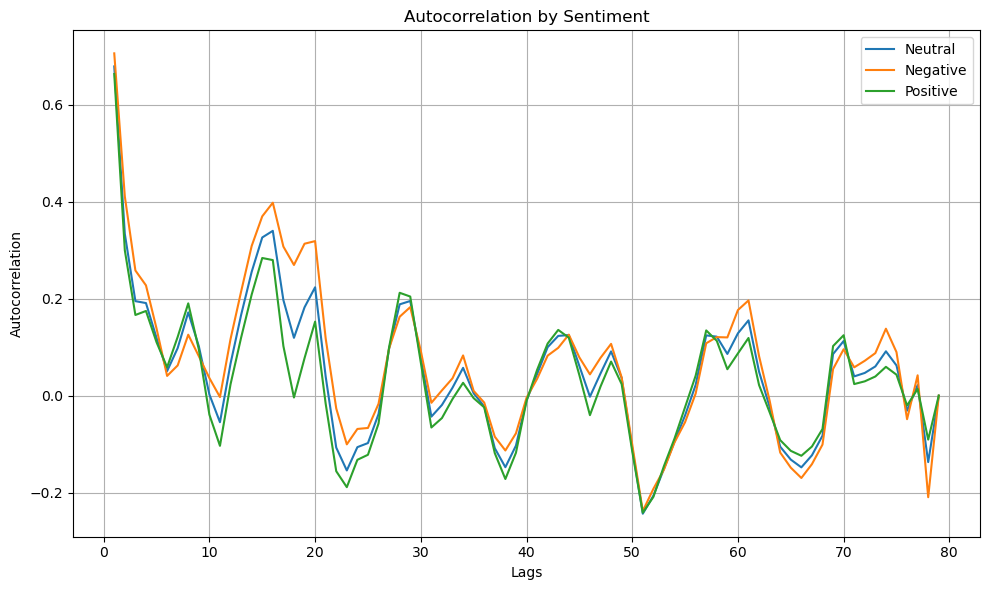
The data had missing dates. These dates were handled using the linear interpolation technique. After linear interpolation the missing values in the sentiments were filled using the modal class of each week.

After handling all the missing data, the daily sentiment counts for each sentiment was computed.

Figure 17: Plot of the sentiments after handling missing sentiments due to handling dates using interpolation

#### AUTOCORRELATION

This is the degree of similarity between a given times series and a lagged version of itself over successive time intervals. It measures the relationship between a current value of a variable and its past value. Auto correlation plots were plotted to determine whether there was a relationship between the sentiment’s current values and its past values. Below is the plot of each sentiment.

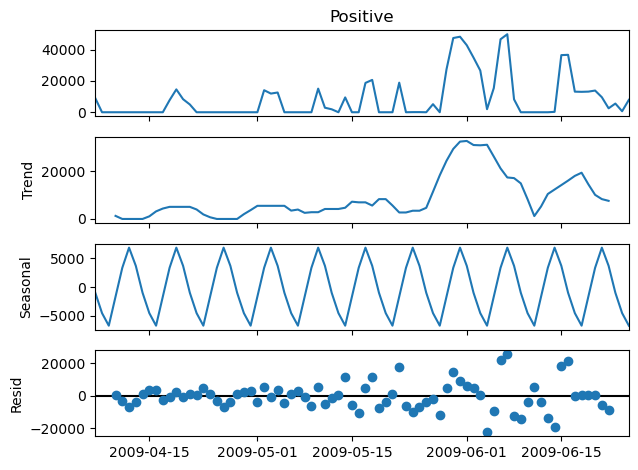
Figure 18:Plot ACF for each sentiment category

From the graph the autocorrelation decreases as the lags increase. This indicates that the influence of past sentiment weakens as we move further back in time.(*Autocorrelation: What It Is, How It Works, Tests*, 2023)

#### DECOMPOSITION

Decomposition in time series involves, breaking down a time series data into individual components such as trend, seasonality and residual. By doing so one is able to establish the patterns and variations that exist in the data. The additive model was used, this is because the seasonal fluctuations were consistent in their amplitude across the data and in the datasets there were sentiments that were neutral which were labelled 0, so multiplying them will result in zero for the whole dataset. Below are the decomposition plots for the three sentiments.(*Interpret all statistics and graphs for Decomposition - Minitab*, no date; ‘Interpret all statistics and graphs for Decomposition’, no date)

Figure 19:Decomposition plot for Positive sentiments



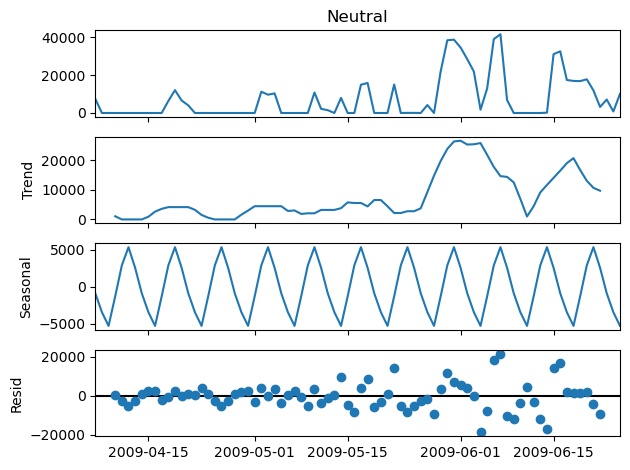
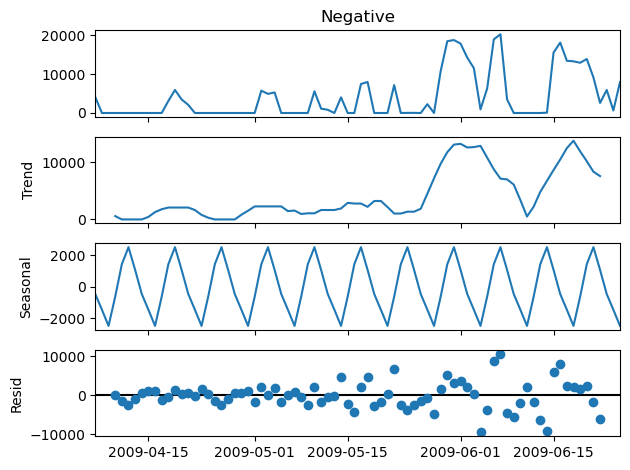


Figure 20:Decomposition plot for Neutral sentiment

Figure 21:Decomposition plot for Negative sentiments



#### DETERMINE THE STATIONARITY OF THE TIME SERIES

A stationary time series is a series where the statistical properties of the data do not depend on time. From the plots above the time series had seasonality and trend.(*time series sationarity - Google Search*, no date).Stationary processes are easy to analyse, hence the need to check for stationarity.

The methods used to determine stationarity, was the ADF statistic, which is a test used to determine whether a time series is stationary or not.

**Using ADF statistic:** for the ADF the hypothesis being tested was

Null: The data is non-stationary (presence of a unit root),

Alternative: The data is stationary

The ADF statistic, p-value and critical values were obtained as follows

Table 2: Table of the ADF test findings

|  |
| --- |
| ADF Statistic: -4.508186637068076  p-value: 0.00019004955919706284  Critical Value (1%): -3.517113604831504  Critical Value (5%): -2.8993754262546574  Critical Value (10%): -2.5869547797501644 |

The p-value is less than 0.05, hence we reject null and conclude the data is stationary.

#### LONG SHORT TERM MEMORY

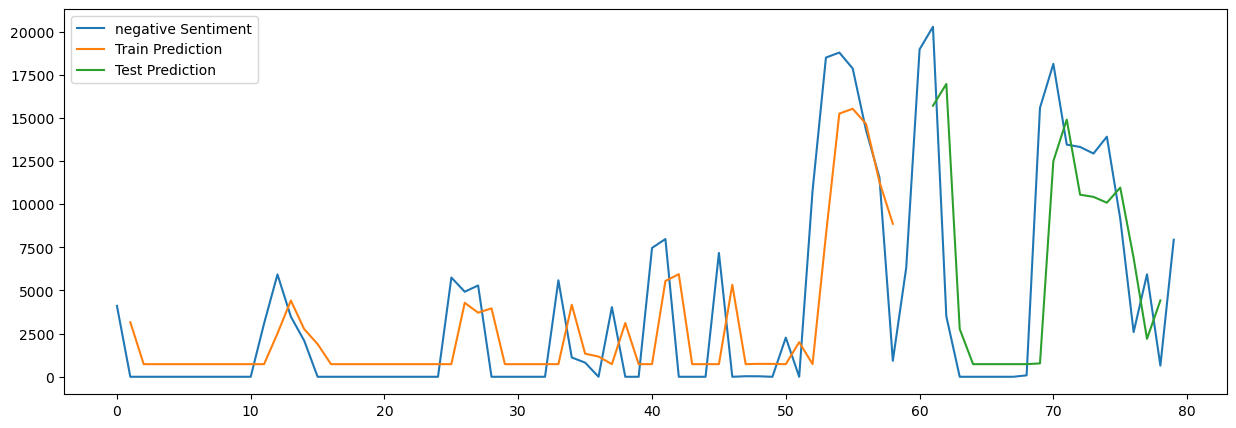
LSTM was utilized so as to model the temporal dependencies in the dataset. The network architecture was constructed using Kera deep learning library. Firstly, a sequential model was initialized, serving as a linear stack of layers. An LSTM layer was then added with 8 Neurons, that was configured to accept input sequences of a defined shape. The input shape was set to (1, look\_back), where look\_back represented the number of timesteps used for predictoon. Lastly a Dense layer was added to the model to produce the final output. produce the final output.

The model was compiled using the mean squared error loss function, which quantifies the difference between the predicted and actual values, and the Adam optimizer, known for its efficiency in training deep neural networks. To train the model, the fit method was employed, specifying the number of epochs, batch size, and verbosity level for monitoring the training progress. This iterative process allowed the model to learn from the training data and adjust its parameters to minimize the prediction error, ultimately resulting in a trained LSTM network capable of making accurate predictions on unseen data.

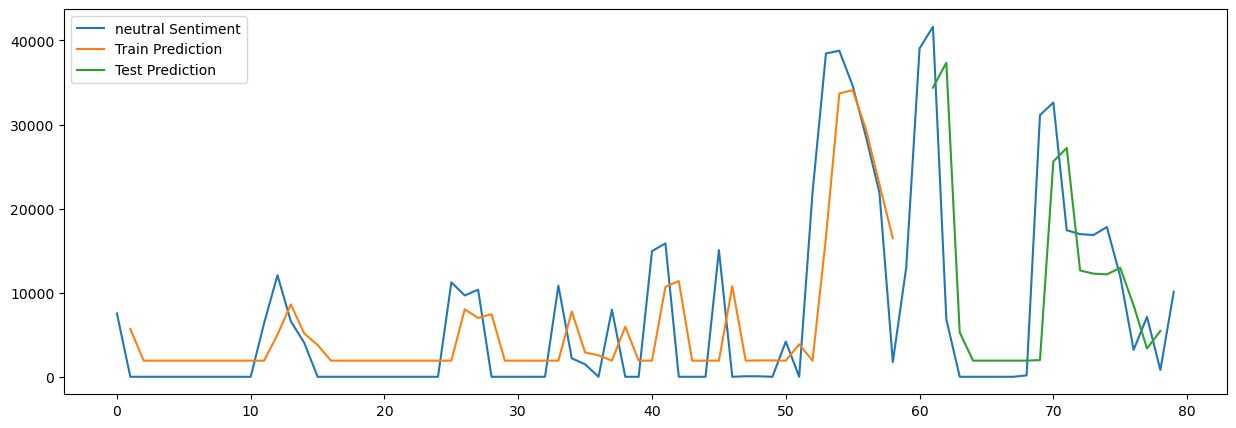
After implementation the model was evaluated using MSE, Mean squared error, which is calculated by squaring the difference between the predicted and actual value and averaging the difference. As the error increases there is an exponential increase in MSE. A model is considered good if the MSE value is closer to zero. MSE is good for weeding out outliers with large errors from the model, by putting more weight on them (M Padhma, 2023).

The best loss and the epoch with the best loss were calculated, see table below:

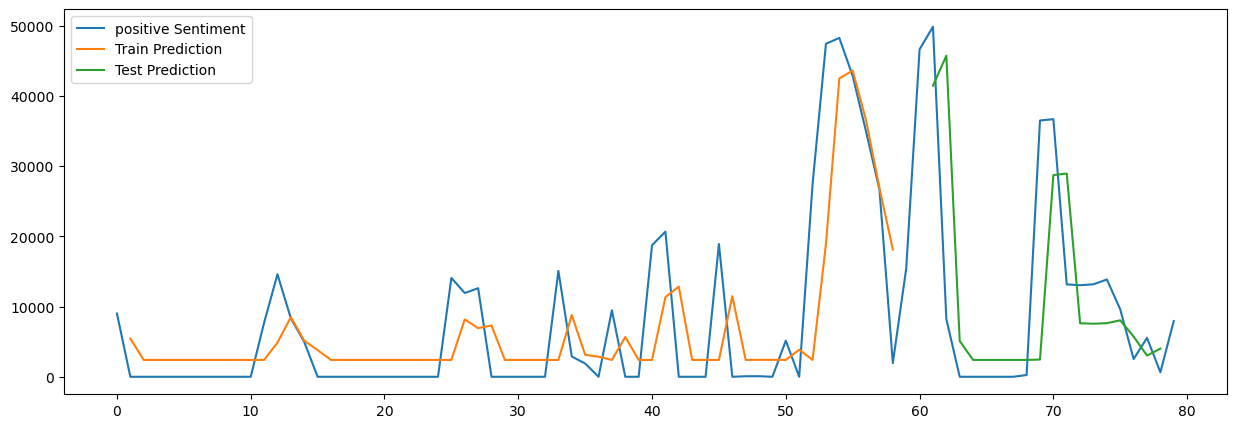
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sentiment** | **Train Score**  **RMSE** | **Test Score**  **RMSE** | **Epoch with the best loss** | **Best Loss** |
| Negative | 3207.73 | 5497.72 | 95 | 0.02523665875196457 |
| Neutral | 6342.04 | 10951.94 | 97 | 0.02339048869907856 |
| Positive | 7805.52 | 13186.76 | 89 | 0.02478545345366001 |

**Figure 22: Plot of the prediction vs baseline negative sentiments**

**Figure 23: The prediction vs Baseline Neutral sentiments**

****

**Figure 24: The prediction vs Positive sentiments**

****

#### AUTOREGRESSIVE INTEGRATING MOVING AVERAGE MODEL

Is a model that predicts future values based on past values. It makes use of lagging moving average to smooth time series data. The model has 3 parameters, p, d and q. Where,

 *p*: the number of lag observations in the model, also known as the lag order.

 *d*: the number of times the raw observations are differenced; also known as the degree of differencing.

 *q*: the size of the moving average window, also known as the order of the moving average. A range of values to determine the best ARIMA for each sentiment. Below is are the findings. The predicted and baseline values were also plotted.

|  |  |  |
| --- | --- | --- |
| **Sentiment** | **Train Score RMSE** | **Test Score RMSE** |
| Negative | 3799.92 | 6808.72 |
| Neutral | 7599.85 | 12905.76 |
| Positive | 9394.95 | 15831.49 |

Figure 25:ARIMA Negative predictions vs baseline

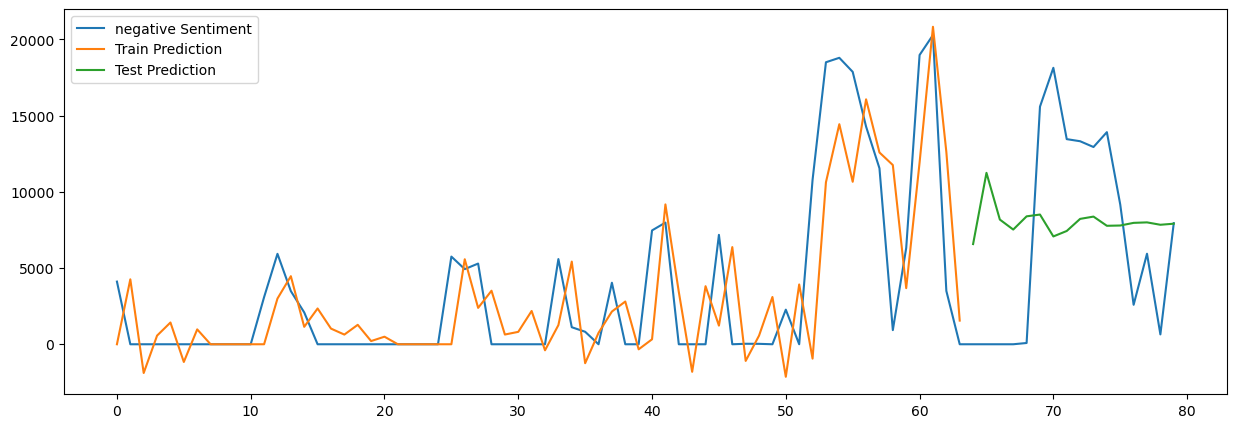


Figure 26:ARIMA Neutral sentiments prediction vs baseline

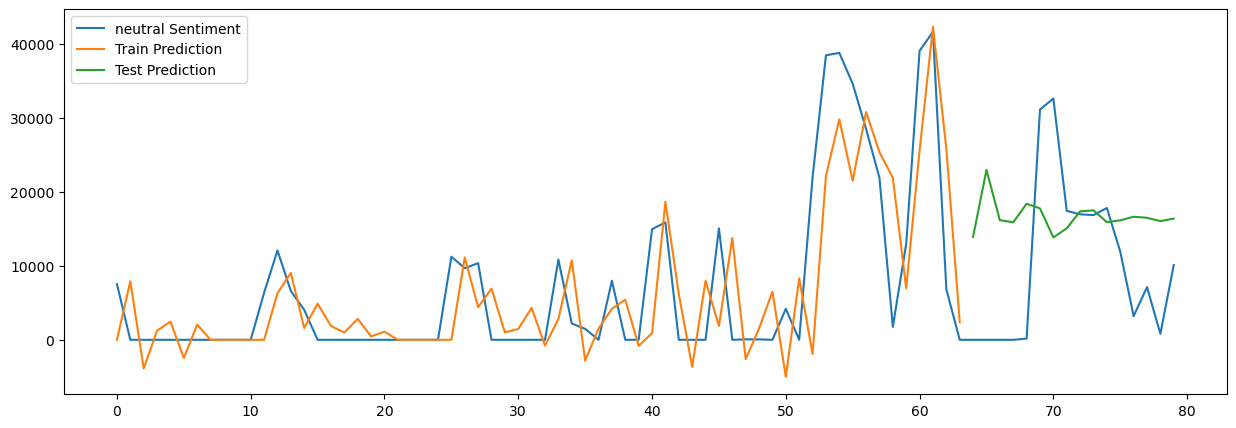
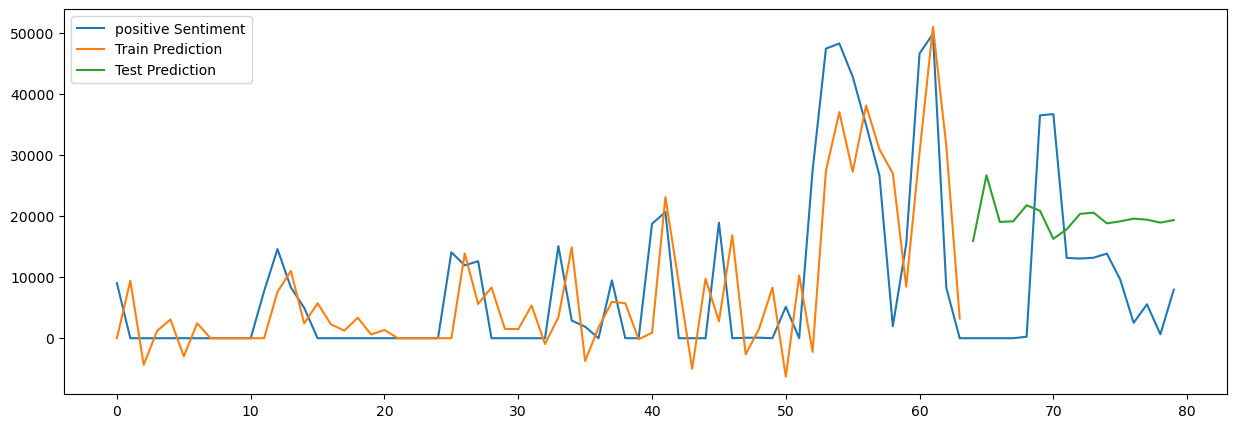


Figure 27: ARIMA Positive sentiments prediction vs baseline



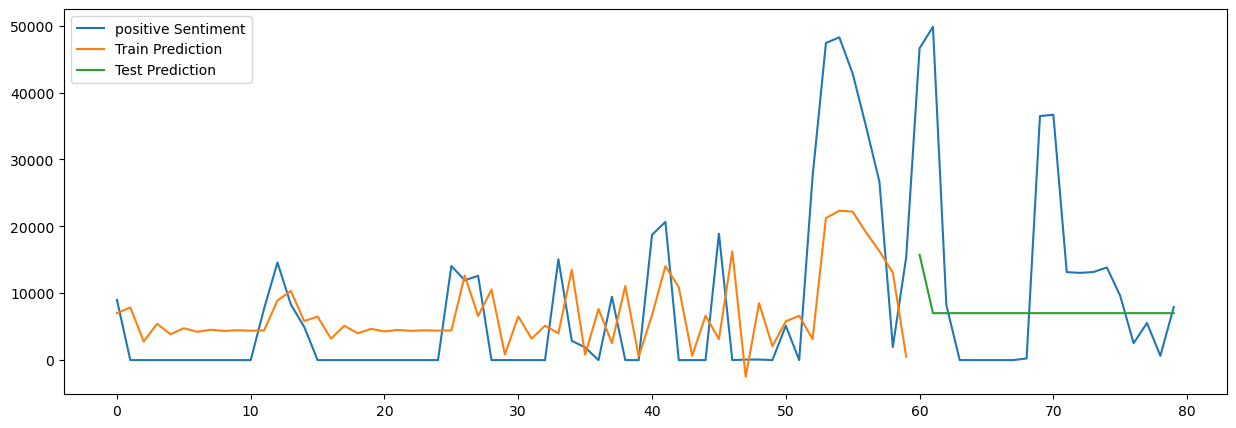
#### HYPERPARAMETER TUNING OF ARIMA MODEL

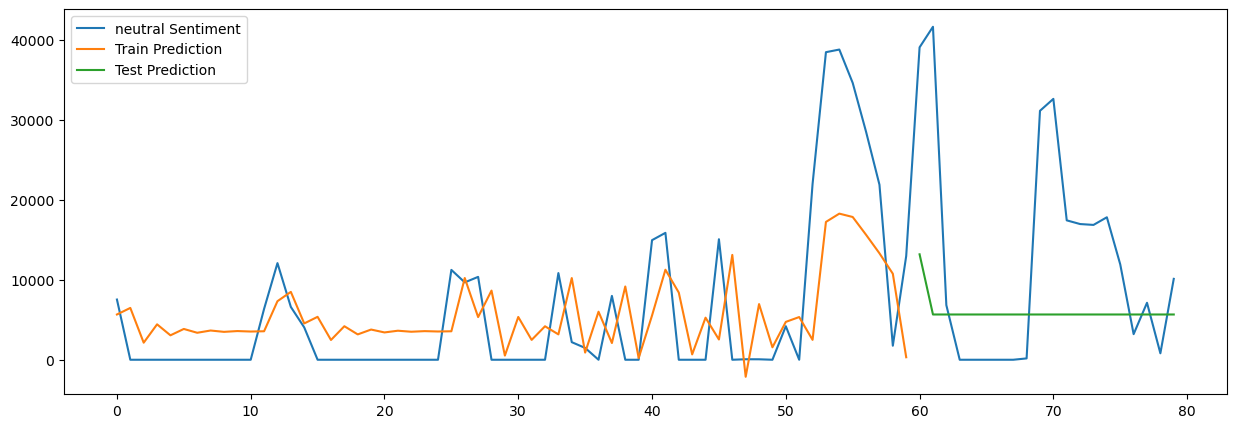
Hyperparameter tuning was done by creating a range of p, q and d values, to test which was the best parameter. The best parameters were:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Best ARIMA | MSE | TRAIN RMSE | TEST RMSE |
| Neutral | (0,0,1) | 123803971.511 | 7584.74 | 14421.35 |
| Positive | (0,0,1) | 158558380.004 | 9444.69 | 15901.63 |
| Negative | (1,0,3) | 36414394.275 | 3169.83 | 7417.46 |

The ARIMA with the best parameters were then plotted, as shown below:-

Figure 28: Positive sentiments prediction vs baseline values





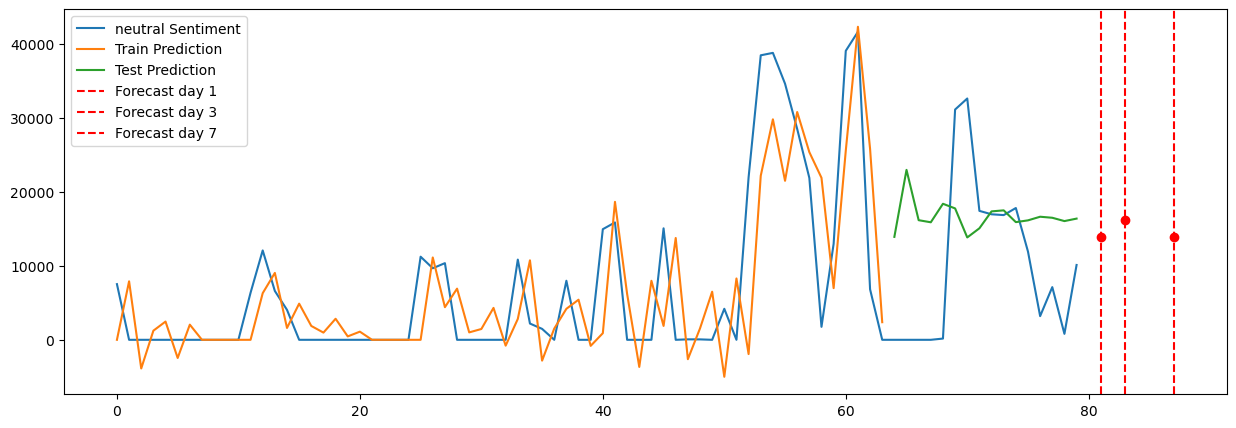
#### FORECASTING DAY 1,3,7 USING ARIMA

The best parameters obtained above were then used to forecast sentiments of Day 1,3, and 7.The forecast below are the sentiment counts of each sentiment

Here are the findings

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | Train score RMSE | Test score RMSE | forecast |
| Neutral | Forecast for day 1 | 7599.85 | 12905.76 | 13912.247660229728 |
| Forecast for day 3 | 16164.82403844619 |
| Forecast for day 7 | 13833.7562252716 |
| Negative | Forecast for day 1 | 3799.92 | 6808.72 | 6570.22894081378 |
| Forecast for day 3 | 8176.679583089519 |
| Forecast for day 7 | 7070.9199326744365 |
| Positive | Forecast for day 1 | 9394.95 | 15831.49 | 15917.206705484197 |
| Forecast for day 3 | 19033.070749851275 |
| Forecast for day 7 | 16263.871082582078 |

Figure 29:Forecasting Day 1,3, and 7 Neutral sentiments



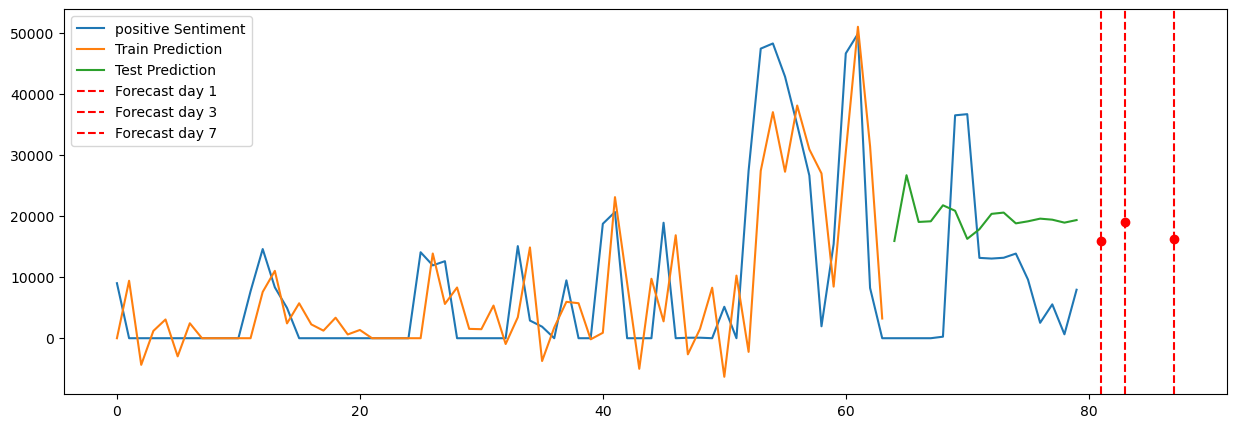


Figure 30:Forecasting Day 1,3 and 7 Positive sentiments

### 

Figure 31:Forecasting Day 1,3 and 7 Negative sentiments

#### Use Tufte Principle- DASHBOARD

According to Edward Tufte, there are 6 principles a visualization should strive toward, that is comparison rather than description, high resolution and utilization of classic designs, Content focus, concepts proven by time, and integrity. (Globus, 2014). The tufte principle was utilized to create a dashboard that communicates findings of the sentiments, ARIMA and LSTM prediction vs baseline values. The dashboard showed a forecasting for ARIMA Model.

# References

Abbas, M. *et al.* (2019) ‘Multinomial Naive Bayes Classification Model for Sentiment Analysis’, *IJCSNS International Journal of Computer Science and Network Security*, 19(3), p. 62. Available at: https://doi.org/10.13140/RG.2.2.30021.40169.

Afshar, M. and Usefi, H. (2022) ‘Optimizing feature selection methods by removing irrelevant features using sparse least squares’, *Expert Systems with Applications*, 200, p. 116928. Available at: https://doi.org/https://doi.org/10.1016/j.eswa.2022.116928.

Aljedaani, W. *et al.* (2022) ‘Sentiment analysis on Twitter data integrating TextBlob and deep learning models: The case of US airline industry’, *Knowledge-Based Systems*, 255, p. 109780. Available at: https://doi.org/10.1016/J.KNOSYS.2022.109780.

*Autocorrelation: What It Is, How It Works, Tests* (2023). Available at: https://www.investopedia.com/terms/a/autocorrelation.asp (Accessed: 19 May 2024).

Bazzaz Abkenar, S. *et al.* (2021) ‘Available online 14’, *Telematics and Informatics*, 57, pp. 736–5853. Available at: https://doi.org/10.1016/j.tele.2020.101517.

Cheng, A.M. (no date) *The Causes, Impact and Detection of Duplicate Observations*.

Dettori, J.R. and Norvell, D.C. (2018) ‘The Anatomy of Data’, *Global Spine Journal*. SAGE Publications Ltd, pp. 311–313. Available at: https://doi.org/10.1177/2192568217746998.

Dogra, V. *et al.* (2022) ‘A Complete Process of Text Classification System Using State-of-the-Art NLP Models’, *Computational Intelligence and Neuroscience*. Hindawi Limited. Available at: https://doi.org/10.1155/2022/1883698.

Gaikwad, M.R. and Goje, A.C. (2015) *A Study of YCSB-tool for measuring a performance of NOSQL databases*, *Journal of Engineering Technology and Computer Research*. Available at: www.ijetcr.org.

Globus, A. (2014) *Principles of Information Display for Visualization Practitioners Principles of Information Display for Visualization Practitioners Principles of Information Display for Visualization Practitioners*. Available at: https://www.researchgate.net/publication/24285628.

‘Interpret all statistics and graphs for Decomposition’ (no date).

*Interpret all statistics and graphs for Decomposition - Minitab* (no date). Available at: https://support.minitab.com/en-us/minitab/help-and-how-to/statistical-modeling/time-series/how-to/decomposition/interpret-the-results/all-statistics-and-graphs/ (Accessed: 19 May 2024).

Javaheri, S.H., Sepehri, M.M. and Teimourpour, B. (2013) ‘Response Modeling in Direct Marketing. A Data Mining-Based Approach for Target Selection.’, *Data Mining Applications with R*, pp. 153–180. Available at: https://doi.org/10.1016/B978-0-12-411511-8.00006-2.

Jianqiang, Z. and Xiaolin, G. (2017) ‘Comparison research on text pre-processing methods on twitter sentiment analysis’, *IEEE Access*, 5, pp. 2870–2879. Available at: https://doi.org/10.1109/ACCESS.2017.2672677.

Kang, H. (2013a) ‘The prevention and handling of the missing data’, *Korean Journal of Anesthesiology*, pp. 402–406. Available at: https://doi.org/10.4097/kjae.2013.64.5.402.

Kang, H. (2013b) ‘The prevention and handling of the missing data’, *Korean Journal of Anesthesiology*, pp. 402–406. Available at: https://doi.org/10.4097/kjae.2013.64.5.402.

Krishnan, H. and Elayidom, M.S. (2016) ‘MongoDB – a comparison with NoSQL databases’, *International Journal of Scientific & Engineering Research*, 7(5). Available at: http://www.ijser.org.

M Padhma (2023) *Evaluation Metric for Regression Models - Analytics Vidhya*. Available at: https://www.analyticsvidhya.com/blog/2021/10/evaluation-metric-for-regression-models/ (Accessed: 4 April 2024).

Mehdipour, F., Noori, H. and Javadi, B. (2016) ‘Energy-Efficient Big Data Analytics in Datacenters’, *Advances in Computers*, 100, pp. 59–101. Available at: https://doi.org/10.1016/bs.adcom.2015.10.002.

Pano, T. and Kashef, R. (2020) ‘A complete vader-based sentiment analysis of bitcoin (BTC) tweets during the ERA of COVID-19’, *Big Data and Cognitive Computing*, 4(4), pp. 1–17. Available at: https://doi.org/10.3390/bdcc4040033.

*Preprocessing and Data Exploration for Time Series — Handling Missing Values | by Data Science Wizards | Medium* (2023). Available at: https://medium.com/@datasciencewizards/preprocessing-and-data-exploration-for-time-series-handling-missing-values-e5c507f6c71c (Accessed: 16 May 2024).

Ridzuan, F. and Wan Zainon, W.M.N. (2019) ‘A review on data cleansing methods for big data’, in *Procedia Computer Science*. Elsevier B.V., pp. 731–738. Available at: https://doi.org/10.1016/j.procs.2019.11.177.

S, S.B. *et al.* (2020) *An Interpretation of Lemmatization and Stemming in Natural Language Processing*. Available at: https://www.researchgate.net/publication/348306833.

Samuels, A. and Mcgonical, J. (2019) *Sentiment Analysis of News Articles: A Lexicon based Approach*. Available at: http://mlg.ucd.ie/datasets/bbc.html.

Das Sarit Chakraborty Student Member, B. and Member, I. (2018) *An Improved Text Sentiment Classification Model Using TF-IDF and Next Word Negation*.

*Secondary Analysis of Electronic Health Records* (2016) *Secondary Analysis of Electronic Health Records*. Springer International Publishing. Available at: https://doi.org/10.1007/978-3-319-43742-2.

*Sentiment Analysis with Textblob and Vader in Python* (2024). Available at: https://www.analyticsvidhya.com/blog/2021/10/sentiment-analysis-with-textblob-and-vader/ (Accessed: 16 May 2024).

Shaikh, E. *et al.* (2019) ‘Apache Spark: A Big Data Processing Engine’, in *2019 2nd IEEE Middle East and North Africa COMMunications Conference, MENACOMM 2019*. Institute of Electrical and Electronics Engineers Inc. Available at: https://doi.org/10.1109/MENACOMM46666.2019.8988541.

Strohbach, M. *et al.* (2016) ‘Big data storage’, in *New Horizons for a Data-Driven Economy: A Roadmap for Usage and Exploitation of Big Data in Europe*. Springer International Publishing, pp. 119–141. Available at: https://doi.org/10.1007/978-3-319-21569-3\_7.

*Text Normalization for Natural Language Processing (NLP) | by Diego Lopez Yse | Towards Data Science* (2021). Available at: https://towardsdatascience.com/text-normalization-for-natural-language-processing-nlp-70a314bfa646 (Accessed: 16 May 2024).

*time series sationarity - Google Search* (no date). Available at: https://www.google.com/search?q=time+series+sationarity&oq=time+series+sationarity&gs\_lcrp=EgZjaHJvbWUyBggAEEUYOTIJCAEQABgNGIAEMgkIAhAAGA0YgAQyCQgDEAAYDRiABDIJCAQQABgNGIAEMggIBRAAGBYYHjIICAYQABgWGB4yCAgHEAAYFhgeMggICBAAGBYYHjIICAkQABgWGB7SAQg0NjUzajBqN6gCALACAA&sourceid=chrome&ie=UTF-8 (Accessed: 19 May 2024).

*What Is Undersampling? | Master’s in Data Science* (2022). Available at: https://www.mastersindatascience.org/learning/statistics-data-science/undersampling/ (Accessed: 18 May 2024).