CCT College Dublin

Assessment Cover Page

Module Title:	Advanced Data Analytics Big Data Storage and Processing
Assessment Title:	Integrated CA2 Sem 2 MSc in Data Analytics
Lecturer Name:	David McQuaid Muhammad Iqbal
Student Full Name:	Usukhbayar Tsendgombo
Student Number:	2022418
Assessment Due Date:	26th May 2023
Date of Submission:	26th May 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.

Group ID - MSc in Data Analytics

Author: Usukhbayar Tsendgombo
E-mail: 2022418@student.cct.ie
Student ID: 2022418
https://github.com/Usukhbayar95/Comprehensive-analysis.git

Abstract

This comprehensive analysis assignment has been tasked the data storage and processing of big data using advanced data analytics techniques. And twitter sentiment analysis and perform forecast of the sentiment at specific time period. The analysis plan was to storage and process the dataset, sentiment analyse and time forecasting.

I have used a random twitter dataset from Kaggle.com and loaded it into MySQL database. Following this, I have loaded the twitter CSV file into a data "table" in my MySQL. After this, transferred data from the MySQL database to a PySpark Dataframe and prepared to the machine learning models and natural language processing.

Sentiment analyse has done by Deep Learning Keras model and time series forecasting with Prophet model and LSTM (Long Short-Term Memory).

List of Figures

Figure 1 - Comparative analysis by YCSB (MySQL and MongoDB) 8
Figure 2 - The databases of MySQL 9
Figure 3 - The Table statement of MySQL database 9
Figure 4 - Sentiment values 12
Figure 5 - Summary of the model 13
Figure 6 - Confusion matrix 14
Figure 7 - Plot of Loss and Accuracy 14
Figure 8 - Plot of Loss and Accuracy 15
Figure 9 - Plot of Sentiment Polarity 15
Figure 10 - Plots of Sentiment Polarity by time period 16
Figure 11 - Plots of the components of the model 16
Figure 12 - Scatter plot of actual values with forecasted values 17
Figure 13 - Summary of the sequential model 17
Figure 14 - Plot of sentiment polarity values from the forecast and the actual values 18
Figure 15 – Dashboard of Daily tweet polarity 18

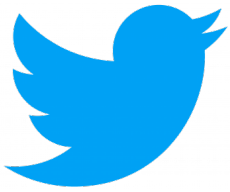
Report plan

The report will be divided into following phases:

- Phase 1: Data storing and processing (MySQL, PySpark, Pandas)
- Phase 2: Data preparation (cleaning, formatting, handling missing value…etc)
- Phase 3: Sentiment analysis ()
- Phase 4: Time series forecasting (Prophet, LSTM)

Twitter

Twitter is an online news and social networking site where people communicate in short messages called tweets. On Twitter users post texts, photos and videos known as "tweets". Registered users can tweet, like, "retweet" tweets, and direct message (DM) other registered users, while unregistered users only have the ability to view public tweets. Users interact with Twitter through browser or mobile frontend software, or programmatically via its APIs. Sentiment analysis in Twitter is a field that has recently attracted research interest. Twitter is one of the most popular microblog platforms on which users can publish their thoughts, emotions and opinions. Sentiment analysis in Twitter tackles the problem of analyzing the tweets in terms of the opinion they express.



With the boom in Internet techniques and computer science, a variety of data in unstructured or semi-unstructured formats have emerged and accumulated, forming *big data* and describing the world from different perspectives. Even without a uniform definition yet, big data have generally been considered to be characterized by the $5V$, i.e., *Volume*, *Variety*, *Velocity*, *Value* and *Ve- racity*. Over the past decades, big data have been introduced into diverse research fields, bringing encouraging innovations to the associated theories and technologies. On the one hand, informative big data have provided new information and knowledge, supporting a new or better understanding for the targeted issues and thereby challenging and even reshaping the basic theories that were based on traditional data. On the other hand, in a different (unstructured or semi-unstructured) format, these big data have called for a substantial renovation of the processing and analysis techniques and even given birth to new methodologies, particularly in the field of computer science.

(Ling Tang, Jieyi Li, Hongchuan Du, Ling Li, Jun Wu, Shouyang Wang. (2021) Big Data in Forecasting Research: A Literature Review)

Architecture of Hadoop

Hadoop is a framework for storing data on large clusters of everyday computer hardware that is affordable and easily available, and running applications against that data. Using networks of affordable compute resources to acquire business insight is the key value proposition of Hadoop. Hadoop clusters typically consist of a few master nodes, which control the storage and processing systems in Hadoop, and many slave nodes, which store the entire cluster's data and is also where the data gets processed. Hadoop comes with two main components: a distributed processing framework named MapReduce (which is now supported by a component called YARN) and a distributed file system known as the Hadoop distributed file system, or HDFS. An application that is running on Hadoop gets its work divided among the nodes (machines) in the cluster, and HDFS stores the data that will be processed. A Hadoop cluster can span thousands of machines, where HDFS stores data, and MapReduce jobs do their processing near the data, which keeps I/O costs low. MapReduce is extremely flexible, and enables the development of a wide variety of applications.

MLA 9th Edition (Modern Language Assoc.)
(Inge, Roger, and Jan Leif. Machine Learning : Advances in Research and Applications. Nova Science Publishers, Inc, 201.)

MySQL

MySQL Relational Database is an assemblage of relational data structured or organized in tables, columns, and rows, where tables represent the objects, columns represent the fields, and rows represent the records. It is the broadly used relational database management system, as it is available for free of cost and available as an open-source for anyone to use. In this application, SQL (Structured Query Language) programming language is used to create, update, delete & manage the tables and their contents, as MySQL is supported with the basic SQL queries. MySQL is the world's most popular open-source database. According to DB-Engines, MySQL ranks as the second-most-popular database, behind Oracle Database. MySQL powers many of the most accessed applications, including Facebook, Twitter, Netflix, Uber, Airbnb, Shopify, and Booking.com. Since MySQL is open source, it includes numerous features developed in close cooperation with users over more than 25 years.

Apache Spark

Spark is a cluster computing engine of Apache and is purposely designed for fast computing process in the world of Big Data. Spark is Hadoop based efficient computing engine that offers several computing features like interactive queries, stream processing, and many others. In memory cluster, computing offered by Spark enhances the processing speed of the applications. Apache Spark is freely available parallel data processing framework which gains increase in attention in the subject of big data analytics and artificial intelligence. Spark is one of the quick and adaptable cluster computing platforms which is made for wide range data processing. It can work using smallest and lowest unit which is known as micro batch process. Spark platform does not use costly disk access as its computation is done on memory which may lead to increased performance in data processing.

Key features are batch/streaming data which is unify the processing of data in batches and real-time streaming, using multiple language: Python, SQL, Scala, Java or R. Execute fast, distributed ANSI SQL queries for dashboarding and ad-hoc reporting and runs faster than most data warehouse. Perform Exploratory Data Analysis (EDA) on petabyte-scale data without having to resort to down sampling.

Apache Spark is a general-purpose, distributed cluster computing, data processing framework that, like MapReduce in Apache Hadoop, offers powerful abstractions for the processing of a large dataset. The Spark core is complemented by a set of powerful and higher-level libraries: SparkSQL, Spark Streaming, MLlib, GraphX, Packages.

Apache data frames are the collection of distributed data. In data frame, the data is organized in columns and optimized tables. Spark data frames can be constructed from various data sources that include data files, external databases, existing RDDs and Spark data frames. Finally, the spark framework provides the ease of use feature, security and phenomenal speed to Big Data.

1. Data storage and process

Comparative analysis by YCSB

For benchmarking, I used two databases (MySQL and MongoBD) and a benchmarking tool (YCSB) to compare their performance.

The Runtime, MySQL performed better in terms of overall runtime, taking 3332 ms compared to MongoDB’s 1528 ms. MongoDB exhibits a higher throughput (2604.24

ops/sec) compared to MySQL (300.12 ops/sec). Both databases experience garbage collections, but the counts and times are relatively low for both. The average latencies for cleanup and insert operations are higher in MongoDB compared to MySQL.

These results shows that MySQL performs better in terms of runtime and average latencies, while MongoDB achieves higher throughput. However, these results are based on the specific test configuration and workload used. Performance can vary depending on factors such as hardware, database configuration, dataset size, and workload characteristics.

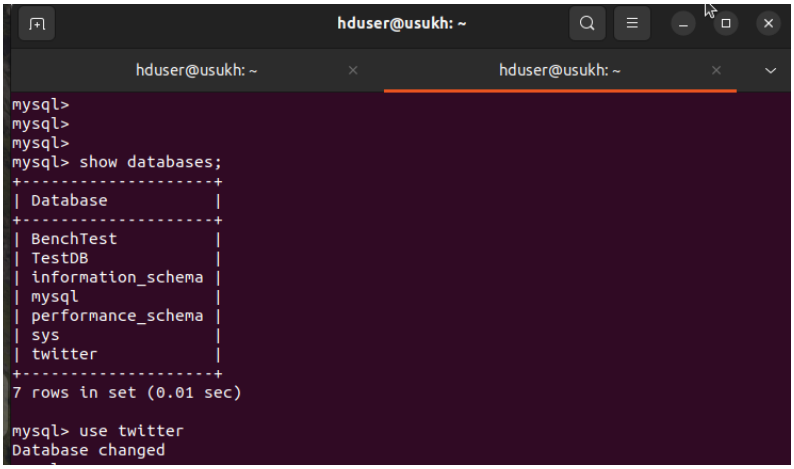
MySQL	MongoBD
[OVERALL], RunTime(ms), 3332	[OVERALL], RunTime(ms), 1528
[OVERALL], Throughput(ops/sec), 300.1200480192077	[OVERALL], Throughput(op 2604.2425
[TOTAL_GCS_G1_Young_Generation], Count, 2	[TOTAL_GCS_G1_Young_Generation], Count, 3
[TOTAL_GC_TIME_G1_Young_Generation], Time(ms), 7	[TOTAL_GC_TIME_G1_Young_Generation], Time(ms), 11
[TOTAL_GC_TIME_%_G1_Young_Generation], Time(%), 0.21008403361344538	[TOTAL_GC_TIME_%_G1_Young_Generation], Time(%), 0.03599476439
[TOTAL_GCS_G1_Old_Generation], Count, 0	[TOTAL_GCS_G1_Old_Generation], Count, 0
[TOTAL_GC_TIME_G1_Old_Generation], Time(ms), 0	[TOTAL_GC_TIME_G1_Old_Generation], Time(ms), 0
[TOTAL_GC_TIME_%_G1_Old_Generation], Time(%), 0.0	[TOTAL_GC_TIME_%_G1_Old_Generation], Time(%), 0.0
[TOTAL_GCs], Count, 2	[TOTAL_GCs], Count, 2
[TOTAL_GC_TIME], Time(ms), 7	[TOTAL_GC_TIME], Time(ms), 11
[TOTAL_GC_TIME_%], Time(%), 0.21008403361344538	[TOTAL_GC_TIME_%], Time(%), 0.035994764397905755
[CLEANUP], Operations, 1	[CLEANUP], Operations, 1
[CLEANUP], AverageLatency(us), 2209.0	[CLEANUP], AverageLatency(us), 5034.0
[CLEANUP], MinLatency(us), 2208	[CLEANUP], MinLatency(us), 5032
[CLEANUP], MaxLatency(us), 2209	[CLEANUP], MaxLatency(us), 5035
[CLEANUP], 95thPercentileLatency(us), 2209	[CLEANUP], 95thPercentileLatency(us), 5035
[CLEANUP], 99thPercentileLatency(us), 2209	[CLEANUP], 99thPercentileLatency(us), 5035
[INSERT], Operations, 1000	[INSERT], Operations, 1000
[INSERT], AverageLatency(us), 2972.783	[INSERT], AverageLatency(us), 2240
[INSERT], MinLatency(us), 1302	[INSERT], MinLatency(us), 253
[INSERT], MaxLatency(us), 21663	[INSERT], MaxLatency(us), 80234
[INSERT], 95thPercentileLatency(us), 4463	[INSERT], 95thPercentileLatency(us), 663
[INSERT], 99thPercentileLatency(us), 6327	[INSERT], 99thPercentileLatency(us), 2456
[INSERT], Return=OK, 1000	[INSERT], Return=OK, 1000
[INSERT-FAILED], Operations, 1	[INSERT-FAILED], Operations, 1
[INSERT-FAILED], AverageLatency(us), 160192.0	[INSERT-FAILED], AverageLatency(us), 3.002368E7
[INSERT-FAILED], MinLatency(us), 160128	[INSERT-FAILED], MinLatency(us), 30015488
[INSERT-FAILED], MaxLatency(us), 160255	[INSERT-FAILED], MaxLatency(us), 30031871
[INSERT-FAILED], 95thPercentileLatency(us), 160255	[INSERT-FAILED], 95thPercentileLatency(us), 30031871
[INSERT-FAILED], 99thPercentileLatency(us), 160255	[INSERT-FAILED], 99thPercentileLatency(us), 30031871

[Figure.1] Comparative analysis by YCSB (MySQL and MongoDB)

1.1. Data Storage

In this comprehensive analysis, I have downloaded a Twitter dataset from Kaggle.com by years. The Twitter dataset is CSV file and includes 38 columns (id, conversation_id , created_at, date, timezone, place, tweet, language, hashtags, cashtags, user_id, user_id_str, username, name, day, hour, link, urls, photos, video, thumbnail, retweet, nlikes, nreplies, nretweets, quote_url, search, near, geo, source, user_rt_id, user_rt, retweet_id, reply_to, retweet_date, translate, trans_src, trans_dest).

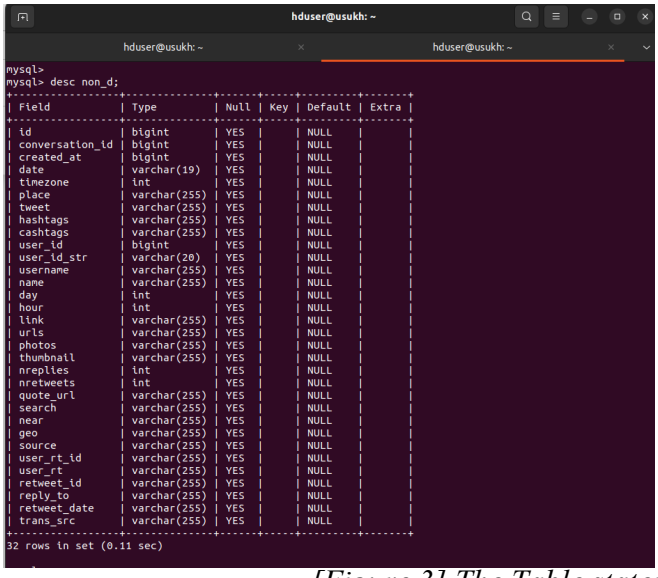
Firstly, I have loaded the dataset into a MySQL database which is named “twitter”. This step involved setting up the necessary database schema and tables to store the dataset.



[Figure.2] The databases of MySQL

1.2. Data Processing

Secondly, I loaded the twitter CSV file into a data table within the MySQL twitter database, the table which is named "non_d". This step allowed me to store and manipulation the twitter dataset for further Sentiment and Time series forecasting analyse. For the load CSV file into a data table, need to SQL statement for creating the table. (Fig 3) shows table statement.



[Figure.3] The Table statement of MySQL database

Following this, I loaded CSV file to the table by using method of "LOAD DATA LOCAL INFILE <path.csv> INTO TABLE table_name".

Data Transfer to PySpark Dataframe:

After successfully load the data into the MySQL database, I connect the data to a PySpark Dataframe. For the connect to the MySQL database, I used “mysql.connector” which allows Python to connect a MySQL database and “pyspark.sql.SparkSession” provides the entry point to PySpark and enables working with DataFrame and “pyspark.sql.types” contains classes for define the schema of a Dataframe.

The “mysql.connector.connect()” method is used to create a connection object by specifying the host, username, password, and database name. And created a Cursor object and Executed the SQL query by using method of “cursor()” and “execute()” and the StructType class is used to define the schema of the DataFrame. The schema defined in the code snippet corresponds to the columns of the table in the MySQL database. Each field in the schema represents a column, and the nullable parameter indicates whether the field allows null values. For example: StructField("id", LongType(), nullable=True).

2. Data cleaning and preparation

Before applying any machine learning or deep learning library for sentiment analysis, it is crucial to do text cleaning and/or pre-processing. It is essential to reduce the noise in human- text to improve accuracy. Data is processed with the help of a natural language processing pipeline. Data cleaning, which tends to identify and remove invalid words, such as misspelling, stop words, punctuation marks, tabs, incomplete words, non-target language and low frequency words from the texts, leaving valid information.

2.1. Data cleaning

For the step of natural language processing (NLP), I have imported the Natural language Toolkit (NLTK) library to remove “stopwords” from the dataset. Downloaded necessary resources (“punkt”, “stopwords”), and the “stopwords” corpus contains a list of common words that often considered irrelevant in text analysis. After this, by using “set()” function, I have created a set called “stop_words” that contains the English “stopwords” from NLTK. And set the functions to perform remove punctuations, HTML tags, URLs, emojis and other Unicode characters from text.

2.2. Tokenization

Tokenization is a process of splitting up a large body of text into smaller lines or words. It helps in interpreting the meaning of the text by analyzing the sequence of the

words. It then initializes a tokenizer object using the “Tokenizer” class and fits it on the cleaned tweet data (“df[‘clean_tweet’]).

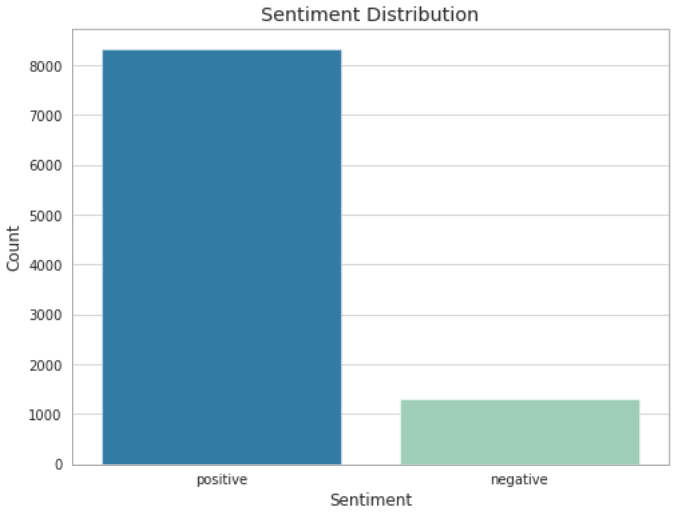
In the first stage, I have settled maximum length of a tweet and maximum number of words to be included in the vocabulary, which the length being 100, the number of word is 10000. The tokenizer method of “fit_on_texts” is called to update the vocabulary based on the cleaned tweet data from my dataset. This method assigns a unique integer value to each word in the vocabulary. Next, the “texts_to_sentences” method has used to convert the cleaned tweet’s text data into sequences of integers, where each integer corresponds to a word in the vocabulary. The resulting sequences are stored in the “sequences” variable. The variable “word_index” is a dictionary that contains the words in the vocabulary as keys and their corresponding integer values as values. Finally, called to pad the sequences to fixed length of maximum length.

2.3. Polarity

Tweets combined with a sentiment score can give you a gauge of your Tweets in a quantitative way. To put some data behind the question of how is the feeling. In sentiment analysis, use polarity to identify sentiment orientation like positive, negative, or neutral in a written sentence. Fundamentally, it is an emotion expressed in a sentence.

For the polarity, I have used the “TextBlob” library to calculate the polarity score of a text which is mean polarity takes a text as input and returns the polarity score of the text using the “TextBlob” library. After this, applied to the cleaned tweet of the DataFrame to polarity.

After the polarity, converted the sentiment polarity of the DataFrame to sentiment labels (positive or negative) and stores the result in a new column which is named “sentiment”. The function is used to apply a condition (df['sentiment_polarity'] >= 0) to the elements in the column sentiment polarity. If the condition is true, the corresponding element in the new column “sentiment” is set to “positive”, otherwise, it is set to “negative”. (Fig4) shows plot of the sentiment values.



[Figure.4] Sentiment values

Finally, through these operations, original twitter dataset's tweets can be transformed into a substantially reduced number of valid, meaningful, important words or phrases.

3. Sentiment Analysis

Sentiment Analysis for the Twitter datasets. A Twitter sentiment analysis determines negative, positive, or neutral emotions within the text of a tweet using NLP and ML models. Sentiment analysis or opinion mining refers to identifying as well as classifying the sentiments that are expressed in the text source. Tweets are often useful in generating a vast amount of sentiment data upon analysis. These data are useful in understanding the opinion of people on social media for a variety of topics.

(Data Science Blogathon.)

In this project, for the sentiment analysis used Keras API of neural network Tensorflow. It allows to define and instantiate a deep learning model by specifying the input and output layers. After applied Train and Test split, the input layer defines the input shape of the model.

The Embedding layer converts the input sequences of integers into dense vectors of fixed size (128 in this model), which is maps each integer value to a dense vector representation. The "len(word_index) + 1" specifies the input dimension of the embedding layer, which is the size of the vocabulary plus one for unknown words. The "input_length=max_len" parameter sets the length of the input sequences.

The “GlobalMaxPooling1D” layer performs global max pooling over the sequence dimension, which means it selects the maximum value from each feature map across the sequence. It reduces the dimensionality of the input and captures the most salient features. And the “Dense” layer is a fully connected layer with 64 units and applies the “ReLU” activation function. The “Dropout” layer randomly sets a fraction of input units to 0 at each update during training, which helps prevent overfitting. In this model, I set 50% of the units to 0. The final layer has 1 unit and applies the sigmoid activation function. It outputs a single value between 0 and 1, representing the predicted sentiment score and the “Model” function is used to create the model, specifying the input and output layers.

Finally, the model architecture represents a simple neural network for sentiment analysis, consisting of an embedding layer, a global max pooling layer, a fully connected layer, a dropout layer for regularization, and a final output layer.

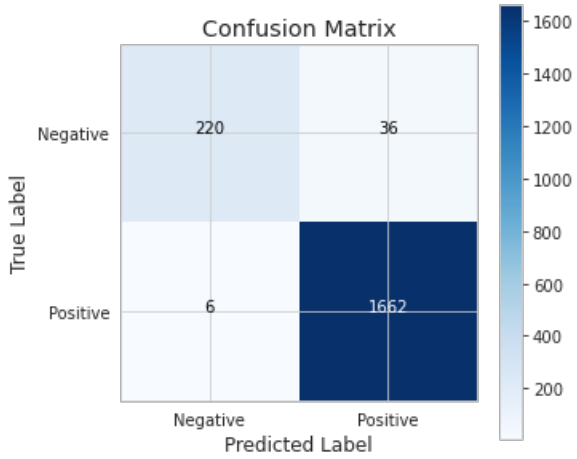
```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 100)]             0

 embedding (Embedding)       (None, 100, 128)          1109248

 global_max_pooling1d (Globa  (None, 128)              0
 lMaxPooling1D)

 dense (Dense)               (None, 64)                8256

 dropout (Dropout)           (None, 64)                0

 dense_1 (Dense)             (None, 1)                 65

=================================================================
Total params: 1,117,569
Trainable params: 1,117,569
Non-trainable params: 0
_________________________________________________________________
```

[Figure.5] Summary of the model

As a result, the model evaluated the test loss is “0.11” and the test accuracy is “0.978” indicates that the model correctly classified approximately 97.8% of the samples in the test dataset. Those results suggest that the model is performing well and has a high level of accuracy in predicting the sentiment of the test data.

Also, by the Confusion matrix classification report defines for the "Negative" class, the precision is 0.97, which means that 97% of the samples predicted as "Negative"

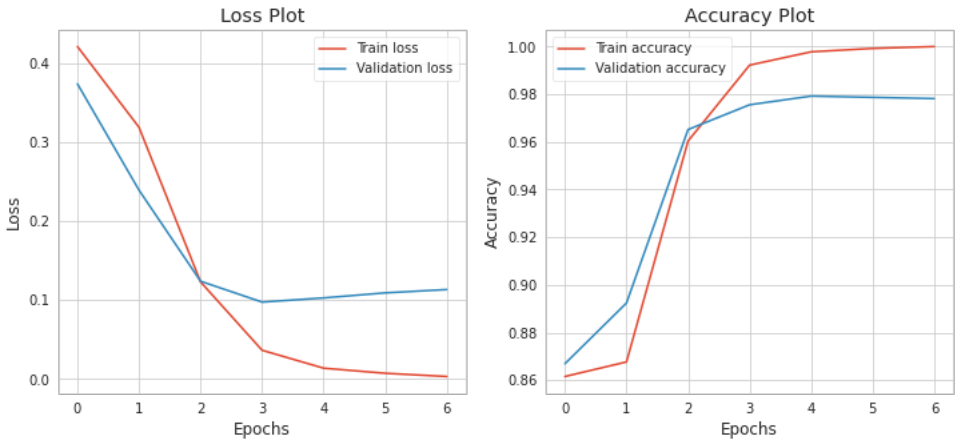
are actually "Negative". The "Positive" class, the precision is 0.98, indicating that 98% of the samples predicted as "Positive" are indeed "Positive".



	precision	recall	f1-score	support
Negative	0.97	0.86	0.91	256
Positive	0.98	1.00	0.99	1668
accuracy			0.98	1924
macro avg	0.98	0.93	0.95	1924
weighted avg	0.98	0.98	0.98	1924

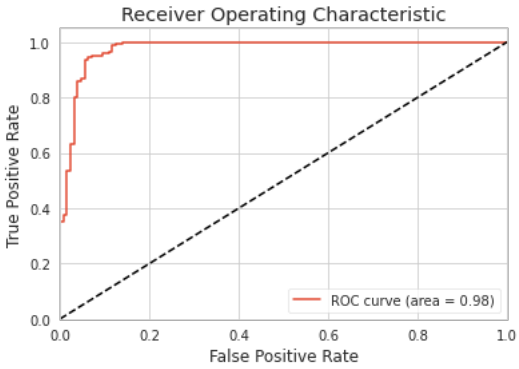
[Figure.6] Confusion matrix

The (Fig 7) shows two subplots of training and validation loss.



[Figure.7] Plot of Loss and Accuracy

The Receiver Operating Characteristic (ROC) curve to evaluate the performance of a binary classification model. (Fig 8)



[Figure.8] Plot of Loss and Accuracy

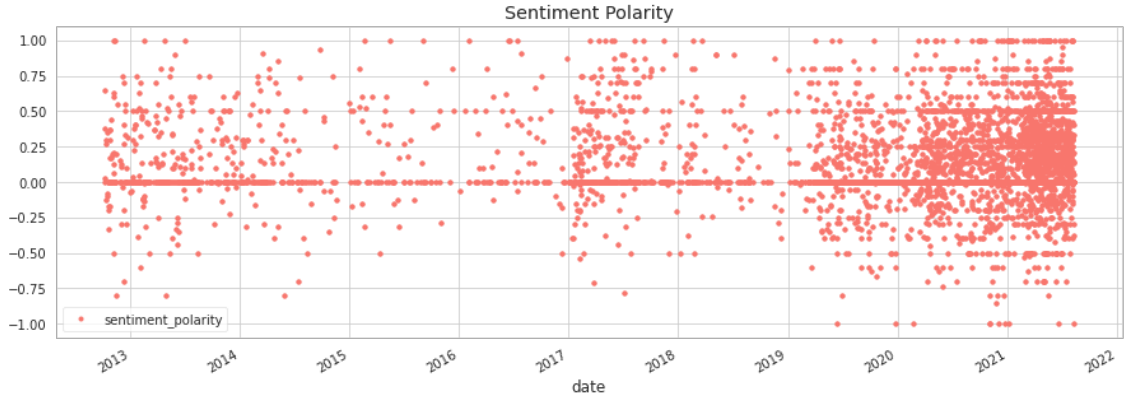
Implementation

By using twiter dataset, I have created a sentiment analyses model using neural network Tensorflow to predict sentiment label (“positive” or ”negative”). The Accuracy of the model is 97.8%.

4. Time Series Forecasting

4.1. Prophet Forecasting Model

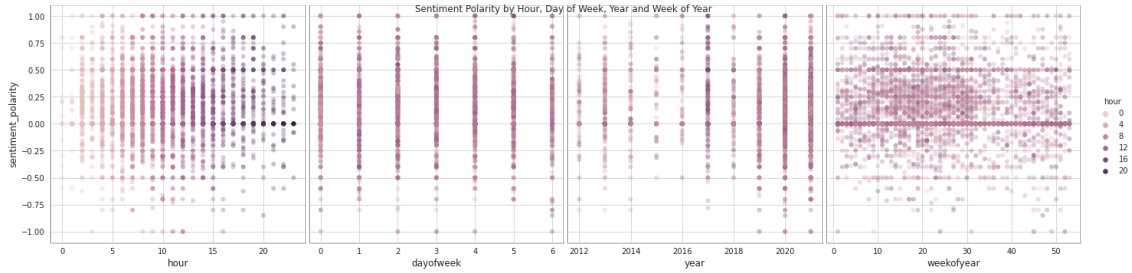
Prophet Forecasting Model is developed by Facebook, available in python and R. Due to its three main features, ie. trend, seasonality, holidays and demand for the high quality of forecasting are the main reason for building this model. By utilizing Prophet forecasting techniques on Twitter datasets, I can gain valuable insights into the behavior of Twitter users related to make predictions, sentiment analysis and more.



[Figure.9] Plot of Sentiment Polarity

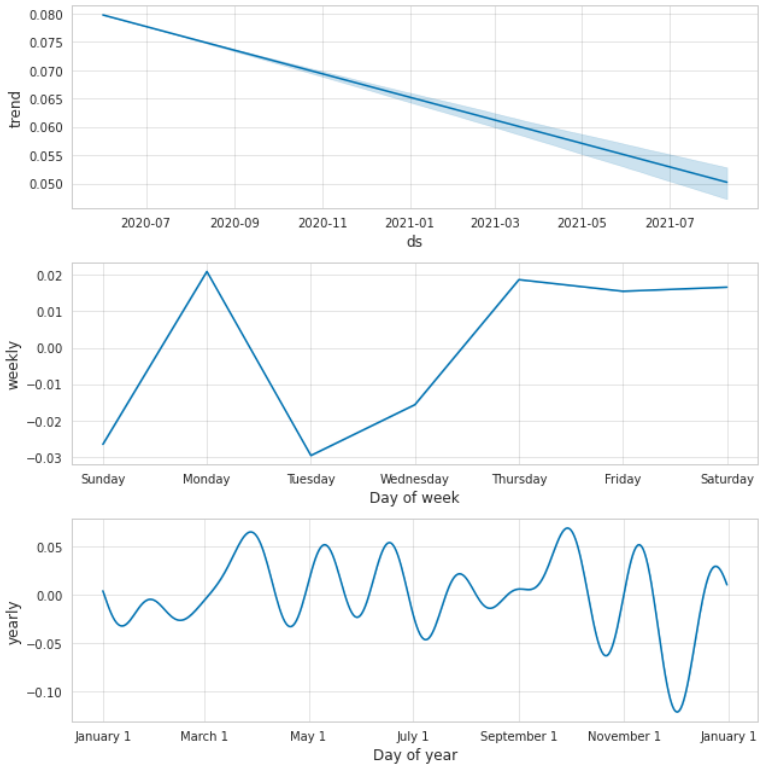
The pair plot (Fig 10) shows scatter plots of 'sentiment_polarity' against 'hour', 'dayofweek', 'year', and 'weekofyear', with different colors representing different hours of

the day. This plot helps visualize the relationships between these variables and the sentiment polarity.



[Figure.10] Plots of Sentiment Polarity by time period

For the time series forecast, I have used average of polarity value of the day. Then split data to less than “2020-06-01” and assigns it to the “d_train” DataFrame. Where the index value is greater than or equal to “2020-06-01” and assigns it to the “d_test” DataFrame. In summary, Split the daily DataFrame into a training set (d_train) that contains rows with index values before "2020-06-01" and a test set (d_test) that contains rows with index values on or after "2020-06-01". And setup and trained the model of Prophet and fit the model.

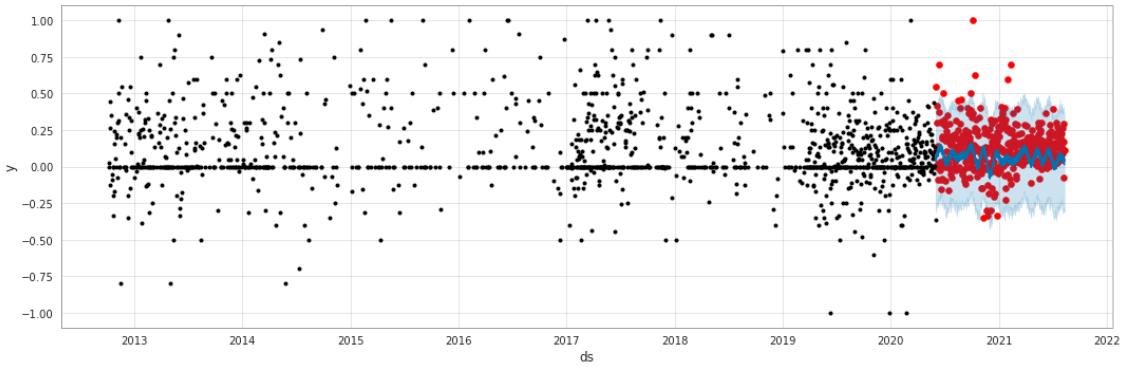


[Figure.11] Plots of the components of the model

Plots of the components of the model (Fig 11) shows:

- The trend showing a tendency to decrease further.

- One the other hand, the results show that the average sentiment of Twitter's tweets tend to increase on weekend, while it tends to be lower on working days.
- Also, the average sentiment of tweets tends to decrease in end of the year.



[Figure.12] Scatter plot of actual values with forecasted values

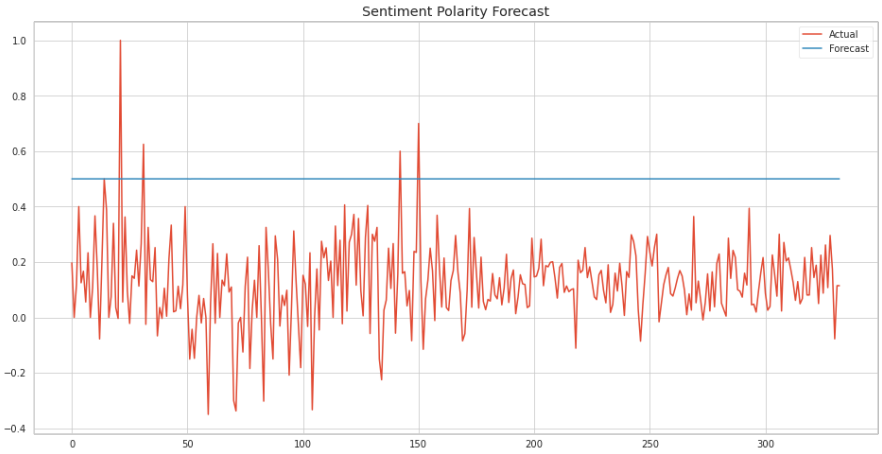
The (Fig 12) provides the forecasted values along with the actual values. This plots the actual values from the "d_test" DataFrame on the scatter plot. The x-values are taken from the index of "d_test", and the y-values are taken from the "average_polarity" column. The points are plotted as red dots.

4.2. LSTM (Long Short-Term Memory).

To begin with, I have split dataset last 30 days to "y" and others to "X". After the reshape data for LSTM input, I have defined the model architecture. The architecture of a sequential model using LSTM for classification. The LSTM layer followed by a dropout layer for regularization and a dense layer with softmax activation for classification.

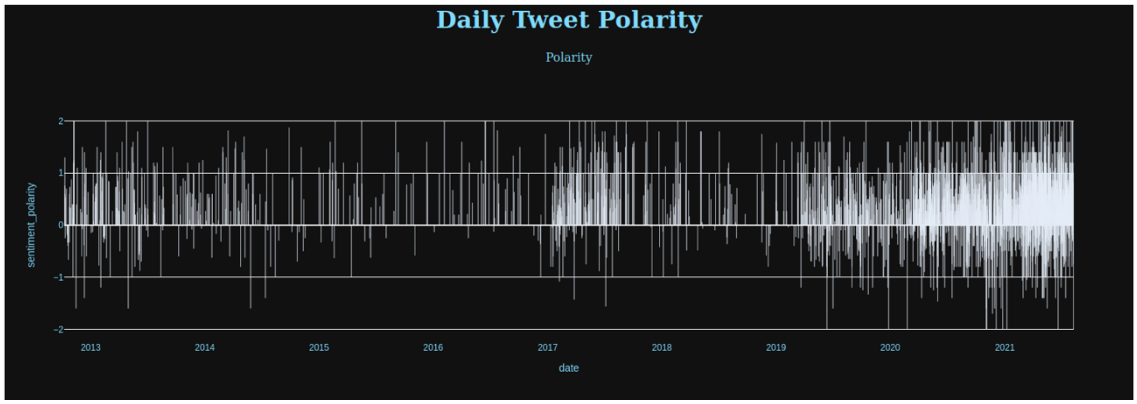
```
Model: "sequential_7"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_7 (LSTM)               (None, 64)                16896

 dropout_8 (Dropout)         (None, 64)                0

 dense_9 (Dense)             (None, 2)                 130

=================================================================
Total params: 17,026
Trainable params: 17,026
Non-trainable params: 0
_________________________________________________________________
```

[Figure.13] Summary of sequential model



[Figure.14] Plot of sentiment polarity values from the forecast and the actual values

Dash



[Figure.15] Dashboard of Daily tweet polarity

Conclusion

In the project, I used a Twitter dataset from Kaggle.com and loaded it into MySQL, and loaded it to the PySpark DataFrame. And prepared the dataset for the machine learning models and natural language processing. And I have created prediction model of sentiment analysis which is 97.8% accuracy. The Sentiment Analyse has done by Deep Learning Keras model and time series forecasting with Prophet model and LSTM (Long Short-Term Memory).

References

Ling Tang, Jieyi Li, Hongchuan Du, Ling Li, Jun Wu, Shouyang Wang. (2021) Big Data in Forecasting Research: A Literature Review.

S.A. Sanchez, H.J. Romero, A.D. Morales. (2020) A review: Comparison of performance metrics of pretrained models for object detection using the TensorFlow framework. *Colombia, Faculty of Engineering Science, GINTEING Research Group*

"What is Apache Spark? Spark tutorial guide for beginner", janbakstraining.com, 2018.

Christos, C., G, Kalliatatakis. And Georgios, S. (2017) Hadoop and What is good for. Machine Learning: Advances in Research and Applications. Chapter 6.

Inge, Roger, and Jan Leif. (2017) Machine Learning: Advances in Research and Applications. Nova Science Publishers, Inc,

K. Krishna Rani Samal, Sontash Kumar Das, Korra Sathya Babu, Abhur Acharaya. (2016) Time Series based Air Pollution Forecasting using SARIMA and Prophet Model.

H. Chiroma, U.A. Abdullahi, S.M. Abdulhamid, A.A. Alarood, L.A. Gabralla, N. Rana, L. Shuib, I. Hashem, D. Gbenga, A. Abubakar, A. Zeki, T. Herawan. (2018) "Progress on Artificial Neural Networks for Big Data Analytics: A Survey*", Federal College of Education (Technical), Gombe, Nigeria,*

Harvard Style *UseIt,* Available at http://www.library.uq.edu.au/training/citation/harvard.html

https://www.socialmediatoday.com/content/twitter-101-what-twitter-really-about#:~:text=Twitter%20is%20a%20social%20network%20and%20real-time%20communication,hence%20the%20bird%20used%20in%20the%20Twitter%20logo.