MSc in Data Analytics

Advanced Data Analytics and Big Data Storage & Processing – Integrated CA4

Author – Darren Smith

Email – [REDACTED]

Student ID – [REDACTED]

Time Series Forecasting of the Sentiment for Elon Musk's Tweets for 2016

GitHub Repository: https://github.com/DarrenCCT/CA4/

Word Count: Approximately 2,700

Contents

Abstract

Tweets from Elon Musk were collected for the year 2016. Using VADERS in the NLTK library, a sentiment analysis was performed on all the tweets he posted on the platform Twitter.

A time series forecasting model, ForecasterAutoreg from the Scikit-learn library, was trained and tested to see how accurately it would perform. A mean squared error of 0.03 was achieved. Attempts to use Hyperparameter tuning ended up not besting the performance of the preliminary model. Elon Musk's future sentiment for one week, one month, and three months were then predicted for the future.

Using only tweets from one individual from a single year has limited potential for conducting time series analysis on their sentiment.

Introduction

Elon Musk is the second-richest person alive. He is the CEO of SpaceX, Tesla Inc., and more recently Twitter Inc. He shocked the world in 2020 when it was announced that he would be buying Twitter for an eye-watering sum of $44 billion (Forbes, 2020).

The aim of this report is to detail my steps in acquiring a year's worth of tweets belonging to Elon Musk. I will then be attempting to use time series forecasting to see if it is possible to fit a model to predict the sentiment of Elon Musk's tweets based on his twitter messages over the year 2016. Then predictions will be made to forecast his expected sentiment for the 1st week of January, the month of January, and the first three months of 2017.

The programming aspect of this project was backed up frequently on Github. A link to the repository can be found on the front page of this report. This report was backed up on OneDrive. I used OneDrive for the report as it was easier for me to access on different machines when I wanted to make changes.

The report has two Jupyter Notebooks attached. One for my work done in Linux for the Big Data aspect and ADA was done in its own notebook once I moved over to Windows.

Data Acquisition

To accomplish the tasks set out in the brief, we were given two avenues to acquire our data. We could either get it through the Twitter API or by downloading an archive through archive.org. In the end, neither of these options were suitable. Due to recent changes in the Twitter API, it is no longer possible to search through tweets unless you pay for a higher tier developer account. It is only possible to get "recent" tweets (past seven days) and a maximum of 50 tweets at a time. It is just not doable to get a year's worth from this. I ruled out archive.org due to the sheer size of the downloads needed. A single month's worth of tweets can be 40-50GB in size. This is not something that I can download. Instead, I have opted to download a dataset from Kaggle to do this assignment. You can refer to section 2.1 of the attached Jupyter Notebook for my attempts at using the Twitter API using the Tweepy library.

I initially intended to conduct a sentiment analysis on tweets regarding the Snooker legend Ronnie O'Sullivan. I wanted to see how sentiments fluctuate between times when he is actively participating in a tournament versus when he is not. Alas, it was not to be.

A link to the dataset I used can be found in the Appendices section.

Big Data Pre-processing

Storage

The tweets were stored in Hadoop Distributed File System (HDFS). I chose this solution as it was built to handle very large datasets and run on low-cost, commodity hardware. It is able to handle processing big datasets quickly and efficiently. It is also compatible with Apache Spark which is what I am going to use for this project (Databricks, 2023).

The brief mentioned that we *can* use a SQL/noSQL database but I decided against it. I did not feel the use of one of these databases would aid in the project in any meaningful way.

Processing

Apache Spark is a multi-language engine for executing data engineering, data science, and machine learning on single-node machines or clusters (Apache Spark, 2023). I chose Apache Spark as it has great compatibility with python and many of its more popular libraries. It can query SQL databases. It can perform Exploratory Data Analytics on huge data sets with little issues. It is also very user friendly. It is much easier to use Apache Spark than Hadoop Mapreduce as it is more intuitive for processing Big Data

Data Exploration & Data Cleaning

I used tools found in Apache Spark to analyse the data. I found out that the dataset contained multiple years of tweets from Elon Musk. Ranging from the years 2010 to 2017. Using said tools, I tallied up the tweets per year to get an idea of how prolific of a tweeter he is. Also, to find out what would be usable for the later forecasting section.

Below is a graphical representation of my findings. The shade of blue's hex value is #1DA1F2. This is the same shade of blue that Twitter uses for its logo. This will be used throughout the report as I found it a fitting colour thematic-wise.

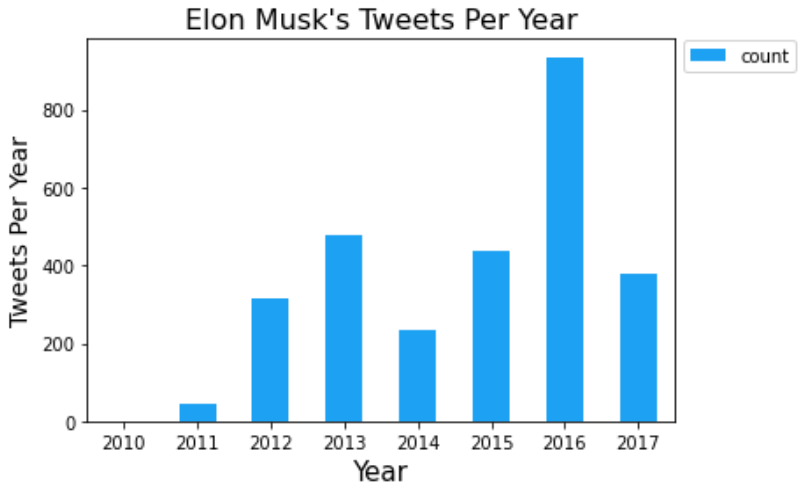

Figure 1 – Bar Chart of Elon Musk's Tweets per Year

With this information, I decided that I would use the year of 2016 as the basis of this project as it has the most tweets. The more tweets I have, the more information I can feed into the machine learning algorithms later to maximise their performance. I removed all the other years. There was also an "id" column for each tweet. I removed this too as I had no

need for it. I then wrote what I had left in my data frame to a csv file so I could continue my work in a Windows environment. Pandas can handle over 800 tweets. Table 1 shows a data dictionary of the data in its original form before any alteration.

Column	Data Type	Description	Example
id	object	Unique identifier number for the tweet	848239928043491328
created_at	object	The date and time for when the tweet was made. In the format YYYY-MM-DD HH:MM:SS	2016-08-17 08:54:58
text	object	The message of the tweet. Up to 160 characters in length	It's a good day to own a Tesla car

Table 1 - Data Dictionary of dataset before pre-processing

Once I had moved over to the Windows machine, I loaded in the data to a Pandas dataframe. I used Pandas as it has a wide range of functions that can perform easy data handling and manipulation. It can also interact well with other libraries such as matplotlib; The library I use for the graphs. I checked the data types of my columns. As we can see in Table 1, the "created_at" column is an object type. I had that converted to datetime type so it would able to be used for time series analysis later. I also renamed the columns "created_at" and "text" to "Date" and "Text" respectively.

Natural Language Processing

Now that I had the relevant data, my next step was to get a sentiment value from the text data. Before I did that, I had to cut out irrelevant sections of the tweets. To do this, I first removed stop words from the tweets. Stop words are words that are considered to have no impact on the sentiment value of the sentence. For example, conjunctions such as "and", "or", "because" are considered stop words. They are removed to help the algorithms that analyse the text by cutting down on the amount of work that needs to be done. Special characters were also removed from the next data. The cleaned data was then attached to the dataframe.

As that was done, I used the VADER model found within the Natural Language Toolkit (NLTK) to perform a sentiment analysis on every cleaned tweet. NLTK measures the polarity of a body of text to see if it is negative, neutral, or positive. These are represented numerically within the range of negative 1 to positive 1. Any number that is below zero is a negative statement. The closer to negative 1, the more negative it is. The same is true in reverse for positive messages. Any message that receives a zero score is a neutral sentence.

The sentiment analysis was performed and the scores were appended to the dataframe. Table 2 below shows the updated data dictionary.

Column	Data Type	Description	Example
Date	datetime64[ns]	The date and time for when the tweet was made. In the format YYYY-MM-DD HH:MM:SS. Now in proper format	2016-08-17 08:56:38
Text	object	The original message of the tweet. Up to 160 characters in length	Someone needs to buy Twitter!!11!
Cleaned	String	The message after removing stop words and extra characters	Someone need buy Twitter
Sentiment	Float64	A number ranging from -1 to +1. Above zero means positive and below is negative	0

Table 2 - Dataset after Data Cleaning and Preparation for Machine Learning

For what I will be doing later, I need daily sentiment values. I currently have multiple tweets per day. Using the Groupby function found in the Pandas library, I can use this in conjunction with the Date column as it is in datetime format. An average was then taken of every sentiment value for every day in the dataframe. This was then collated in to a new dataframe consisting of just the date and the average sentiment value. The old dataframe was discarded as it had no more use.

I then checked for missing dates throughout the year and there was some found. To tackle this, I filled in the missing dates and gave them a zero. You can only be neutral if you say nothing on a particular day.

Preparation for Machine Learning

I decided against normalising the sentiment data (converting all the values to be between zero and one). While it is true that normalising the dataset can aid the machine learning algorithms by reducing the calculation crunching required, the dataset is not particularly large and there are no extreme outliers present. The sentiment values I have also only go from negative one to positive one. The time series forecaster can handle negative values just fine. In my opinion, the graph that will be generated will be easier to read with the negative values left in. The reader can quickly tell if a sentiment is negative or positive based upon whether it goes below or above the neutral line of zero.

Time Series Forecasting

For the forecasting prediction, I will be using the ForecasterAutoreg from the Scikit-learn library. The forecasting algorithms attempt to predict future values by modelling the time series based on previous known values in the past. Due to it being from Scikit-learn library, it also has access to the entire Scikit-learn environment. This will provide access to tools from this library which can leveraged for hyperparameter tuning and performance evaluations. For training the time series forecasting model, a training/testing split of 80:20 was chosen. This is standard.

Below is a line graph of Elon Musk's sentiment values throughout 2016. With the contrasting colour of a dark orange to clearly show what data is being used for testing and training.

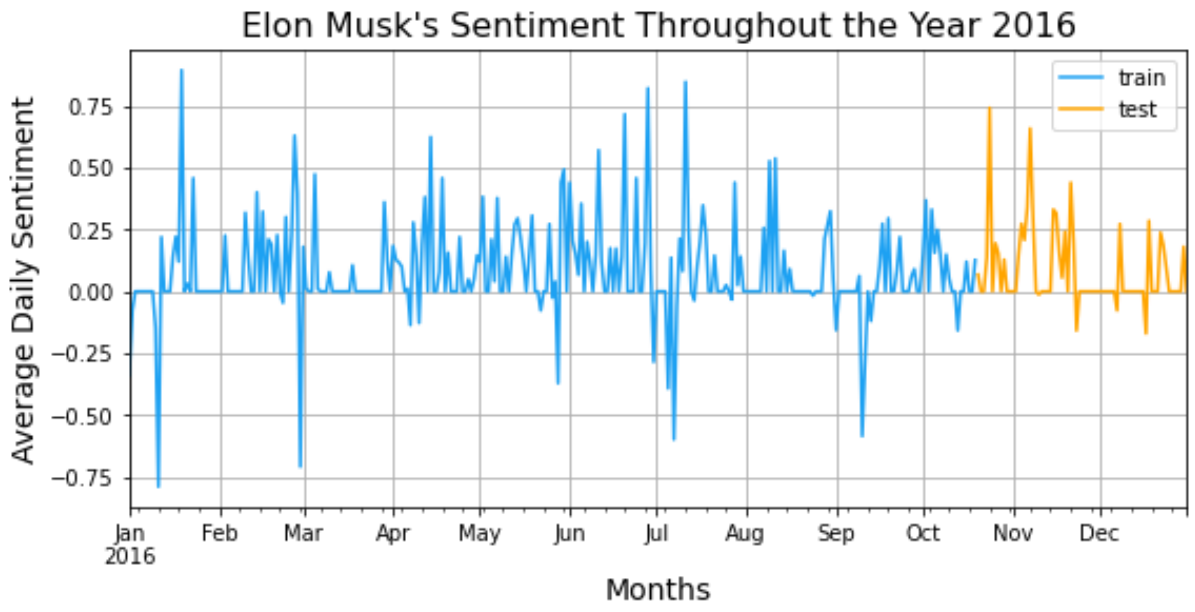

Figure 2 - Elon Musk's Sentiment for 2016

What we can learn from this is that Elon Musk is a highly positive individual on Twitter. Days in which he tends to be negative are few but do tend to be extreme. He had several negative days towards the end of June. This could be attributed to him dealing with the controversy of when a person had died when using the autopilot feature in a Tesla car. The Guardian reported that Elon Musk went on a rant during this time and it could even be used as a case study of how to not deal with a crisis (Woolf, 2016).

With the time series forecasting model trained, it was now time to test the predictions on the testing split.

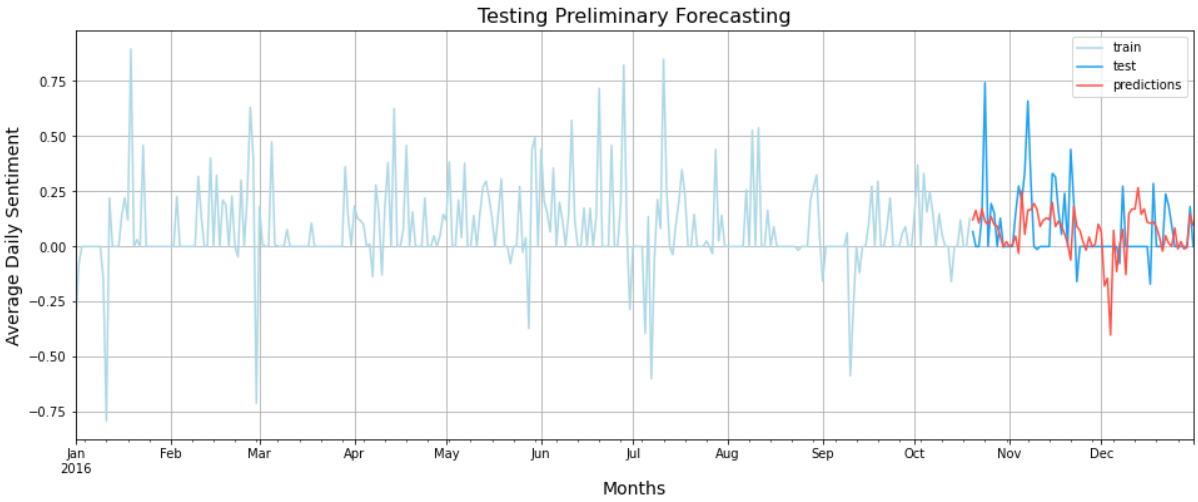

Figure 3 - Testing Preliminary Forecasting Model

Yeah, it is not great but I do not think that it is bad either. It is a solid starting base. The mean squared error for this model was 0.030133824103468796.

I will now move on to hyperparameter tuning to try make the model more accurate.

Hyperparameter Tuning

As mentioned earlier, ForecasterAutoReg has access to the Scikit-learn library that this is what I used to tune the hyperparameters. For my first round of hyperparameter tuning, I tried multiple values for the window size and had the optimiser look for the best max depth and n estimators. The preliminary model had a window size of 6 so I chose values in and around that. The window size is the number of units before the predicted value that the model attempts to predict the next unit value of. In this case, the model uses the previous 6 days to predict the next day's sentiment value. The figure below is what the optimiser picked for the best fitting model. Which was a window size of 6, max depth of 3, and n estimators to be 100. The metric being measured to evaluate the performance is the mean squared error. The closer to zero it gets, the better the model's predictions fit the actual data.

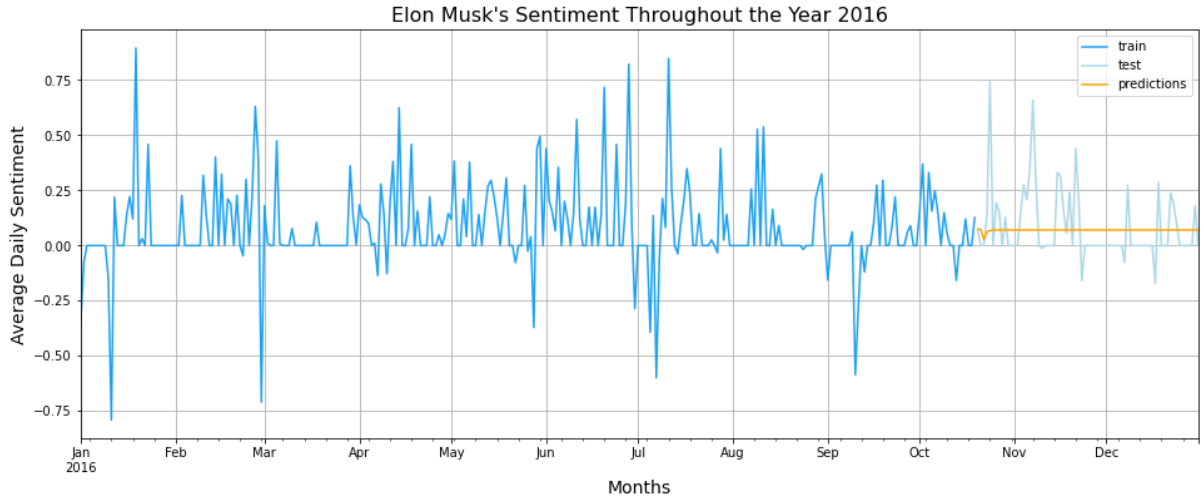

Figure 4 - Hyperparameter Tuning results

This looks worse in my opinion and the mean squared error of 0.0408 confirms as such. I tried many different window sizes and different parameters to try to improve upon the original model but none of them came close. They usually had the same mean squared error as this model. To share the results of all this testing in this report would be tedious. You can refer to section 2.5 of the attached Jupyter Notebook for more graphs and of all my testing mentioned.

Either way, as my original model was the best one, that is the one I will do predictions of the future with.

Predictions

The brief set out to have us make predictions 1 week, 1 month, and 3 months into the future. The Figure below show Elon Musk's expected sentiment for the week.

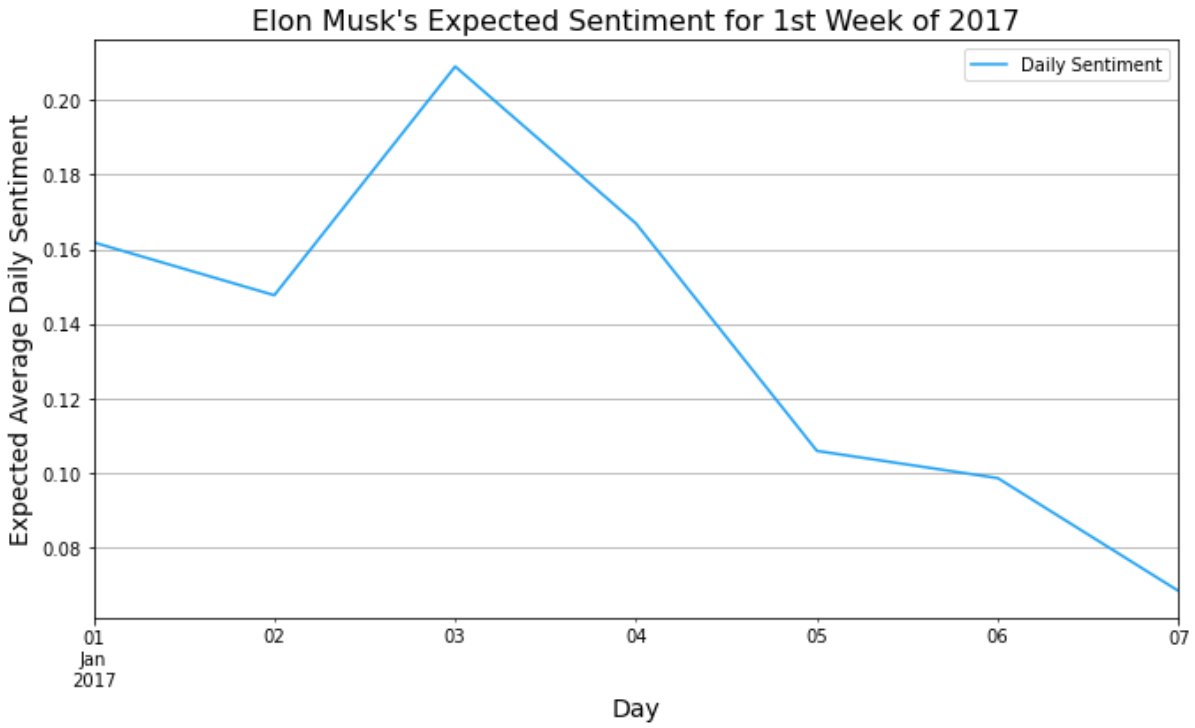

Figure 5 - Elon's Expected Sentiment for 1st Week of 2017

Elon Musk is expected to have positive tweets in the first week of January 2017 with no negative views being shared.

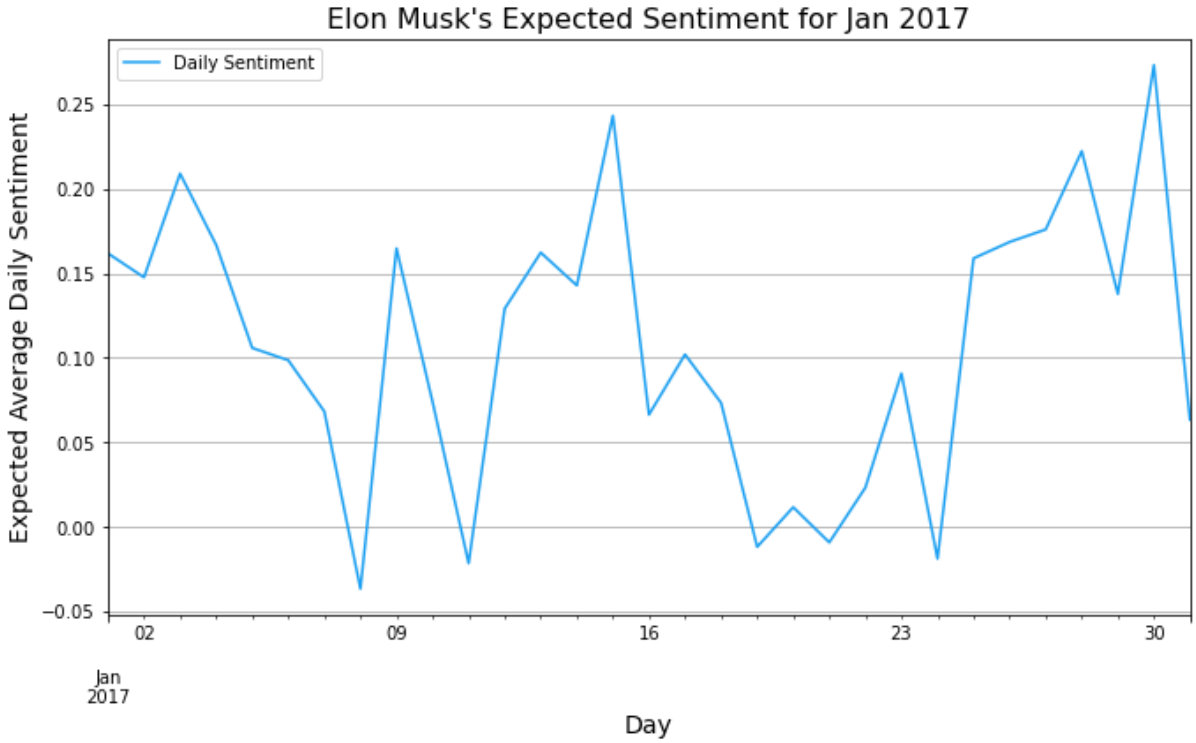

Figure 6 - Elon's Expected Sentiment for Jan 2017

For the first month of 2017, Elon is expected to have a mostly positive month. There are a few slightly negative days but no extremely bad days.

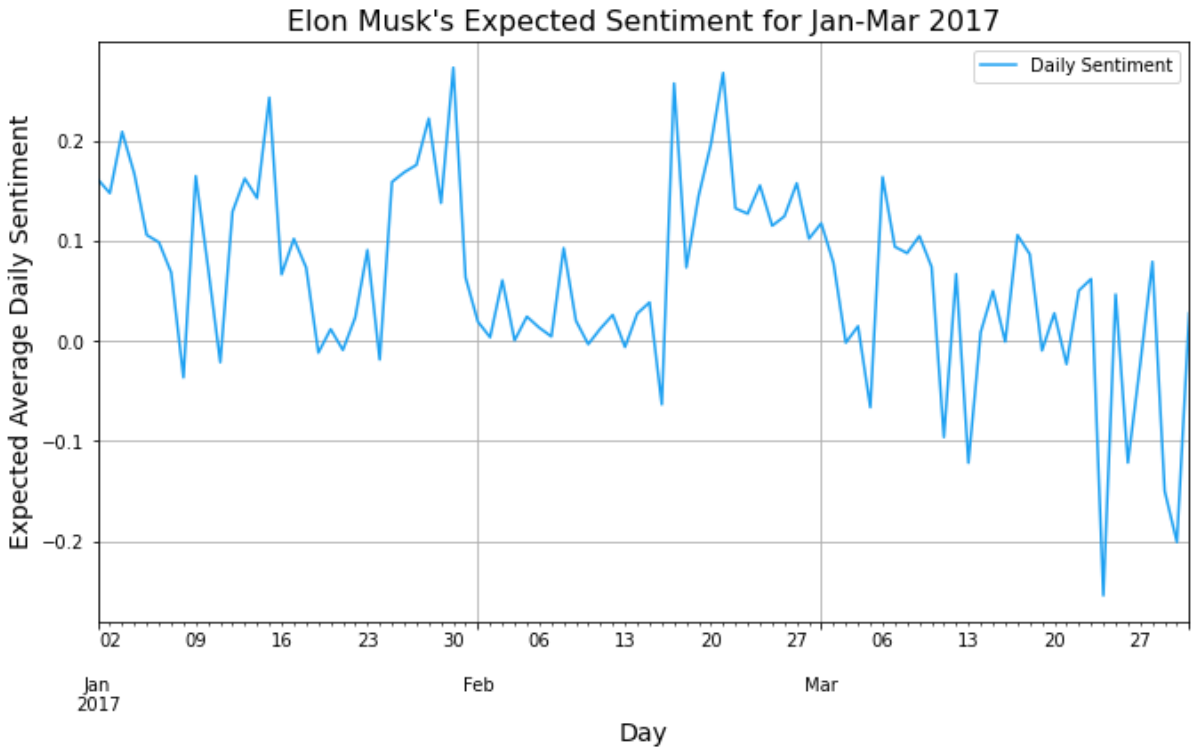

Figure 7 - Elon Musk's Expected Sentiment for the First Three Months of 2017

Elon continues to have a positive February but starts getting more negative as the month of March ends.

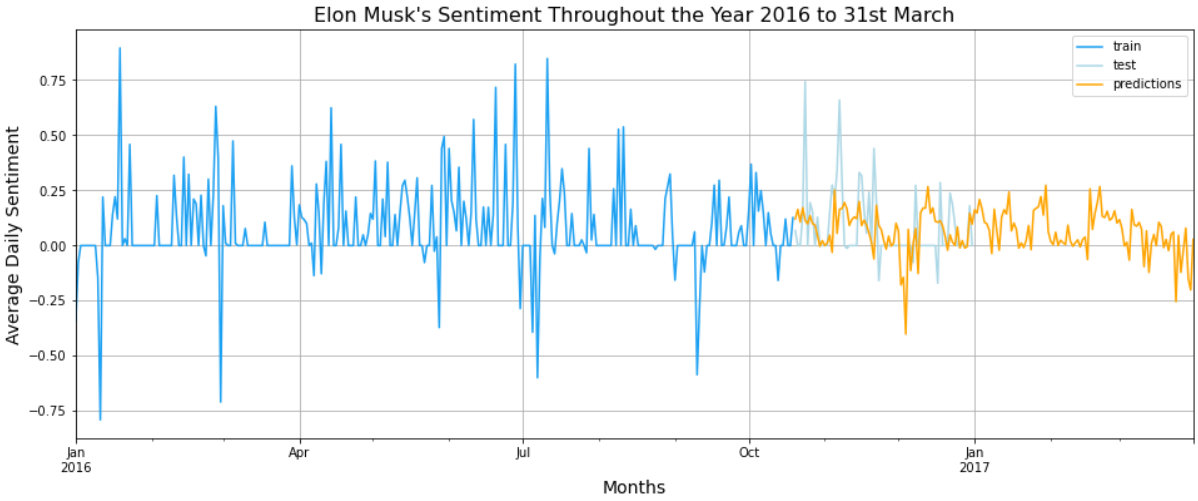

Figure 8 - Elon Musk's Sentiment from 2016 to the end of Mar 2017

Lastly, a final graph to put it all together with the actual data.

Conclusion

What I have learned from this is that it is possible to use a sentiment analyser to gather tweets from an individual to predict their future sentiments. However, using solely tweets from one person to predict their sentiment going forwards has limited potential when confined to using only one year's worth of data. Elon Musk could be considered to an active

tweeter and even data om him, in one of his most active years, was not enough to build an accurate model.

I would have much preferred to have done this assignment with the Twitter API as I could have collected so much more data on a topic rather than using an individual's own tweets. That proves impossible as of May 2023 due to changes in the Twitter platform since the man himself took control.

References

Apache sparkTM - unified engine for large-scale data analytics (no date) *Apache SparkTM - Unified Engine for large-scale data analytics*. Available at: https://spark.apache.org/ (Accessed: 20 May 2025).

Elon Musk (no date) *Forbes*. Available at: https://www.forbes.com/profile/elon-musk/ (Accessed: 19 May 2025).

What is Hadoop Distributed File System (HDFS) (no date) *Databricks*. Available at: https://www.databricks.com/glossary/hadoop-distributed-file-system-hdfs (Accessed: 20 May 2025).

Woolf, N. (2016) *Elon musk twitter rant a 'case study' in how not to handle a crisis, experts say*, *The Guardian*. Available at: https://www.theguardian.com/technology/2016/jul/07/tesla-elon-musk-autopilot-death-crisis-management (Accessed: 23 May 2025).

Appendices

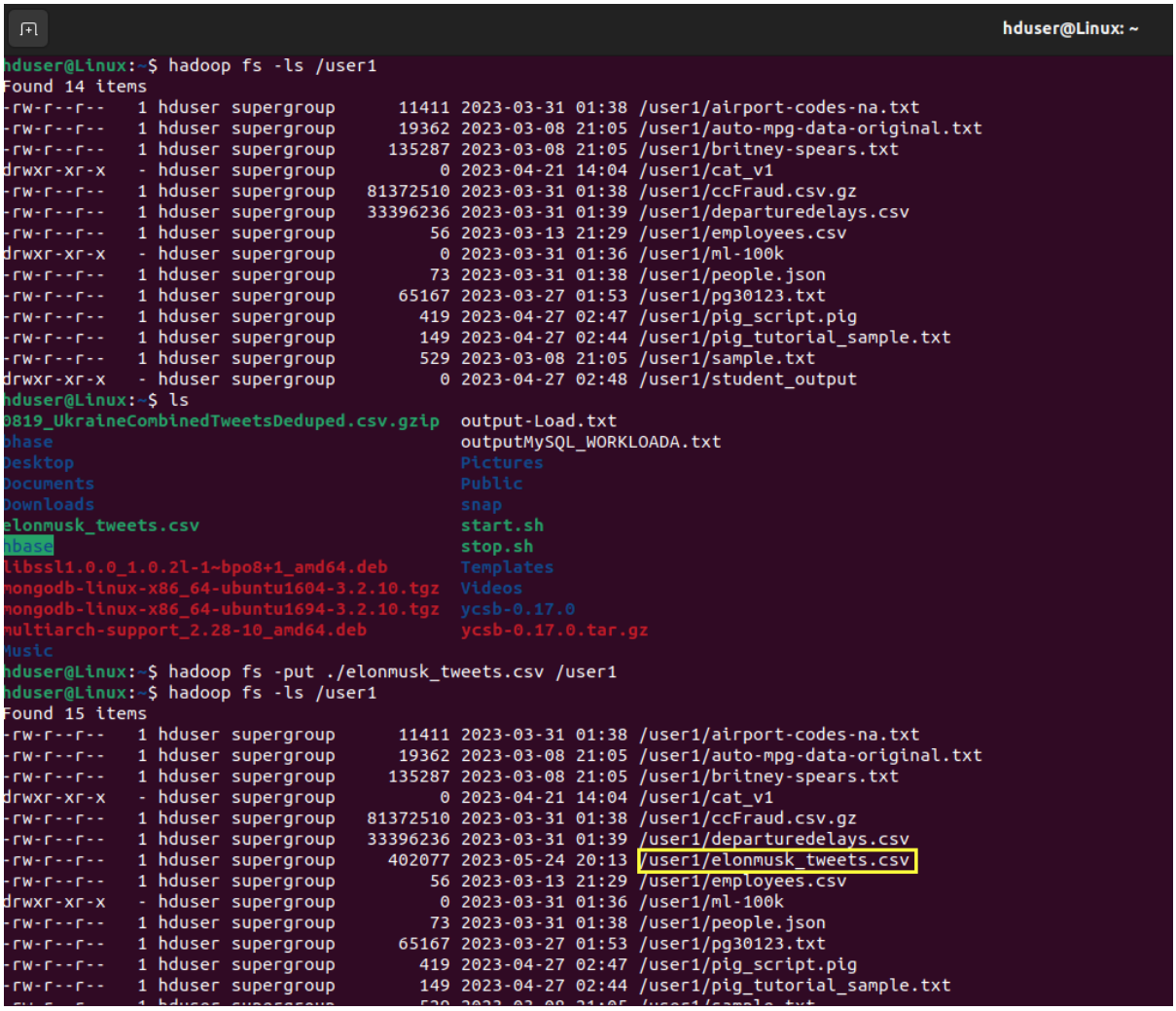

Figure 9 – HDFS

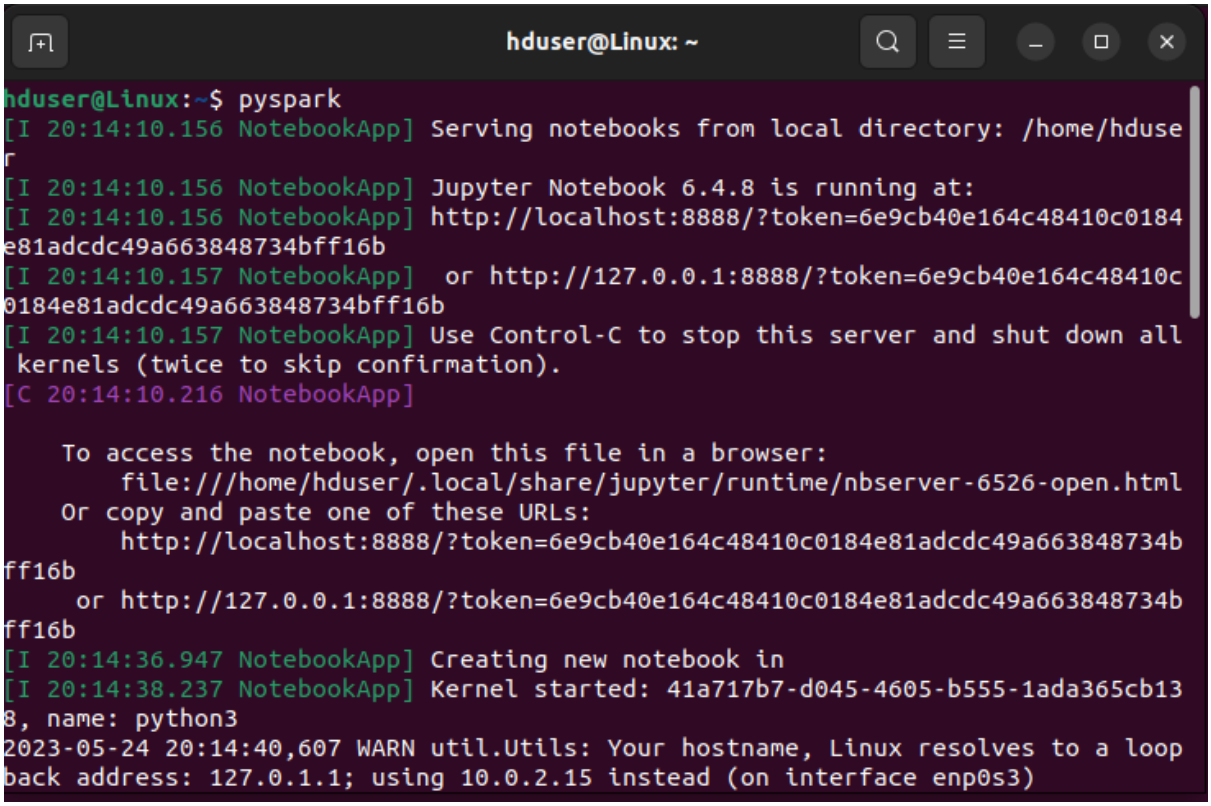

Figure 10 – Pyspark

Link to dataset used: https://www.kaggle.com/datasets/kingburrito666/elon-musk-tweets