**Sentiment analysis of one year of twitter data from the Ukraine war.**

**Word count 2719**

**Twitter\_Archive-Copy5.ipynb had most of the work the dashboard is at the end of the copy6**

*Data sourcing:*

My intention was to source the data from twitter using their API. But recent changes at twitter meant regular twitter developer account holders no longer had access to historical data , however it is still open to students doing academic research or developers willing to pay over 40K . I applied for Academic research status shortly after CA2 was released and unfortunately it is still under review. (“Twitter API Academic,” n.d.) There have been a lot of staff cuts at twitter recently and combined with every student interested in twitter data now applying at the same time which may have something to do with this delay.

See code in notebook for the anticipated API process in jupyter notebook

I attempted to download twitter data from the Internet archive but don’t have fast internet speeds at home or work to download the data required from this source. This would have been my preferred dataset .

I reluctantly sourced my data from Kaggle , it is 18.2G of tweets in relation to the Ukraine war.(Wando, n.d.) . The data is open source and not subject to licence.

The data was downloaded as a folder containing a .csv.gzip file for each days tweets gathered in 2022. The 18G was too big for my Ubuntu virtual machine so I stored it on an external hard drive.

My second dataset also came from Kaggle it’s the Ukraine weather dataset which goes back to the , again there was no licence to us this data.

*Loading the data:*

I wrote a function to loop over the gzip files, each file was unzipped and temporarily loaded as a pandas dataframe. I then filtered the dataframe for English language and location Ukraine, saving out only the date and tweet text fields by appending the result to a pyspark dataframe. I saved to a PySpark Dataframe , which is equivalent to a SQL table in a relational database.

I chose this data structure because

* Being a distributed computing platform it can stores and work with very large datasets or databases across a distributed storage enviroument such as Hadoop HDFS where as Pandas dataframes reside in memory and while faster are limited to the size of available memory in your computer.
* PySpark takes advantage of parallel processing across many machines , speeding up the process.
* PySpark works well with other big data processes , ie streaming and SparkSQL

But these advantages only work for big datasets , for small datasets pandas will outperform Pyspark.

*Data Storage:*

I stored the data as a paraquet file to the HDFS in /user1 directory as Ukraine\_twitter.parquet . I used the parquet format because it’s optimised for use with Hadoop HDFS, this makes it an efficient storage type when working with large and complex datasets. Its storage format is by columns.

*Advantages of using parquet:*

* It uses columnar storage structure , when selecting columns in analytic process is faster to read over row based csv formats.
* Integration and compatibility: Parquet is used by Hadoop MapReduce, Apache Spark, and Apache Impala allowing easy sharing between these processes.
* Parquet stores metadata with its files, this can make for faster queries on large datasets.
* Repeated occurrences of a value in a column are stored only once, this can save a significant amount of space, making it a cost-effective solution for storing large datasets.

On inspection the saved file appears to have zero K in storage but it’s actually a directory and the data is contained within a series of part files (part-00001-xxx.snappy.parquet) inside the directory.

PySpark does this to take advantage of the distributed file system , big files are broken up and saved to the multiple machines in the Hadoop system. This is an efficient process as the parts of the file are saved in parallel to each machine. It also makes the saving process more fault tolerant over all , because If one machine fails to write , then only that part of the data is lost , all the other data will be ok. The parts become one file again when read back into PySpark. Similarly with a x.csv file saved to the HDFS, it creates a .csv directory containing the actual CSV files split out in parts.

It is possible to write as one file with the .coalesce(1) command.

*Programming Language: (pyspark)*

I chose PySpark over Pig and Hive for the following reasons.

* PySpark uses Python as its primary language and it’s the language im most familir with .Its also the most popular languages in the data science community because it easy to work with. In contrast, Hive uses a language similar to SQL called HiveQL, Pig uses its own language Pig Latin.
* PySpark has many libraries for machine learning giving it a distinct advantage over Pig and Hive which have only limited ML capabilities .
* PySpark is compatible with Parquet, CSV, JSON and other file formats, and big data tools like Hadoop which gives it greater flexability.
* Its use of RDD (Resilient distributed datasets) This is an immutable distributed collection of objects that allows data to be distributed across a cluster and processed in parallel.

While PySpark certainly has its advantages, the choice between PySpark, Pig, and Hive should be made based on the needs of a given project. Hive and Pig are great for structured and semi-structured data, respectively. If your project requires handling complex data types or unstructured data, PySpark is be a better choice.

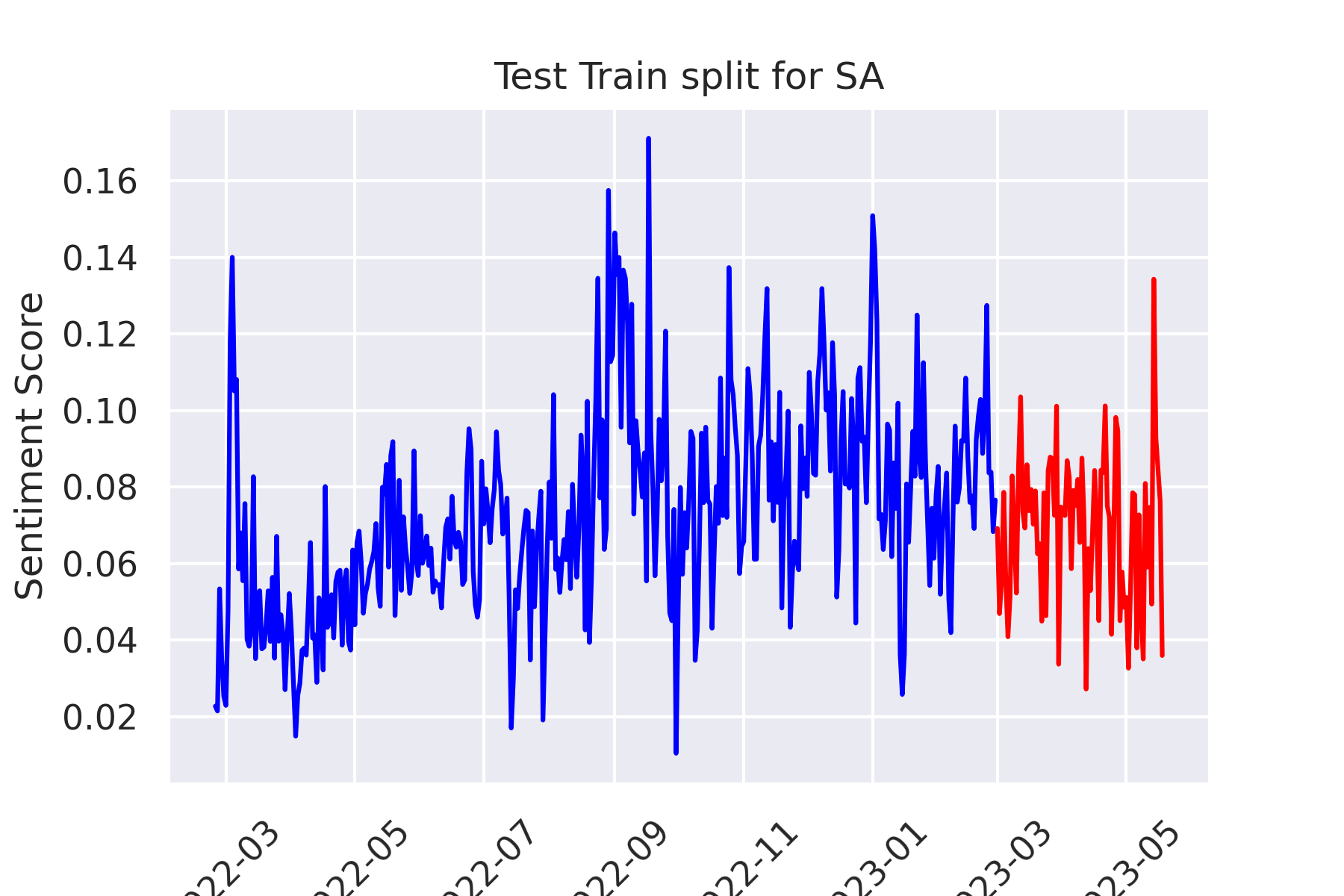
*AR Model Implementation:*

*My initial dataset consists of 1 year and 3 months of tweets from Ukraine. After sentiment analysis is performed the resulting dataset is a univariate time series consisting of just tweet sentiment at regular intervals of one day.*

*Auto regressive models are particularly suited to this type of data which is why I use them here.* *The aim of this models is to evaluate the underlying pattern of the time series (like seasonality and trend) and from this to make forecasts.*

*In this project I compare the results from four auto regressive models (AR) with the use of the statsmodels module with SARIMAX AND ARIMA*

*I split my data into one year for training and the remaining 3 months for testing.*

**

*Checking for Stationary:*

When the statistical properties (Mean,Variance,covariance) of a time series don’t change over time it’s said to be stationary. This suggests there is no trend or seasonal effect just constant fluctuation. Many predicting models require data to be stationary in order to make an accurate prediction. If data is non-stationary it’s possible to detrend it before using it for predicting.

The Augmented Dickey-Fuller (ADF) test and the Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test are both used to assess the stationarity of a time series. However, they have different null hypotheses and interpretations. The twitter time series was tested (see jupyter notebook) and the results are analysed below. Larger datasets typically give more accurate ADF results but a year and 3 months of data will be enough a rule of thumb is to be over 50 data points.

**Augmented Dickey-Fuller (ADF) Test:**

Null Hypothesis (H0): The time series has a unit root, i.e., it is non-stationary.

Alternative Hypothesis (H1): The time series does not have a unit root, i.e., it is stationary.

When the p-value of the ADF test is less than the significance level (usually 0.05), you reject the null hypothesis and conclude that the time series is stationary. In this case, the ADF test p-value is 0.0015, which is less than 0.05, indicating that the time series is likely stationary.

**Kwiatkowski-Phillips-Schmidt-Shin (KPSS) Test:**

Null Hypothesis (H0): The time series is stationary or trend-stationary.

Alternative Hypothesis (H1): The time series is non-stationary.

When the p-value of the KPSS test is less than the significance level (usually 0.05), you reject the null hypothesis and conclude that the time series is non-stationary. In this case, the KPSS test p-value is 0.01, which is less than 0.05, indicating that the time series is likely non-stationary or trend-stationary.

The results of the ADF and KPSS tests are conflicting in this case. The ADF test suggests the time series is stationary, while the KPSS test suggests it is non-stationary.

The results are not clear at this stage, To make a judgment I will consider the following steps:

Examine the time series plot visually to identify trends, seasonality, or any other non-stationary behaviour. On inspection it is hard to determine.

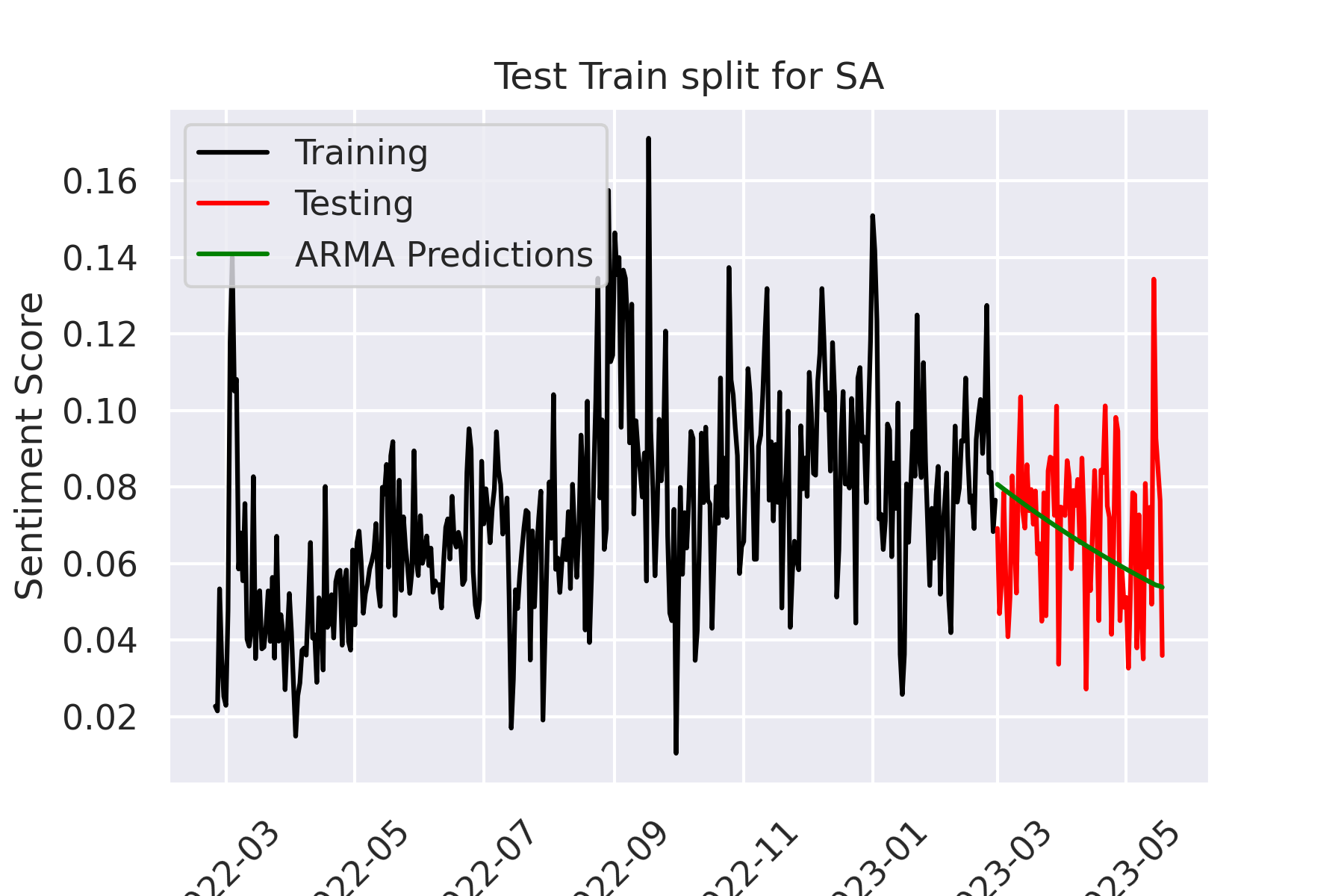
Further investigation could involve applying a Box-Co transformations.

*Forecasting:*

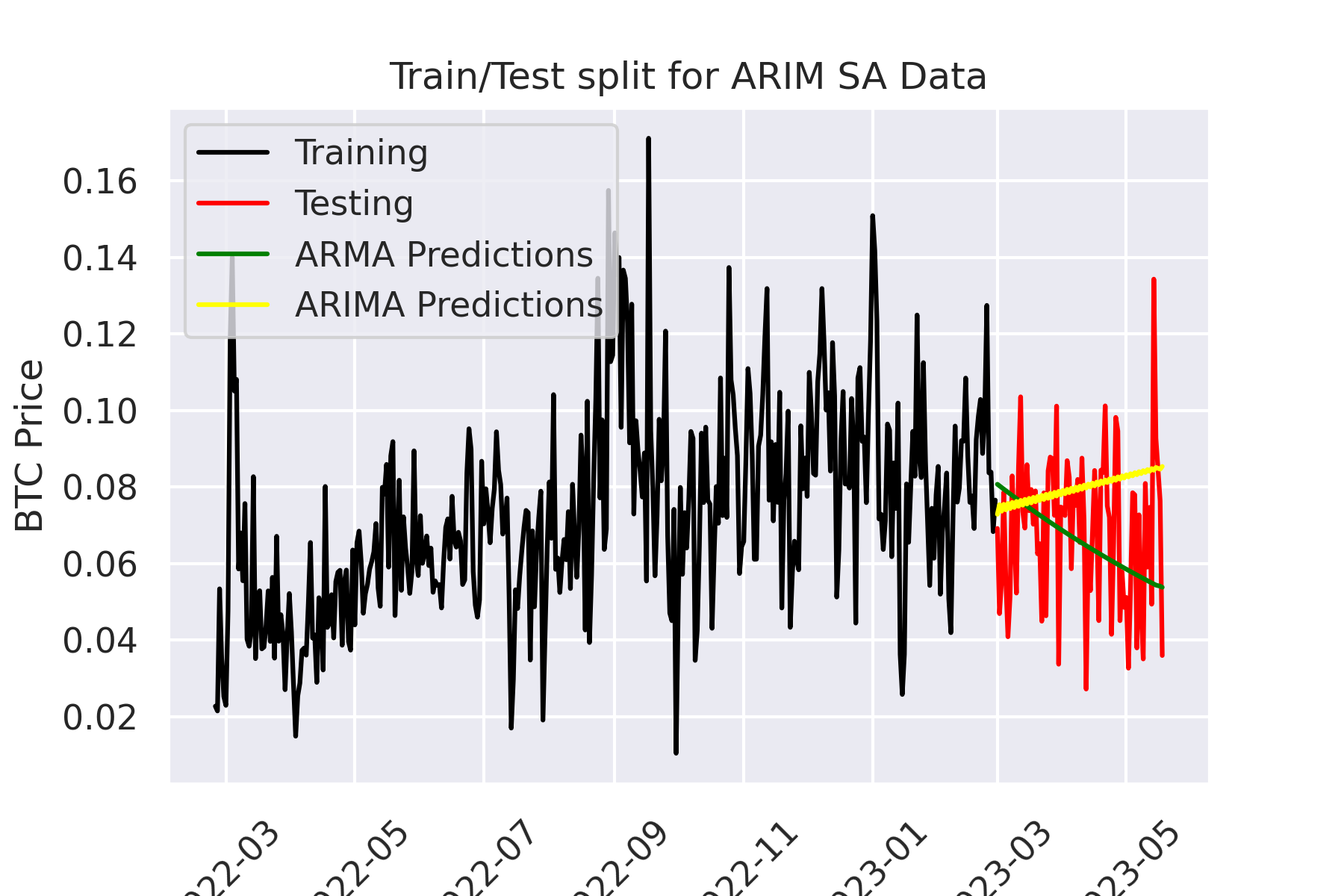
*1. ARMA (AutoRegressive Moving Average): this models is a blend of Autoregressive (AR) and Moving Average (MA) models. Auto Regressive refers to the use of past values to predict future values. The predicted values are a linear weighted combination of the past values . AR models use the dependency between an observation and the previous time period observations. This is similar to a linear regression, but where the feature inputs are historical values.*

*To predict the next step in a sequence , the MA models use the dependency between an observation and a residual error (white noise) from a moving average model applied to lagged observations The idea here is that ARMA uses a combination of past values and white noise in order to predict future values. Autoregression models historical data while the white noise models shock events like pandemics, market crashes , ect.*

.ARMA models are suited for univariate time series without trend and seasonal components.



2. ARIMA (AutoRegressive Integrated Moving Average): This is similar to ARMA except there is a trend component .However if a time series has a trend component , it needs to be made stationary , ie , the mean, variance, and covariance are all constant over time. This can be achieved by differencing the data (i.e., subtracting the previous observation from the current observation). Hence, ARIMA includes the differencing step in the model itself. It's typically applied to non-seasonal time series data.



3. SARIMA (Seasonal Auto-Regressive Integrated Moving Average): This model builds on the previous two by adding seasonal parameters: SAR (seasonal autoregressive order), SD (seasonal differencing order), and SMA (seasonal moving average order).

SARIMA models are best suited for time series with that have trend and seasonality.

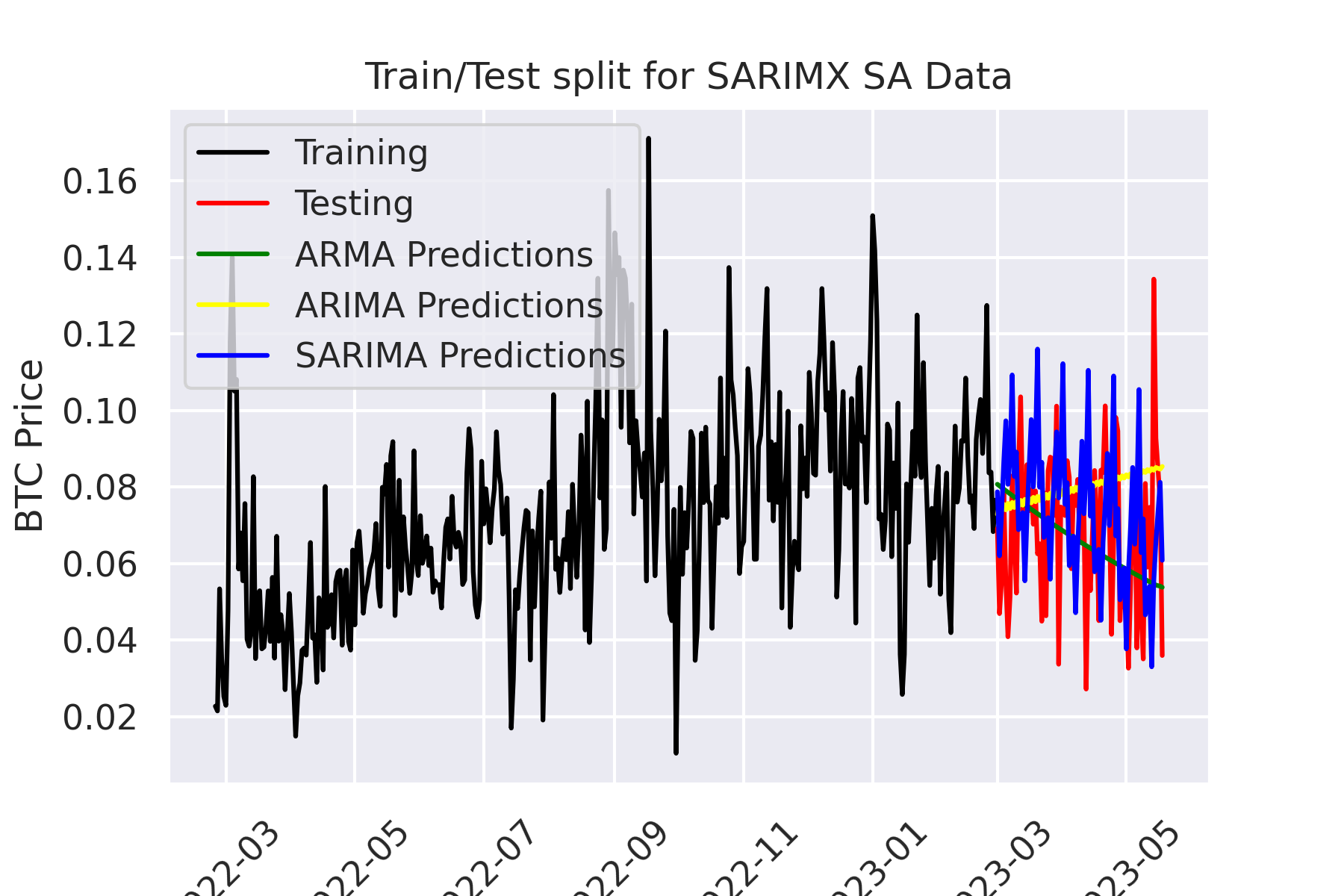
The order parameter in the SARIMAX model has three components: (p, d, q), and each component stands for the following:

p is the order of the AutoRegressive (AR) term. It refers to the number of lags of the dependent variable to be used as predictors.

d is the order of differencing. Differencing is the method of transforming a time series dataset. Its used to remove the series dependence on time, including structures like trends and seasonality.

q is the order of the Moving Average (MA) term. It refers to the number of lagged forecast errors that should go into the ARIMA model.

In this instance the best rmse score was obtained from this model when the order performance was p=2 , d=2 , q=2



*Analysis of change sentiment:*

fig xx below is a histogram of the sentiment over the entire year , it appears to be normally distributed , with a strong slightly positive median. This is remarkable give the country was plunged into war over this period of time.

A picture containing square, rectangle, line, screenshot

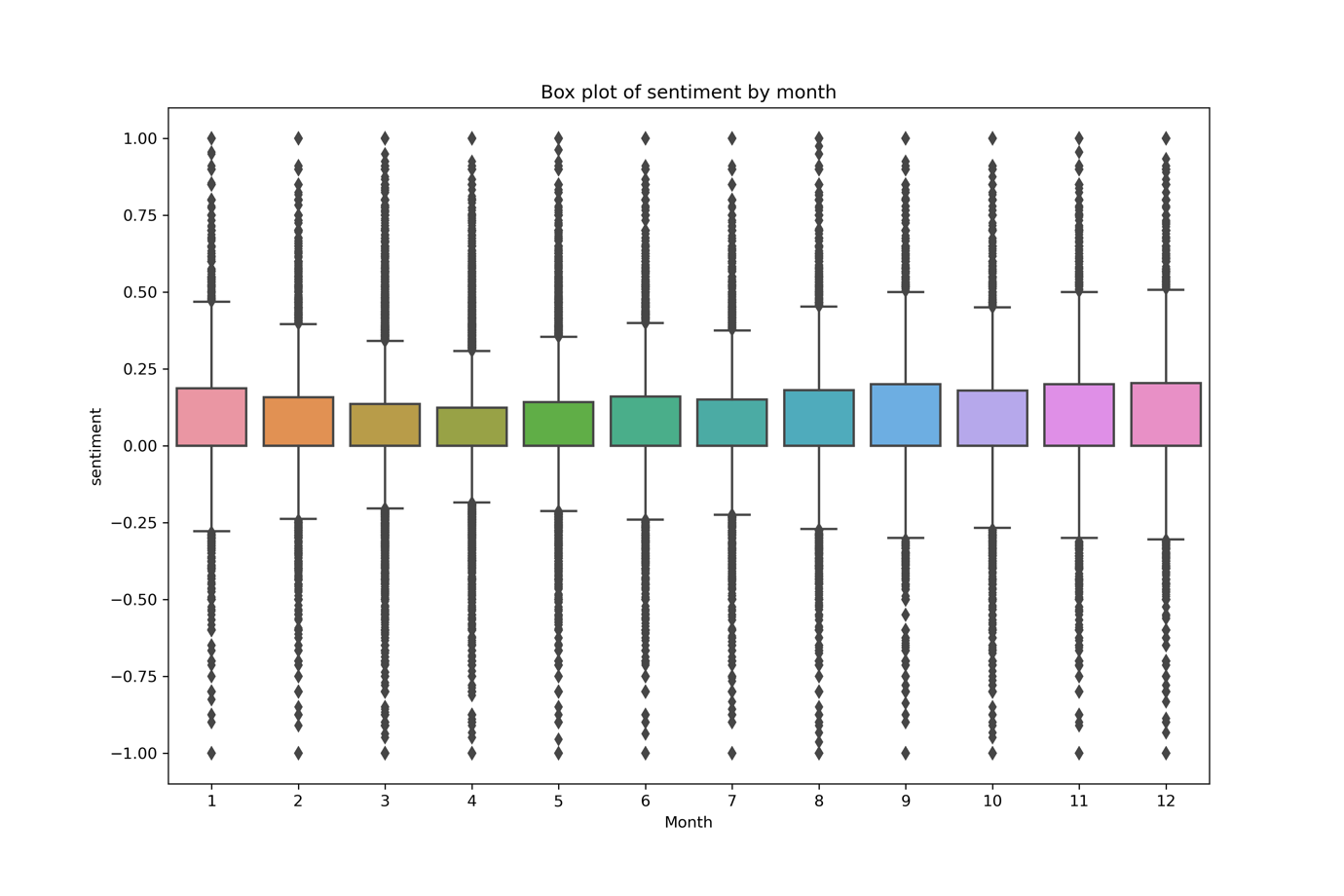
Description automatically generated

To get a visual idea of the change in sentiment over this period see fig xx below.

The box in the box plot represents 50% of the data i.e. the interquartile range with the remaining 25% above the box and 25% below the box.

The first thing to note is that 75% of tweets are neutral to positive in nature for the entire year. Defiance in the face of adversity ? by the time the war gets going in M2 and M3 Positivity can be seen to drop with the maximum and minimum interquartile levels moving closer to the box , while at the same time more extreme outlier sentiment is being expressed . As the year progresses the plots slowly return to where they started the year.

Key dates in the war can be tracked in this sentiment as the initial negativity of the war gives way to more positive stories as the Russians get pushed back. A new normal ?



**Key dates in the war.** (News, 2023)

Feb. 24, Russian President Vladimir Putin launches an invasion of Ukraine

March 2, Russia claims control of the southern city of Kherson

April The Russian pullback from Kyiv reveals hundreds of bodies of civilians

April 9, a Russian missile strike on a train station in the eastern city of Kramatorsk kills 52 civilians and wounds over 100.

On May 18, Finland and Sweden submit their applications to join NATO in a major blow to Moscow over the expansion of the military alliance.

On June 30, Russian troops pull back from Snake Island,

On July 22, Russia and Ukraine, with mediation by Turkey and the United Nations, agree on a deal to unblock supplies of grain stuck in Ukraine’s Black Sea ports

Aug. 9, powerful explosions strike an air base in Crimea. More blasts hit a power substation and ammunition depots there a week later

On Sept. 6, the Ukrainian forces launch a surprise counteroffensive in the northeastern Kharkiv region, quickly forcing Russia to pull back

On Sept. 21, Putin orders mobilization of 300,000 reservists

On Oct. 8, a truck laden with explosives blows up on the bridge linking Crimea to Russia’s mainland

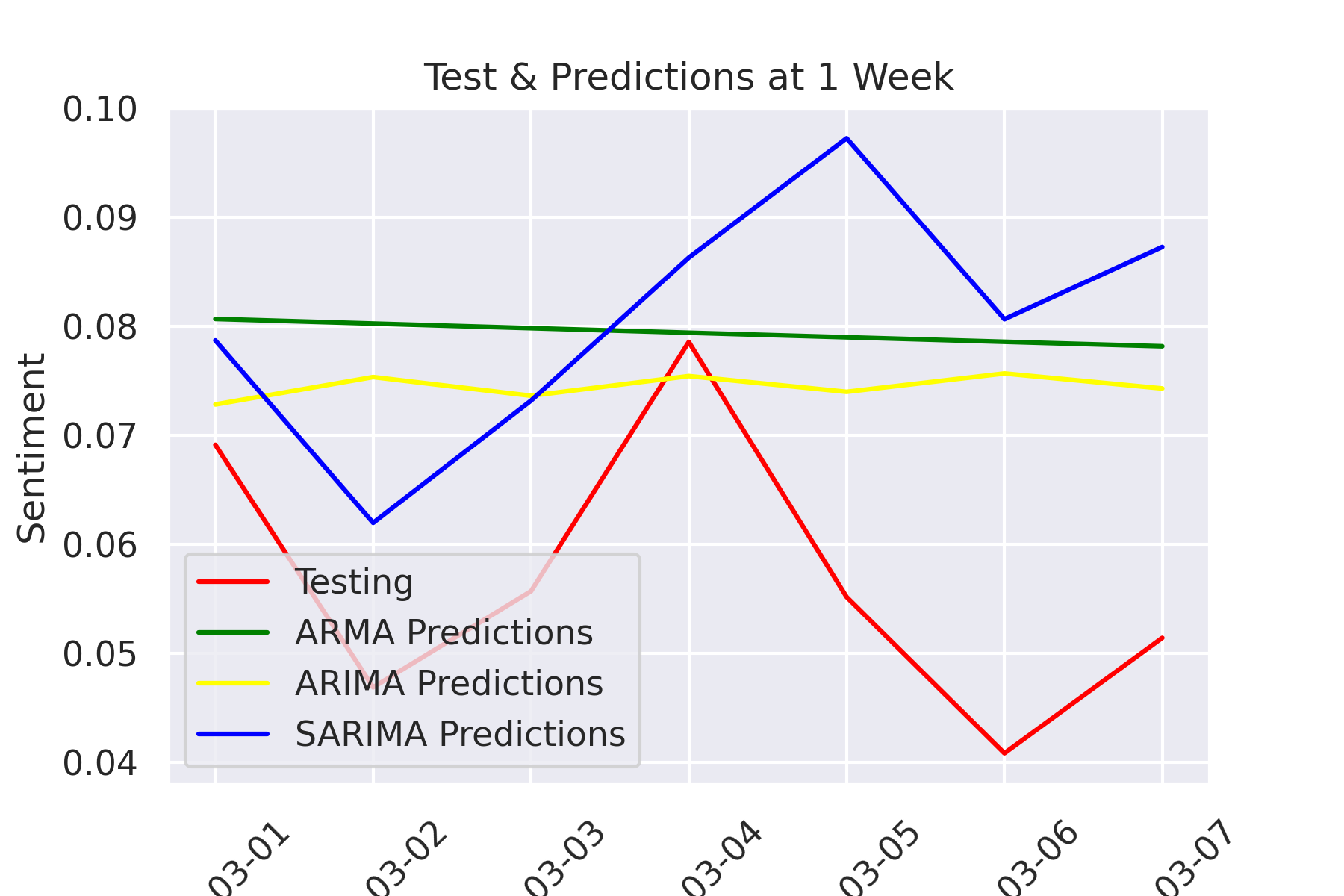
On Nov. 9, Russia announces a pullback from the city of Kherson

On Dec. 5, the Russian military says Ukraine used drones to target two bases for long-range bombers deep inside Russian territory

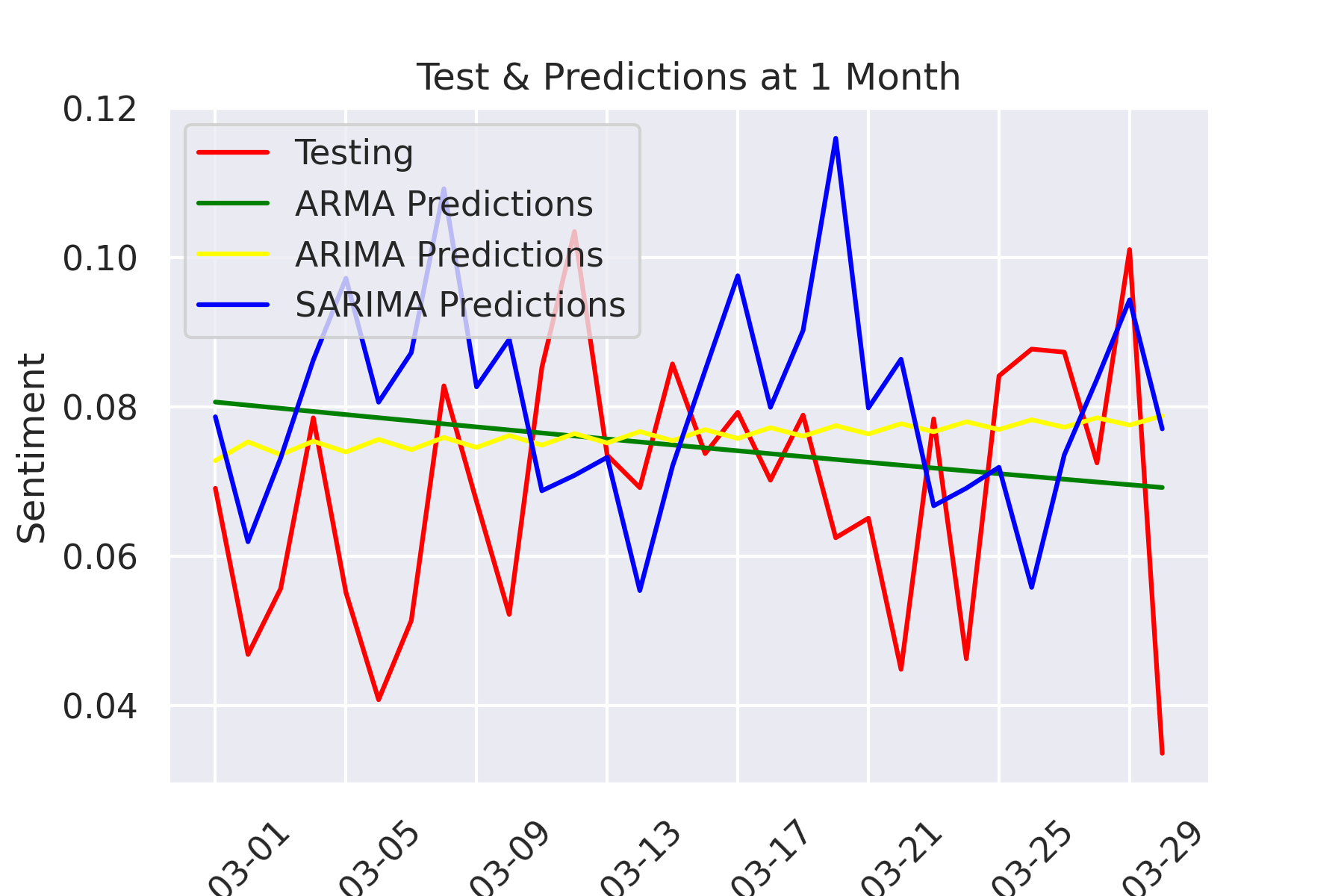
On Dec. 21, Zelenskyy visits the United States on his first trip abroad since the war began

*Presentation of findings:*

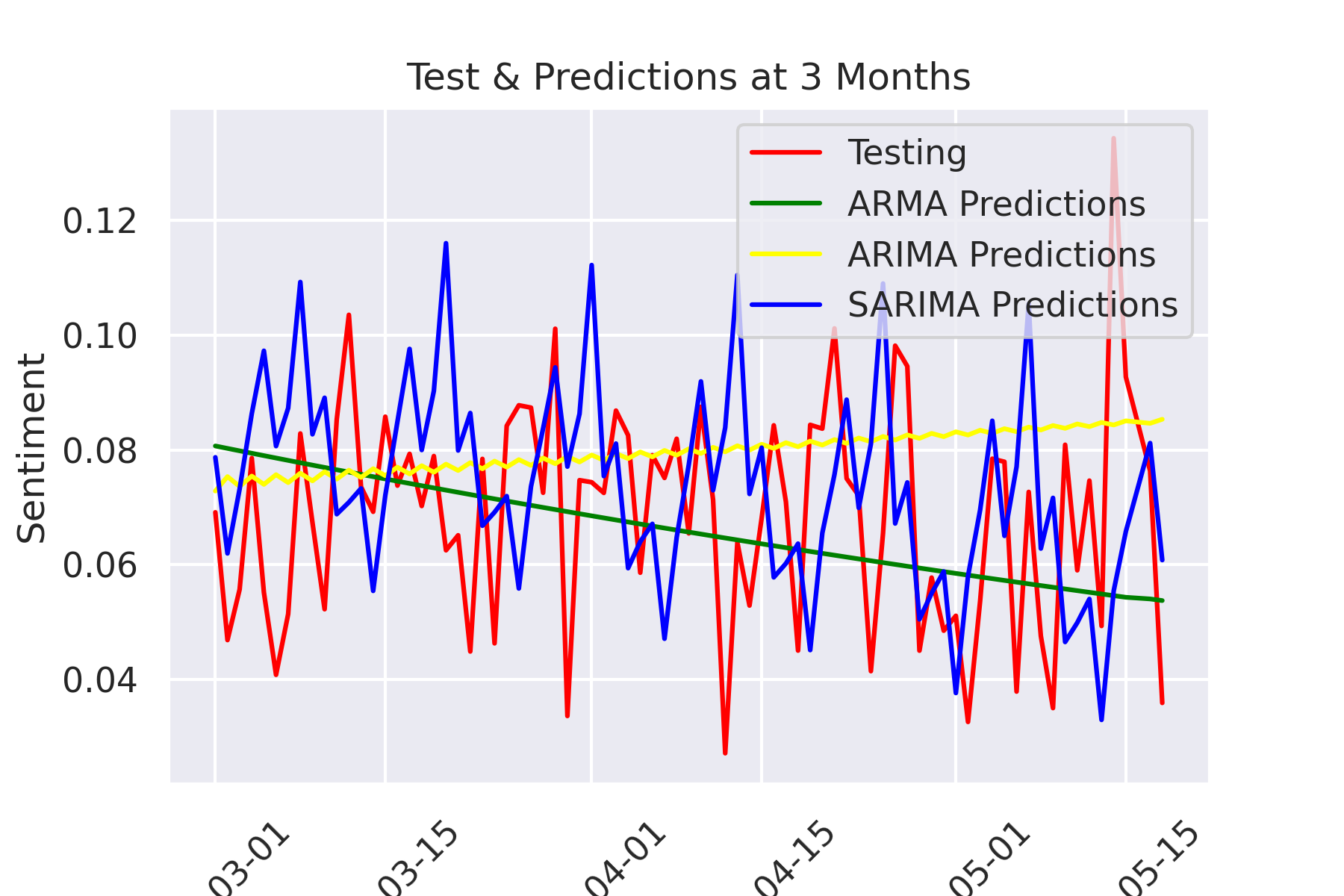
1 Week forecast. All 3 models forecast the sentiment higher than the actual data.



1 Month Forecast: It becoming clear ARMA has the better prediction. SARIMA is out of step with the actual predictions.



3 Month Prediction: it’s a close call on visual inspection which is the better predictor. Is better lo look at the RMSE results.



Model Results for rmse indicate ARMA has the lowest rmse and is therefore the best performing model in this case. This result would confirm the stationary property of the time series as ARMA models are best suited for univariate time series without trend and seasonal components.

A green rectangle with black text

Description automatically generated with medium confidence

*Dashbord:* See jupyter notebook for code. This Python script identify outliers in a given series of data. I uses a function that generates a plot with given series `avg` and `highlight`. `avg` is a series of average values, and `highlight` contains the outliers to be plotted.

The find\_outliers function detects the outliers. Where

`variable` is the data column from which we want to find outliers

`window` is the window size for the moving average and standard deviation calculations

`sigma` is the number of standard deviations away from the mean to consider a point an outlier, and `view\_fn` is the function used to view the result.

It first calculates the moving average (`avg`) of the specified variable with the given window size. The `residual` is calculated by subtracting `avg` from the original variable data, which gives the deviation from the moving average.

The moving standard deviation (`std`) of the `residual` is calculated with the given window size. Outliers are detected where the absolute value of `residual` exceeds `sigma` times `std`. These are points that are `sigma` standard deviations away from the mean.

Finally, the function calls `view\_fn` (defaulted to `mpl\_plot`) to generate a plot of the average values and the outliers, and returns the figure.

Conclusion: In conclusion : The access of Twitter data for this project was very difficult. I had worked on the API section first in anticipation of getting access to twitter which never materialised , Then failed to down load the big datasets due to my internet speedsfinally settling on a dataset from Kaggle that was more manageable . My analysis was just on sentiment of people tweeting from Ukraine. I didn’t filter for any other variable and the biggest variable , the war was playing out over this period. The sentiment can be seen to match to key periods in the war . The best prediction result came form the ARMA model but in reality the other models ARIMA and SARIMAX performed well also all with a rmse of around

A picture containing line, diagram, plot, screenshot

Description automatically generated

News, A., 2023. Key moments in a year of war after Russia invaded Ukraine [WWW Document]. AP NEWS. URL https://apnews.com/article/russia-ukraine-war-one-year-anniversary-timeline-a1304c6fb319bf1c0e93635f6f6a2633 (accessed 5.26.23).

Twitter API Academic [WWW Document], n.d. URL https://developer.twitter.com/en/products/twitter-api/academic-research (accessed 5.24.23).

Wando, B., n.d. 🇺🇦 Ukraine Conflict Twitter Dataset [WWW Document]. URL https://www.kaggle.com/datasets/bwandowando/ukraine-russian-crisis-twitter-dataset-1-2-m-rows (accessed 5.24.23).