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 Storage and Processing Advanced Data Analytics
Assessment Title:	CA2 integrated assesment
Lecturer Name:	**David McQuaid** **Dr. Muhammad Iqbal**
Student Full Name:	Faelan Redmond
Student Number:	SBA22190
Assessment Due Date:	26/05/23
Date of Submission:	26/05/23

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.

Sentiment analysis of tweets related to Ukranian refugees

Name: Faelan Redmond

Student #: SBA22190

Github: https://github.com/faelanred/MSc-Data-Analytics/tree/main/CA2_SEM2

Introduction

In the age where social media use is rampant, society and therefore public opinions are more visible than they have ever been before. With current events in mind, an interesting area to study is the changes, if any, in public opinion towards refugees from Ukraine. Since the beginning of the conflict in February of 2022 this is a topic that has sparked debates from both sides and seems to have been subject to increased anti-refugee opinions in recent months. This paper proposes a method to quantify any changes in the publics opinion in this area by using the social media platform as a real time probe.

The vastness of twitter, having 450M active users at the time of writing this, means that it has become an invaluable go to tool when attempting to track public opinion in real time and/or at scale. Taking advantage of this, our study analyses the publics sentiment towards Ukrainian refugees through an exhaustive study of tweets that span over a year since the beginning of the conflict. The insight gained from this tweet will highlight some of the societal struggles faced by this group and may be a valuable tool for informing policies in accepting countries or charitable efforts.

The understanding of this sentiment will provide only part of our solution, Far better than acting reactively, is being able to act proactively. In light of this tools such as SARIMA and Prophet will be used in order to forecast public sentiment over the short term, this provides us with a fresh and more relevant approach to sentiment analysis problems and will be more valuable to stakeholders when it comes to proactive policy making through future glimpses at public sentiment towards this group.

In essence, in this paper we are studying an interesting intersection of data science and social science. Using timeseries methods, we aim to arrive at a technique that can accurately forecast fluctuations in public opinion, the idea is that this work will provide insight into public sentiment and give stakeholders a greater opportunity to navigate it effectively. As such we hope this paper will prove to be a useful tool in greater understanding the publics feeling towards refugees now and into the near future and become a tool for helping them be better accepted into communities.

Data

Although there were many data sources proposed for this project, many of them proved to be unusable for many reasons. Due to recent changes in the twitter API none of the ‘hacks’ that allow users to gain access to historical tweets are functioning at the moment so this option was unavailable. Usually one would apply for access to the academic version of the twitter api, which does give historical access, however due to recent changes in staffing at twitter the customer service is uncharacteristically slow meaning that we could not gain access in time regardless of the fact that we applied on.

The twitter backup on archive.org was also an option, however to some this was unavailable due to restrictions in bandwidth and processing. Due to each day being approximately 3gb of compressed data, to collect a years' worth of data would be close to 900gb which isn't feasible on a rural internet connection. For those that do have access to fast and reliable broadband, this would have been an excellent option.

The data for this project was sourced from Kaggle (UA Ukraine Conflict Twitter Dataset | Kaggle), this is updated daily and has over 17gb (compressed)of tweets related to the conflict in Ukraine. We were able to filter these for a list of key word matches relevant to our subject area, netting approximately 500,000 tweets in the end. Due to long processing times experienced when using all the data, a random subset of 20% was retrieved using pyspark, this enabled much better performance on the virtual machine (VM), and allowed us to do much more with the data as before even group by operations took a very long time. This data was free to use under a CCO public domain licence (https://creativecommons.org/publicdomain/zero/1.0/).

Big Data Methods Used

This project was built on top of big data processing platforms, and NoSQL databases with big data processing in mind. These include Apache spark and Cassandra which were used to process and store/query the data respectively. This section will briefly discuss the technologies we used to fulfill this project, highlighting the miost notable features in the process.

Ycsb:

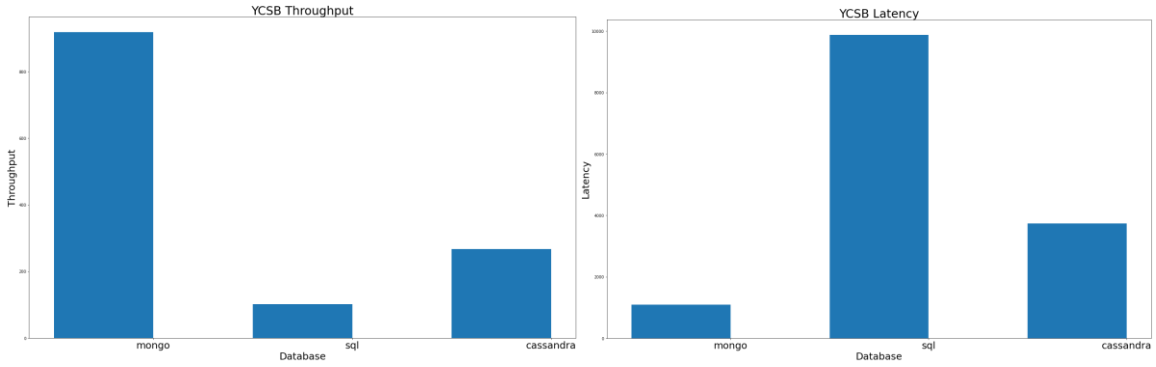
The Yahoo! Cloud Serving Benchmark (YCSB) is an open-source framework developed by yahoo, obviously, to facilitate performance benchmarking of data storage solutions under a variety of standard user specified workloads. This allows users to critically evaluate a variety of database technologies for their use case using these pre-specified workloads, evaluating strengths and trade-offs of each.

Workload	Ratio (in %) of each operation			
	Read	Modify	Scan	RMW
(A) Write Heavy	50	50	-	-
(B) Read Heavy	95	5	-	-
(C) Read-Only	100	0	-	-
(D) Read Latest	95	5	-	-
(E) Short Scans	0	5	95	-
(F) Read-Modify	50	0	-	50

This natively supports a multitude of database technologies, in our case it has been used for mySQL, mongoDB, and Cassandra, but it is easily extended by developers to facilitate new workloads or database systems. The fact that YCSB provides standard measures for performance across each system it that is tested with it is what grants its popularity. Users are now empowered to make informed decisions regarding what systems are right for their application, and what trade offs they can afford to make in the development process. It are these pros listed above that make this such a widely used tool in industry.

The operation of this tool is broadly defined by two phases, The workload phase, and the benchmarking phase. The workload phase defines one from a set of 6 workloads seen above, each of these will test a specific characteristic of the database such as read or write performance. The benchmarking phase takes place when YCSB runs these workloads on a selected database, and records performance metrics such as throughput and latency in the process.

Below are the YCSB Benchmark scores for mongodb, sql, and cassandra. Cassandra was chosen for the database in this project as it provided a good comprimise between the efficiency of mongo db, and the ease of use of sql. It is clear that mongo would be the best preformer, but cassandra has the advantage of being veryu sililar to sql in use.



PySpark:

Pyspark is an open-source python interface for Apache spark, this tool allows for the processing of large datasets on distributed computing clusters which has benefits for both performance and fault tolerance. This is achieved by breaking the dataset up into chunks called Resilient Distributed Datasets (RDD) which then get spread across the frameworks clusters, ensuring that no single machine is relied upon for data processing. This means that the possibility of processing failures and data loss is greatly reduced.

PySpark relies on an innovation of googles MapReduce boilerplate called the directed acyclic graph (DAG), which is responsible for breaking the data down and parallelising the processes. This segments the data into its RDD's and allocates them to cluster nodes for processing, it then takes the result of each computation and combines them to present a single final output. This technology makes spark extremely scalable, allowing for its capabilities to be increased by scaling horizontally, simply adding more cluster nodes. The allocation of resources is determined by a component called spark scheduler, which is similar to hadoops YARN.

Overall the PySpark architecture can be broadly broken into two components, The driver program and the executor nodes following a leader-follower architecture. The driver program coordinates the assignment of tasks to nodes while keeping track of RDD locations. Meanwhile the executor nodes serve to execute these allocated tasks, store data, and manage any computation processes. This results in a resilient storage solution & distribution of tasks meaning that the system has a high fault tolerance.

Cassandra:

Cassandra is an open-source NoSQL database which can efficiently manage large volumes of unstructured data across servers. The risk of data loss is reduced by distributing data across nodes in

a cluster, which can avoid mass data loss if a single node fails. Cassandra is ideal for write heavy workloads such as IoT applications. A hashing method is used to enable parallel data reading/writing across nodes for increased performance when compared to more traditional relational databases such as SQL, this also replicates data to other nodes adding with data availability in the case of a node failure. Like spark, its capacity can be easily increased through horizontal scaling.

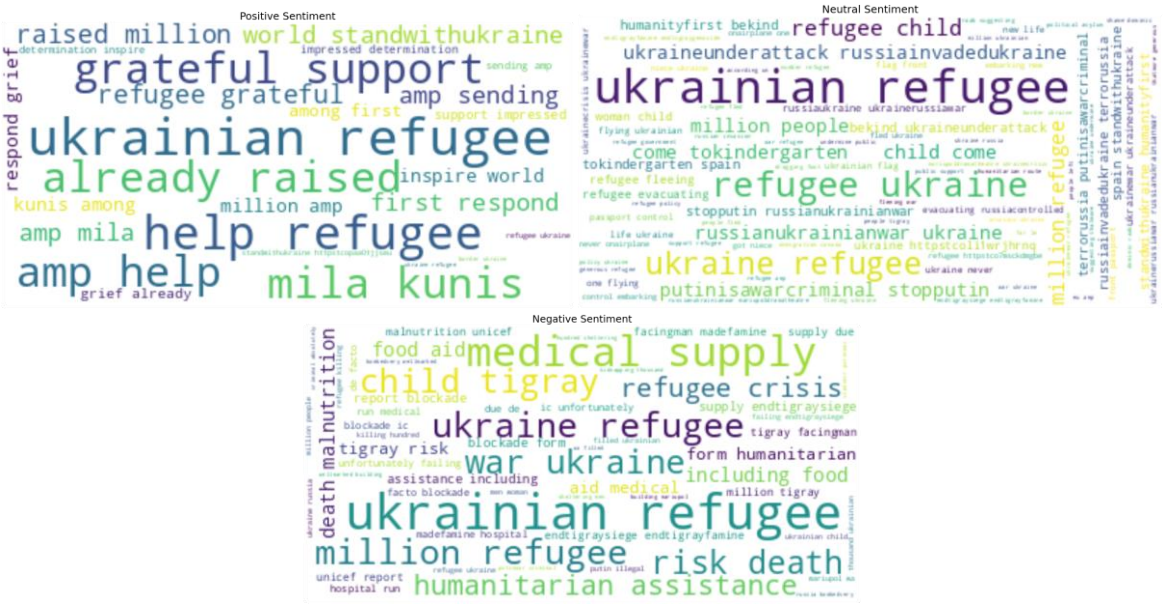
Communication between nodes is handled by a component called gossip, this is a communication protocol that shares the location and information relating to different nodes in the database system, this may be considered similar to YARN in Hadoop or spark scheduler in pyspark. The simple node, datacentre, and cluster architecture of Cassandra proves extremely effective. This system follows a ring or peer to peer architecture with each node being effectively identical and communicating equally with others. Regardless of data location, each node can handle read and write operations which ensures high availability and fault tolerance. These qualities have mentioned make Cassandra databases a robust, fault tolerant, and highly scalable solution for use cases that require real time constant data transactions.

Results & Discussion

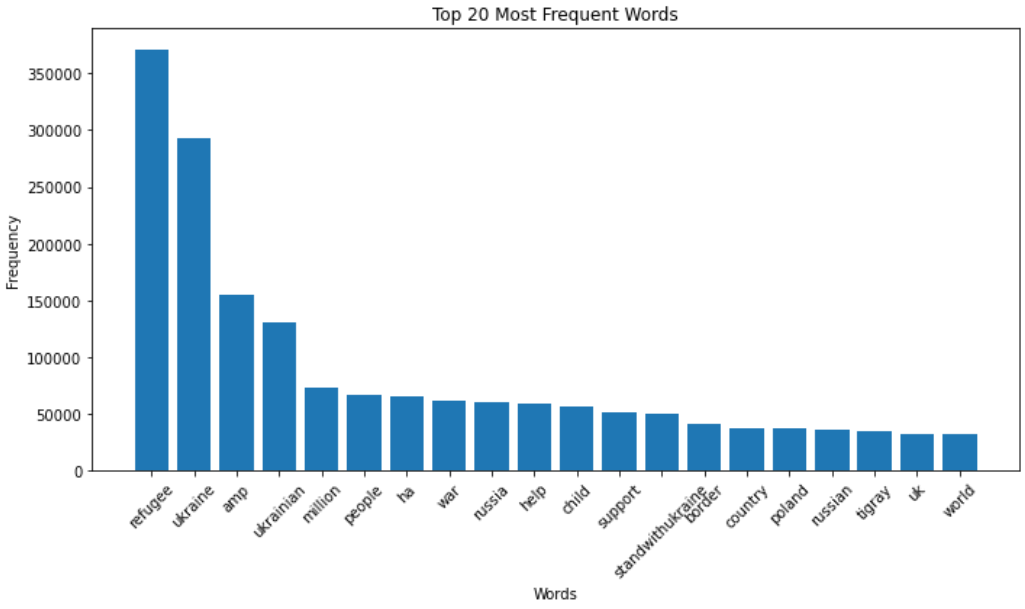
Sentiment analysis & Exploratory Data Analysis

The most important step of any data analysis project is that of Data preparation and eda. The data was filtered using a simple loop in python which parses through each file in the dataset and checks every tweet within for a key word match. This part of the process ensured that we only had comments in english, and that were relevant to the subject in question, refugees. Following this, standard text analysis steps were carried out to remove any information within the tweet that would be of no use to us. This includes special characters, twitter usernames, retweets, standardising the text to lower case, carrying out lemmatization, and removing stop words. After the text was clean, a popular tool called vader was used to analyse the sentiment of each tweet, assigning a number between –1 and 1, representing most negative and most positive respectively. Vader was a good choice as it uses pre trained models that are designed to deal with social media data, and understands things like sarcasm or context in text.

After we had obtained a sentiment score for each tweet, we could begin to look at the distribution of people feelings towards this group. The plot below shows the word cloud obtained for tweets categorized into the positive, neutral, and negative categories respectively.

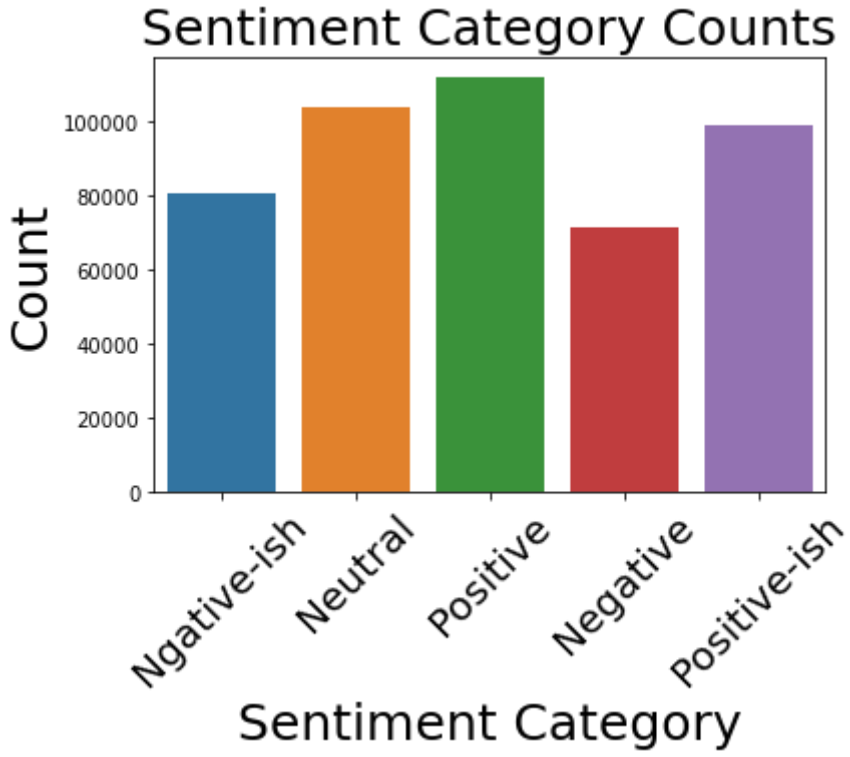


These show the most commonly used words in each category, unfortunately here they dont provide a lot of context. The positive tweets seem to focus on helping, raising support or funds, mila kunis (possibly praising her for some actions) and inspiration. The negative tweets focus on the war, food, humanitarian assistance, death, and medical aid. The neutral tweets focus on the number of people, humanity, kindness, and the war. It appears as though the attitudes towards the refugees themselves are not negative here, but the negative tweets stem from people criticising the current situation and having it cause issues such as a refugee crisis. In total the top 5 words were found to be refugee, Ukraine, amp, Ukrainian, and million. The top 20 words may be seen on the plot below.

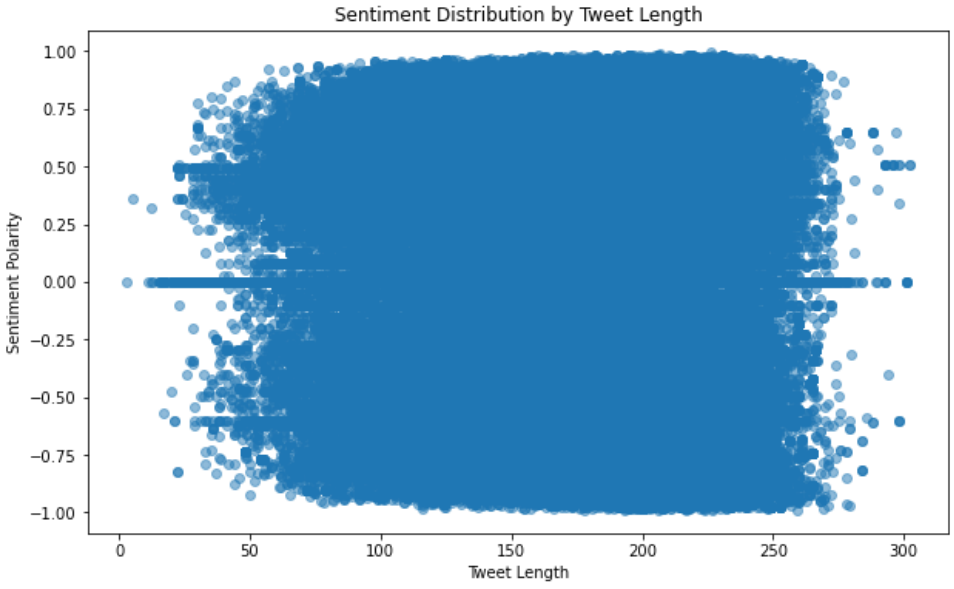


Lets next look at the proportion of tweets in the dataset that fall into each category, this can also be useful in order to help us understand whether positive, negative, neutral or so on feelingsd are mmore popular in the dataset. From the plot below we can see that there is a relatively even distribution of sentiment within the dataset. This could be for many reasons but as mentioned above it may arise from the dataset containing tweets of both people praising the efforts humanitarian societies are making to help those affected by the war, and those heavily criticising the

war and the agressors. This conflict in subject may be having a cancelling effect of sorts which has caused such an even distribution, with positive having the most counts by only a small margian. This fact may complicate the project as ewe progress, especially in the timeseries forecasting section as the data when plot may just appear to be noise.

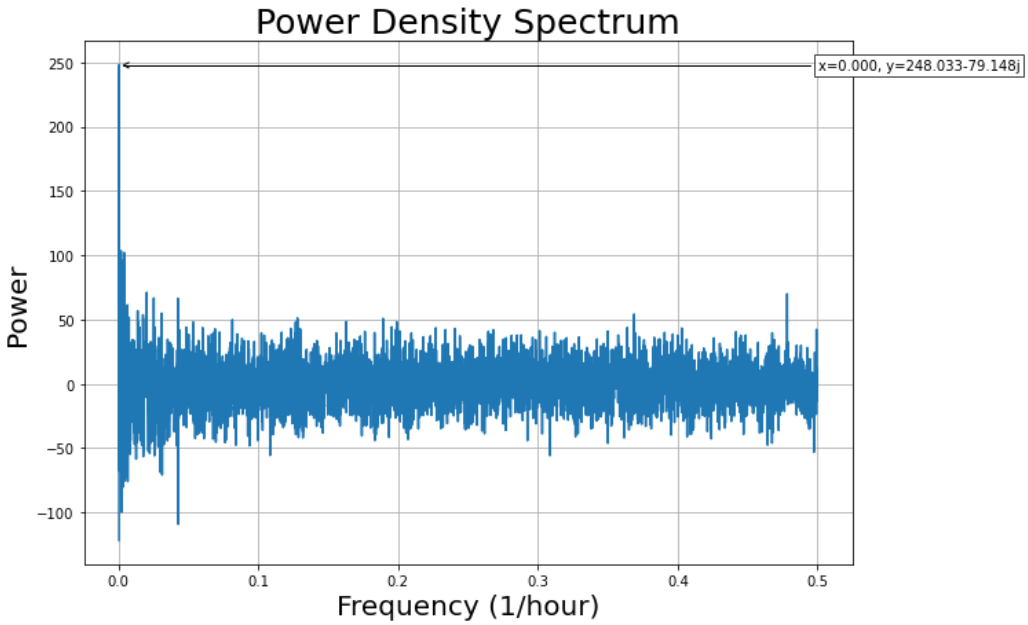


Finally, before we move onto the timeseries analysis, we thought it interesting to investigate the relationship between sentiment score and text length, curious if tweet length has any effect on the assigned sentiment. This may be useful to know as there could be a specific format of tweet that is related to positive/negative tweets. From the plot below we can see the results of this investigation where tweet length and sentiment have absolutely no relationship as points seem to just be evenly distributed across the area of the plotm meaning that based off the length of a tweet there is no way to predict its associated sentiment.

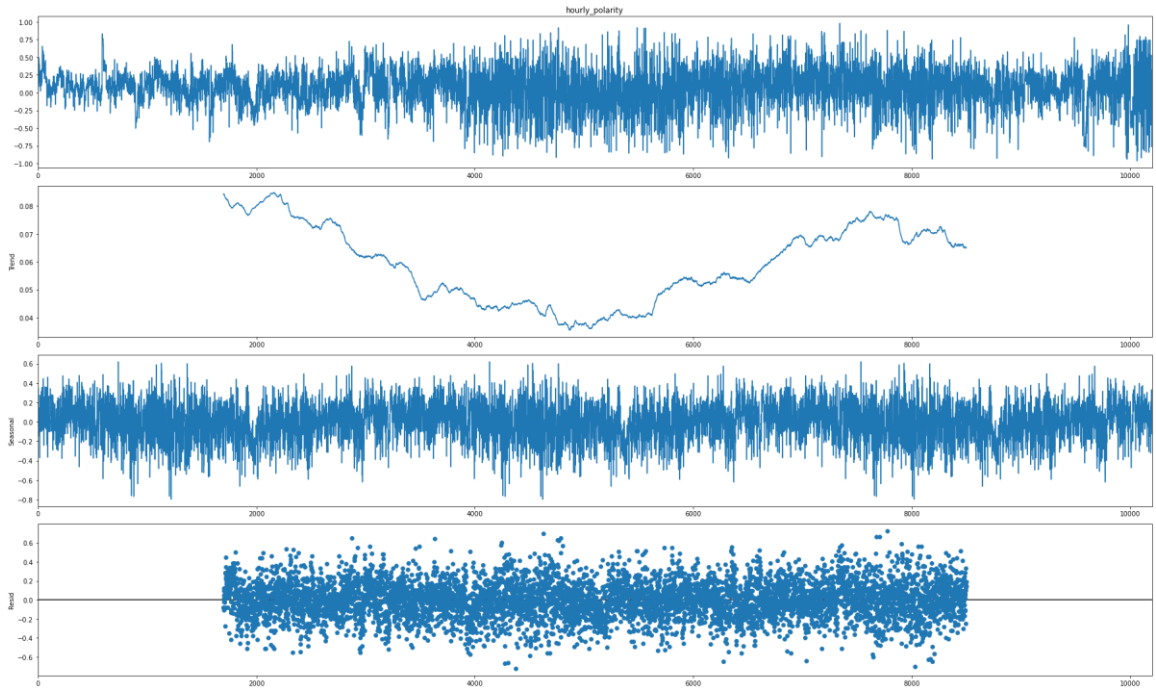


Timeseries Analysis and forecasting

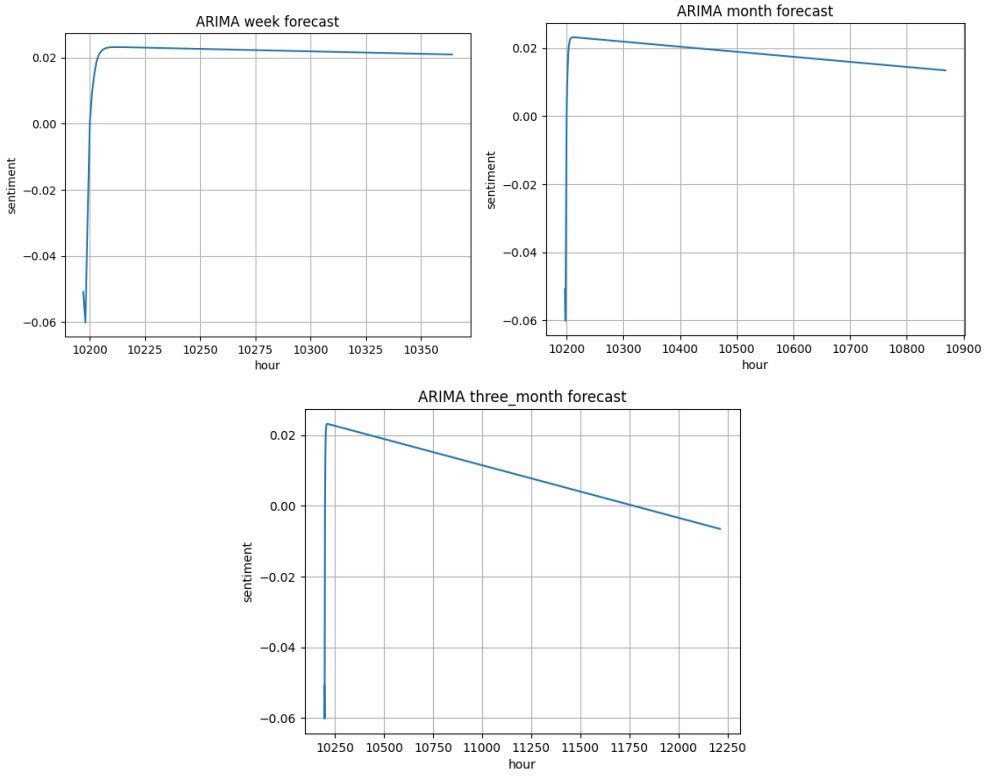
Now that we have explored the sentiment in the dataset, we move over to attempting to predict it in the near future. In order to allow this, it is important that we first investigate the properties of the timeseries so that we can make more informed decisions as we move forward. As the seasonal decompose function requires a period in order to run, we first ran a fast fourier transform on our timeseries in order to get the power density spectrum. We found from this that the data has a period of 3398.999 hours (141.625 days), this will be used to decompose the signal in the next step.



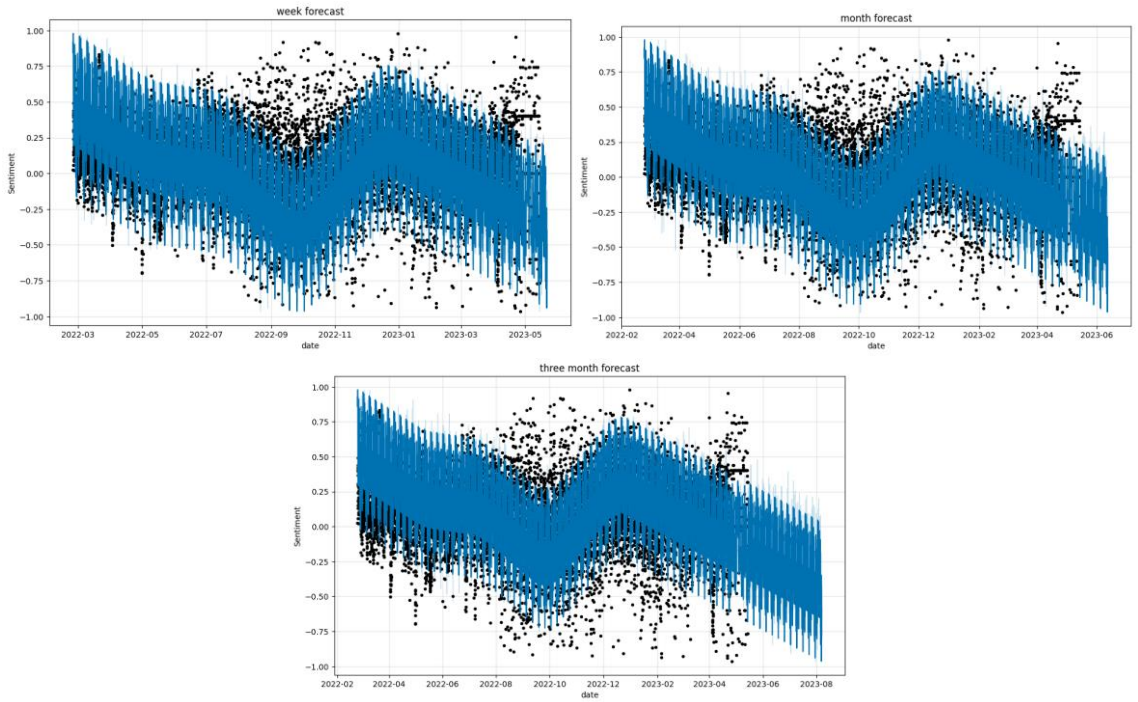
As the data appears to be additive, we decomposed the data using the additive model of seasonal_decompose from statsmodels, and with a period of 3399 hours. Based on the plots below we can see a couple of things. The timeseries itself appears to be very ‘noisy’ with there being no very obvious trend in the raw dataset. When the trend is extracted from the data there appears to be a U shape (polynomial trend) to the data, this is somewhat surprising as this is not at all obvious in the raw data. It is possible that this is some strange anomaly in the seasonal decomposition, or noise in the data somehow masks this but looking at it, i dont know if this is the case as the raw spectrum seems to oscillating around a centre line. Finally there is absolutely no seasonaluity in the data, very slight ungulations are seen buit that is hardly noticable.



Now that we have some insifht into the data, we can start forecasting. Due to the apparent trend in the data we must first check the stationarity, according to a KPSS test we get a score of 0.0477 which is less than the significance level of the test meaning that the data is not stationary. Due to this we must be careful with our choice of forecasting model, ARIMA has a differencing factor so we will use that, and we will also use pythons prophet library. Unfortunately we will see that neither of these methods worked particurlarly well despite trying to tune the parameters. Testing found that the best parameters for ARIMA could be estimated as (1,1,2) using the pcf and acf functions (see notebook) but the auto_arima function from pmdarima has them placed at (4,1,1) so this is what we went with.



As we can see, the arima model gave us a pretty bad forecast. Its just a straight line on a downward trajectory and obviously does not follow the pattern of the input data very well. Accordinmg to the summary report it had an AIC of –98.356. Although still not ideal, it seems as though the Prophet model had a better preformance. The model still struggled with the data, however the forecasts do look somewhat more reasonable. Seen below are the weekly, month, and three moponth forecasts respectively.



Visually this seems like a far more reasonable fit. Unfortunately the model still seems to really struggle with the timeseries giving a fit that does not really match the trend or seasonality of the input despite experimentating with different parameters or fit types. Note that in order to get data on the same scale as our imput min-max scaling was used as a 'hack'. We see that both of these models has the timeseries strictly decreasing in the future whicvh does not seem correct from the inpput data. Each of these model outcomes just tell me that more work is needed on this forecasting activity. Had i not used an LSTM model in the previous asignment this would have been tested.

Dashboard

The dashboard from this project can be found in a python notebook that is attached to this submission. Pannel was used due to its ease of use, nice clean interface, and its fantastic interaction with pandas dataframes. The dashbvoard is very straightforward, showing only a plot with the original timeseries. Using a slider the user may sellect any date, up to 3 months, for which a forcast will be shown on the plot. This is an interactive bokeh plot that has pan and zoom features, which m,akes it even more user friendly.

Conclusion

The objective of this project was to use sentiment analysis and time series methods to analyse the attitude towards ukranian refugees in tweets, and try to forecast sentiment over the next 3 months. Upon initial analysis of the tweet sentiment we recopognised that there may be an issue with context, meanwhile some tweets are positive and in support of the refugees, it appears as though the negative ones arent negative towardss the refugees themselves but the war that has caused the

situation. The result of this is a relativvely even distribution of positive, negative, and neutral teweets in the dataset which we realized may cause issues as we progress. There was no relationship between tweet lenfgth or sentiment either, so there was no way to predict sentiment basaed off this. Based off of this information, it appears as though more care should have been taken with tweet sellection, specifically in regards to the tweet context.

We then carried out timeseries analysis on the tweet sentiment. The timeseries is additive, and there was a poeeriod of 3399 hours (although there was no obvious seasonalitry. A seasonal decomposition revealed that the data was very noisy and asurprisingly displayed a pasrabolic type trend with no obvious seasonality. A KPSS test revealed that the data was not stationary, so based on this we chose an ARIMA model based off of its differencing term, and Prophet to carry out the forecasting. Both did not preform very well with Arima returning a straight line sloping downwards, and prophet doing the same aside from some fluctuations, possibly short period seasonality, in the data. This likel;y stems from the data effectively being noise and coulkd have been rectified as mentiohned above by taking tweet context into account.

Overall the project was a biut of a disaster. Due to a lack of context in the tweets the sentiments nearly contridicted themselves at times and resulted in a dataset that was effectively noise. As a result the distribution of positive, neutral, and negative tweets was almost identical, and attempts to run timeseries fortecasting were fruitless. Going forward more thought should be put towards tweet sellection, and different models should be used for forecasting. LSTM's may be a good choice as they can be very effective in capturing data but may be prone to overfitting.