UKRAINE WAR

SENTIMENT AND BOT ANALYSIS REPORT

MAY 2023

PREPARED BY:
JUSTYNA JAKUBOWSKA
Student

PREPARED FOR:
UNIVERSITY OF CALIFORNIA BERKELEY
AI & ML Course Final Assignment

THE PURPOSE

UNDERSTANDING THE PUBLIC SENTIMENT WHILE IGNORING BOTS

CONFLICT SUMMARY
On 24 February 2022, Russia invaded and occupied parts of Ukraine in a major escalation of the Russo-Ukrainian War, which had begun in 2014. The invasion has resulted in tens of thousands of deaths on both sides, and instigated Europe's largest refugee crisis since World War II. About 8 million Ukrainians were displaced within their country by June, and more than 8.2 million had fled the country by May 2023. (Wikipedia)

DATA
I decided to use a dataset that contains almost 48,000 tweets collected in one day in August last year to do a sentiment analysis of the Ukraine war along with the sentiment prediction algorithm. In addition I have used that dataset to design an ML model for predicting if the tweet was posted by a human or a bot.
It is worth keeping in mind that the tweets were in many languages. I chose to concentrate on English tweets and therefore the data decreased to a little bit over 23,000 tweets.

The dataset is from Kaggle and is located here: https://www.kaggle.com/datasets/bwandowando/ukraine-russian-crisis-twitter-dataset-1-2-m-rows?resource=download

PURPOSE OF SENTIMENT ANALYSIS
Sentiment analysis applied to the Ukrainian war can provide valuable insights for multiple stakeholders. For example:

- Government and Policy Makers. Sentiment analysis can reveal public opinion towards the conflict, which can guide policy decisions and strategies. It can also detect shifts in sentiment, possibly indicating changes in the political landscape.
- Humanitarian Organizations. These organizations can use sentiment analysis to understand the feelings of the people affected by the conflict, helping them better target their efforts and communication.
- News Organizations and Journalists. They can use sentiment analysis to gauge public opinion and tailor their reporting accordingly. It can also help identify misinformation or bias in reporting.
- Academics and Researchers. Sentiment analysis can provide a rich dataset for studying the societal and psychological impacts of the conflict. It can also be used to study the propagation of sentiments in social networks.
- Peacekeeping and Mediation Groups. Sentiment analysis can provide insights into the emotional state and attitudes of the parties involved, which could be useful in negotiations and peace talks.

Machine Learning algorithms I explored can be used to automate this sentiment analysis on a large scale. They can process large amounts of tweet data and classify them into sentiment categories:

THE PURPOSE

UNDERSTANDING THE PUBLIC SENTIMENT WHILE IGNORING BOTS

positive, negative, neutral.
The potential of sentiment analysis and ML is huge, but one of the key challenges is the complexity and nuance of human language, including sarcasm, idioms, and cultural references as well as native speakers vs. non native speakers.

PURPOSE OF IDENTIFYING BOTS
Creating an ML algorithm to identify bot-generated content is extremely useful in the context of the Ukrainian war. Bots can be used to manipulate public opinion, spread disinformation or propaganda, and exacerbate tensions. Identifying and mitigating these influences has several benefits:

- Disinformation Combat.
  Bots can be used to spread false information at a high speed and scale. Identifying bots can help slow the spread of such disinformation.
- Public Discourse Integrity Maintenance.
  Bots can artificially inflate certain viewpoints or narratives, distorting the real public opinion. By identifying these bots, we can ensure a more accurate representation of human sentiments.
- National Security.
  Bots can also be used as a tool of information warfare by adversarial nations. Identifying these bots can help protect national security interests.
- Academic Research.
  Understanding the extent and nature of bot activity can provide insights for academic research in fields like media studies, political science, and cybersecurity.
- Social Media Platforms. These platforms have a responsibility to ensure the integrity of their platforms. Identifying bot activity can help them uphold this responsibility.

Creating an ML algorithm to detect bots can be a complex task. It often involves analyzing various aspects of the content and behavior, such as the frequency and timing of posts, the ratio of followers to following, the use of certain keywords or phrases, and the network of interactions.

THE PROCESS

STEPS TO CREATING ML MODELS FOR DEFINING A USER AS A BOT OR A HUMAN

Since the dataset didn't define the label for each tweet as either "human" or "non-human," it is not possible to create a supervised learning model to predict whether a tweet was made by a human or not.
Therefore I decided to explore:
1. unsupervised learning techniques to identify patterns or anomalies in the data that may suggest who created a tweet. I:
* used clustering algorithms to group similar tweets together based on their features,
* used anomaly detection algorithms to identify tweets that deviated significantly from the norm,
2. a hybrid approach that combines unsupervised and supervised learning. I:
* used unsupervised learning to identify potential non-human tweets and then manually labeled a subset of these tweets to train a supervised learning model.

The first step was to understand what each column means and then I moved to cleaning up the dataset:
1. dropped some columns, because they:
- had to much missing data,
- had same variable for all rows,
- ware not relevant to the objective.

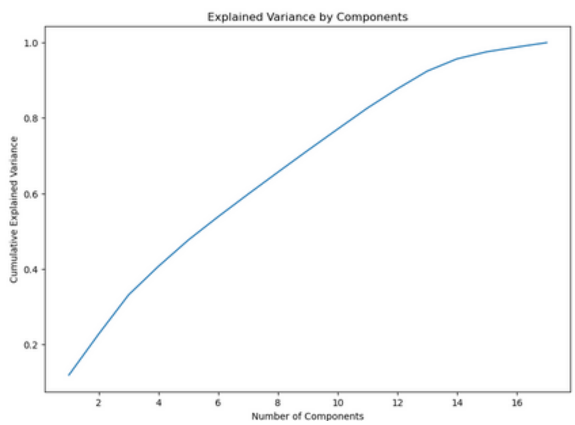
2. dropped rows for tweets in other languages since some tools I use serve only English and also I wouldn't be able to manually asses if the tweets in foreign language were made by human or bot since I speak only 2 languages.
3. created columns based on existing columns that would serve my analysis better (e.g. tweet frequency)
4. Created columns through:
A.) searching for anomalies in spelling and grammar:
- Checked for common errors made by automated systems, such as:

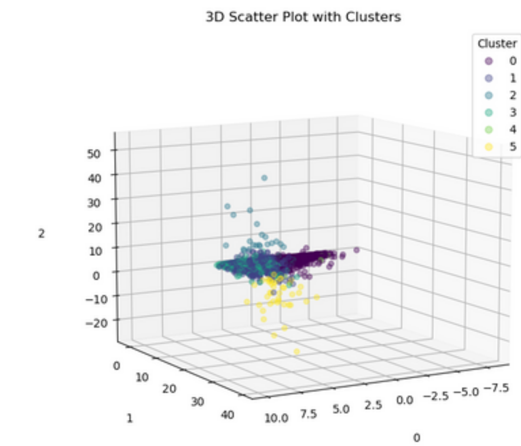
*) inconsistent capitalization,
*) unusual punctuation,
*) misspellings.
Humans are generally more accurate in their language usage.
B.) searching for language patterns anomalies:
*) Analyzed the text for repetitive structures that humans are less likely to use.
*) Analyzed the text for formulaic structures that humans are less likely to use.
C.) Finding URL shorteners, generic hashtags and automation tools, because they suggest that the tweet was created by an automated system.
D.) Checking context and relevance, because humans are more likely to engage in real-time conversations and refer to specific events, whereas automated tweets might lack timely references.
E.) Analyzing the sentiment and emotional tone of the tweet. Humans tend to express a wider range of emotions and nuances, while automated systems tend to produce generic or robotic sentiments.

Based on the cleaning of the data and creating new relevant information I got the final dataframe ready for creating the ML model that would cluster users with similar features.
I started with PCA analysis to select the best number of components (see chart below). After careful consideration I chose to go with 12 components, because adding more components wouldn't provide much more useful information.

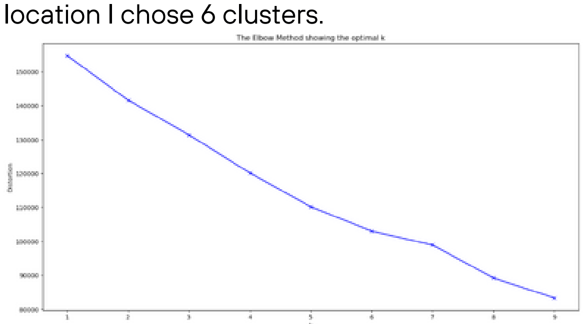
THE PROCESS

STEPS TO CREATING ML MODELS FOR DEFINING A USER AS A BOT OR A HUMAN





Then I applied clustering algorithms to the selected features in hopes to isolate a group of users that are bots. I used DBSCAN and K-MEANS, but DBSCAN results were poor so I didn't even include them here. On the other hand I feel K-Means performed fantastic and here is how I went about it:
I researched how many clusters are optimal and based on the chart below and elbow location I chose 6 clusters.



Then I fit the model on 6 clusters.
To present clusters properly I chose a 3 D chart (see next column).
Now that I had actual clusters, all I had to do is to reengineer the points to get the user IDs along with their combined tweets. Since the clusters were different sizes:

cluster 1: 5700 points
cluster 0: 2439 points
cluster 3: 2155 points
cluster 5: 42 points
cluster 2: 21 points
cluster 4: 16 points

and my goal was to study manually 1000 users, I chose the whole cluster 5, 2, 4 and then randomly selected 307 users from the remaining 3 clusters. As a result I was able to download 6 .csv files to study manually to determine if the user is a bot or a human.
It is important to note that bots these days can write in a human way and on another hand some human tweets can sound like bots since some of them are not written by native speakers or were written under the influence...
I tried my best to decide on users' category. Unfortunately the ratio of bots to humans was 1:4 which wouldn't create a good ML algorithm. At that point I decided to settle on 2:3 ratio. To achieve it I had to randomly select from all files 500 human records and discard them. I still tried to do it proportionately according to the cluster size.

THE PROCESS

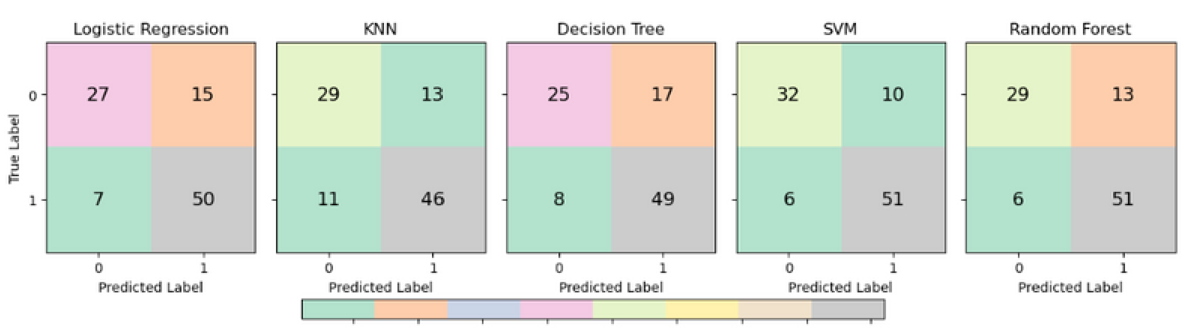
STEPS TO CREATING ML MODELS FOR DEFINING A USER AS A BOT OR A HUMAN

Now I merged the 6 csv files into 1 file in Jupyter Notebook. The result was a file with 282 human users and 210 bot users that was ready for creating an ML model for predicting user being a human or not. I split the data to testing and training, double checked if the data was balanced, double checked the distribution (Shapiro-Wilk test). Since majority of features ware not normally distributed I used MinMaxScaler.
Since this is a binary classification problem, that can be solved by several machine learning algorithms, I explored:
* Logistic Regression,
* Random Forest,
* Support Vector Machines
* KNN
* Decision Tree
Results:

The best algorithm turned out to be SVM with:
*) cost function (C) set to 10,
*) the gamma value set to 'scale', and
*) the kernel function set to'rbf' (Radial basis function).

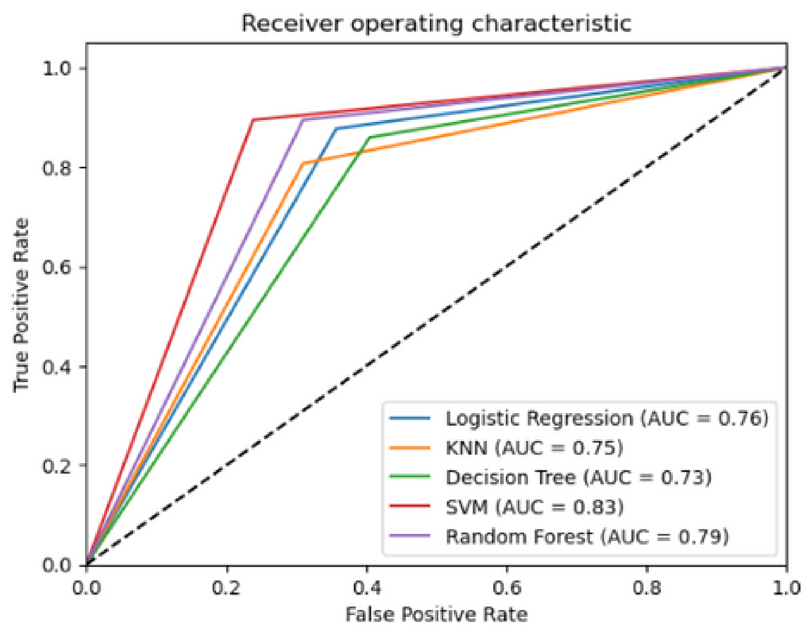
- The model correctly classified approximately 86.77% of the training data.
- The model correctly classified approximately 83.84% of the testing data, which is slightly lower than the training accuracy, and may suggest a slight overfitting to the training data.
- The average of the cross-validation scores is approximately 78.89%, which is lower than both the training and testing accuracy. This suggests that the model's performance might not be as strong when introduced to completely new, unseen data.

	Training time	Training accuracy	Testing accuracy
Logistic Regression	0.09676	0.814249	0.777778
KNN	0.09676	0.865140	0.757576
Decision Tree	0.09676	0.890585	0.747475
SVM	0.09676	0.867684	0.838384
Random Forest	0.09676	1.000000	0.808081



THE PROCESS

STEPS TO CREATING ML MODELS FOR DEFINING A USER AS A BOT OR A HUMAN



- The standard deviation of the cross-validation scores is 0.0298, indicating a small spread around the mean cross-validation accuracy, which suggests a fairly consistent model performance across different data subsets.
- The precision of the model is 0.8361. This means that when the model predicts a Twitter account to be human (1), it is correct about 83.61% of the time.
- The recall score of the model is 0.8947. This implies that the model is able to correctly identify approximately 89.47% of all the human-operated Twitter accounts in the dataset.
- The AUC (Area Under the Curve) score is 0.8283. This is a measure of the trade-off between the true positive rate and the false positive rate. An AUC score closer to 1 means the model has better classification performance. Here, it suggests a
- reasonably good balance between sensitivity (true positive rate) and specificity (true negative rate).
- The training time was the same for all algorithms and took less than 0.1 second. But we should keep in mind that the dataset was very small and training time might be showing major differences for big datasets.

Overall, these metrics suggest that the SVM classifier performs reasonably well at choosing between bot and human-operated Twitter accounts.

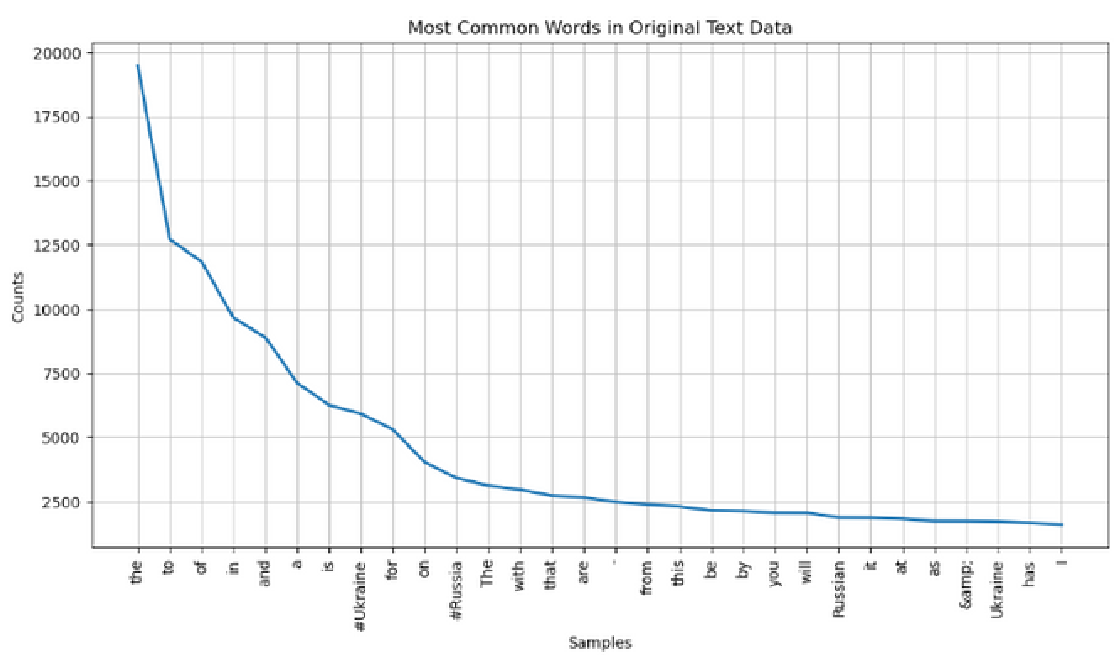
THE PROCESS

STEPS TO CREATING ML MODEL FOR TWEET SENTIMENT CLASSIFICATION FOR ALL USERS

The first step was to understand what each column means and then I moved to cleaning up the dataset:
1. dropped rows for tweets in other languages since some tools I use serve only English and also I wouldn't be able to manually asses if the tweets in foreign language was made by human or bot since I speak only 2 languages.
2. Text cleaning:

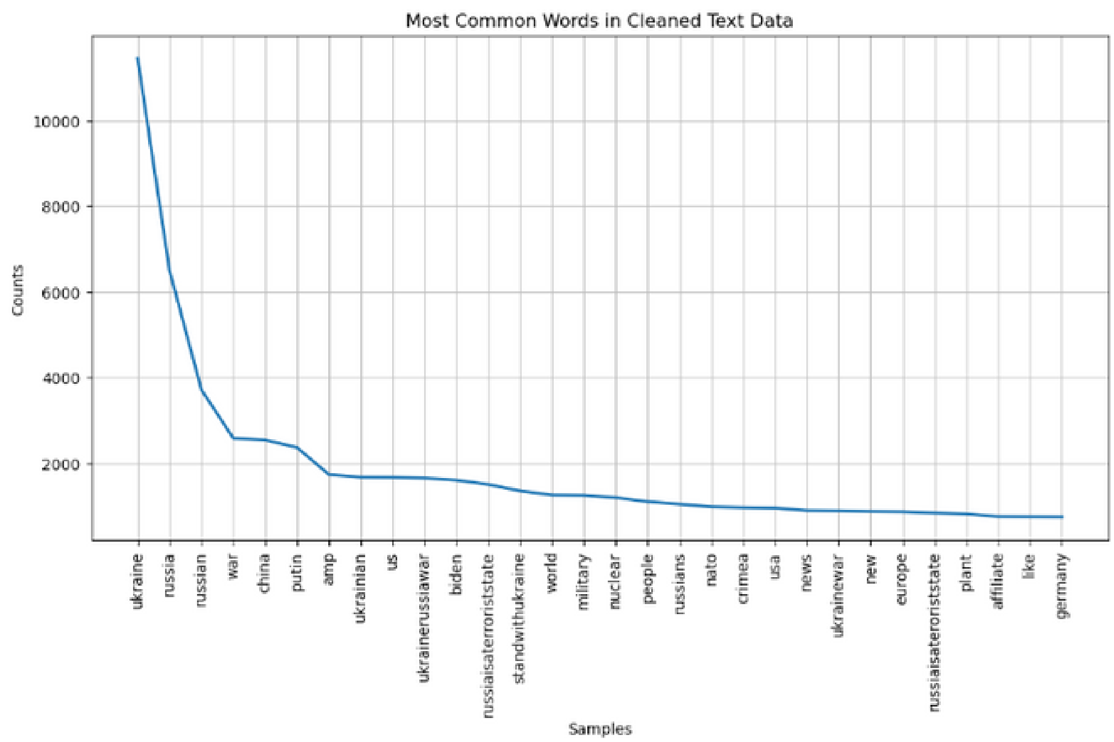
- Removed unwanted characters such as special characters, punctuation, and digits from the text data using regular expressions.
- Converted all the text data to lowercase using the lower() function.
- Removed stop words from the text data using NLTK.
- Removed extra whitespaces from the text data using the strip() function.
- Tokenized each word in the text data into individual tokens.
- Performed lemmatization first, because it produced a more accurate base form of the word compared to stemming, as it considers the part of speech and the context of the word in the sentence.
- Performed stemming.

Then I compared the frequency and type of most used words of the original text data and the cleaned text data (view charts below).



THE PROCESS

STEPS TO CREATING ML MODEL FOR TWEET SENTIMENT CLASSIFICATION FOR ALL USERS



At this point the data was ready for using a pre-trained sentiment analysis model TextBlob to analyze the sentiment of each tweet and assigning a positive or negative label.
Now the dataframe is ready for creating the ML model for tweet sentiment prediction. I ran 3 types of algorithms:

- Logistic Regression
- Decision Tree
- MultinomialNB

All algorithms were run 2 times. This was done for 2 different methods of converting text data into numerical vectors which can be understood by machine learning models:

- Count Vectorization and
- TF-IDF (Term Frequency-Inverse Document Frequency) .

Here are the results for Count Vectorization:

Model	Best Parameters	Best Score	Time to Fit Data
Logistic Regression	{'cvect__max_features': 2000, 'cvect__stop_words': None, 'lgr__C': 1, 'lgr__penalty': 'l2'}	0.852105	0.488190
Decision Tree	{'cvect__max_features': 2000, 'cvect__stop_words': None, 'dtc__criterion': 'gini', 'dtc__max_depth': None}	0.828405	1.379136
MultinomialNB	{'cvect__max_features': 2000, 'cvect__stop_words': None, 'mnb__alpha': 10}	0.792395	0.237737

THE PROCESS

STEPS TO CREATING ML MODEL FOR TWEET SENTIMENT CLASSIFICATION FOR ALL USERS

Here are the results for TF-IDF:

Model	Best Parameters	Best Score	Time to Fit Data
Logistic Regression	{'lgr__C': 10, 'lgr__penalty': 'l2', 'tfidf__max_features': 2000, 'tfidf__stop_words': None}	0.854464	0.510296
Decision Tree	{'dtc__criterion': 'gini', 'dtc__max_depth': None, 'tfidf__max_features': 2000, 'tfidf__stop_words': None}	0.819777	3.763280
MultinomialNB	{'mnb__alpha': 0.1, 'tfidf__max_features': 2000, 'tfidf__stop_words': None}	0.801311	0.245283

RESULTS
The best model turned out to be a Logistic Regression model run through TF-IDF with a score of 0.854464, which indicates that with the best parameters, the model correctly predicted the sentiment of about 85.4% of the tweets in the validation set. The best parameters are:

- 'gr__C (the inverse of regularization strength): 10. It suggests a relatively weaker regularization, allowing the model to fit the training data more closely.
- 'lgr__penalty': 'l2'. L2 regularization discourages large coefficients by adding a penalty equal to the square of the magnitude of coefficients to the loss function.
- 'tfidf__max_features': 2000: This parameter is used in the TF-IDF Vectorizer, it sets a limit to the top 2000 features ordered by term frequency across the corpus.
- 'tfidf__stop_words': None: This means that the TF-IDF Vectorizer is not removing any stop words from the text.

Logistic regression with TF-IDF had a second time when it comes to training.

THE RESULTS

UNDERSTANDING THE SENTIMENT FOR ALL USERS

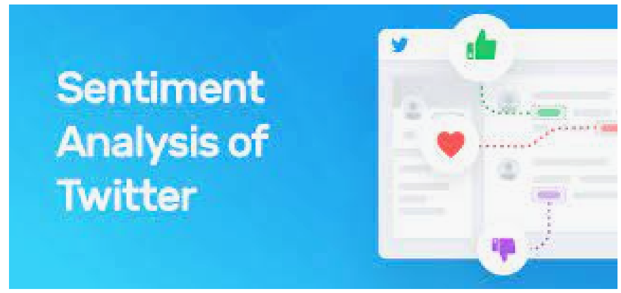
Here are the sentiment results for the original data:

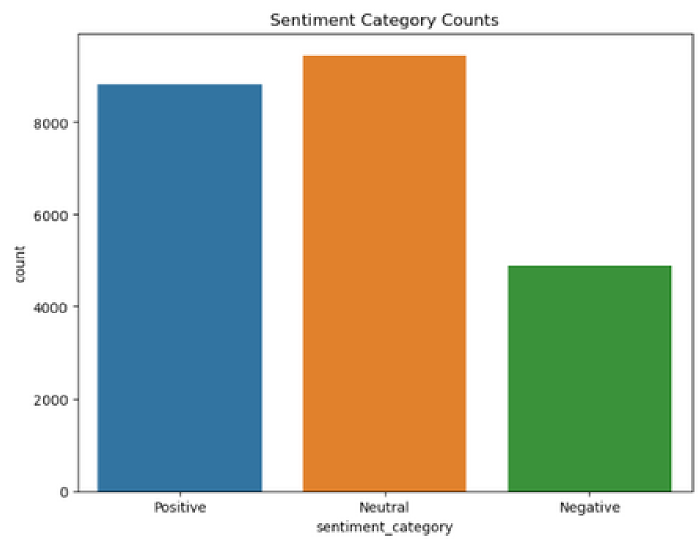
- Neutral sentiment: 9458 tweets (40.804176%)
- Positive sentiment: 8832 tweets (38.103456%)
- Negative sentiment: 4889 tweets (21.092368%)

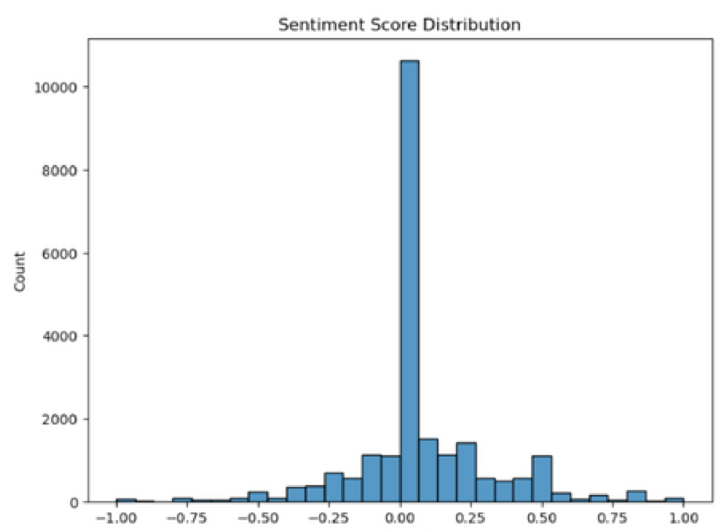
It is best to present the sentiment analysis in a chart form.

In the bar plot "Sentiment Category Counts" on the right, the x-axis represents the sentiment category: Positive, Negative, Neutral and the y-axis represents the count of tweets in each category. The most tweets belong to neutral sentiment category (orange color bar) followed by positive sentiment category (blue color bar) and negative sentiment (green color bar).

In the histogram "Sentiment Score Distribution" on the right, the x-axis represents the sentiment score and the y-axis represents the count of tweets for each sentiment score. The sentiment score ranges from -1 (most negative) to +1 (most positive), with 0 being neutral. The histogram shows how these scores are distributed across the tweets in my dataset.
It looks like overwhelming majority of tweets is somewhere between -0.5 and 0.5 with neutral tweets at 0 leading. That means that there are to many extremely positive tweets (1) or negative tweets (-1).





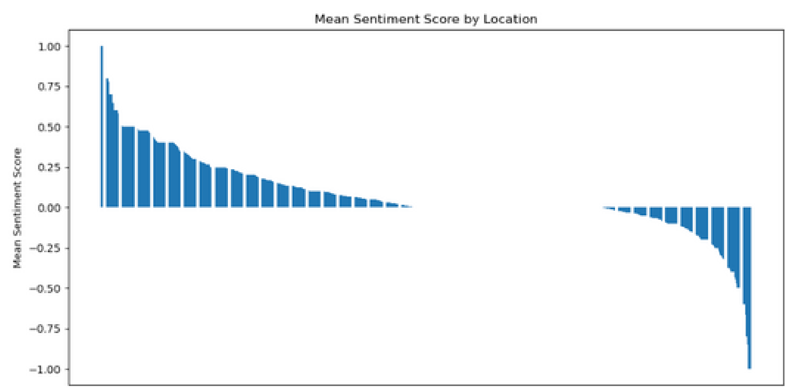


THE RESULTS

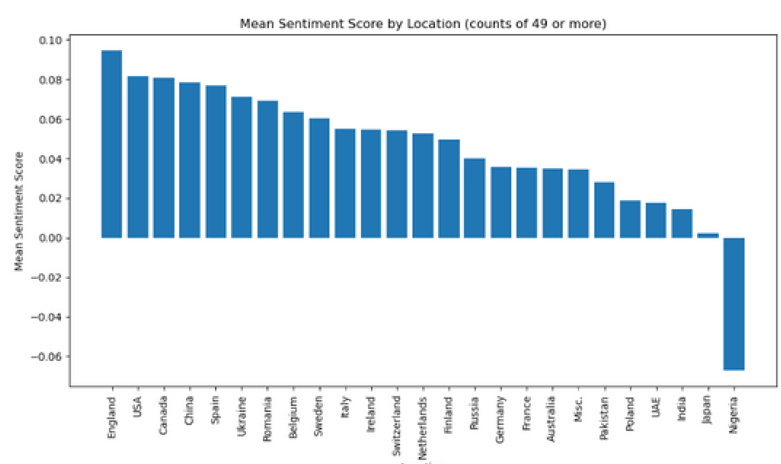
UNDERSTANDING THE SENTIMENT BY LOCATION FOR ALL USERS

SENTIMENT ANALYSIS BY LOCATION
We need to keep in mind that this analysis is more accurate for the parts of the world where English is the native language like USA, England, Canada...
I decided to present the data on 2 charts.

Since there were over 3000 values on x axis I decided not to list them on the chart "Mean Sentiment Score by Location" that is on the left. This chart's goal is to give us a general idea of the sentiment in the world from English speaking population. We clearly can see that majority (about half) of the world has positive sentiment (values above 0). A little bit more than 25% of the world is neutral (value is 0) and slightly less than 25% has a negative sentiment (values below 0). It is also obvious that majority of sentiment is between values of -0.5 and 0.5 and very small population has extreme sentiment about the subject of Ukraine war (values of 1 or -1).



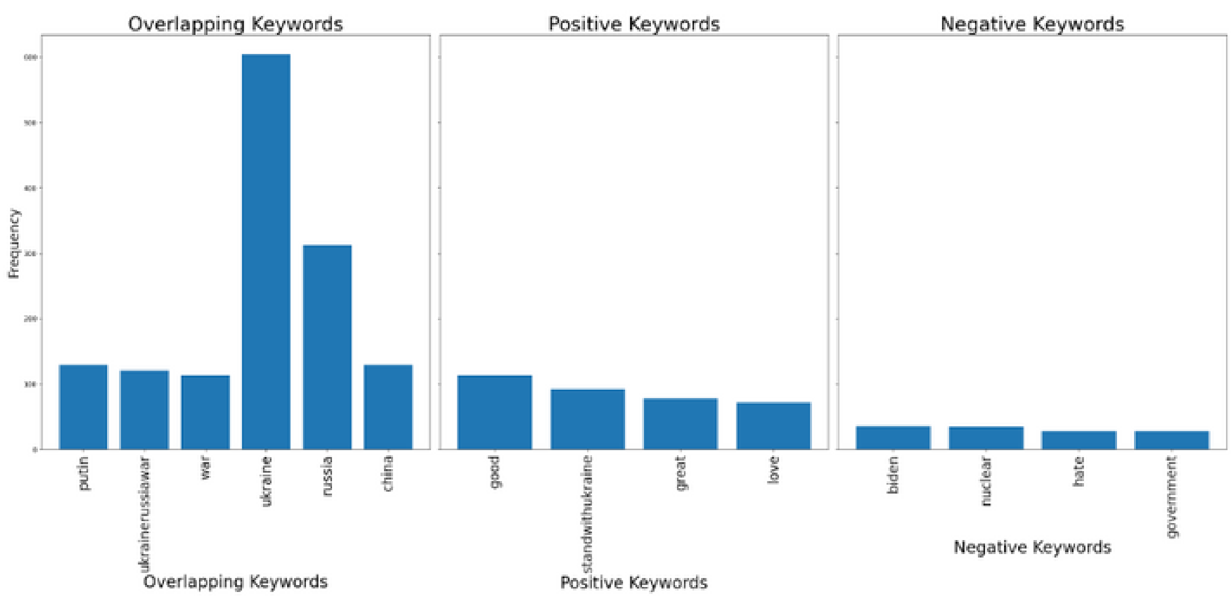
The plot "Mean Sentiment Score by Location (counts if 49 or more)" on the right shows countries' sentiment as well but only for the countries that had 49+ combined frequency. The most positive sentiment shows England that is followed by USA, Canada and China. It is very interseting that the positive sentiment is not very strong and is captured below 0.09 where the maximum scale is 1.0
The most neutral appears Japan that is almost 0.
The most negative is Nigeria, but even then their sentiment is around -0.06 where the maximum scale is -1.



THE RESULTS

UNDERSTANDING THE SENTIMENT BY KEYWORDS FOR ALL USERS

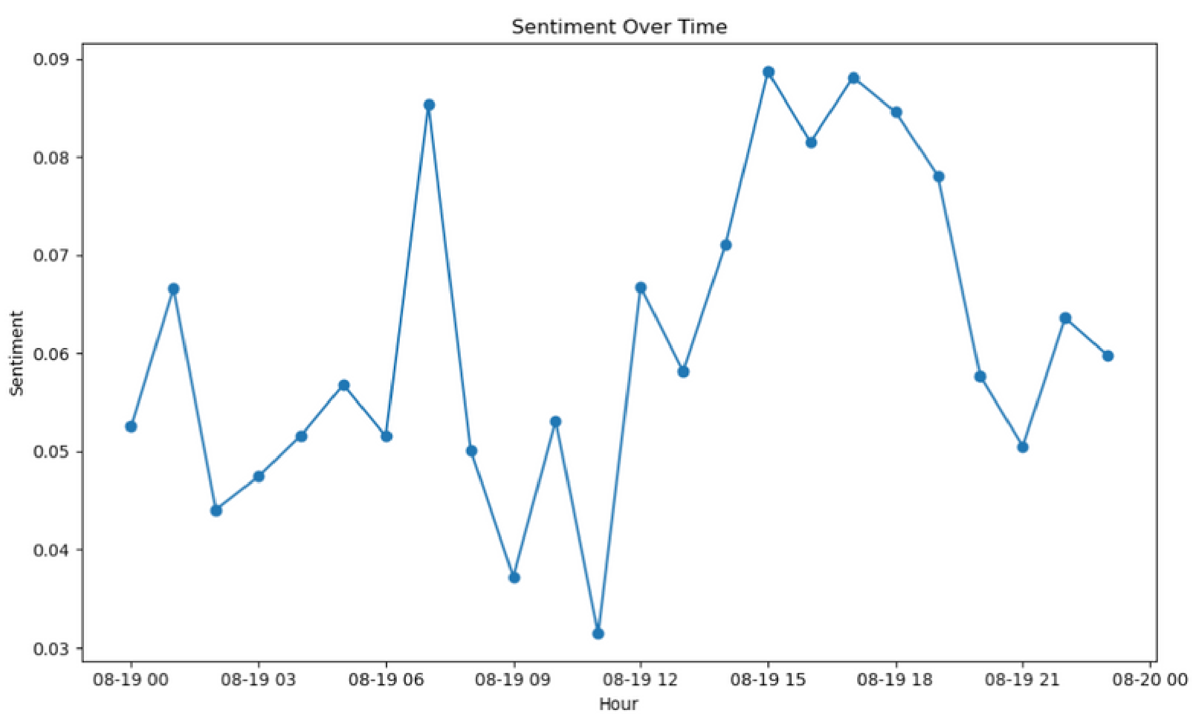
SENTIMENT ANALYSIS BY KEYWORDS RESULTS
The overlapping keywords that repeat in the tweets regardless of their sentiment score are: Ukraine, war, Putin, Russia, ukrainerussianwar and china.
The tweets with positive sentiment have following keywords showing up the most: good, standwithukraine, great, love.
The tweets with negative sentiment have following keywords showing up the most:: Biden, nuclear, hate and government.



THE RESULTS

UNDERSTANDING THE SENTIMENT OVER TIME FOR ALL USERS

SENTIMENT OVERTIME ANALYSIS
I grouped the data by hour. The plot shows how the mean sentiment changes over time in the dataset. It looks like the sentiment mean for the day was positive and fluctuating between 0.03 and 0.09. The sentiment mean was the closest to the neutral state around 11AM and most positive around 3PM.
If we divided data by the time zones we might be able to observe patterns around people daily activities, for example the sentiment can be its highest when people are about to live work since they might be excited about their afternoon plans...



COMPARISON

SENTIMENT ANALYSIS FOR ALL USERS VS. HUMAN USERS

Once the ML model was ready to define if the user is human or bot the users were assigned to one of these groups. After completing it, a dataset with humans only was separated and I was able to run a sentiment analysis on it as well as create an ML model for sentiment prediction.

We still need to keep in mind that even-though I say "dataset for human", the prediction model was right only 83.61% and that assuming that I classified the tweets correctly.

Now I decided that it would be interesting to compare the sentiments and ML results for both "all users" dataset and "human" dataset. Below we explore it.

First, the data was cleaned in a similar manner as described in previous chapter. Then the data was ready for using a pre-trained sentiment analysis model TextBlob to analyze the sentiment of each tweet and assigning a positive or negative label.
Now the dataframe was ready for creating the ML model for tweet sentiment prediction. I ran 3 types of algorithms:

- Logistic Regression
- Decision Tree
- MultinomialNB
- All algorithms were run 2 times. This was done for 2 different methods of converting text data into numerical vectors which can be understood by machine learning models:
- Count Vectorization and
- TF-IDF (Term Frequency-Inverse Document Frequency) .

Here are the results for Count Vectorization for all users:

Model	Best Parameters	Best Score	Time to Fit Data
Logistic Regression	{'cvect__max_features': 2000, 'cvect__stop_words': None, 'lgr__C': 1, 'lgr__penalty': 'l2'}	0.852105	0.488190
Decision Tree	{'cvect__max_features': 2000, 'cvect__stop_words': None, 'dtc__criterion': 'gini', 'dtc__max_depth': None}	0.828405	1.379136
MultinomialNB	{'cvect__max_features': 2000, 'cvect__stop_words': None, 'mnb__alpha': 10}	0.792395	0.237737

Here are the results for Count Vectorization for humans:

Model	Best Parameters	Best Score	Time to Fit Data
Logistic Regression	{'cvect__max_features': 2000, 'cvect__stop_words': None, 'lgr__C': 1, 'lgr__penalty': 'l2'}	0.844764	0.387347
Decision Tree	{'cvect__max_features': 2000, 'cvect__stop_words': None, 'dtc__criterion': 'gini', 'dtc__max_depth': None}	0.821165	1.020091
MultinomialNB	{'cvect__max_features': 2000, 'cvect__stop_words': None, 'mnb__alpha': 0.1}	0.790044	0.181062

The results for Count Vectorization above show that the models for humans were not performing as good as for all users.

COMPARISON

SENTIMENT ANALYSIS FOR ALL USERS VS. HUMAN USERS

Here are the results for TF-IDF for all users:

Model	Best Parameters	Best Score	Time to Fit Data
Logistic Regression	{'lgr__C': 10, 'lgr__penalty': 'l2', 'tfidf__max_features': 2000, 'tfidf__stop_words': None}	0.854464	0.510296
Decision Tree	{'dtc__criterion': 'gini', 'dtc__max_depth': None, 'tfidf__max_features': 2000, 'tfidf__stop_words': None}	0.819777	3.763280
MultinomialNB	{'mnb__alpha': 0.1, 'tfidf__max_features': 2000, 'tfidf__stop_words': None}	0.801311	0.245283

Here are the results for TF-IDF for humans:

Model	Best Parameters	Best Score	Time to Fit Data
Logistic Regression	{'lgr__C': 10, 'lgr__penalty': 'l2', 'tfidf__max_features': 2000, 'tfidf__stop_words': None}	0.848304	0.395577
Decision Tree	{'dtc__criterion': 'entropy', 'dtc__max_depth': None, 'tfidf__max_features': 2000, 'tfidf__stop_words': None}	0.816150	2.028070
MultinomialNB	{'mnb__alpha': 0.1, 'tfidf__max_features': 2000, 'tfidf__stop_words': 'english'}	0.796534	0.199477

The results for TF-IDF above show that the models for humans were again not performing as good as for all users.

The best model for all users turned out to be a Logistic Regression model run through TF-IDF with a score of 0.854464, which indicates that with the best parameters, the model correctly predicted the sentiment of about 85.4% of the tweets in the validation set.

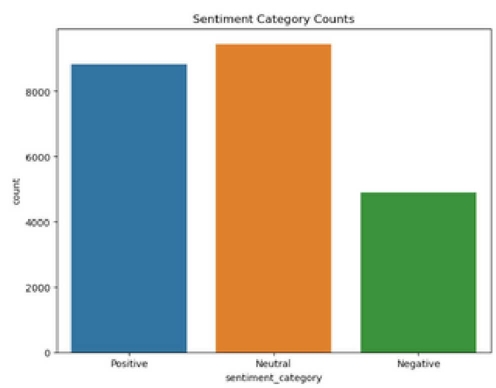
The best model for humans turned out to be also Logistic Regression model run also through TF-IDF with a score of 0.848304, which indicates that with the best parameters, the model correctly predicted the sentiment of about 84.83% of the tweets in the validation set.

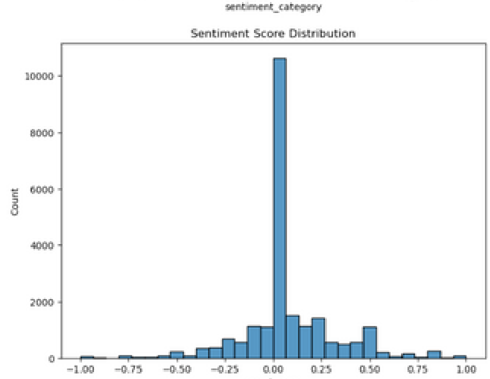
COMPARISON

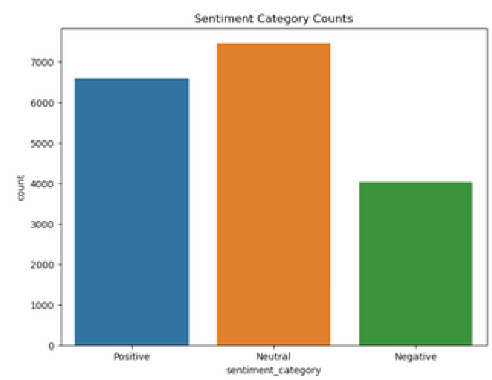
UNDERSTANDING THE SENTIMENT FOR ALL USERS VS. HUMANS

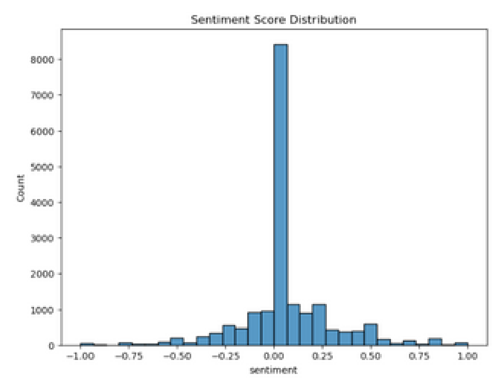
Here are the sentiment results for all users:
- Neutral sentiment: 9458 tweets (40.804176%)
- Positive sentiment: 8832 tweets (38.103456%)
- Negative sentiment: 4889 tweets (21.092368%)

Here are the sentiment results for humans:
- Neutral sentiment: 7459 tweets (41.255531%)
- Positive sentiment: 6592 tweets (36.460177)
- Negative sentiment: 4029 tweets (22.284292)









Above we can see that the charts for all users and humans look pretty much identical except for the scale. So the trends were actually very similar but we should keep in mind that the discrepancy might be way more visible on larger datasets.

COMPARISON

UNDERSTANDING THE SENTIMENT BY LOCATION FOR ALL USERS VS HUMANS

SENTIMENT ANALYSIS BY LOCATION

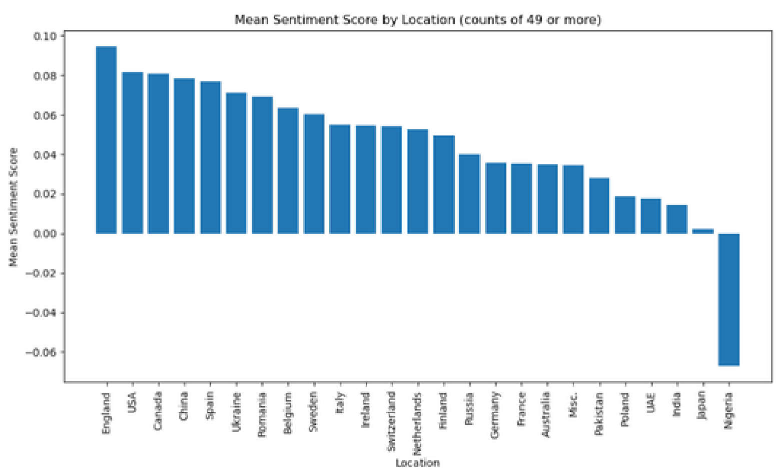
We need to keep in mind that this analysis is more accurate for the parts of the world where English is the native language like USA, England, Canada...

The plots "Mean Sentiment Score by Location (counts if 49 or more)" below show countries' sentiment but only for the countries that had 49+ combined frequency.

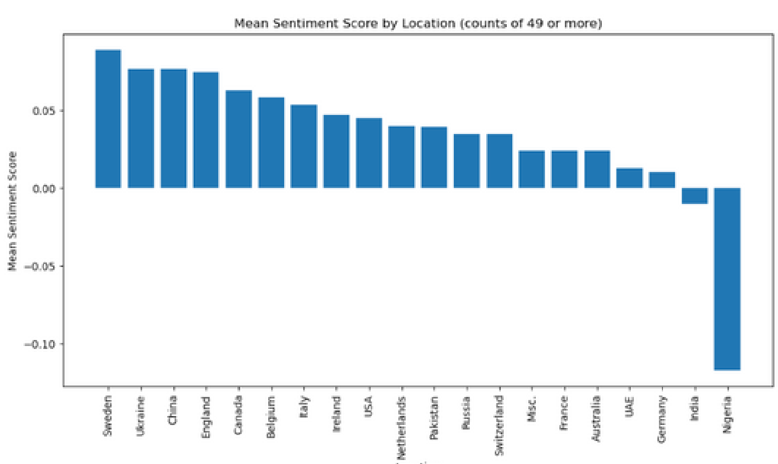
For all users chart shows that the most positive sentiment is in England and is followed by USA, Canada and China. While for just humans it shows Sweden followed by Ukraine, China and Canada. It is very interesting that the positive sentiment is not very strong and on both charts it is captured below 0.09 where the maximum scale is 1.0

The most negative is Nigeria on both charts, but even then their sentiment is not very negative since the maximum scale is -1. The Nigeria negative sentiment is stronger for human data vs. all users...

All users:



Humans:

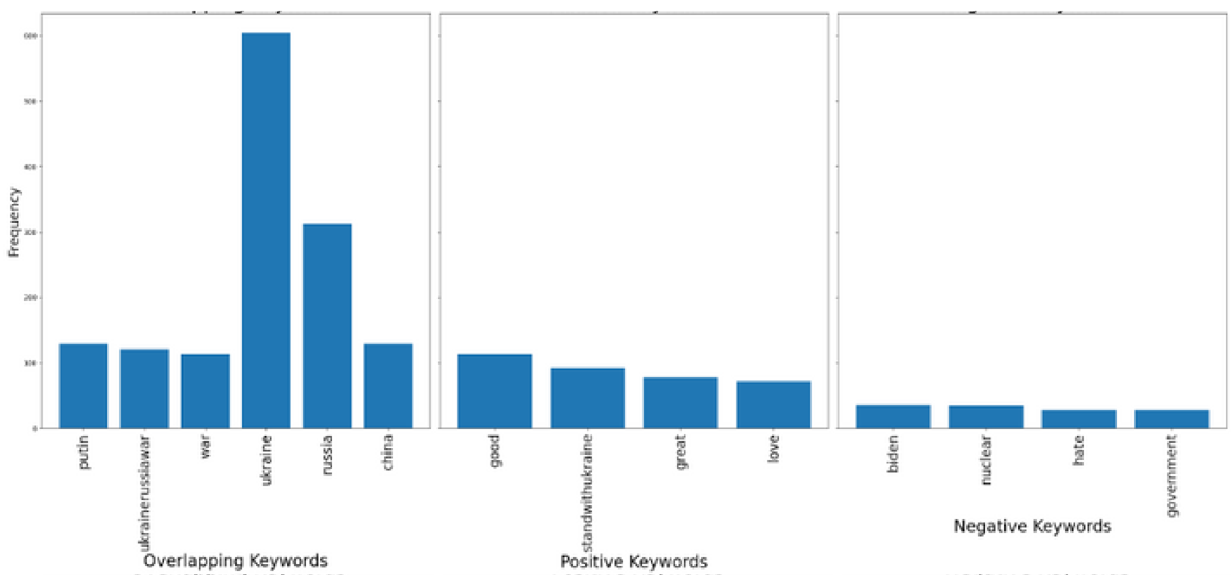


COMPARISON

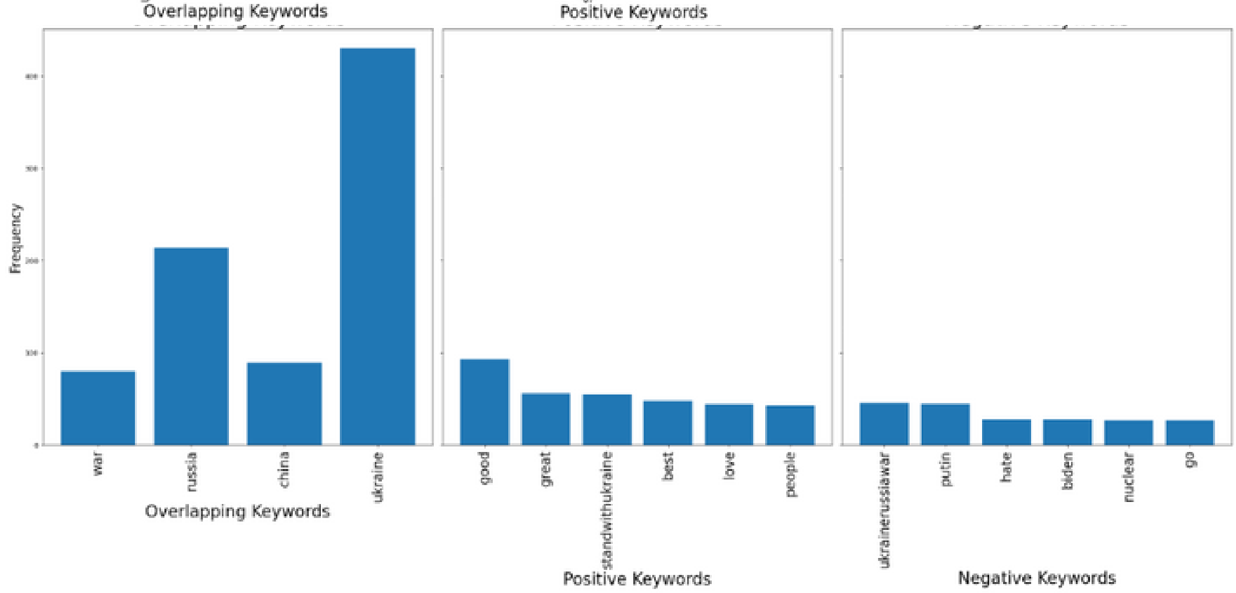
UNDERSTANDING THE SENTIMENT BY KEYWORDS FOR ALL USERS VS. HUMANS

Only human data has only 4 words overlapping vs. 6 (additional putin and ukrainerussiawar) for all users and the situation is reversed for positive and negative keywords. The additional positive words for human data are best and people and for negative words they are putin and ukrainerussiawar and go.

All users:



Humans:



COMPARISON

UNDERSTANDING THE SENTIMENT OVER TIME FOR ALL USERS

SENTIMENT OVERTIME ANALYSIS
I grouped the data by hour. The plots below show how the mean sentiment changes over time in the dataset. It looks like the sentiment mean for all users for the day was positive and fluctuating between 0.03 and 0.09. The sentiment mean was the closest to the neutral state around 11AM and most positive around 3PM.
The sentiment mean for humans for the day was positive and fluctuating between 0.01 and 0.09. The sentiment mean was the closest to the neutral state around 9AM and most positive around 7AM.

All users:

Humans: