10 Academy

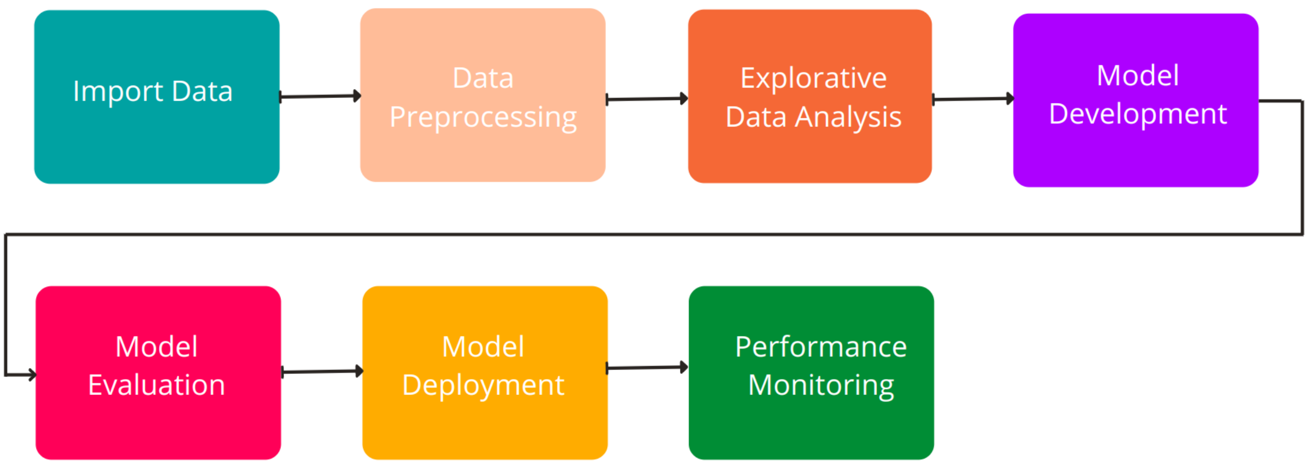
Week 0

Report 3 - Final Report

Name: Yishak Tadele

ML Workflow Chart and Description

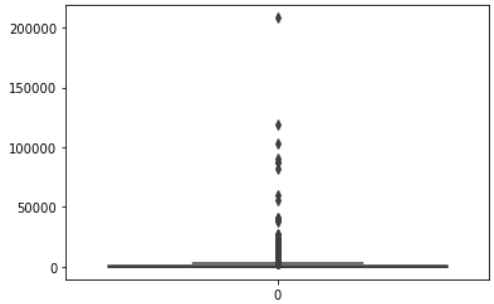
Diagram

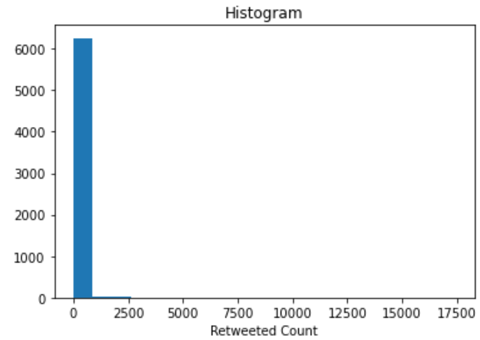


In the following section, I will explain the work I have done and the new insights I obtained.

1. Data Importing
    - After loading the JSON dump into the pandas' data frame I have saved it to the pickle file to make it available for later access.
    - The pickle file is directly loaded and further preprocessed in the next stages.
2. Data Preprocessing
    - In this stage, I have cleaned the dataset further, especially the Twitter content extracted further. For example
        - Stop words, usernames and links and non-English characters are removed.
        - Lemmatization and vectorized representation is applied to the text feature.
3. Data Exploration /EDA

   Plots: most of the features with numerical values have the following distribution in a boxplot. These features are followers_count, friends_count, retweet_count and favorite_count. The distribution reflects that a small number of users get a lot of likes and retweets and most of the users account likability is below average. The same argument can be built for the histogram plots which can be shown.





-

Analytical Results:

```
1 df.describe()
```

	polarity	subjectivity	favorite_count	retweet_count	followers_count	friends_count
count	6326.000000	6326.000000	6326.000000	6326.000000	6.326000e+03	6326.000000
mean	0.062901	0.313011	0.399621	45.417325	1.016427e+04	1543.008536
std	0.229075	0.279022	1.534209	353.598765	2.162880e+05	4968.226632
min	-1.000000	0.000000	0.000000	0.000000	0.000000e+00	0.000000
25%	0.000000	0.000000	0.000000	0.000000	5.500000e+01	95.250000
50%	0.000000	0.300000	0.000000	0.000000	2.960000e+02	390.500000
75%	0.150000	0.500000	0.000000	4.000000	1.273000e+03	1427.500000
max	1.000000	1.000000	59.000000	17409.000000	1.024910e+07	208360.000000

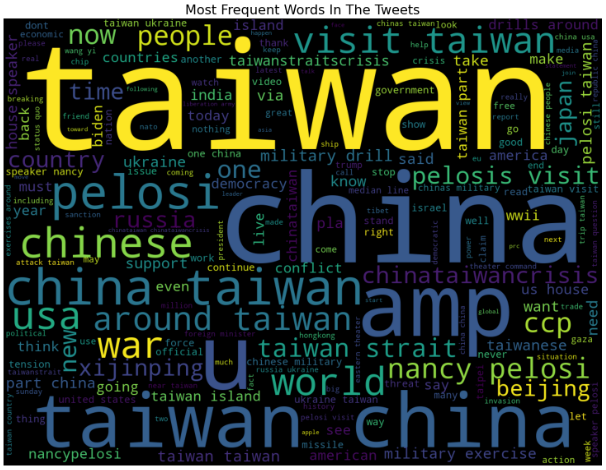
- The following picture shows the range, average and different quartiles for the numerical features.
- The possibily_sensitive feature shows missing values for 3286 entries

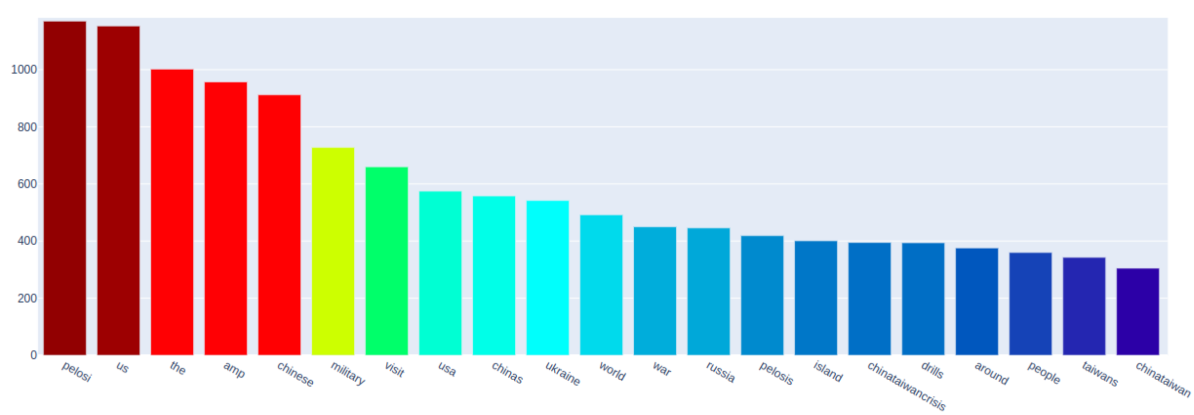
```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 6326 entries, 0 to 21997
Data columns (total 15 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   created_at          6326 non-null   datetime64[ns, UTC]
 1   source              6326 non-null   object
 2   original_text       6326 non-null   object
 3   polarity            6326 non-null   float64
 4   subjectivity        6326 non-null   float64
 5   lang                6326 non-null   object
 6   favorite_count      6326 non-null   int64
 7   retweet_count       6326 non-null   int64
 8   original_author     6326 non-null   object
 9   followers_count     6326 non-null   int64
 10  friends_count       6326 non-null   int64
 11  possibly_sensitive  3040 non-null   object
 12  hashtags            6326 non-null   object
 13  user_mentions       6326 non-null   object
 14  place               6326 non-null   object
```

- The created_at feature consists of only two days, which means that the tweets in the dataset are two-day tweets.
- After cleaning the text feature (tweet content) the following picture is generated using word cloud and the next picture shows the top 20 words found on the cleaned dataset. Also, the top 50 words are shown as follows

Most Frequent Words In The Tweets





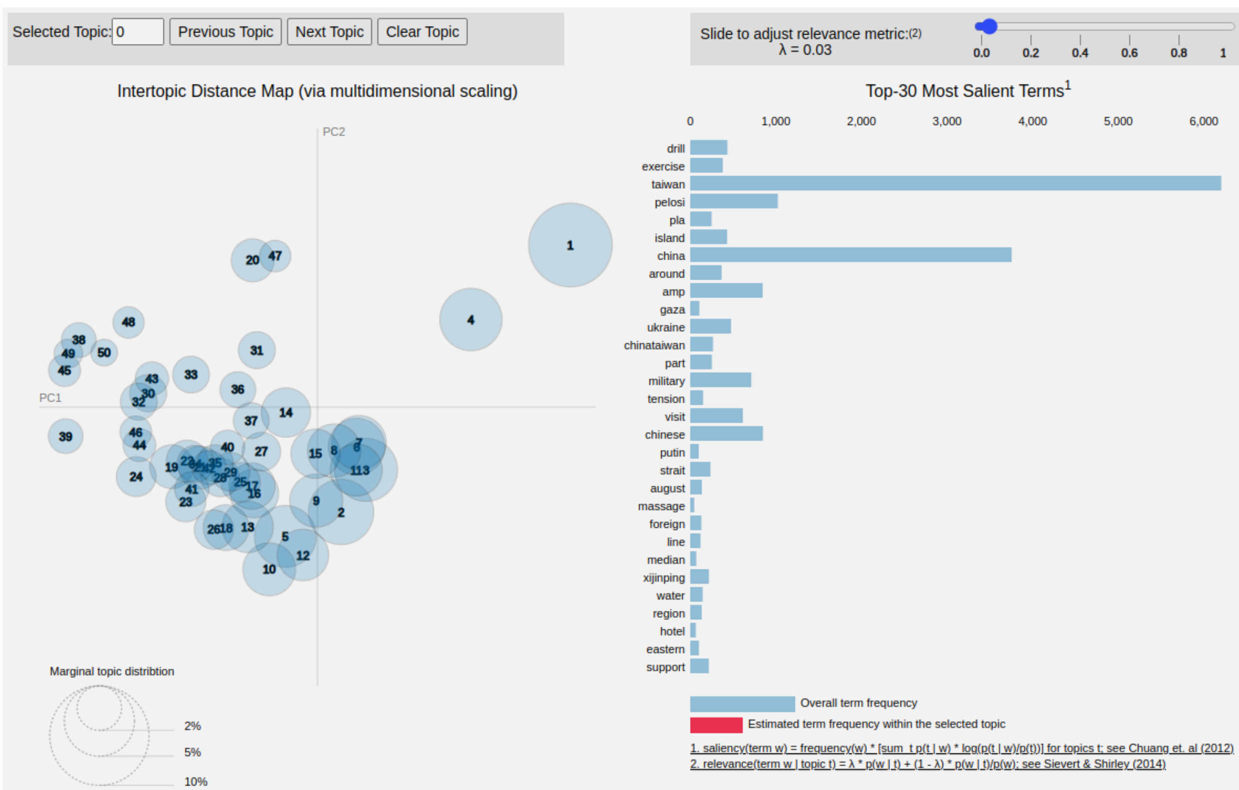
4. Model Development

a. Topic Modelling: here I have used Latent Dirichlet Allocation/LDA to model the particular topics. It is popularly used to extract topics from a given corpus. LDA categorizes the text into a document and the words per topic, these are modelled based on [Dirchilet distributions and processes](#).

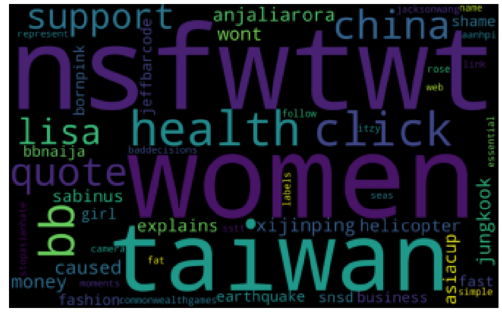
LDA makes two key assumptions the first is the documents are a mixture of topics and the second is topics are a mixture of words.

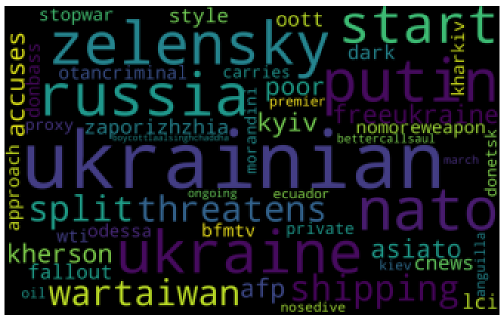
After developing the model with LDA I have shown the results using pyLDAvis package with an interactive display.

- Each circle is a topic and the size represents the abundance of that topic in the corpus.



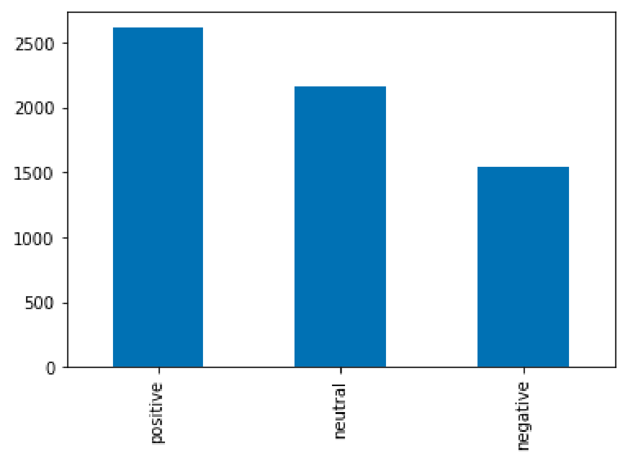
- After the LDA model training, the generated topics are shown in the notebook I have been working on. But to grasp the content I have plotted word clouds using the four topics as follows.





b. Sentiment Analysis: using the feature polarity I was able to model the sentiment analysis as a classification problem. Since the polarity value range is [-1,1] negative values are labelled as negative sentiments and vice versa.

- So the data feature will be the cleaned text and the label will be the polarity feature converted to binary values. The following picture shows the label distribution.



- The dataset is split into train-test with a 0.6/0.4 ratio and the text is vectorized with a unigram, bigram and trigram vectorizer to evaluate which choice could perform better. These represent the stack of words that can be found in a single vectorized set.

- I have used several classification algorithms to experiment with the results. These are SVM, Naive Bayes, Decision Tree, Random Forest, KNN and Logistic regression classification algorithms.

Sentiment Analysis Additional work

- I have tried the bigram and trigram vectorization methods along with TF-IDF. TF-IDF stands for Term Frequency – Inverse document frequency.
    - TF = frequency of the term in a document/ total number of terms in documents
    - IDF = log(Total number of documents/Number of documents with a term

      So TF-IDF = TF.IDF
- In the experiment, I tried 6 ML algorithms by splitting the dataset to train-test with a 0.6/0.4 ratio. The following pictures show the results I obtained.

a. Bigram Vectorization: as can be seen in the picture Logistic regression Classifier shows a higher performance than the rest of the models.

```
# Bigram vectorization
bigram_models = evaluation(X_data = X_train_bigram,y_data = y)
```

Model	Cross Validation	Train Accuracy	Test Accuracy
SVM	0.688	0.977	0.739
NBC	0.636	0.876	0.692
DTC	0.695	1.0	0.74
RFC	0.699	1.0	0.747
KNN	0.645	0.726	0.685
LRC	0.735	1.0	0.779

b. Bigram Vectorization with TF-IDF: here Scalable Vector Machine outperforms the other algorithms.

```
# Bigram Tf-IDF vectorization
bigram_tfidf_models =  evaluation(X_data = X_train_bigram_tf_idf,y_data = y)
```

Model	Cross Validation	Train Accuracy	Test Accuracy
SVM	0.657	0.999	0.701
NBC	0.615	0.865	0.661
DTC	0.651	1.0	0.726
RFC	0.67	1.0	0.732
KNN	0.66	0.802	0.674
LRC	0.657	0.853	0.693

c. Trigram vectorization: here Logistic regression outperforms the other models.

```
1 # Trigram vectorization
2 trigram_models = evaluation(X_data = X_train_trigram,y_data = y)
```

Model	Cross Validation	Train Accuracy	Test Accuracy
SVM	0.68	0.978	0.747
NBC	0.632	0.856	0.692
DTC	0.691	1.0	0.743
RFC	0.68	1.0	0.768
KNN	0.645	0.717	0.645
LRC	0.719	1.0	0.794

d. Trigram vectorization with TF-IDF: here random forest outperforms the other.

```
1 # Trigram Tf-IDF vectorization
2 trigram_tfidf_models =  evaluation(X_data = X_train_trigram_tf_idf,y_data = y)
```

Model	Cross Validation	Train Accuracy	Test Accuracy
SVM	0.656	0.999	0.692
NBC	0.611	0.849	0.659
DTC	0.647	1.0	0.7
RFC	0.679	1.0	0.725
KNN	0.663	0.809	0.66
LRC	0.659	0.838	0.685

5. Model Deployment

- I have deployed a streamlit dashboard that shows various analytical results and a sentiment computer interface.
- The interface takes in a sentence as an input and predicts the sentiment as positive or negative. It has been discussed in detail in the deployment submission.

Conclusion

In this project, the Twitter dataset is explored in various ways and the results are presented in this report and also in the notebook provided. The topic modelling and sentimental analysis were the major sections in which the project was focused. The sentiment analysis result shows a Logistic regression model with Trigram vectorization outperforms the other combination of models.