Table of Content:

Abstract:

Living in the age where information quickly accessible on social media platforms exposes us and makes us susceptible to believing fake news articles. These articles underlying tone incites xenophobia and racism leading to polarized views fueled by lies. This problem is plentiful on this social media platform, the needs of which require us to experiment with machine learning and deep learning models to classify fake news to enhance user experience. This paper explores the methods, results and conclusions drawn from the analysis. It also makes recommendations for the future.

Introduction:

Social Media companies need to have satisfied customers for it to be successful and this report looks at our ability to predict fake news articles on a social media platform and flag them as fake or real. From a business perspective, the social media giant is the client and they are concerned about potentially losing current users from their platform. This is because current users have complained of fake news articles on the platform that are eroding their trust in the platform. The client is interested in predicting which news articles circulating on their platform are real or fake, and to flag the fake articles so users on the platform are aware of whether the article they are reading is real or fake. This project is time-sensitive because of the increase in misinformation being spread on the client's platform and increasing pressure from regulatory authorities to contain and identify fake news on its platform.

Methods:

Data Wrangling:

The dataset for news articles containing real and fake news articles was obtained from Kaggle in a csv format. The shape of the dataset was 20800 rows and 5 columns. There were 10413 real articles and 10387 fake articles in the dataset indicating that the dataset is balanced. The column names of the dataset were checked and **'title'** was converted to **'article_title'.** The **'id'** column was dropped as it was redundant in the dataset.

For visual purposes, the response variable, in this case ‘label’, was moved to the left side of the table. Also, it makes the dataset splitting into train/test set easier later on.

	label	article_title	author	text
0	1	House Dem Aide: We Didn’t Even See Comey’s Let...	Darrell Lucus	House Dem Aide: We Didn’t Even See Comey’s Let...
1	0	FLYNN: Hillary Clinton, Big Woman on Campus - ...	Daniel J. Flynn	Ever get the feeling your life circles the rou...
2	1	Why the Truth Might Get You Fired	Consortiumnews.com	Why the Truth Might Get You Fired October 29, ...
3	1	15 Civilians Killed In Single US Airstrike Hav...	Jessica Purkiss	Videos 15 Civilians Killed In Single US Airstr...
4	1	Iranian woman jailed for fictional unpublished...	Howard Portnoy	Print \nAn Iranian woman has been sentenced to...

To help with text preprocessing, a helper function was created. This helper function removes line breaks, new lines, hyperlinks, ampersand, greater than sign, less than sign, non breaking space, emails, new line characters and distracting single quotes. A new column **‘length’** was created that contains the length of the article text.

	label	article_title	author	text	length
0	1	House Dem Aide: We Didn’t Even See Comey’s Let...	Darrell Lucus	House Dem Aide: We Didn’t Even See Comey’s Let...	4886
1	0	FLYNN: Hillary Clinton, Big Woman on Campus - ...	Daniel J. Flynn	Ever get the feeling your life circles the rou...	4143
2	1	Why the Truth Might Get You Fired	Consortiumnews.com	Why the Truth Might Get You Fired October 29, ...	7670
3	1	15 Civilians Killed In Single US Airstrike Hav...	Jessica Purkiss	Videos 15 Civilians Killed In Single US Airstr...	3223
4	1	Iranian woman jailed for fictional unpublished...	Howard Portnoy	Print An Iranian woman has been sentenced to s...	934

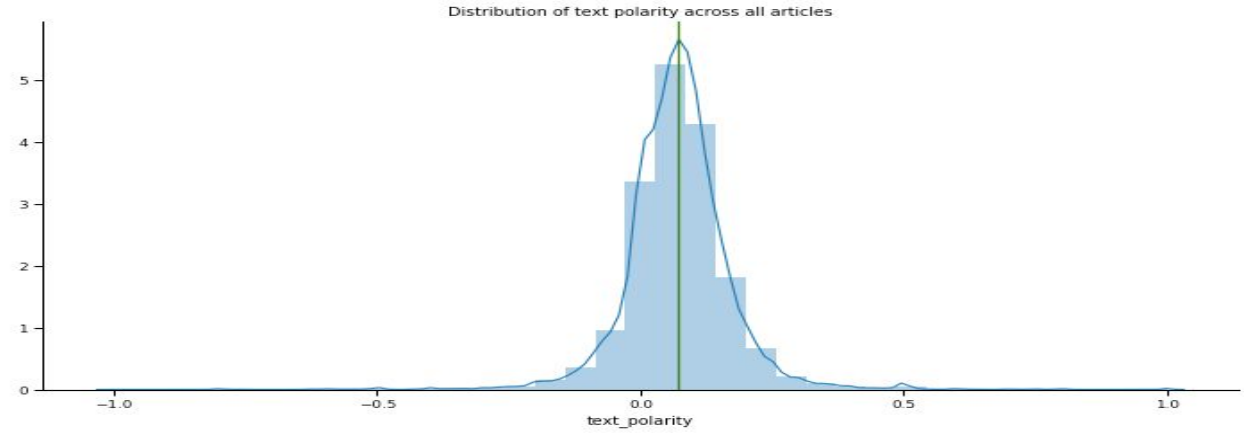
Next, null values were checked in the dataset. The columns ‘author’ and ‘article_title’ contained 1957 and 558 null values respectively. The null values in these columns were replaced by ‘Unknown’. To remove articles that are not really articles as they contain really few characters, I picked an arbitrary number of 50 characters. The length of text in articles that was less than 50 were dropped. The new shape of the dataset was 20554 rows and 5 columns.

Exploring the Data:

Before making any visualizations, another column called ‘text_polarity’ was added to the dataset. Using the TextBlob library’s sentiment polarity function on the ‘text’ column returned polarity of text on scale -1 to 1 indicating negative to positive.

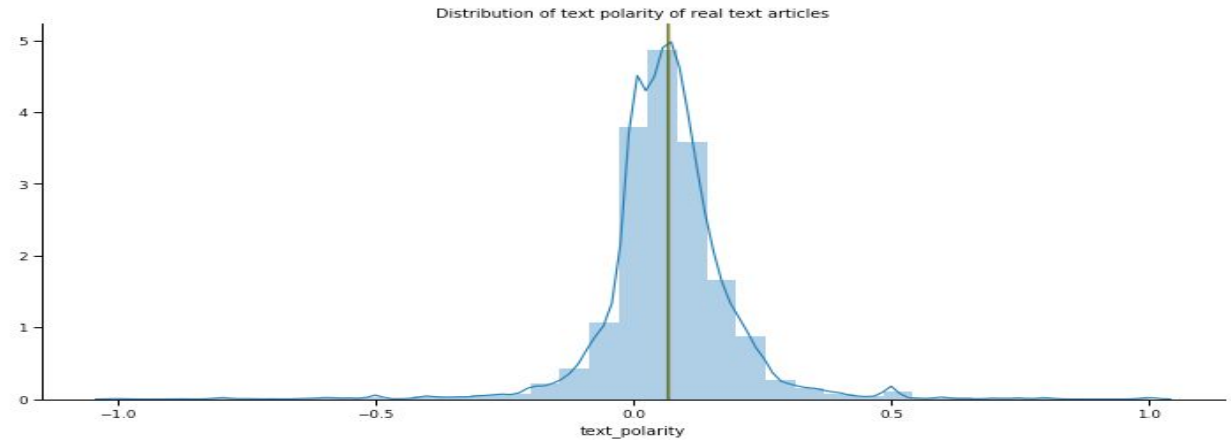
It would be interesting to see how the distributions of text polarity in all articles, real articles and fake articles differ. The distribution of text polarity across all articles seems to be evenly distributed with some articles being completely negative and completely positive.

```
The mean of text polarity in all articles is: 0.071
The median of text polarity in all articles is: 0.071
```



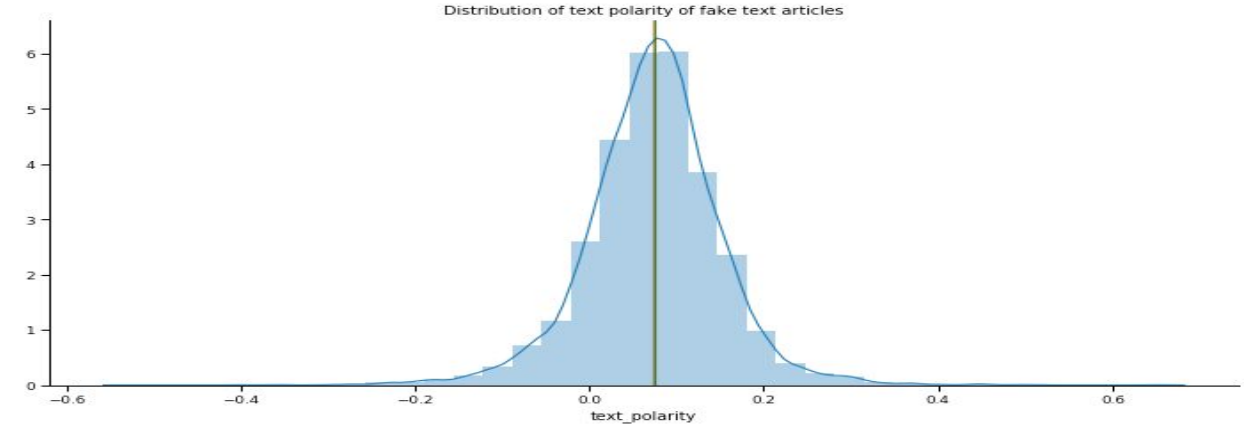
The distribution of text_polarity of real text articles again seems to be evenly distributed. It also contains articles that have completely negative or positive polarity.

```
The mean of text polarity in real articles is: 0.069
The median of text polarity in real articles is: 0.065
```

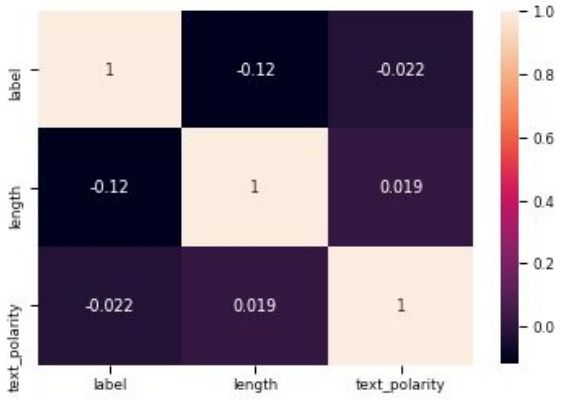


The distribution of text polarity of fake news articles seems normally distributed. An interesting thing to note is that there are no perfectly negative or positive text polarity (i.e. -1 and +1) in the fake news articles.

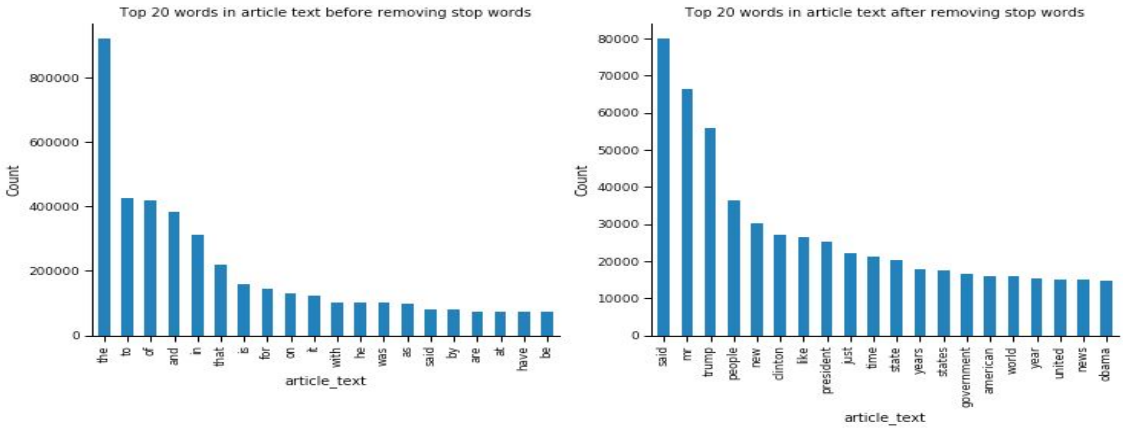
```
The mean of text polarity in fake articles is: 0.074
The median of text polarity in fake articles is: 0.075
```



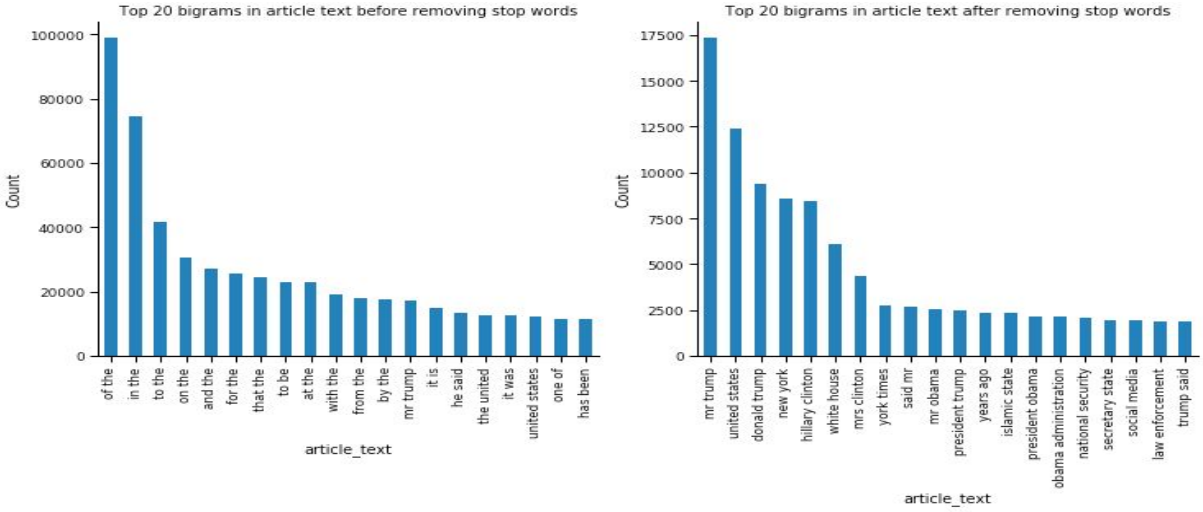
To check for any correlations between the different features in the dataset, a heatmap was used. There did not seem to be any strong correlations in the dataset. The greatest correlation value in the dataset was -0.12 between length and label, indicating a very weak negative correlation.



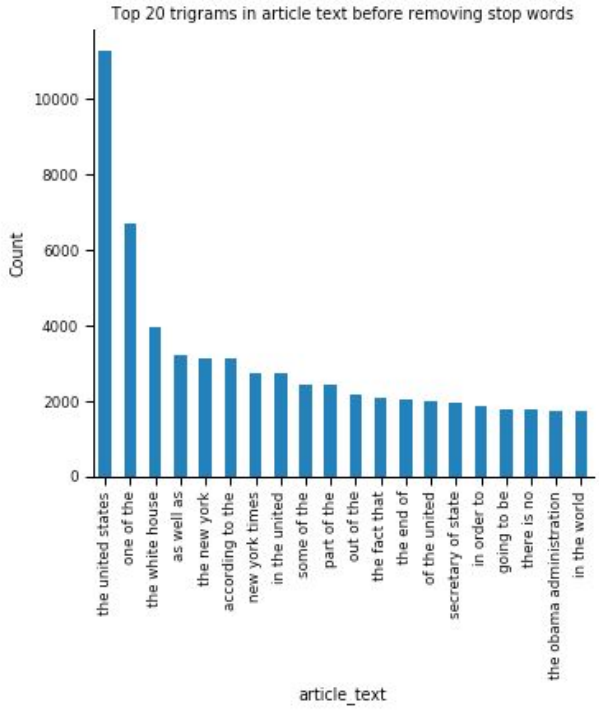
For all news articles text, the top 20 unigrams, bigrams and trigrams with and without removal of stop words were plotted. Without removal of stop words, the top word that occured in all articles was ‘the’. With the removal of stop words, the top word that occured was ‘said’.

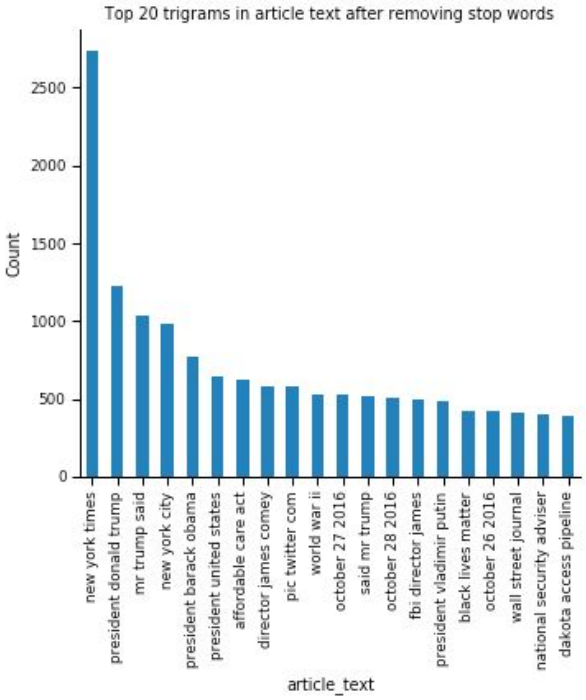


Without removal of stop words, the top bigram that occured in all articles was ‘of the’. With the removal of stop words, the top bigram that occured was ‘mr trump’.

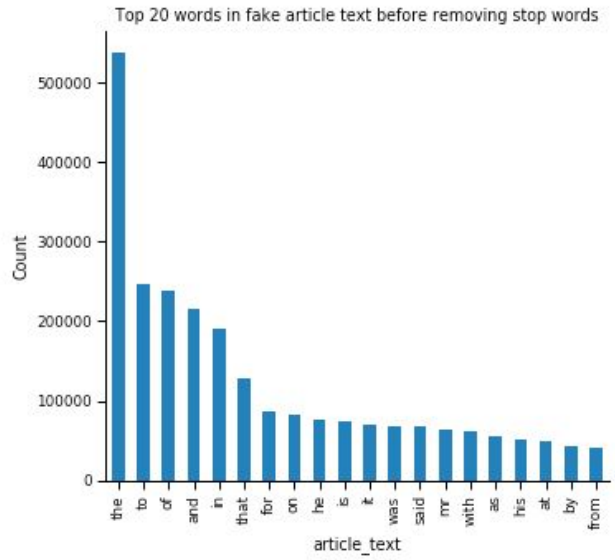


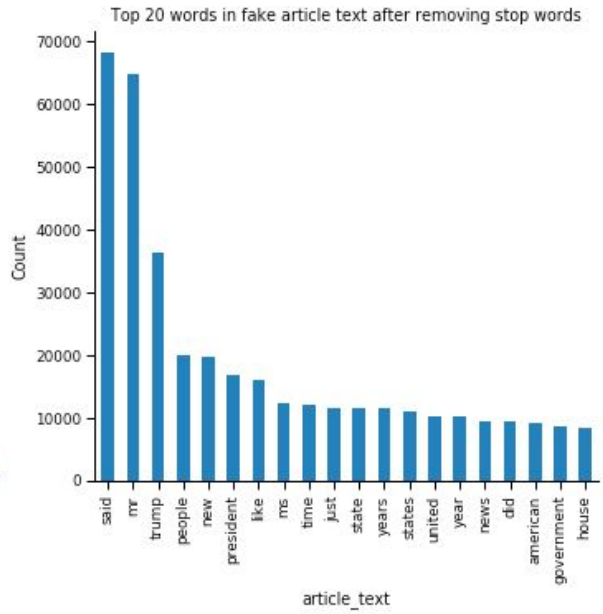
Without removal of stop words, the top trigram that occured in all articles was ‘the united states’. With the removal of stop words, the top trigram that occured was ‘new york times’.



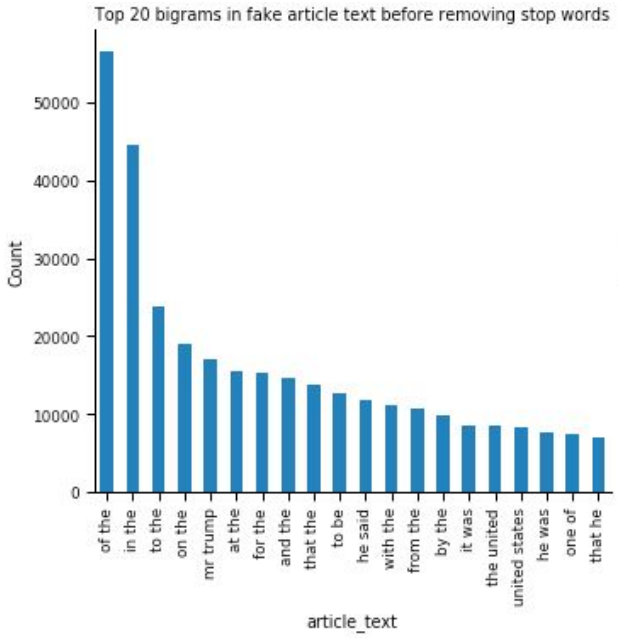


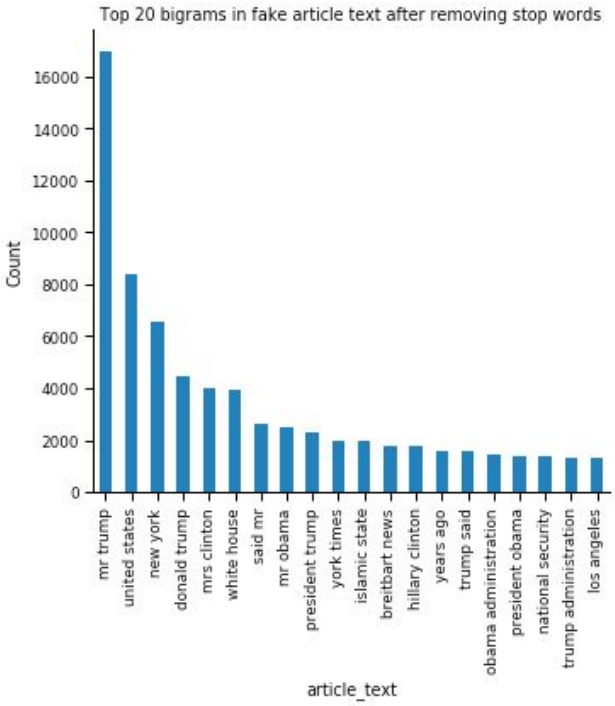
Next, for all fake articles text, the top 20 unigrams, bigrams and trigrams with and without removal of stop words were plotted. Without removal of stop words, the top word that occured in fake articles was ‘the’. With the removal of stop words, the top trigram that occured was ‘said’.



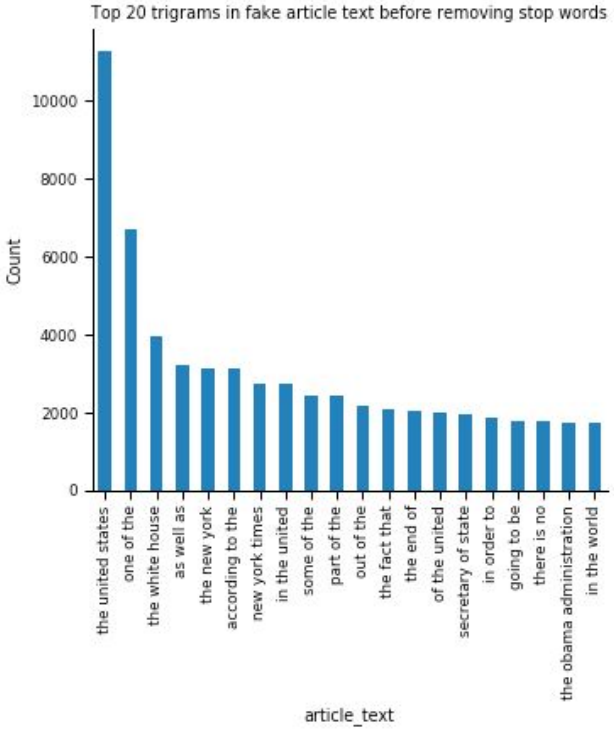


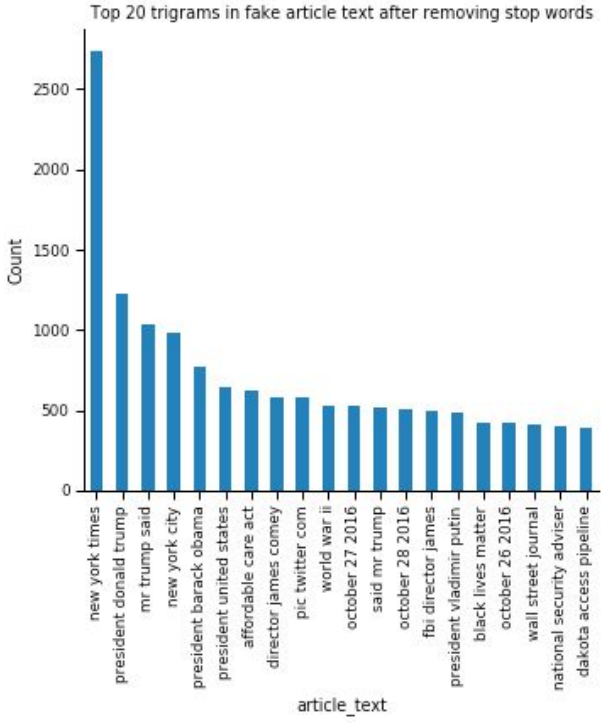
Without removal of stop words, the top bigram that occured in fake articles was ‘of the’. With the removal of stop words, the top bigram that occured was ‘mr trump’.



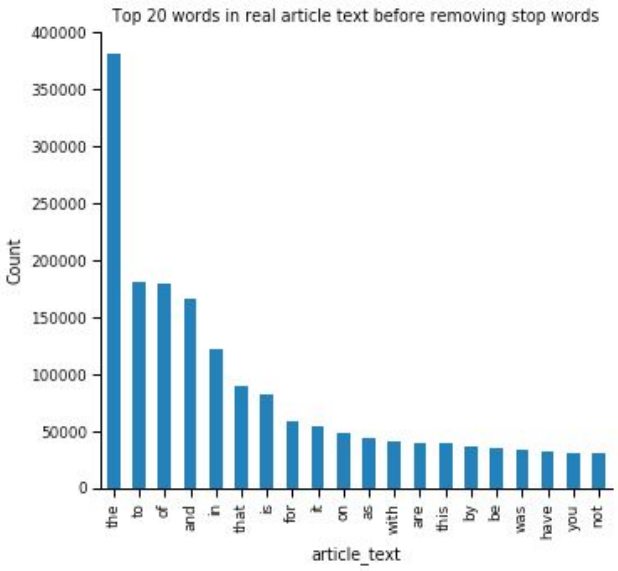


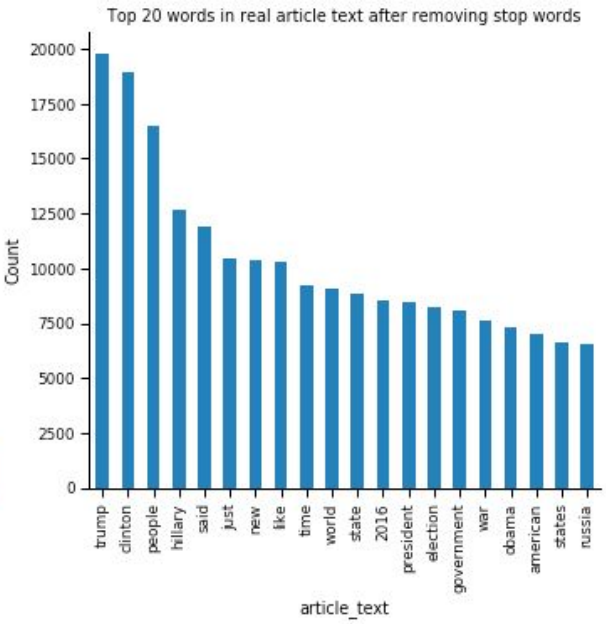
Without removal of stop words, the top trigram that occured in fake articles was ‘the united states’. With the removal of stop words, the top trigram that occured was ‘new york times’.



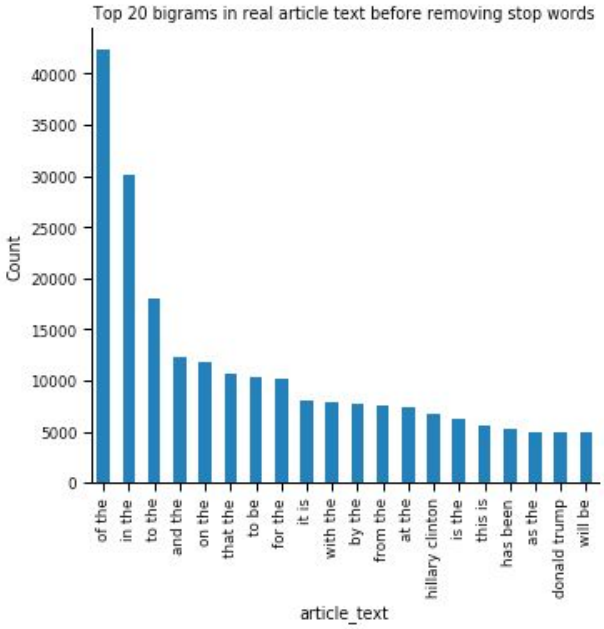


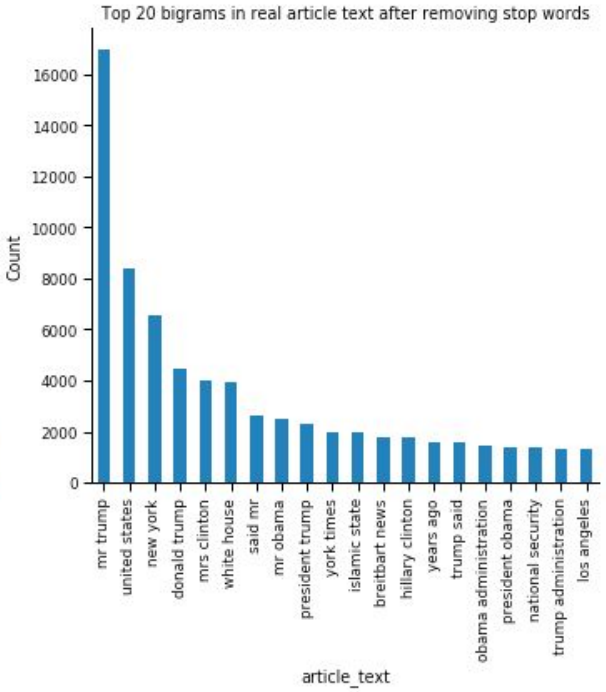
Finally, for all real articles text, the top 20 unigrams, bigrams and trigrams with and without removal of stop words were plotted. Without removal of stop words, the top word that occured in real articles was ‘the’. With the removal of stop words, the top word that occured was ‘trump’.



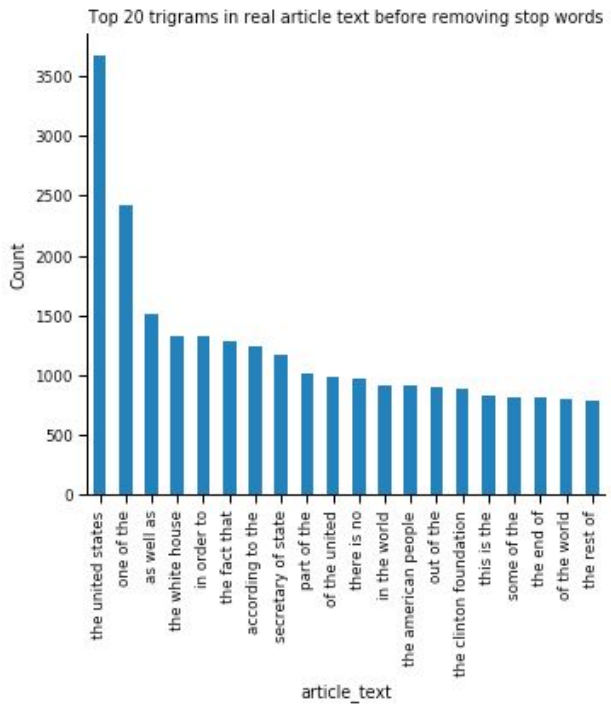


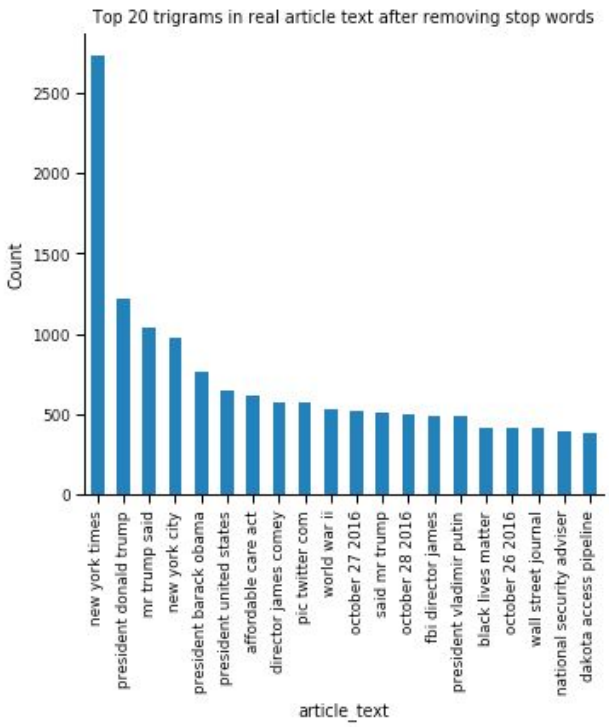
Without removal of stop words, the top bigram that occured in real articles was ‘of the’. With the removal of stop words, the top bigram that occured was ‘mr trump’.





Without removal of stop words, the top trigram that occured in real articles was ‘the united states’. With the removal of stop words, the top trigram that occured was ‘new york times’.





Inferential Statistical Analysis:

Next, a one-sample t-test was performed to test whether a population mean is significantly different from some hypothesized value. We are going to test to see whether the average text polarity of fake articles differs from the average text polarity of all articles.

Hypothesis Testing: Is there significant difference in the means of text polarity between articles that are fake and all text articles?

Null Hypothesis: The null hypothesis would be there there is no difference in text polarity between articles that are fake and all text articles.

Alternate Hypothesis: The alternative hypothesis would be that there is a difference in text polarity between fake articles and all articles.

The t-distribution left quartile range was -1.96112925575354 and right quartile range was 1.9611292557535396. The test result showed the test statistic 't' is equal to 3.0200. T is simply the calculated difference represented in units of standard error and tells us how much the sample mean deviates from the null hypothesis. The null hypothesis can be rejected if the t-statistic lies outside the quantiles of the t-distribution corresponding to the chosen confidence level and degrees of freedom.

```
Out[229]: Ttest_1sampResult(statistic=3.020018793001539, pvalue=0.0025337349388929752)
```

A p-value of 0.00253 means we'd expect to see data as extreme as our sample due to chance way less than 5% of the time if the null hypothesis was true. In this case, the p-value is lower than our significance level α (equal to 1-conf.level or 0.05) so we should reject the null hypothesis.

Based on the statistical analysis of a one sample t-test, there is a significant difference between the mean text polarity of all articles and the mean text polarity of fake news articles. The low P-value of 0.0025 at a 5% confidence interval is a good indicator to reject the null hypothesis.

Modeling:

This next step, which is the crux of our analysis, requires training models to predict whether a news article is fake or real. Before we can train the models, we need to convert the text into suitable input for these algorithms. To do this, we will use two methods that are called Count Vectorizer and TF-IDF Vectorizer.

The CountVectorizer provides a simple way to both tokenize a collection of text documents and build a vocabulary of known words, but also to encode new documents using that vocabulary. Conversely, TF-IDF (Term Frequency – Inverse Document Frequency) are word frequency scores that try to highlight words that are more interesting, e.g. frequent in a document but not across documents. The TfidfVectorizer will tokenize documents, learn the vocabulary and inverse document frequency weightings, and allow you to encode new documents.

Each of these two methods will act as input for both a Multinomial Naive Bayes Classifier and a Logistic Regression Classifier. Logistic Regression was used as the benchmark model. A benchmark model is a model that is used for reference to compare how better other models are against it. No hyper-parameter tuning was performed for logistic regression and it was trained with a C value of 0.01. Alternatively, for the Multinomial NB model, hyper-parameter tuning was performed and found alpha = 0.1 to be the best value in both the CountVectorizer and TF-IDF Vectorizer models.

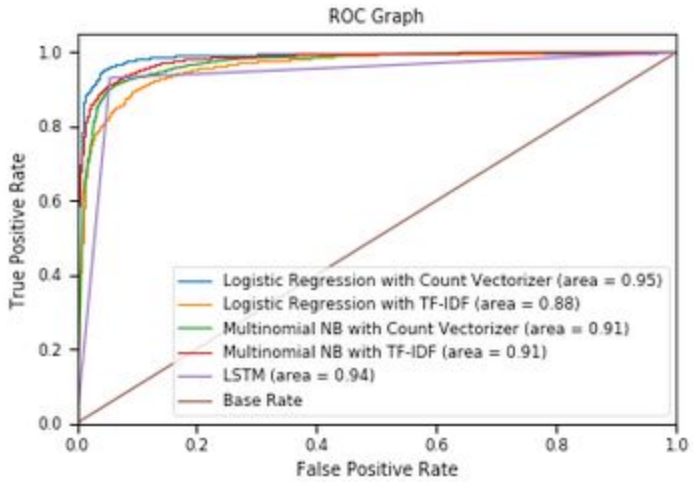
The LSTM (Long Short Term Memory) Neural Networks, are a particular type of recurrent neural networks. "*In a simple way, LSTM networks have some internal contextual state cells that act as long-term or short-term memory cells. The output of the LSTM network is modulated by the state of these cells. This is a very important property when we need the prediction of the neural network to depend on the historical context of inputs, rather than only on the very last input.*"

The LSTM model was created with the maximum number of words to be used to be 50,000 and the max number of words in each text to be 250. The LSTM model had a SpatialDropout1D value of 0.2, dropout value of 0.1 and recurrent dropout value of 0.1.

Results:

AUC - ROC curve is a performance measurement for classification problems at various thresholds settings. ROC is a probability curve and AUC represents degree or measure of separability. It tells how much a model is capable of distinguishing between classes. Higher the AUC, better the model is at predicting 0s as 0s and 1s as 1s. By analogy, Higher the AUC, better the model is at distinguishing between articles that are real or fake. Below is a table of the classifiers used and their respective AUC-ROC scores.

Classifier	AUC-ROC Score	Accuracy
Multinomial NB (Count Vectorizer)	0.91	0.91
Multinomial NB (TF-IDF)	0.91	0.91
Logistic Regression (Count Vectorizer)	0.95	0.95
Logistic Regression (TF-IDF)	0.88	0.88
LSTM	0.94	0.94



Logistic Regression with Count Vectorizer to tokenize the data gave us the best results. It achieved an accuracy of 95% and an AUC-ROC score of 95%. Although the LSTM model achieved a higher AUC-ROC score, it would be an overkill to use it since Logistic Regression performs better in accuracy and has a great AUC-ROC score. Also, Logistic Regression is less computationally expensive.

The worst model performance was also by Logistic Regression with TF-IDF Vectorizer achieving an accuracy and AUC-ROC score of 0.88.

Model Evaluation:

Apart from just using accuracy as an evaluation metric, we also use the AUC-ROC score which plots the True Positive Rate against the False Positive Rate. We should also take into consideration the False Positive and False Negative Errors to evaluate our models' performances.

False Positives (Type I Error): You predict that the article is Fake but is Real.
False Negatives (Type II Error): You predict that the article is Real but is Fake.

```
 -- Logistic Regression Model --
-- Logistic Regression Model AUC = 0.95 --
-- Logistic Regression Model with Count Vectorizer Accuracy = 0.95 --
[[1942  126]
 [  72 1971]]
              precision    recall  f1-score   support

           0       0.96      0.94      0.95      2068
           1       0.94      0.96      0.95      2043

    accuracy                           0.95      4111
   macro avg       0.95      0.95      0.95      4111
weighted avg       0.95      0.95      0.95      4111
```

```
 -- Logistic Regression Model --
-- Logistic Regression Model AUC = 0.88 --
-- Logistic Regression Model with TFIDF Accuracy = 0.88 --
[[1684  384]
 [ 102 1941]]
              precision    recall  f1-score   support

           0       0.94      0.81      0.87      2068
           1       0.83      0.95      0.89      2043

    accuracy                           0.88      4111
   macro avg       0.89      0.88      0.88      4111
weighted avg       0.89      0.88      0.88      4111
```

```
 -- Multinomial NB Model with Count Vectorizer --

-- Multinomial NB Model with Count Vectorizer AUC = 0.91 --
-- Multinomial NB Model with Count Vectorizer Accuracy = 0.91 --

[[1991   77]
 [ 282 1761]]

              precision    recall  f1-score   support

           0       0.88      0.96      0.92      2068
           1       0.96      0.86      0.91      2043

    accuracy                           0.91      4111
   macro avg       0.92      0.91      0.91      4111
weighted avg       0.92      0.91      0.91      4111
```

```
 -- Multinomial NB Model with TF-IDF --
-- Multinomial NB Model with Count Vectorizer AUC = 0.91 --
-- Multinomial NB Model with Count Vectorizer Accuracy = 0.91 --

[[2023   45]
 [ 306 1737]]

              precision    recall  f1-score   support

           0       0.87      0.98      0.92      2068
           1       0.97      0.85      0.91      2043

    accuracy                           0.91      4111
   macro avg       0.92      0.91      0.91      4111
weighted avg       0.92      0.91      0.91      4111
```

```
-- LSTM Neural Network on Tokenized Text --
-- LSTM Neural Network on Tokenized Text AUC = 0.94 --
-- LSTM Neural Network on Tokenized Text Accuracy = 0.94 --


[[1945  113]
 [ 142 1911]]


              precision    recall  f1-score   support

           0       0.93      0.95      0.94      2058
           1       0.94      0.93      0.94      2053

    accuracy                           0.94      4111
   macro avg       0.94      0.94      0.94      4111
weighted avg       0.94      0.94      0.94      4111
```

Conclusion:

The distribution of text polarity of all articles (both fake and real) seemed evenly distributed while some articles were completely negative and completely positive in their polarity. Similarly for real articles, the text polarity appeared evenly distributed and contained articles with completely negative or positive polarity. For fake articles, the distribution of text polarity also appeared to be normally distributed. However, an interesting thing to note is there are no articles that have perfectly negative or positive text polarity (i.e. -1 and +1).

A heatmap was used to check for any correlations between the different features in the dataset. There did not seem to be any strong correlations in the dataset. The greatest correlation value in the dataset was -0.12 between length and label, indicating a very weak negative correlation.

In all news articles, without removal of stop words, the top word that occured was ‘the’. With the removal of stop words, the top word that occured was ‘said’. Similarly, without removal of stop words, the top bigram that occured was ‘of the’. With removal of stop words, the top bigram that occured was ‘mr trump’. Lastly, in all news articles, without removal of stop words the top trigram that occured was ‘the united states’. With the removal of stop words, the top trigram that occured was ‘new york times’.

In all fake articles, without removal of stop words, the top word that occured was ‘the’. With the removal of stop words, the top word that occured was ‘said’. Similarly, without removal of stop words, the top bigram that occured was ‘of the’. With removal of stop words, the top bigram that occured was ‘mr trump’. Lastly, in all news articles, without removal of stop words the top trigram that occured was ‘the united states’. With the removal of stop words, the top trigram that occured was ‘new york times’.

In all real articles, without removal of stop words, the top word that occured was ‘the’. With the removal of stop words, the top word that occured was ‘trump’. Similarly, without removal of stop

words, the top bigram that occured was ‘of the’. With removal of stop words, the top bigram that occured was ‘mr trump’. Lastly, in all news articles, without removal of stop words the top trigram that occured was ‘the united states’. With the removal of stop words, the top trigram that occured was ‘new york times’.

Of all the models, Logistic Regression with Count Vectorizer to tokenize the data gave us the best results. It achieved an accuracy of 95% and an AUC-ROC score of 95%. Although the LSTM model achieved a higher AUC-ROC score, it would be an overkill to use it since Logistic Regression performs better in accuracy and has a great AUC-ROC score. Also, Logistic Regression is less computationally expensive.

Potential Solutions:

Now that we have built our classifier and also trained a LSTM network, both of which can be improved over time as we get more data and tweak our model settings, we can incorporate this model into the social media website. This can be done on the backend, where the model can be deployed to run against news stories that people share on the platform and flag the content as 'fake' or any other suitable title. We can also discuss with the data engineering team what the best way to incorporate the model into the site would be. There is a need to be cautious of causing downtime or disruption of service to users when updating the model. It could be possible to use Kubernetes to productionize the model and update it potentially alleviating us from any disruption of service.

It is important that rather than just predicting if the news article is fake or real, we also have an estimate of the probability that the news article is fake or real. We could flag the users that share fake articles consistently, and shut down these accounts in hopes of retaining the loyal user base of the social media platform.

We can run into two sorts of problems with this approach. Firstly, consider that the news article is flagged as fake by the model, but in reality the news article is real. This is called a false positive and this mistake could be expensive, as it could lead to embarrassment for the social media company and also potential loss of users. Though, this could be rectified quickly by building a mechanism to report articles that have been falsely flagged as fake. This could be one of the many directions the evolution of this project can head in.

On the contrary, consider that the news article is flagged as real but in reality is actually fake, and continues to get shared unchecked with more and more people being exposed to the fake news while thinking it is real. This is called a false negative and it has the ability to seed distrust and polarize online users on sensitive issues using racist and hyper-nationalistic rhetoric. This is more harmful to the validity of our model as well

as the reputation of the social media company, as it leads to people distrusting the platform and its ability to separate truth from lies. To mitigate this issue, a mechanism should be built that allows people to report if the article is fake and then a team assigned to fact checking the article make the ultimate decision based on their analysis. These are just some solutions that can be implemented, neither are they meant to be perfect nor do they claim to be.

Since, this is a relatively new area and we will learn more about how to counter the different tactics employed by individuals and states to spread disinformation on social media platforms. The most important thing for our future accuracy of predictions is tied to how much more data we collect to enhance the model.

To summarize all of the above mentioned points:

Solution 1:

We can rank the articles with a probability estimate of it being real or fake, indicating how confident we think a particular article is real or fake. This will provide the readers some context and enough to decide for themselves to be skeptical of the news or not. The social media platform isn't making the decision for their users but giving them the information to decide for themselves so it does not feel like the user is being coerced into believing something or not.

Solution 2: Provide free training to the users of the social media platform to identify fake news. For example, tips and resources can be shared that discuss common ways one can fact check a news article. It can be as simple as copy pasting information from the article into google, and seeing if the information adds up.

Where do we go from here?

This problem is about equipping the social media company with actionable knowledge regarding their ability to identify fake news on their platform and countering it. When modeling the data, we should not use the predictive metric as our final solution. Instead, we should use the information we get from modeling and arm the social media platform users so they can carry out informed decision making.

Once the model is deployed, as more and more articles get shared on the platform, our dataset will grow. This will help make our model more accurate over time by allowing us to even test out different techniques or other models that may perform better or are more computationally efficient for our use case. For now, this model will do as it is better to have some knowledge about an article's validity than to be completely in the dark in this age of misinformation.

References:

https://medium.com/datathings/the-magic-of-lstm-neural-networks-6775e8b540cd