Building a Fake News Detector – final project report

Jonathan Pearce (July 2023)

Problem statement

The reliability of information is key to businesses, governments and media organisations around the world. The rise of misinformation and fake news has a massive impact on business and government to effect change or develop appropriate responses/behavioural change within society, as well as sell products and provide services. With the rise of social media and online forums, and with customers becoming content generators, businesses have even more to contend with.

They have to manage business and product/service reputation issues linked with fake news or misinformation, along with dealing with an increasing number and range of regulatory and compliance requirements covering legal and liability responsibilities to do with managing fake news, false information and abuse/offensive content/materials

Thus, knowing what is true and what is fake in terms of protecting your own business (either as a publishing organisation or as a product producer or service provider) and enhancing your reputation and profile is crucial.

So, reliable and effective tools that can help governments, businesses and media/publishing organisations manage fake news and false/misleading information will only become more and more important. This project set out to create a fake news detector to address such issues.

Context

The reliability and credibility of information/content is an issue that cuts across all content/subject areas, and is ultimately one of trust and credibility. While trust and credibility could be seen as possibly a slightly intangible value, or something that is secondary to commercial or pure business issues, one only has to look at countless examples where lack of trust or credibility has had a catastrophic impact on business interests and concerns. For example, the international banking/financial crises of 2008 were largely about trust and credibility in financial information regarding consolidated mortgage debt and the financial instruments used to package them. More recently in the last decade, the Volkswagen/Audi group has had to feal with the multimillion fallout from the cover-up regarding its falsification of the emissions data for its vehicles.

With the rise of social media and online forums, and with customers becoming content generators, businesses have even more to contend with. On one level, the number and type of companies that publish information has increased massively as technology has grown and the internet has developed and expanded. It’s not just traditional publishers or news/media organisers that publish information and therefore have to ensure the accuracy and reliability of their information, and how trusted it is. Any organisation that runs any aspect of social media or an online community now falls into that category (which at one end of the spectrum can be a simple as having a comments feature on your company content).

On another level, businesses need to be aware of and manage their reputations and faith/trust in their products from user/purchaser reviews of their products and services on a whole range of internet and social media sites.

On yet another level, there are a whole range of legal and compliance issues regarding content and content management. Traditional legal issues and liabilities affect anyone responsible for false, abusive, misleading, offensive, etc, information. Regulation and liability will only increase with international authorities around the world all considering new and enhanced responsibilities to manage and control falsification of information, particularly in the light of reported developments in machine learning and artificial intelligence.

Data Wrangling and pre-processing

There are a whole range of readily available datasets available on the internet to test the development of such models, but after a review of a number of them, it was decided to focus on ones with a decent amount of text so that a reliable model could be developed. In practice many of the available datasets were too small or limited.

The project used the following dataset from https://www.kaggle.com/datasets/jruvika/fake-news-detection?select=data.csv, which contains publication URLs, headlines, full content and binary ratings on over 4000 news articles.

The essence of building a fake news detector is a natural language processing/classification problem, so we needed to ensure that the textual data to be used in the model is cleaned, optimised and standardised so as to give the classification algorithms the best chance of understanding the numerical/vector relations between the words and phrases in the dataset. There are various options on how to vectorize data and which classification algorithms to use, but first the data need to be prepared.

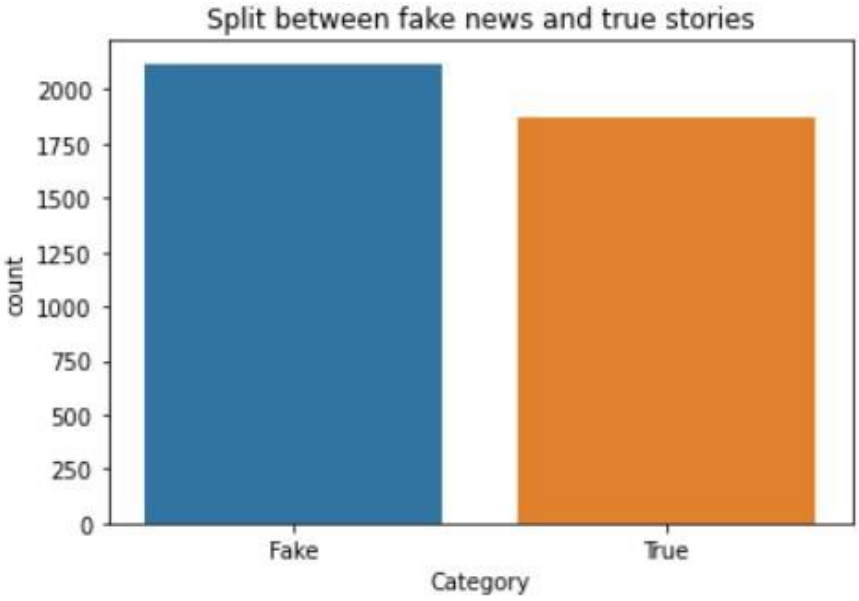
First, it was decided to ignore the URLs, as these would give too much information on whether or not the article was from a trusted source (or not) and so, if used, would defeat the purpose of the project, which is to build a detector than can work from textual data only, without obvious clues as to the provenance of the data. Then, it was also decided to merge the headlines with the body text to create a new combined data column, which would form the basis of the text to be analysed.

Next basic standardising and data cleaning methods were used to:

- Remove punctuation.
- Remove non-alpha-numeric characters.
- Change all the text to lower-case
- Consider any other content issues, eg, removing website addresses, etc.

A review of null values was carried out, which revealed 21 articles where there was no article body text, but just a headline. Various options were considered, eg, just using the headings (to act as a proxy for the text) or looking up the stories from the URLs, or dropping the relevant rows. Looking up the articles was quickly discounted, as some had been discontinued and others were behind pay walls. The proxy content option didn't seem very sound given the apparent length of many of the other articles and also because the heading wouldn't be fully representative of the full article content. In the end, given that the null values represented only a small subset of the dataset (about 0.5%) it was decided to drop them, accepting a small drop in the size of the dataset, but having improved data quality.

The split between stories labelled as fake and those labelled as true is shown in the figure below:



Exploratory data analysis

The initial data analysis explored the following aspects of the dataset:

- lengths, distributions and averages of the different components of each article, such as characters, words, etc.
- analysis according to label subset
- reviewing the above with and without stopwords (ie, those commonly used, generally small words that add little to the overall meaning of a piece of text).

It also looked at the bi-grams and N-grams in the text to see what could be learned, before then building some word clouds.

By plotting histograms on the number of text characters and number of words in each article, it initially looked like the fake news articles were slightly shorter than the true stories. Looking more closely it became clear that this was definitely the case, as follows:

- The average number of characters in fake news stories was 2395 compared to 3620 for the true stories.
- The average number of words in fake news stories was 417 compared to 618 for the true stories (and 511 for all the articles in the dataset).

So, as a rule of thumb, the fake news stories were about one-third shorter than the true stories.

In addition, interestingly, the histograms on the average word length show that the fake news articles had a lower average word length than the true stories (see figure below).

Fake **True**

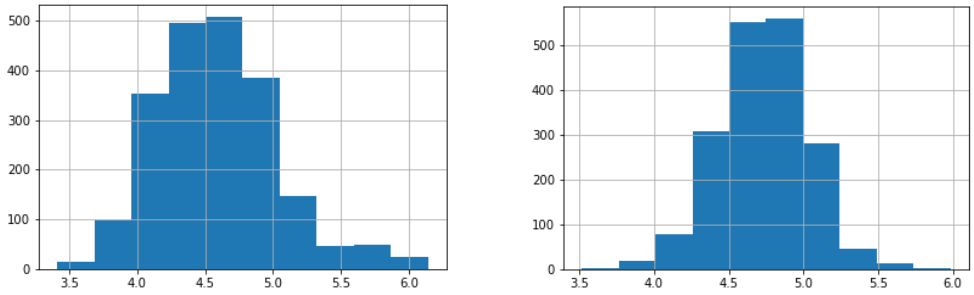


Figure: average word lengths of article for fake news (left) and true stories (right) subsets

This all suggests that the fake news stories were characterised by shorter articles, with simpler, shorter words. Overall we saw that the, despite this the majority of articles are under 5000 characters or 1000 words in length, with the data being right-skewed with a long tail, accounting for articles up to c22k characters or 4000 words in length, but with the true stories tending to account for more of the longer articles.

We also removed stopwords from the text that is being used in natural language processing as these words do not provide any useful information to help us understand the content or nature of the text. They might not have any useful meaning (eg, though being prepositions or conjunctions) or simply because they occur too frequently. We then analysed the most commonly used non-stopwords, subsetting this by fake and true stories. The biggest notable difference was the large use of the word “said” in the true stories compared the fake news – 8,000+ uses compared to 1,000+ uses. This suggested that validation, attribution and accurate citing of sources played a strong role in the true stories. See figure below.

Fake **True**

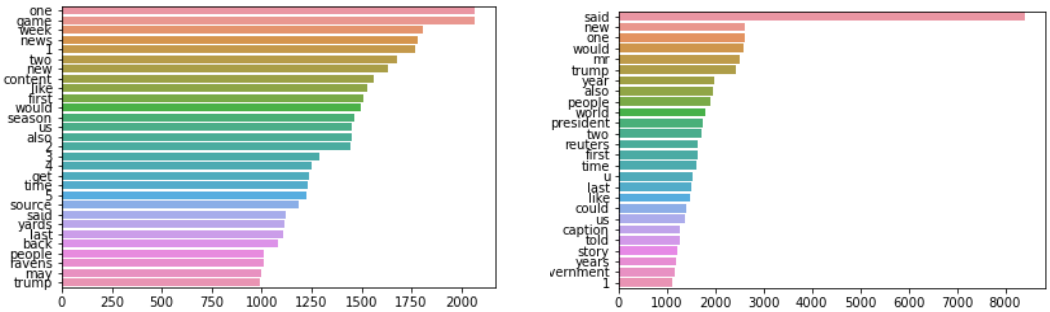


Figure: top non-stopwords in fake news and true stories, respectively

The analysis of bi-grams and tri-grams was less rewarding although it did highlight the preponderance of words or phrasing directed at the reader of the fake news articles,

including exhortations to engage them or give their view, or think of what other people think, etc. See figure below.

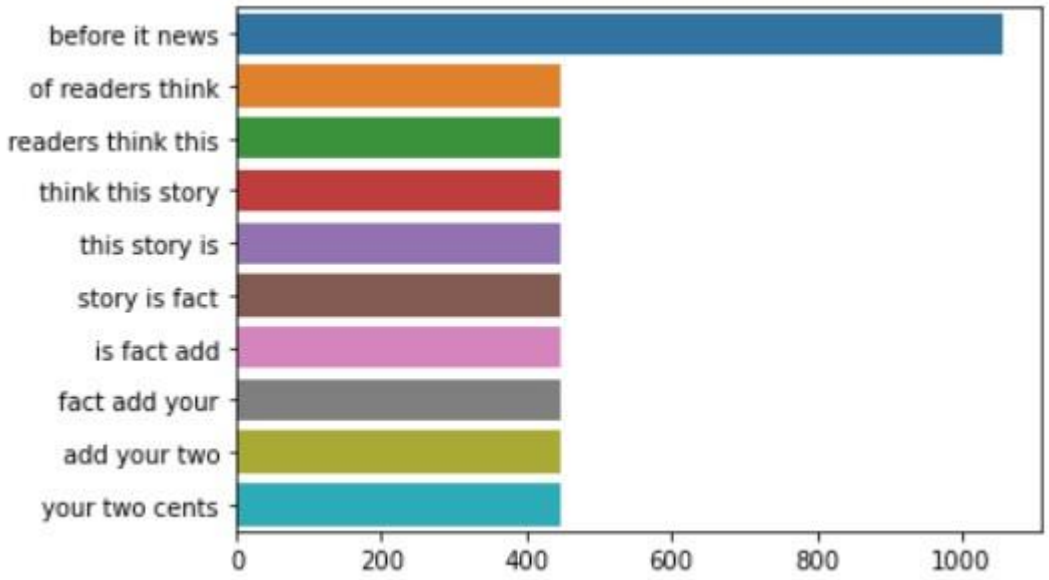
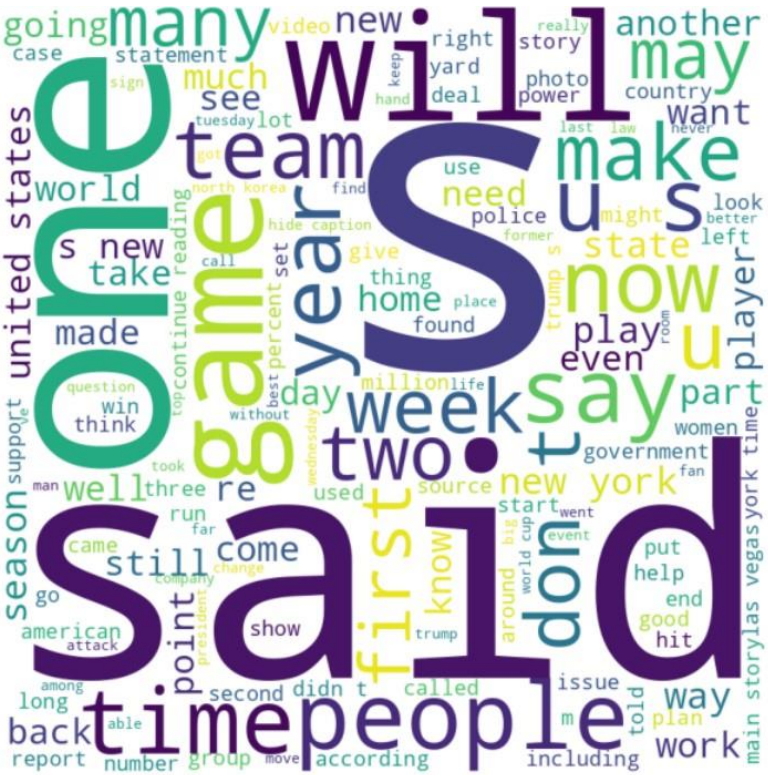


Figure: top tri-grams in fake news stories

Finally, word clouds were compiled for the dataset as a whole and subsetted by story type. See below for the whole dataset.



Modelling and selection

Vectorization

Before building and testing models for a natural language processing task, the text to be used has to be vectorized, ie converting the text in each article into numerical entries in the form of vectors/matrices. The are a number of vectorization techniques, including:

- count vectorization
- N-Grams
- Bag of Words (BOW)
- Term Frequency-Inverse Document Frequency (TF-IDF)
- Word2Vec

Each have their pros and cons.

Count vectorization generates a document term matrix where each cell is the count corresponding to the news article indicating the number of times a word appears in the document, also known as the term frequency. The document term matrix is a set of dummy variables that indicates if a particular word appears in the document. A column is dedicated to each word in the corpus. The count is directly proportionate to the correlation of the category of the news title. This means, if a particular word appears many times in fake news titles or true news titles, then the particular word has a high predictive power of determining if the news title is fake or true. The vectorizer produces a sparse matrix output, but only the non-zero values are stored in order to save memory.

TF-IDF is similar to count vectorization - a document term matrix is generated and each column represents a single unique word. The difference in the TF-IDF method is that each cell doesn’t indicate the term frequency, but the cell value represents a weighting that highlights the importance of that particular word to the document. The sparse matrix output for this method displays decimals representing the weight of the word in the document. High weight means that the word occurs many times within a few documents and low weight means that the word occurs fewer times in a lot of documents or repeats across multiple documents.

Bag of Words (BoW) is the simplest technique, involving tokenization, vocabulary creation, and vector creation. BoW is only concerned with the frequency of the words in a particular text, meaning there is limited distinction between different types of words, so that articles, prepositions, conjunctions, etc, may be elevated in importance compared to, say, adjectives, even though the former might not be that relevant.

TF-IDF avoids that issue (in that often-used words don't overpower less frequently used, but important words), and offers more nuance than count vectorization, hence why this form of vectorization was decided upon in the first instance.

Word2Vec offers an alternative approach. It creates high-dimensional vectors where words are assumed to be completely independent of each other. By using a neural network with only a couple of layers, Word2Vec tries to learn relationships between the words and then embed them in a lower dimensional space. In this way, the representation of the words is said to be contextually aware because the words that have similar meanings are mapped closely together via their vectorization "rating".

So it was decided to use Word2Vec also, but only after building and testing models with TF-IDF.

Model-building

Using TF-IFD vectorization

Because the problem is a binary classification problem, it was decided to try the following different classifier algorithms, namely:

- Passive Aggressive Classifier
- Random Forest Decision Tree Classifier
- Logistic Regression Classifier
- Linear Subject Vector Classifier
- Naive Bayes Multinomial Classifier

In terms of evaluation, a classification table and confusion matrix were produced for each model. While accuracy was used as an initial indicator (and because the dataset was reasonably well-balanced between true and false, it was a reliable first indicator), the number of false positives was also seen as important, ie, the success of the model in reducing the number of fake news stories that got through and were classed as true.

All models had good accuracy results:

- Passive Aggressive - 98.37%
- Random Forest - 96.49%
- Logistic Regression - 97.37%
- Linear SVM - 98.5%
- Multinomial Naive Bayes - 93.86%

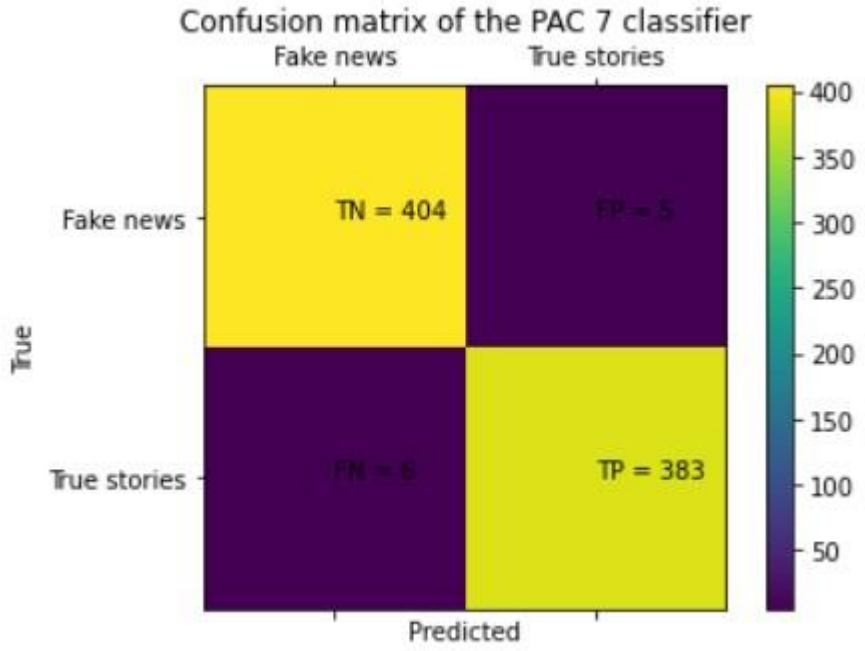
However, the two that stood out were the PassiveAggressive Classifier and the Linear SVM Classifier. Not only were their accuracy scores slightly higher, but the number of false positives (ie, the number of fakes news stories that they let through) were significantly lower than the other algorithms (6 and 8, respectively, compared to double figures for the others and sometimes into the high 20s).

Using Word2Vec vectorization

The models were retried using Word2Vec vectorization, but this proved less successful, with accuracy levels between 4 and 8% lower than with TF-IFD vectorization. This may be because Word2Vec is better suited to larger datasets.

Model selection

Given the results, model selection and development continued with the PassiveAgressive and Linear Subject Vector Machine classifiers. Both manual tuning and GridSearch tuning were carried out. It was not possible to improve the Linear SVM model, but further tuning slightly improved the PassiveAggressive classifier, resulting in a final accuracy level of 98.62% and only 5 fake news stories incorrectly classified as true. See full results below.



Classification_report

	precision	recall	f1-score	support
0	0.99	0.99	0.99	410
1	0.98	0.99	0.99	388
accuracy			0.99	798
macro avg	0.99	0.99	0.99	798
weighted avg	0.99	0.99	0.99	798

Conclusion and next steps

Having tried a number of classifiers, it's clear that once you have a cleaned, standardised and vectorized dataset of articles/text, it's possible to build a passable model with many of them. However, in order to refine that model and focus on things that really matter - including in the context of fake news, reducing the number fake articles that slip through the net - then only the PassiveAggressive Classifier and the Linear Subject Vector Machine Classifier were really up to the job. Parameter tuning offered only marginal improvements, but was worth doing for the PassiveAggressive Classifier.

In terms of word vectorization, it was interesting to see that TF-IFD performed well, in marked contrast to Word2Vec. This may be due to the size of the dataset. Even though the dataset was of a decent size (4000 articles), it may be that Word2Vec works better on much larger datasets.

The next steps would be try out the model in practice and to develop refined models with larger datasets and different categories of news/content.