* Rejected datasets bc only one new source
* Rejected bc not full text
* <https://www.kaggle.com/ruchi798/source-based-news-classification>
* <https://www.kaggle.com/mdepak/fakenewsnet?select=PolitiFact_fake_news_content.csv>
* <https://www.aclweb.org/anthology/P17-2067/>

<https://github.com/sfu-discourse-lab/MisInfoText>

* <https://github.com/sfu-discourse-lab/Misinformation_detection/blob/master/snopes_checked_v02.csv.zip>
* <https://github.com/sfu-discourse-lab/Misinformation_detection/blob/master/buzzfeed-v02-originalLabels.txt.zip>
* <https://github.com/sfu-discourse-lab/Misinformation_detection/blob/master/buzzfeed-top.csv.zip>

<http://www.fakenewschallenge.org/#stageone>

* <https://github.com/FakeNewsChallenge/fnc-1>

<https://journals.sagepub.com/doi/full/10.1177/2053951719843310>

Methods used for text classification vary from classic machine learning algorithms using a set of pre-defined linguistic features to modern neural network models which mainly rely on pre-trained word vectors and embedded representations resulting from processing large amounts of textual data.

In NLP, the feature-based approach, which involves the extraction and analysis of linguistic cues for identification of specific target phenomena (e.g., fake product reviews from real ones) has been a very powerful model with relatively interpretable results. Features such as n-grams, subjectivity and polarity markers, lexical semantic classes, syntactic or discourse-level features have been explored in previous work on deception detection in general and on news classification in particular. These features can be used with a variety of traditional supervised algorithms. Feature-based modelling usually involves feature engineering and a feature selection phase. Based on comparative experiments in different machine learning applications, it has also been shown that the performance of these classic models plateaus at some point as the training data size increases ([Ng, 2011](https://journals.sagepub.com/doi/full/10.1177/2053951719843310)). Thus, in problems where Big Data is available, deep neural network models are being preferred, as they usually achieve impressively better results (for a recent overview of the NLP trends see [Young et al., 2018](https://journals.sagepub.com/doi/full/10.1177/2053951719843310)).

Deep learning has taken over most NLP tasks but usually in domains where large-scale training data is available. In text classification, recurrent neural networks (RNNs), convolutional neural networks (CNNs) and Attention models have been competing with feature-based models ([Conneau et al., 2017](https://journals.sagepub.com/doi/full/10.1177/2053951719843310); [Le and Mikolov, 2014](https://journals.sagepub.com/doi/full/10.1177/2053951719843310); [Medvedeva et al., 2017](https://journals.sagepub.com/doi/full/10.1177/2053951719843310); [Yang et al., 2016](https://journals.sagepub.com/doi/full/10.1177/2053951719843310); [Zhang et al., 2015](https://journals.sagepub.com/doi/full/10.1177/2053951719843310)). CNNs are composed of convolution and pooling layers, which provide an abstraction of the input. These models are employed in specific NLP tasks where the presence or absence of features is a more distinguishing factor than their location or order. For example, presence of specific words and phrases in a product review is usually indicative of it being a positive or negative review. Therefore, CNNs are well suited for the purpose of longer text classification. Neural network models have also been applied in previous work within the domain of misinformation and fake news ([Rashkin et al., 2017](https://journals.sagepub.com/doi/full/10.1177/2053951719843310); [Wang, 2017](https://journals.sagepub.com/doi/full/10.1177/2053951719843310); [Yang et al., 2017](https://journals.sagepub.com/doi/full/10.1177/2053951719843310)).

<https://arxiv.org/pdf/2006.03644.pdf>

For these task, stance detection has been used as a key feature for checking the credibility of a piece of news. As discussed in section 5.3, stance in comments toward the news is measured to detect if these comments are confirming or denying the news, which, in turn, is used to detect if the news is a rumor or authentic.

The Fake News Challenge initiative (FNC-1) adopted this approach and proposed a stance detection task to estimate the stance of articles toward a given headline (i.e., claim). The best performing system at the FNC was proposed by [11]. In their study, they used gradient-boosted decision trees and a convolutional neural network (CNN), along with many textual features. Another study by [90] achieved relatively similar results to the best system with a feature-light memory network model enhanced with LSTM and CNN. A more recent study by [107] used three features extracted from users’ interactions, news authors, and the contents of news article to better detect the fake news. In the study by [48], a combination of lexical knowledge, word embedding, and n-gram was to detect the stances in two datasets, FNC-1 and Emergent. The study by [21] proposed a new text representation that incorporated the first two sentences of the article along with the news headline and the entire article to train a bidirectional RNN model using an FNC-1 dataset

<https://arxiv.org/pdf/1808.02831.pdf>

However, the ability to inspect and evaluate the contribution of individual, interpretable features is particularly desirable for an emergent task like fake news detection, because it can provide valuable insight into the nature of the task and the inherent idiosyncrasies that might exist in the data. For this reason, we introduce a new system developed using only traditional feature engineering. By incorporating hand-crafted features, we hope to gain a better understanding of what exactly constitutes fake news, and which kind of text-based features contribute the most to successful stance detection. Overall, our system outperforms the official baseline by 3.43% and would rank #13/50 on the official leaderboard

Riedel et al. (2017) introduce a NN architecture based on term frequency-inverse document frequency (TF-IDF) transformed bagof-words (BOW) representations as input to a multi-layer perceptron (MLP).

Due the high imbalance of class labels in the original data set, we also experiment with simple oversampling of the disagree category to match the number of agree samples in the original dataset. Table 2 shows the data distribution of the dataset after resampling.

Our system is built with XGBoost4 , an optimized distributed gradient boosting library. Gradient boosting is a popular technique that can solve complex regression or classification tasks by producing and combining a number of weaker and smaller prediction models in the form of decision trees. The model is built in stages and generalized by optimizing a differentiable loss function. As a result, gradient boosting combines a number of weak learners into a single, strong learner on an iterative basis. In contrast to linear classifiers (such as logistic regression) decision tree models are capable of capturing non-linear relationships in data as well.

We estimate the best hyperparameter settings for each model using a grid search with cross-validation on the training set. Carefully tuning the tree-related hyperparameters (such as the maximum depth of a tree) results in the largest increase of cross-validation accuracy.

A screenshot of a social media post

Description automatically generated

“Intro” features are computed on the first 250 character of the original body text. Most of the features assume basic preprocessing of the input text, including removing stop words and punctuation and lemmatizing all remaining tokens in the resulting sequence. All preprocessing is done using spaCy5 , an emerging open source NLP toolkit written in Python.

<https://www.kaggle.com/kumudchauhan/fake-news-analysis-and-classification>

* [3.2 Fake news detection from title of the news article:](https://www.kaggle.com/kumudchauhan/fake-news-analysis-and-classification#fake-news-detection-from-title-of-the-news-article)
  + [3.2.1 Naive Bayes Classifier as Base line model](https://www.kaggle.com/kumudchauhan/fake-news-analysis-and-classification#naive-bayes-classifier-as-base-line-model)
  + [3.2.2 Logistic Regression Classifier](https://www.kaggle.com/kumudchauhan/fake-news-analysis-and-classification#logistic-regression-classifier)
  + [3.2.3 Random Forest Classifier](https://www.kaggle.com/kumudchauhan/fake-news-analysis-and-classification#random-forest-classifier)
* [3.3 Fake news detection from body of the news article:](https://www.kaggle.com/kumudchauhan/fake-news-analysis-and-classification#fake-news-detection-from-body-of-the-news-article)
* [3.4 Fake news detection using terms appearing either in title or body of the news article:](https://www.kaggle.com/kumudchauhan/fake-news-analysis-and-classification#fake-news-detection-using-terms-appearing-either-in-title-or-body-of-the-news-article)