Source Based Fake News Detection

Notebook (code) [15]: https://drive.google.com/file/d/10dYxFgk5NdsiEX4_CpG4Opvg2wMz_BLz/view?usp=sharing

1 Problem Area

Problem

Fake news and its implications

Fake news refers to fabricated or misleading information presented as news, often with the intention of deceiving or manipulating the audience. Its spread can erode trust in legitimate sources, hinder informed decision-making, and even incite violence [8]

The spread of fake news online poses a significant challenge in today's digital world. The pervasive issue of fake news, particularly in the context of major global events like the COVID-19 pandemic, presents significant challenges across both social media platforms and web search engines. Misleading information can have serious consequences, impacting public opinion, swaying elections, and even endangering public health. Manually identifying fake news is time-consuming, subjective, and often impractical.

Contribution of text classification methods

Fake news detection

Text classification methods, leveraging advanced machine learning techniques, are pivotal in addressing this challenge by automatically detecting and mitigating the spread of false information.

- **Content-Based Detection:** Existing research on fake news detection often focuses on analysing the content itself. Techniques like sentiment analysis, named entity recognition, and identifying stylistic features can help flag suspicious messages [8].
- **Source-Based Detection:** Studies have also investigated the credibility of the source as a factor in fake news detection. A paper published in IJIRSET (2020) explores "Source Based Fake News Classification using Machine Learning". This approach complements content analysis by considering additional source-based features such as the source (publishing website) of the articles, authors, publishing date, etc. to glean possible insight on the reputation and past history of publishing misinformation [10].

Using machine learning models in conjunction with feature engineering to classify news articles based not only on content but also source features of articles like the source website url, author, title, and text content, is a new realm of research as presented by Patil et al. (2020).

Prior Work

1. The first paper by Patil et al. (2020) focuses on the use of machine learning to classify fake news by examining the sources of the articles, alongside other attributes such as authors, titles, and the body text of articles. The study underscores the difficulty in manually classifying news due to the labor-intensive nature of the process and the inherent bias that manual classification can introduce. This work highlights the necessity of automated systems to enhance the accuracy and efficiency of fake news detection [10].

2. Building on this, the second paper by Mazzeo et al. (2021) extends the scope of detection to the realm of web search engines, which despite being trusted sources of information, are not immune to being exploited for spreading misinformation. This paper emphasises the dual use of textual content and URL features to improve the classification processes. Their approach involves not just traditional text-based features but also explores the structural aspects of URLs, which are commonly exploited in phishing and malware distribution. The authors apply various machine learning algorithms to handle class imbalances inherent in real-world data, suggesting the necessity of sophisticated methods to deal with the nuances of fake news detection in complex datasets [11].

2 Objectives

The spread of fake news online undermines trust in media, fuels polarisation, leading to several real-world consequences. Existing manual methods for identifying fake news are slow and subjective. Text classification using machine learning offers a powerful solution, as demonstrated by Patil et al. (2020). However, there is still scope for development in this field in leveraging a more comprehensive set of features surrounding such data especially with regards to their sources and in utilising more advance classification techniques. This project aims to refine the effectiveness of text classification methods by incorporating such methods as proposed in studies in tackling the critical issue of fake news detection.

Building on existing work

- **Enhanced Feature Engineering:**
  - While Patil et al. (2020) focused on source-based features, this project might explore additional data sources like web search engine results. This aligns with the approach taken by Mazzeo et al. (2021) who analysed both textual content and URL information displayed by search engines for COVID-19 related queries.
  - Explore advanced techniques to extract richer features from news articles. This could involve more nuanced feature extraction, lexical analysis, identifying suspicious keywords or patterns, and incorporating URL information from search engine results (inspired by Mazzeo et al., 2021)
- **Addressing Class Imbalance:**
  - Real-world datasets often have an imbalance between real and fake news articles. Methods like oversampling or under-sampling can be employed to address this, ensuring the model does not become biased towards the majority class (real news).
  - Experiment with a more effective method as compared to under-sampling and reducing data size as in in [10], utilising strategies recommended in [11] .
- **Binary classification:**
  - While Patil et al. (2020) is based on multi-class classification of articles by their types that attribute to their authenticity, (as tagged during the curation of the dataset), this project could focus on targeting binary classification of articles as real or fake only, eliminating this additional factor to streamline analysis and possibly configure a more

generalisable and adaptable classification model suited for real-world applications of fake news detection.

3 Dataset : *Source Based Fake News*

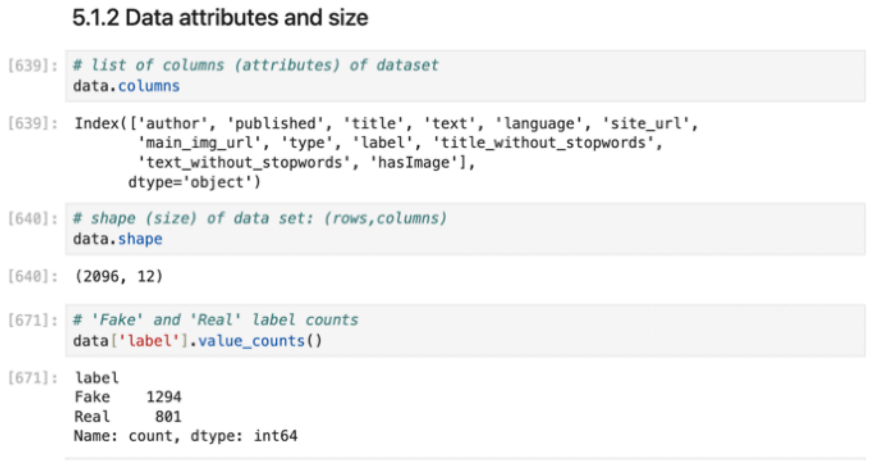
Source [13]: *https://www.kaggle.com/datasets/ruchi798/source-based-news-classification*

Dataset description

The dataset used for this project is the Source Based Fake News (SBFN) dataset which is derived from the curation and preprocessing of the Getting Real about Fake News (KaggleFN) dataset, containing attributes that include various aspects of the news such as the content and source of the news. It provides information about the article itself, the author of the article, the website that has published it and when it has been published, etc. [9]. This dataset also contains the columns 'text_without_stopwords' and 'title_without_stopwords' which have been generated by preprocessing and stopword removal from the text and title components of the KaggleFN dataset.

Size

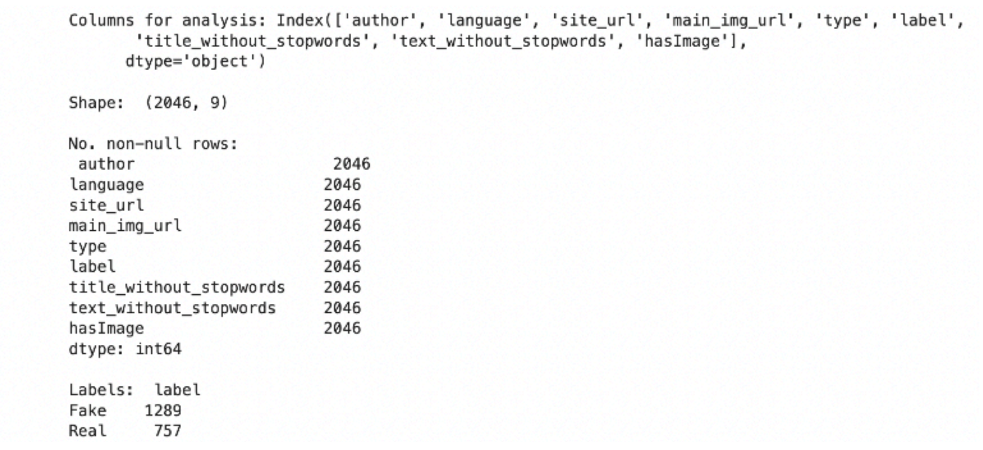
The downloaded SBFN dataset (used for this project) contains 2096 rows with 12 columns, consisting of 801 Real and 1294 Fake news (rows) as shown above in 5.1.2 [15].



5.1.2 Data attributes and size

```
[639]: # list of columns (attributes) of dataset
       data.columns

[639]: Index(['author', 'published', 'title', 'text', 'language', 'site_url',
              'main_img_url', 'type', 'label', 'title_without_stopwords',
              'text_without_stopwords', 'hasImage'],
             dtype='object')

[640]: # shape (size) of data set: (rows,columns)
       data.shape

[640]: (2096, 12)

[671]: # 'Fake' and 'Real' label counts
       data['label'].value_counts()

[671]: label
       Fake    1294
       Real     801
       Name: count, dtype: int64
```

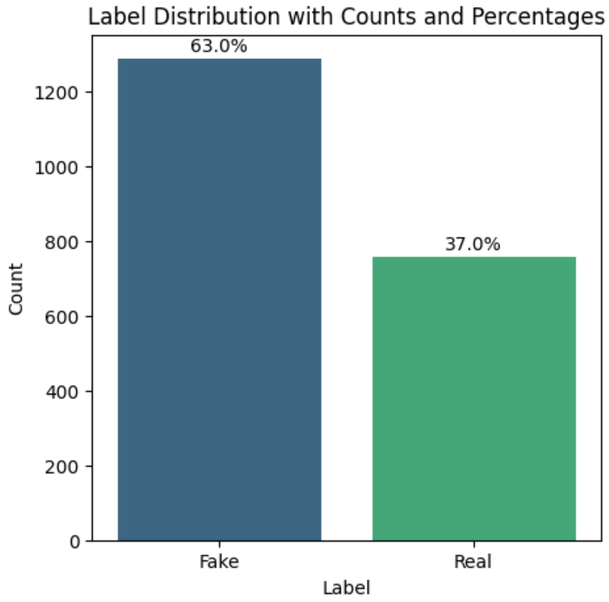
After replacing and removing null values where applicable, and dropping rows for streamlined analysis (5.2 [15]), the cleaned dataset contains **2046 rows with 9 columns**, consisting of **757 Real and 1289 Fake news** (rows) as shown below in 5.2.3 [15].



```
Columns for analysis: Index(['author', 'language', 'site_url', 'main_img_url', 'type', 'label',
       'title_without_stopwords', 'text_without_stopwords', 'hasImage'],
      dtype='object')

Shape:  (2046, 9)

No. non-null rows:
 author                      2046
language                    2046
site_url                    2046
main_img_url                2046
type                        2046
label                       2046
title_without_stopwords     2046
text_without_stopwords      2046
hasImage                    2046
dtype: int64

Labels:  label
Fake    1289
Real     757
```

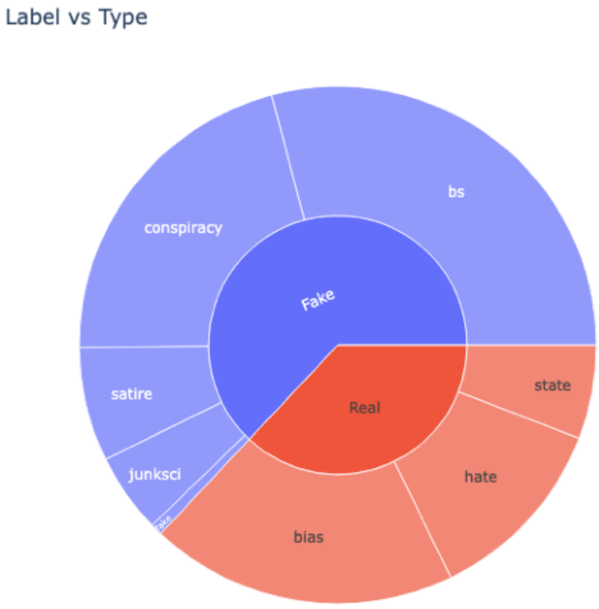
Distribution (after cleaning)

I. **Labels** (5.3.1 [15]):



The distribution of the Fake and Real labels in the dataset as shown above, is imbalanced with a significantly higher proportion of Fake labels (63.01%) compared to Real labels (36.99%).

II. **Types** (5.3.2 [15]):



These articles have also been classified classified into various types such as bs (bullshit), bias, conspiracy, hate, satire, state, junksci (junk science) and fake. The bs, conspiracy, satire, junksci and fake news articles are labelled as fake news and bias, hate and state news articles are labelled as real news. As shown above, there is a considerable amount of skew towards the label, ‘bs’ [10]. These are the labels on which classification in Patil et al. (2020) are carried out on.

III. **Languages** (5.2.2 ii [15])

The distribution of languages that the articles are in are as follows (after cleaning):

'english':	*1974*
'german':	*72*

Data types

Highlighted and bold rows signify the columns utilised for analysis and classification respectively

Column Name	Data type	Description
author	object (text)	author of post
published	object (text)	date on which the article was published
title	object (text)	title of post
text	object (text)	article text
language	object (text)	language of post
site_url	object (text)	url of source website
main_img_url	object (text)	url of main image of the post
type	object (text)	article type of 8 categories: bias, bs, hate, state, conspiracy, satire, junksci, fake
title_without_stopwords	object (text)	title without stopwords (generated attribute without stop words after preprocessing)
text_without_stopwords	object (text)	text without stopwords (generated attribute without stop words after preprocessing)
hasImage	float64	boolean value indicating whether the article has an image or not (generated attribute to check if the source contains images)
label	object (text)	Fake' or 'Real' label of article

Data Acquisition

The Getting Real about Fake News (KaggleFN) Dataset from which the Source Based Fake News (SBFN) dataset [13] is derived from, consists of text and metadata scrapped from 244 websites that were tagged as 'bullshit' by the BS Detector Chrome Extension. The skewness in the KaggleFN dataset has been removed in the SBFN dataset. The dataset contains news articles that were collected from 68 unique websites, among which 50 sites published only real news, 8 sites published only fake news, and 10 sites published both fake and real news [9].

4 Evaluation Methodology

Metrics

Accuracy

A balanced dataset mitigates the impact of class distribution on predictive accuracy, which represents a misleading indicator that reflects the underlying class distributions. As accuracy measures the number of correct predictions divided by the total number of predictions, the significant proportion of correct predictions of the majority class can skew this measure.

False positives and Negatives in Fake News Detection

Despite achieving a balanced dataset, relying solely on accuracy as an evaluation metric might still not be wholly appropriate in the context of fake news detection. Accuracy measures the proportion of true results (both true positives and true negatives) among the total number of cases examined, which does not fully address the different costs of false positives and false negatives. For example, a model might show high accuracy but still fail to detect a significant number of fake news articles, if it tends to misclassify these as real (a high number of false negatives). In fake news detection, the ability to detect as many fake articles as possible (high recall) is more crucial than ensuring every article flagged as fake is indeed fake (high precision). This emphasis is due to the high costs associated with failing to identify and stop the spread of fake news. Therefore other metrics will also be considered to provide a more holistic view of model effectiveness, as follows:

- **Recall** (Sensitivity, True Positive Rate):
  - Measures the proportion of actual positives (fake news articles) that are correctly identified by the model. It is calculated as the number of true positives divided by the sum of true positives and false negatives.
  - High recall ensures that the system minimises the number of fake articles that are misclassified as real, addressing the critical need for comprehensive detection.
- **F1 score** (Balance of precision and recall)
  - It combines both precision (the accuracy of positive predictions) and recall into a single metric by calculating the harmonic mean of precision and recall [*= 2×(precision×recall)/(precision+recall)*].
  - Ensures that the model does not overly favour recall over precision or vice versa, promoting a balanced approach to both catching fake news and maintaining credibility.
- **Area under the receiver operating characteristic curve** (ROC AUC):
  - Measures the ability of a model to discriminate between classes (real and fake news) at various threshold settings. It is calculated from the ROC curve that plots the true positive rate (recall) against the false positive rate at different classification thresholds.
  - The ROC AUC provides a comprehensive measure of a model's effectiveness across all possible classification thresholds. In fake news detection, a high AUC value indicates that the model accurately distinguishes between real and fake news, allowing flexibility in adjusting the threshold to meet different operational needs without sacrificing overall performance.

5 Preprocessing

Code [15]: https://drive.google.com/file/d/10dYxFgk5NdsiEX4_CpG4Opvg2wMz_BLz/view?usp=sharing

Following the subsections of the Jupyter Notebook,

5.2 Data Cleaning

- **Stopword removal:**
  - 4 rows of the generated column 'text_without_stopwords' that were null were replaced with their respective text after stop word removal. This removes common words that do not carry significant meaning (e.g., "the", "is", "and"), reducing noise to focus on meaningful words. This was carried out using the Natural Language Toolkit (nltk) library in Python by downloading the readily available English stop words library and *punkt* tokenizer from nltk.
- **Other null and discarded values:**
  - Resulting rows with NaN values and un-required columns (published, text and title without removal of stopwords) were dropped as they do not contribute to analysis. The values "ignore" in the language column were replaced with "english" upon inspection of rows and the 2 rows in French and 1 in Spanish were also dropped as they do not compose of much data and to simplify lemmatisation and pos tagging in later processing steps (that are language specific).

5.4 Text processing

- **Text cleaning:**
  - Text was converted to lowercase, unwanted characters and numbers were removed from the text, and URLs were split into tokens using specified delimiters after removal of protocols such as http://, www., and file extensions, etc (detailed in 5.4.1[15]).
  - This process was curated to retain some tokens (capped at 10 to avoid noise from too many tokens) that represent aspects of the source such as the presence of .com, .org, etc, and words found in the image URL, to incorporate suggestive URL-based features as inspired by Mazzeo et al. (2021) which states that:
    - URLs were found to contain several suggestive word tokens
    - The use of dots for adding an extension (i.e., .co) could suggest a fake website, whereas the proportion of http and https did not provide relevant information, as https secured protocol now is commonly used
- **Replacing missing values and Grouping:**
  - Columns with missing values such as "no author"and "title" were replaced with "anonymous" and "untitled" to provide singular terms that will not be affected by tokenisation, thus ensuring that the absence of information is clearly and consistently represented, to provide more accurate lexical analysis and feature representation
  - "no image url" was removed as the boolean variable 'hasImage' provides this information.
  - The text and title columns were combined as content, and the site url and image url columns were combined as source, for effective and distinguished processing and text representation.

- **Tokenisation and Lemmatization:**
  I. **Source**
    - Consisting of author, site and image URL data, this column was just tokenized to retain the integrity of the tokens that are proper nouns representing information on the source of the article.

  II. **Content**
    - Consisting of the title and text data that hold semantic value, this column was tokenized and lemmatized using language-specific models provided by spaCy [14].
      - Lemmatization processes tokens to reduce them to their base or dictionary form (lemma), normalising the words in text data.
      - This also involves looking at the Part of Speech (POS) of a word to determine its base form, as lemmatization in spaCy is context-aware, utilising both morphological analysis and POS tags to determine the most accurate lemma for a word
      - Extracting only the lemmas from the text data places focus on the essential parts of words which are useful for machine learning, and improving the understanding of the linguistic structure of the content.

5.5 Preparing data for Classification

- **Encoding categorical data and labels with label encoders:**
  - 'hasImage' and 'label' are converted from float values [1.0, 0.0] and categories ['Real', 'Fake'] respectively into into numerical forms suitable for machine learning models using LabelEncoder from Scikit-learn.
- **Feature extraction with TF-IDF Vectorization:**
  - Textual data (content and source) were converted into a TF-IDF matrices, which count the term frequencies across documents and adjusts them by the inverse document frequency. This highlights words that are more unique to a document, which can be more informative.
  - TF-IDF provides a way to quantify the importance of words (features) in a set of documents, thus enhancing the effectiveness of classification and clustering algorithms
  - The use of separate vectorizers allows a customised vectorization process for the content and source data, catering to their specific contributions in classification.

5.6 Balancing the Dataset with Data Augmentation:

- **Re-sampling**
  - **Class imbalance:** As mentioned earlier, the datasets presents a significant class imbalance, with a greater prevalence of fake news articles (63%) as compared to real ones (37%). This would likely make classifiers biased towards the majority class leading to classification of all the instances in the dataset as belonging to the majority class [11], thus undermining the model's ability to effectively identify fake news articles.
  - **Over-sampling with SMOTE:** To mitigate this, the data has been augmented in this project to increase samples of the 'Real' class using Synthetic Minority Over-sampling Technique (SMOTE). SMOTE works by creating synthetic samples rather than simply

duplicating examples from the minority class in the training dataset. It selects points that are close in the feature space, drawing a line between the points in the feature space and drawing a new sample at a point along that line [12]. This approach helps in achieving a more balanced dataset, while providing valuable information to the model, which facilitates a fairer ground for training classifiers, thus improving the classifier's ability to generalise.

- As stated in Mazzeo et al. (2021), with "the under-sampling technique, where instances from the majority class were removed, the score of the classifier models was very poor compared to the over-sampling technique", which has proven this method to be more successful. This is especially the case considering the small size of the dataset.

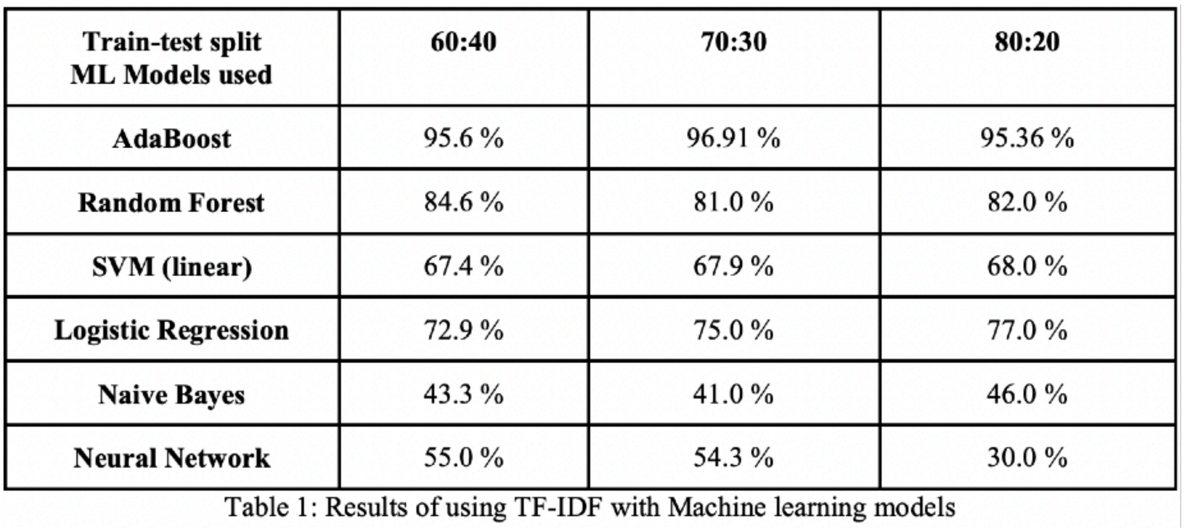
Text Representation (TF-IDF Vectorization)

TF-IDF Vectorization was considered as the most suitable text representation for this project considering the following:

- **Comparison with Word Embeddings**
  - The paper published on this dataset (Patil et al., 2020), compares the performance (accuracy) of various machine learning models on features represented by TF-IDF vectorization, as well as word embedding techniques such as Word2Vec and GloVe. The results clearly indicated superior performance with TF-IDF as compared to Word2Vec and GloVe.
  - While TF-IDF focuses on word importance within documents, Word2Vec and GloVe capture semantic relationships between words. Given the limited data and the use of URL features that do not hold semantic value, TF-IDF might be better suited for this task since it excels when specific keywords hold strong classification power, which also aligns with the success of AdaBoost in the study [10].
  - Word2Vec and GloVe, however, might have struggled due to the limited dataset and lack of correlation to semantic context. Their embeddings may have fallen short for accurate fake news detection without a rich amount of contextual data.
- **Inclusion of URL based features**
  - TF-IDF is able to accommodate the inclusion of URL based features by transforming both text and URL data into a shared numerical format as compared to word embedding techniques that focus on semantic relationships.
  - This integration allows machine learning models to effectively learn from both content-specific signals (words and their importance) and contextual signals (source credibility and characteristics).
  - The results that verify the effectiveness of incorporating URL features in ML classifiers from Mazzeo et al. (2021), are shown to be more evident in the classifiers that used TF-IDF.

6 Baseline performance

The following table in (Patil et al., 2020) presents the classification results (accuracy) obtained by various machine learning models trained on features extracted by TF-IDF vectorization. Since my project is based on the same dataset and follows similar data preprocessing and representation techniques, these results will be used as a baseline for comparison.



Train-test split ML Models used	60:40	70:30	80:20
AdaBoost	95.6 %	96.91 %	95.36 %
Random Forest	84.6 %	81.0 %	82.0 %
SVM (linear)	67.4 %	67.9 %	68.0 %
Logistic Regression	72.9 %	75.0 %	77.0 %
Naive Bayes	43.3 %	41.0 %	46.0 %
Neural Network	55.0 %	54.3 %	30.0 %

Table 1: Results of using TF-IDF with Machine learning models

Patil et al., 2020

Some factors to consider are:

- These results are based on multi-class classification of the articles as per their types (8 categories), while mine will be based on binary classification on the Real and Fake labels. This simplifies the learning task for my model, focusing on differentiating just two classes, especially with the equally distributed balanced dataset achieved through oversampling. Thus comparison of generalisability is also limited as my model is not trained with the real-world imbalanced distribution present in the benchmark data.
- In this study, the class imbalance among types was resolved by removing a considerable skew towards the class ‘bs’, which means the size of training data is reduced and the model had to learn to identify a wider variety of categories within a dominant class. This could lead to overfitting or accuracy being skewed towards the dominant class.
- Direct comparison is also in-effectuated to some extent because my model's performance on specific non-fake news categories (like conspiracy or satire) cannot be directly evaluated against the benchmark.

Therefore it is vital to consider additional metrics as mentioned in the evaluation methodology to provide a more nuanced understanding of the model's performance. However as a guideline, the above table will be used as the baseline for model comparison.

7 Classification approach

Features

I. Content (Title and Text of Articles):

These features are fundamental for text-based classification tasks, as they contain the core information and context of the articles, such as:

- **Linguistic Cues:** Fake news often employs specific language patterns to manipulate emotions or mislead readers. Certain word frequencies extracted from the text could highlight cues like unusual word choices or repetitive phrasing [1].
- **Factual Inconsistencies:** Factual errors or inconsistencies in a Fake news article can be revealed through text analysis by identifying contradictions or the absence of keywords expected in legitimate news articles [2].

II. Source (Author and Tokenized URLs of source website and image source):

Features of the source, including author and URL components, significantly aid fake news detection by providing context beyond the content itself. Here is how they contribute:

- **Author Credibility:** Established news outlets with journalistic integrity often have articles by identifiable authors with a history of responsible reporting. Conversely, fake news may lack author information or have authors with a history of spreading misinformation [3].
- **Website Credibility:** The name of the website itself is a strong indicator of authenticity. Reputable news organisations typically have well-known websites with clear and established branding. Fake news sites, on the other hand, may use names that mimic legitimate outlets or employ sensationalised or misleading domain names [4].
- **URL-based Features:**
  - **Lexical Cues:** The presence of suspicious keywords in the URL (e.g., "breakingnews", "shockingtruth") can signal potential deception [5, 11].
  - **Subdomain Analysis:** Subdomains associated with known fake news outlets raise red flags [5], or additional tokens representing domain suffixes (e.g., .com, .com.co) could indicate malicious websites [11].

III. HasImage:

Although the impact of this feature might be less significant compared to textual or source features, it was considered as the presence of an image might have a role to play in aspects such as the following:

- **Image Relevance:** Incongruence between the image content and the article's text can be a red flag. Fake news sometimes uses unrelated or emotionally charged images to grab attention but may not be truly representative of the article's content [10].
- **Image Source:** The origin of the image could be contained in its URL. If the image originates from a known source unrelated to the source website or the article's topic, it might raise suspicion about the article's legitimacy [7].

Labels

I. Real/Fake Label of Article:

These labels are essential for supervising the learning process, guiding the classifier to discern patterns associated with articles labeled Fake vs Real. This binary classification aligns with the project's goal to determine the authenticity of news articles based solely on their source and content.

Chosen Approach

AdaBoost with default base estimator (Decision Tree)

AdaBoost (Adaptive Boosting) is an ensemble learning technique that enhances the performance of classifiers by combining multiple weak learners (simple models) into a single strong learner. Decision trees are typically used as weak learners, referring to a model that performs slightly better than random guessing.

AdaBoost works by training these weak learners sequentially, each time focusing more on the training instances that were misclassified by the previous models. Each learner is assigned a weight based on its accuracy, and these weights are used to make the final decision.

The key features of AdaBoost include its ability to adapt to the errors of the previous learners and its effectiveness in reducing both bias and variance in the model. It is commonly used with decision trees as its default, and is mainly applied to binary classification tasks.

I. Rationale:

- AdaBoost achieved the best results (accuracy) as published in (Patil et al., 2020), with an accuracy of 96.91% with a 70:30 split ratio of train-test data. As this project is based on the same dataset utilised in the study and follows similar preprocessing and feature representation techniques, the superior performance of this model served as a key rationale in choosing this approach.
- Other factors considered were:
  - AdaBoost is more robust to noisy data and can learn complex relationships between features, which is useful in the case of fake news detection, especially in the context of leveraging the sparse, high-dimensional data created by TF-IDF.
  - As AdaBoost focuses on instances that are harder to classify, overall model is improved, especially on complex classification tasks like fake news detection.
  - The use of ensemble methods like AdaBoost can also handle imbalanced datasets better, which is the case in (Patil et al., 2020) but not in this project as data distribution is balanced by oversampling.
- Choosing the default base estimator Decision Trees is favoured by its interpretability and efficiency in managing both numerical and categorical data.
- The parameters for the classifier were derived from favourable results obtained in [7], also based on the same dataset.

II. Benefits and Drawbacks of Alternative Approaches:

1. **Random Forest:**
   - Benefits:
     - As an ensemble of decision trees, Random Forest generally performs well with sparse, high-dimensional data (like those from TF-IDF)
     - Reduces overfitting by averaging multiple decision trees and handles variance better than a single decision tree
   - Drawbacks:
     - More complex and less interpretable than a single decision tree
     - Can be computationally expensive
2. **Support Vector Machines (SVM) and Logistic Regression:**
   - Benefits:
     - Effective in high-dimensional spaces if carefully tuned and leverages dense representations of word embeddings better.
     - Works well with clear margin of separation
   - Drawbacks:
     - Linear nature of these models which may not capture complex boundaries required for tasks like fake news detection.
     - Less effective on noisier datasets with overlapping classes, requiring careful kernel choice.
3. **Neural Networks:**
   - Benefits:
     - Excellent at capturing nonlinear relationships and highly flexible in modeling complex patterns
   - Drawbacks:
     - Requires large datasets for sufficient training
     - Prone to overfitting
     - Less interpretable
4. **Naive Bayes:**
   - Benefits:
     - Simple and efficient to train, interpretable
   - Drawbacks:
     - May struggle with complex datasets
     - Assumes features are independent, which might not be true for fake news data

---

9 Evaluation

Evaluating the Fake News Classifier

- The binary classification model achieved an accuracy of 98.37%, successfully identifying fake news articles in 393 out of 399 cases (Recall: 0.981). This indicates that the classifier not only accurately identifies fake news but also maintains a high detection rate of true fake news articles with minimal misclassification.

- The F1 score of 0.976 further emphasises this effectiveness, indicating a good balance between precision and recall. Additionally, the AUC (Area Under the Curve) of 0.983 signifies a strong ability to differentiate real from fake news instances across various classification thresholds.
- While a direct comparison to the benchmark (96.91% accuracy for multi-class classification) is limited due to the different problem settings, our model's performance suggests high competitiveness, particularly considering the high AUC value.

10 Summary and conclusions

Overall evaluation and development from prior work

The enhanced performance of the classifier can be attributed to several key modifications and improvements over the baseline (Patil et al., 2020):

1. **Binary Classification:**

The focus on binary classification (real vs. fake), compared to multi-category classification is more specific allowing for targeted feature engineering and model tuning, which can lead to more accurate predictions in the context of fake news detection.

2. **Incorporation of URL Features:**

Including features from website and image URLs adds a layer of analysis that leverages source credibility and context, potentially capturing deceptive practices not evident from content alone.

3. **Separate TF-IDF Vectorizers:**

By utilising separate TF-IDF vectorizers for content and source data, the approach finely tunes the input features to the unique characteristics of textual and source information, enhancing the model's ability to distinguish between real and fake news based on both content and origin.

4. **Language-Specific Lemmatization:**

Using SpaCy for lemmatization, which is tailored to the specific language of the content, provides more accurate text preprocessing compared to NLTK used in (Patil et al., 2020). This can lead to better semantic understanding and, consequently, better model performance.

5. **Oversampling with SMOTE:**

Opposed to the approach of under-sampling in (Patil et al., 2020) which might lead to loss of valuable data, the use of SMOTE to balance the dataset helps in maintaining the integrity of minority class characteristics, thus enhancing model learning and generalisation.

Contribution to the problem area and transferability of solution to other areas

The contribution of this project to the problem area includes providing a method that not only improves accuracy in detecting fake news but also offers insights into the reliability of news sources based on URL analysis. This method is highly transferable to other areas where source credibility is crucial, such as in academic plagiarism detection, the verification of online user-generated content and other malicious content such as phishing as mentioned in (Mazzeo et al., 2021).

Replicability

The methods employed are replicable in various programming environments. The use of Python and libraries like SpaCy and SMOTE can be adapted or replaced in other programming languages that support similar machine learning and natural language processing libraries. For instance, equivalent libraries are available in R, such as tm for text mining and ROSE for data balancing. The approach is versatile enough to be applied using different algorithms or development tools, which encourages experimentation and further innovation in the field.

References

- [1] Role of Contextual Features in Fake News Detection: A Review [1] (https://ieeexplore.ieee.org/iel7/9051861/9071521/09071524.pdf)
- [2] Electronics | Free Full-Text | Sentiment Analysis for Fake News Detection (https://www.mdpi.com/journal/applsci/special_issues/Sentiment_Social_Media)
- [3] Detecting Fake News at its Source - MIT News http://fakenews.mit.edu/
- [4] Evaluating the effectiveness of publishers' features in fake news detection on social media https://link.springer.com/article/10.1007/s11042-022-12668-8
- [5] The Impact of Social Media on Trust and Democracy Pew Research Center
- [6] *IFCN Code Of principles*. (n.d.). https://www.ifcncodeofprinciples.poynter.org/
- [7] Ruchi. (2020, October 23). *How do you recognize Fake News?* Kaggle. https://www.kaggle.com/code/ruchi798/how-do-you-recognize-fake-news#Label-vs-Type
- [8] Allcott, Hunt, and Matthew Gentzkow. "Social Media and the Spread of Misinformation." American Economic Review, vol. 109, no. 6, 2019, pp. 2139-2182. https://www.aeaweb.org/issues/652
- [9] Shakya, N. and Poudyal, P. 2022. Detection of fake news using deep neural networks. *Kathmandu University Journal of Science Engineering and Technology*. 16, 2 (Dec. 2022).
- [10] Patil, Vikas, and S. B. Patil. "Source Based Fake News Classification Using Machine Learning." International Journal of Innovative Research in Science, Engineering and Technology (IJIRSET), vol. 7, no. 6, June 2020, pp. 121-124. (Note: This source might be difficult to locate due to the nature of the publishing platform)
- [11] Mazzeo, V., Rapisarda, A., & Giuffrida, G. (2021). Detection of fake news on COVID-19 on web search engines. *Frontiers in Physics*, *9*. https://doi.org/10.3389/fphy.2021.685730
- [12] *SMOTE for Imbalanced Classification with Pytho*. (2021, March 17). https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/
- [13] *Source based Fake News Classification*. (2020, August 29). Kaggle. https://www.kaggle.com/datasets/ruchi798/source-based-news-classification
- [14] *Lemmatizer · SPACY API Documentation*. (n.d.). Lemmatizer. https://spacy.io/api/lemmatizer
- [15] NLP_CW1, Jupyter Notebook on Colab: https://drive.google.com/file/d/10dYxFgk5NdsiEX4_CpG4Opvg2wMz_BLz/view?usp=sharing