# FAKE NEWS PROJECT

REPORT

**Anton Ankjær Barslund**[*]
qjr900@alumni.ku.dk

**Jeppe Bonde Bakkensen**
slr105@alumni.ku.dk

**Oskar Windfeld**
pqw926@alumni.ku.dk

**Tue Kauffmann**
bgr626@alumni.ku.dk

March 27, 2025

https://github.com/AntonBarslund/GDSGroup

[*]Use footnote for providing further information about author (webpage, alternative address)—*not* for acknowledging funding agencies.

## Part 1: Data Processing

When working with large text datasets, cleaning and preprocessing the data is essential to ensure consistency and accuracy for the best performing classifier. In this process, we utilized the `cleantext.clean` function to systematically normalize the text corpus. This involved removing new lines, tabs, and unnecessary whitespaces; converting all text to lowercase; performing Unicode character normalization; and transliterating non-standard characters to ASCII for maximum compatibility (e.g. replacing accented characters such as *áŷë* with *aye*). As our corpus is considered English language only, we determined that accent removal would not compromise semantic integrity.

We have used regex patterns to replace the most common ways of writing dates; accounting for both American and European syntax. Titles such as *Friday the 13th*, a movie, were being matched by the date patterns. This could have been avoided using a compiled list of movie titles, but in general we considered this to be a minor problem for our dataset.

After the date removal, we performed an additional cleanup by replacing all URLs, emails and numbers with the placeholders URL, EMAIL and NUM, respectively. Throughout this cleaning process, all punctuations was removed from the text. We then used the `NLTK` (Natural Language Toolkit) module to tokenize the text with the `word_tokenize` function. Further preprocessing involed removing stopwords using the `NLKT`'s stopword corpus and applying the `PorterStemmer` function to stem the words to their root forms (e.g. *funky, funks, funking* $\rightarrow$ *funki, funk, funk*).
This step ensures that the analysis focuses purely on content-bearing terms.

The previous processes proved to be time consuming, due to the large dataset. We therefore implemented some optimizations in our code, such as compiling our regex patterns, to lower the overall running time. Going through the process of replacing extraneous data (e.g. numbers and urls) from the content-row, we do `tokenize()`, `stopword removal()` and `stemming()` all using the `NLTK` module.

### Data exploration

We have chosen to use the `Pandas` library to represent the Fake News Corpus dataset (FNC). This library is widely recognized for its ability to efficiently handle large datasets and offers powerful features, such as dataframes, which streamline data management and processing. Note that the scraping of the dataset has timestamps: $2017 - 2018$; thereby an almost decade old dataset, which has to be taken into consideration, given how speech evolves over time.

Before processing the full dataset of approximately one million articles, we conducted an initial linguistic analysis of the parts of speech (POS) associated with the two labels: **fake** and **reliable** (see figure 2 and 3 in the Appendix). We had before our POS analysis a hypothesis that fake articles would exhibit a higher frequency of descriptive language—particularly *adjectives* and *adverbs*, but the distributions shows that both the fake and reliable datasets are structurally quite similar. The top seven POS categories appear in the same rank order for both subsets, suggesting that almost all articles use language in similar proportions.

However, a notable difference emerges in the relative emphasis on specific modifiers and verb forms: *adverbs* are more prevalent in fake articles, while *past-tense verbs* dominate in reliable ones. This may suggest that fake articles lean toward more *present-focused, descriptive, and emotive* language while the reliable articles may adopt a more *retrospective and reportorial* tone, rooted in past events.

Although our initial expectation of higher usage of adjectives in fake articles was not supported, the increased use of *adverbs* in fake news may still align with our broader hypothesis about emotional and rhetorical manipulation. These early findings indicate that more nuanced linguistic features such as verb tense and modifier patterns could be more indicative of article credibility than simple POS counts alone. However, a more thorough analysis is necessary before drawing a definitive conclusions.

We also found it interesting to analyze the relationship between total words and unique words in the dataset, aiming to understand lexical diversity. We computed the ratio of unique words to total words as a measure of how frequently new vocabulary is introduced. The dataset contains approximately 263 million words in total, of which about 1.73 million are unique. This mean that the unique-to-total word ratio is $0.0066$, meaning that only $0.66\%$ of the words in the corpus are unique. This low ratio reflects Heaps' Law, which states that the growth of unique words slows as the corpus expands.

## Part 2: Simple Logistic Regression Model

When determining which types of articles should be classified as reliable or unreliable, there are certain article classifications that are inherently clear. Articles labeled as **fake**, **conspiracy**, or **junksci** are undoubtedly fake, and those explicitly labeled **reliable** can be considered trustworthy.

The article types of **hate**, **bias**, **clickbait**, **rumor**, **satire** and **political** present greater ambiguity, as they inherently blur the lines between objective reporting and subjective narrative, revealing fundamental uncertainties about the content's true purpose and target audience. In the absence of truly objectivity, we have chosen to classify all of these article types as fake. Consequently, the only article type we have chosen as reliable is the one explicitly labeled **reliable**.

After examining the categories' descriptions, we have chosen to exclude the category labeled **unreliable** from our dataset. This category is described as: *"Sources that may be reliable, but whose contents require further verification"*[2]. This implies that the category is unknown and given the binary structure of our classification we would be forced to label them as either *fake* or *reliable*, a process that inevitable would result misclassifications without a thorough and potentially unfeasible manual investigation. Therefore excluding the **unreliable** category allows us to preserve the overall integrity of our dataset by avoiding introducing incorrect labels.

### Splitting the data

Using the recommended $80\%$ training, $10\%$ validation and $10\%$ test set split strategy, we aim to reserve the test set for a single final evaluation of model performance. We decided first to split the data after constructing bag-of-words (BoW) representation—on the full dataset. Performing these operations on the combined dataset simplifies implementation and avoids redundant processing. The splitting itself will later be done using the `train_test_split` function from `sklearn`.

While preparing to split the FNC dataset, we encountered a critical challenge related to data leakage. The dataset lacks publication dates and is not ordered chronologically, making it difficult to ensure a clean split. This structural issue increases the risk that similar or near-duplicate examples could end up in both the training and test sets, compromising the integrity of the evaluation.

When preparing to split the FNC dataset into three respective sets: **training**, **validation** and **test**, for our logistic regression model, we identified a critical issue related to data leakage, caused by the dataset's structure. Unlike an ideal dataset, the FNC does not include a publication date column, and the articles are not arranged in sequential order based on when they were published. This creates a significant problem for splitting the data properly.

If we were to split the dataset randomly, data leakage would be inevitable. News articles have a temporal nature; newer articles often contain information about past events, either through direct references or contextual knowledge. A random split would likely place some newer articles in the training set and older ones in the test set. This means the model could "learn" from future information during training, undermining its ability to generalize to truly unseen data.

[2]GitHub: `https://github.com/several27/FakeNewsCorpus`

To prevent this, the data must be split in a way that respects the chronological order of the articles. Ideally, the training set should include the oldest articles (first $80\%$ of the timeline), with the validation set containing the second oldest (next $10\%$), and the test set consisting of the most recent articles (final $10\%$). This ensures that the model is trained only on past data and evaluated on future data (post 2018), mimicking real-world conditions and avoiding leakage.

The absence of a publication date column makes a true chronological split impossible. To address this, we have made the assumption that the articles were added to the corpus in the order they were scraped, this order might roughly approximate their publication sequence. While not guaranteed, it is a reasonable starting point given the lack of explicit timestamps.

**Training the simple logistic regression**

When training the simple logistic regression classifier using the $10{,}000$ most frequent words from the **content** field as input features, and we assigned labels according to the scheme described above. The performance metrics from this experiment are presented below:

- F1 Score: $0.957$
- Accuracy: $0.936$
- Mean Squared Error: $0.064$

These results are unusually high, indicating a strong likelihood of data leakage. The root cause is likely tied to how the dataset was split, given that we assumed the articles were presented in true chronological order. If newer articles appear too early in the dataset while older ones end up in the test set, the model could inadvertently learn from future information, resulting in inflated performance scores.

We have chosen to exclude additional metadata features such as **url, author** and **title**, due to the possibility of introducing bias into the model. Most of the true news comes from well-known sources, and most of the fake news comes from unknown sources. This can lead the model to associate big news agencies with reliable and unknown sources with fake news. This would mean that the model would not detect fake news, but only identify well-known sources.

The goal of the `FakeNewsDetector` is to evaluate articles based on their content, not their contextual or source information. By excluding the metadata features, we ensure that the model is forced to rely solely on language and textual patterns. This focus on textual evidence, rather than external metadata features, prevents us from introducing unnecessary bias into our model.

**Implementing the BBC Dataset in the Training Set**

We applied the same preprocessing pipeline to the BBC News dataset scraped in Exercise 2, labeling all articles as reliable before incorporating them into the training set. After including approximately 600 BBC articles, the performance metrics of the logistic regression classifier were as follows:

- F1 Score: $0.957$
- Accuracy: $0.936$
- Mean Squared Error: $0.064$

Despite the addition, the model's performance remained virtually unchanged. This is expected, as the BBC articles make up less than $0.003\%$ of the reliable training dataset, exerting a negligible influence on the training distribution the

first measurable difference appears only at the fifth decimal place. In effect, their inclusion has no meaningful impact on the model's behavior.

For the remainder of this project, we have opted not to include the BBC News dataset, primarily due to concerns about temporal data leakage, as noted in the Splitting the Data section. The FNC corpus spans a specific timeframe (09/11/2017 to 01/01/2018), whereas the BBC articles date from 18/11/2024 to 08/03/2025, effectively post-dating the FNC by several years. Including these newer articles could expose the model to "future" information—such as references to contemporary political figures—that did not exist at the time of the FNC collection. In doing so, the model's performance might improve for the wrong reasons, leveraging knowledge unavailable in the original timeframe. By excluding these articles, we maintain a consistent temporal framework and ensure that our model's evaluation is based solely on content relevant to the period covered by the FNC.

## Part 3: Advanced Model

In our advanced model for our neural network model, we have chosen the `Keras` model through the `TensorFlow` library. We developed an advanced model to classify the "fakeness" of a given text using a fully connected neural network implemented in `TensorFlow`. The model takes a $10,000$-dimensional input vector and produces a single binary output, "1" if fake and "0" if reliable. This classification is achieved using the sigmoid activation function, which maps a continuous value to 0's or 1's. We opted using binary cross-entropy as the loss function due to its efficiency in classification problems. We considered the mean squared error (MSE), but the output is discrete, and therefore cross-entropy proved more suitable.

For optimization, we used the Adam (Adaptive Moment Estimation) optimizer, as it is suitable for many applications, especially given our limited knowledge of other optimizers. To evaluate the model's performance, we monitored precision, recall, and F1-score. We did this to make sure the model did not just predict the majority class, which would happen if we just looked at accuracy. We chose to combat the unbalanced data, by giving the model some class weights, that were calculated from the count of the different classes.
Our initial approach involved a trial-and-error process to determine the number of layers and neurons. We started with a model featuring four hidden layers containing: $1000$, $500$, $100$ and $50$ neurons, respectively, each with a ReLU activation function, except for the last layer with a sigmoid. To combat overfitting, we incorporated dropout layers with values of $0.4$ after the first hidden layer and $0.2$ after the second and third. Using the bag-of-words representation from earlier in the assignment, the model achieved an F1 score of $0.893$ on its first run. We noticed an issue with the "unreliable" category, which was initially treated as *fake*. Since it is a category of unknown reliablity, we decided to exclude this category as uninformative data, that the model should not learn from, rather than forcing it into the fake or reliable class. After several iterations and adjustments to the model's configuration, we discovered a data leakage problem caused by improper shuffling in the splitting function. Once corrected, the model's F1-score dropped to $0.79$.

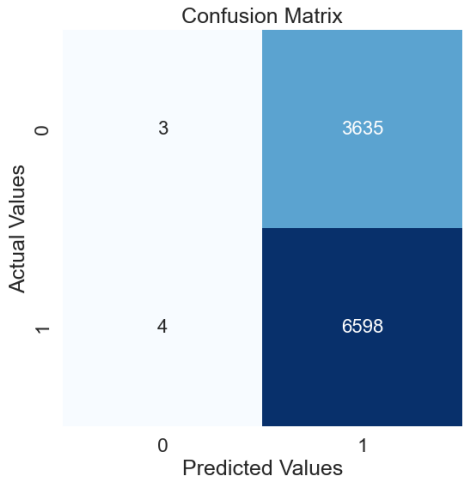
To improve performance, we shifted our focus to optimizing the model and experimenting with different structures. We then implemented a Term Frequency-Inverse Document Frequency (TF-IDF) representation using a vectorizer from `scikit-learn`, instead of the BoW approach. This adjustment significantly boosted the model's performance. Our final model retained the $4$ hidden layers with $1000$, $500$, $100$ and $25$ neurons, respectively, and dropout rates of $0.4$ after the first layer and $0.2$ after the second and third. We chose this because we did not find any drastic improvements in the score when changing the weights. This configuration gave an F1-score of $0.97$, which we found satisfactory in. We chose to use the built-in balancing function, the class weights parameter in the `NN.fit` function.

## Part 4: Evaluation

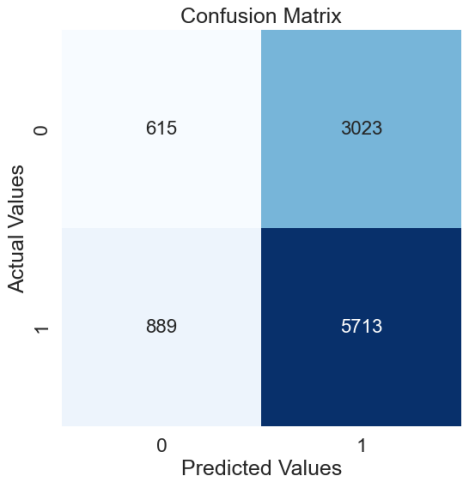
As seen in the different parts, we have achieved an F1 score of $0.94$. We think this is the best metric to evaluate the models on, and thus we have chosen models that fit this. The reason we have focused on F1 is that it combines precision and recall. Precision is good when we want to measure the proportion of true positives (TP); in other words, it focuses on the quality of the positive predictions. Recall, or sensitivity, focuses on the model's ability to capture all positive instances. This can be made into a confusion matrix that, in combination with the F1 score, tells a great deal about the model's performance.

When we created the models, we tested them and found that the F1 scores for the two models were closer to each other than we thought. To examine further how the models differ, we imported the LIAR dataset. This dataset is made up of many statements that are marked with labels. The labels are a range of different levels, from *super fake* to *true*; we forced them to be binary. We chose to believe that both statements labeled *true* and *mostly-true* were *reliable*, and the rest of the labels were *fake*.

To evaluate the models on this dataset, we took all the preprocessing that we made in Part $1$ and processed the new data through that pipeline. Then we used the two best-performing models that we had gotten in Parts $2$ and $3$. When running both of the models, we got an F1 score of about $0.78$ for both models, which surprised us, since we thought the advanced model would perform better in a cross-domain assignment. We made the confusion matrices, which tell a more complete story about how the models operate in much different ways.




Confusion matrix for logistic regression model
6598 TP and 3635 FP; 3 TN and 4 FN




Confusion matrix for neural network model
5713 TP and 3023 FP; 615 TN and 889 FN

As we can see in the figures above, the models behave in almost diametrically opposite ways. The logistic model almost always guesses that the text is fake and only in two instances guesses—falsely—that the statement is true. This might be because the statements are much shorter, and the model has "learned" that a shorter article is more often fake. The advanced model, somehow, has learned the opposite, or it has at least guessed much more "reliable" than "fake." This is an interesting fact because we would much rather have a model that takes some kind of informed guess than a model that just always guesses the same.

It makes sense that the models perform worse when using them on a dataset that is differently structured. It is likely the length that confuses the models and makes them perform worse. We are surprised that they perform this well, since we can see that the logistic model only guesses "fake" all the time.

## Part 5: Conclusions

The results of this report highlight the importance of data structure in processing and classification. The problem in this dataset is the lack of a publication timestamp, which has implications when we want to preserve the temporal structure that news articles have. This is evident when we look at the performance of the two models and how they compare to eachother. This makes it impossible to sort the data appropriately and maintain its structure, forcing us to create a model that learns from leaked data.

Our cross-domain performance showed us that a model that excels in one area does not always perform well when applied to another domain. We hypothesize that this is largely due to the lack of words in the statements in the new domain, since the models were trained on texts much longer than the data in the LIAR dataset. This claim is supported by the fact that the logistic model almost always predicted "fake". We suspect the model has learned to classify brevity as a fake news trait. This idea might hold some merit, since an article that does not support its claims is often shorter. We think the neural network might be better at these kinds of cross-domain tasks because it seems able to learn more complex behaviors. This behavior stems from the model's more intricate structure, and we can observe in the confusion matrix that this is true. To build a better model, we would need to find a way to structure the input data with its temporal structure in mind.

## Appendix

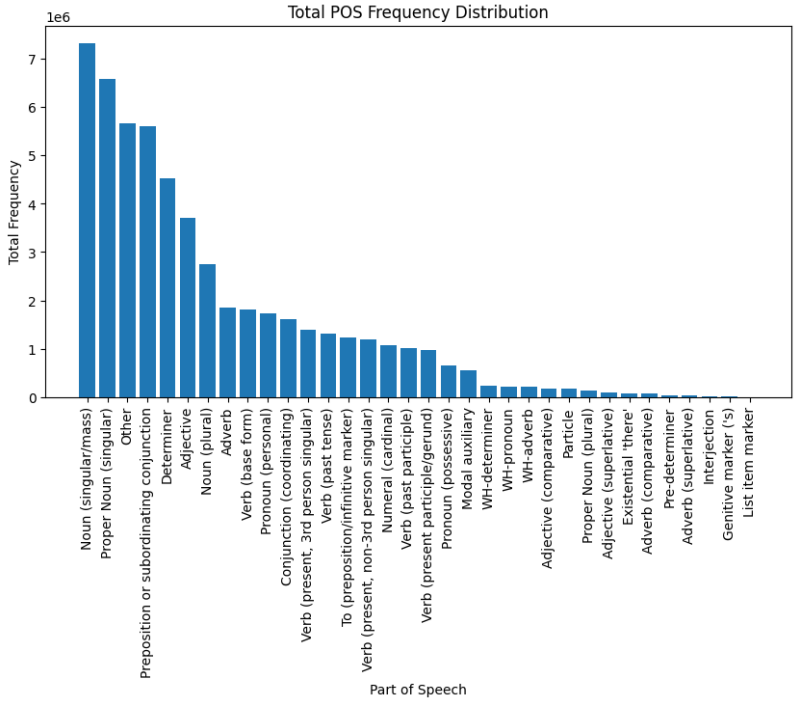



Figure 2: Fake dataset parts of speech, where adverbs and base verbs are used more frequently compared to reliable POS

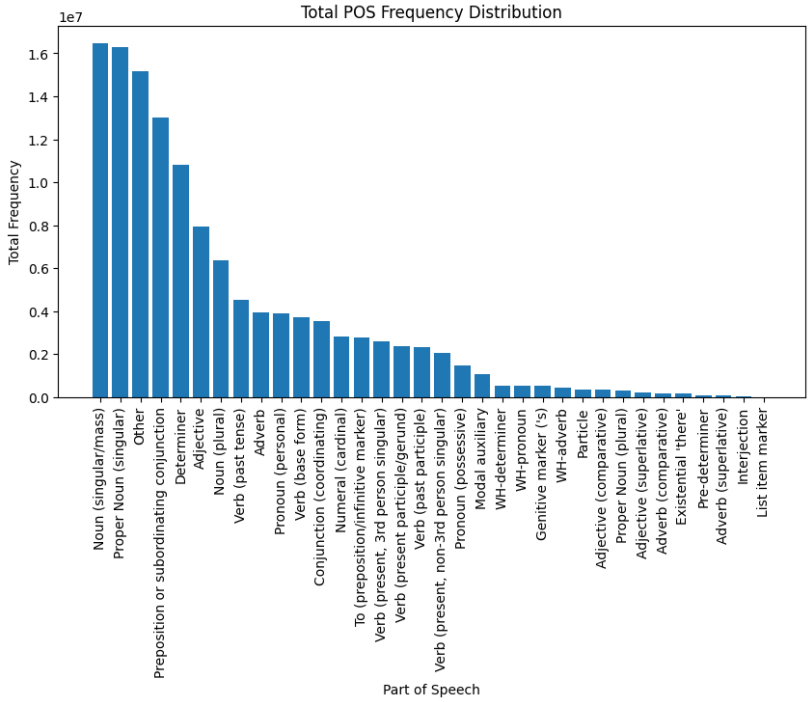



Figure 3: Reliable dataset parts of speech, where past tense verbs are more frequently used compare to fake POS