29-05-2025

NLP – Fake News Detection on Twitter

Authors: Thea Brock Reinhold Jensen (176854), Hans-Henrik Hansen (153689)

Group Name: FRI-161558-34

Course: Natural Language Processing and Text Analytics (CDSCO1002E)

Number of pages: 14

Number of characters (incl. spaces): 34.050

Abstract

The rise of social media has transformed global communication, enabling real-time information sharing while also facilitating the rapid spread of misinformation and fake news. This study addresses the critical need for effective and scalable fake news detection on platforms like Twitter. The aim is to develop and compare various machine learning and deep learning models to classify tweets as either representing fake or real news. A combination of traditional classifiers, Naive Bayes, Logistic Regression, and Random Forest, as well as a deep learning-based Bidirectional LSTM (biLSTM) model is applied. These models are trained and evaluated using two text representation techniques: TF-IDF and pre-trained Word2Vec embeddings. Emphasis is placed on optimizing the F1-score to address a slight class imbalance and to ensure balanced performance across categories.

Using the large-scale, fact-checked TruthSeeker2023 dataset comprising over 130,000 tweets from 2009 to 2022, this study also incorporates exploratory data analysis to uncover linguistic patterns in fake versus real content.

The EDA revealed subtle differences in part-of-speech usage, with fake news tweets showing slightly more assertive language through higher verb usage, and real news tweets containing more nouns, reflecting a more formal tone. Named Entity Recognition showed that fake news tweets more frequently mentioned individuals and controversial topics, while real tweets focused more on policy and institutional references. Topic modeling further highlighted that fake news tended to cluster around misinformation themes such as election fraud and vaccine denial, whereas real news centered on policy issues like taxation, healthcare, and minimum wage. The biLSTM model achieves the highest performance (F1-score 93%), while Logistic Regression offers comparable results with lower computational cost.

The results of the analysis are then considered in relation to the importance of the continuously changing nature of language on social media, and the ethical implications of deploying deep-learning models in contexts that might influence free speech in the public debate.

Keywords

NLP, Binary Classification, Fake-News, Twitter Moderation, Deep-Learning, Machine Learning.

Table of contents

1. Introduction

Social media has fundamentally reshaped how information is created, shared, and consumed, and it has allowed for real-time global communication between people from all over the world. However, this shift has also facilitated the rapid spread of misinformation and fake news, with significant societal consequences. The traditional news media no longer has monopoly on the news, and consequently, the truth is no longer a black or white fact but instead something that is increasingly up to debate due to the large amount of misinformation being distributed on online platforms.

Effectively identifying and classifying fake news is crucial not only for academic and technological development but also for practical applications in content moderation. The ability to automatically flag potentially misleading or harmful content can assist platforms in swiftly responding to problematic posts, enabling early intervention and reducing the risk of public misinformation. This is especially critical in high-stakes domains such as public health, politics, and crisis communication, where false narratives can have potentially devastating consequences.

Detecting fake news on platforms like Twitter poses distinct challenges due to the posts being short, the dynamic nature of twitter content in general, and the nuanced and often informal use of language.

To address this, the present study leverages the TruthSeeker2023 dataset, a large-scale, publicly available collection of over 130,000 tweets spanning from 2009 to 2022. The dataset draws from 1,400 news statements where half are labeled as fake and half as real and fact-checked by Politifact, a credible fact-checking organization. Tweets were collected using topic-relevant keywords via an API and subsequently labelled, through crowd-sourced consensus using Amazon Mechanical Turk, based on the level of compatibility between the given tweet and the news statement the tweet was being evaluated against (Dadkhah et al., 2024).

1.2. Formulation of problems

R1: Are there any systematic differences between fake news and true news that can be illustrated through an EDA process and what are the most important features when classifying real and fake news?

R2: How accurately can traditional machine learning models and deep learning architectures classify fake and real news, and can these be generalized and scaled to have a use case in real life applications?

2. Related work

Khalil and Azzeh (2024) conducted an extensive comparative analysis using the TruthSeeker dataset to evaluate the effectiveness of various text representation techniques such as TF-IDF, Word2Vec, BERT, and RoBERTa combined with classic machine learning algorithms including K-Nearest Neighbors (KNN), Support Vector Classifiers (SVC), and Random Forest. Their preprocessing pipeline involved standard NLP steps such as stop word removal, tokenization, and lowercasing. They concluded that while TF-IDF and Word2Vec perform well with traditional classifiers, transformer-based models such as BERT and RoBERTa offer the best performance in fake news detection.

Dadkhah et al. (2024) also utilized the TruthSeeker dataset. They employed traditional classifiers like Logistic Regression, Random Forest, and SVM, achieving moderate accuracy scores (60–70%). Their evaluation extended to transformer-based models, where BERTweet achieved the highest accuracy (~96%) in binary classification. However, performance dropped significantly in more granular (multi-label) classification tasks, revealing the limitations of current models in handling nuanced fake news content.

Iqubal et al. (2024) expanded this comparative framework by testing both traditional and deep learning models, including Naive Bayes, Logistic Regression, Random Forest, and a multilayer perceptron (MLP) on the TruthSeeker dataset. Their findings showed that Logistic Regression outperformed more complex models in both accuracy and computational efficiency, with an F1-score of 0.97. The MLP, while nearly as accurate, was more resource intensive. Their study highlighted that simple models can still yield strong results, particularly when coupled with effective vectorization like TF-IDF.

Harika and Hussain (2024) explored hybrid deep learning models using pre-trained Word2Vec embeddings and a combined BiLSTM-LSTM architecture highlighting the potential of advanced sequential architectures in handling the temporal and contextual dependencies in fake news narratives.

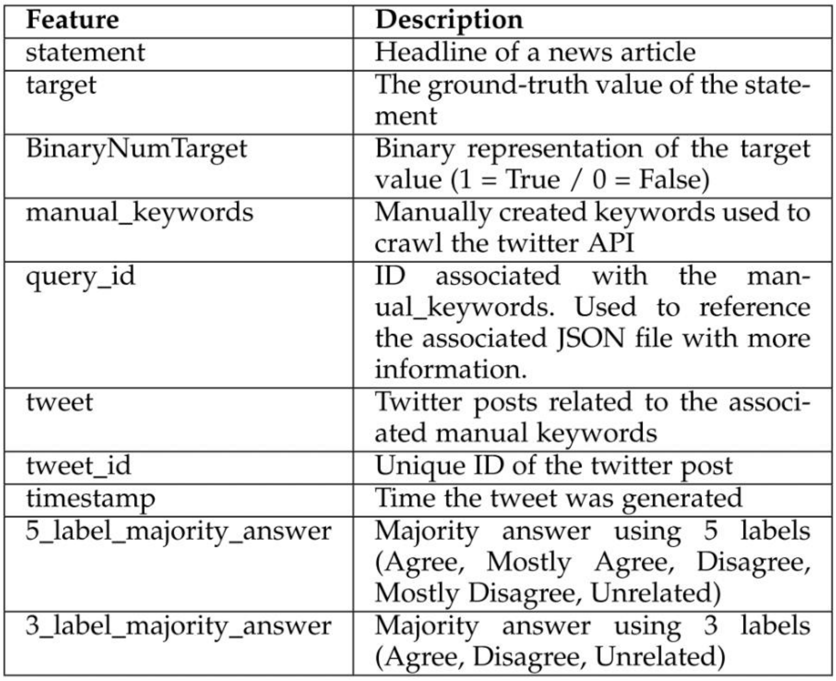
Kumar et al. (2025) introduced a novel Hashtag Context-aware Fake News Detection (HCFND) framework to overcome the limitations of sparse social media content and outdated external references. HCFND enriches the verification process by retrieving related posts from hashtags and named entities mentioned in a tweet. This approach mimics human behavior in validating information through contextual discourse and was shown to outperform existing best practice models on multiple Twitter benchmark datasets.

Collectively, these studies showcase the value of integrating external context, advanced text representation techniques, and hybrid deep learning models in fake news detection. While transformer-based models generally outperform classical approaches, their effectiveness can vary based on classification granularity and computational demands. Simpler models like Logistic Regression remain competitive, especially in well-balanced binary tasks, while architecture like BiLSTM and HCFND hold promise for more complex or dynamic detection settings.

3. Methods

3.1 Data description

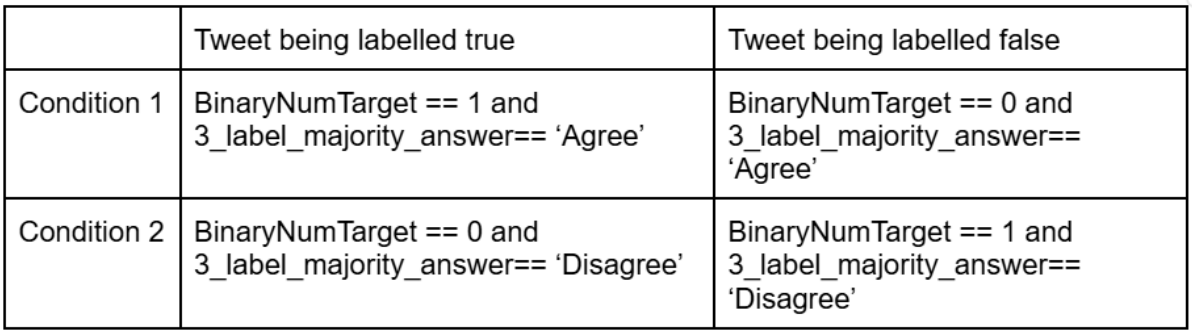
The dataset TruthSeeker2023 consists of more than 130,000 tweets from 2009 to 2022. The tweets were collected by gathering 1,400 news statements from 700 news articles containing fake news and 700 news articles containing real news (Dadkhah et al., 2024, p. 3379). The statements were labelled by Politifact which is a fact checking organization run by editors and reporters from the Tampa Bay Times (University of California Berkeley Library, n.d.). The statements were manually labeled with two to five keywords to describe their respective topics. These keywords were then used as queries via an API to collect tweets relevant to each news article (Dadkhah et al., 2024, pp. 3379–3380). The tweets were manually annotated via Amazon Mechanical Turk based on their alignment with the corresponding news statement and included only if at least two out of three annotators agreed (Dadkhah et al., 2024, p. 3381). This results in the dataset having the following columns:



Feature	**Description**
statement	Headline of a news article
target	The ground-truth value of the statement
BinaryNumTarget	Binary representation of the target value (1 = True / 0 = False)
manual_keywords	Manually created keywords used to crawl the twitter API
query_id	ID associated with the manual_keywords. Used to reference the associated JSON file with more information.
tweet	Twitter posts related to the associated manual keywords
tweet_id	Unique ID of the twitter post
timestamp	Time the tweet was generated
5_label_majority_answer	Majority answer using 5 labels (Agree, Mostly Agree, Disagree, Mostly Disagree, Unrelated)
3_label_majority_answer	Majority answer using 3 labels (Agree, Disagree, Unrelated)

(Dadkhah et al., 2024, p. 3382)

From the dataset, it is not labelled in a column whether a tweet contains fake news or not. To have the tweets being labelled, we created a column ‘target_tweet’ where 0 means that the tweet contains fake news, and 1 means that the tweet doesn’t contain fake news. The values in the column were given based on the following conditions:



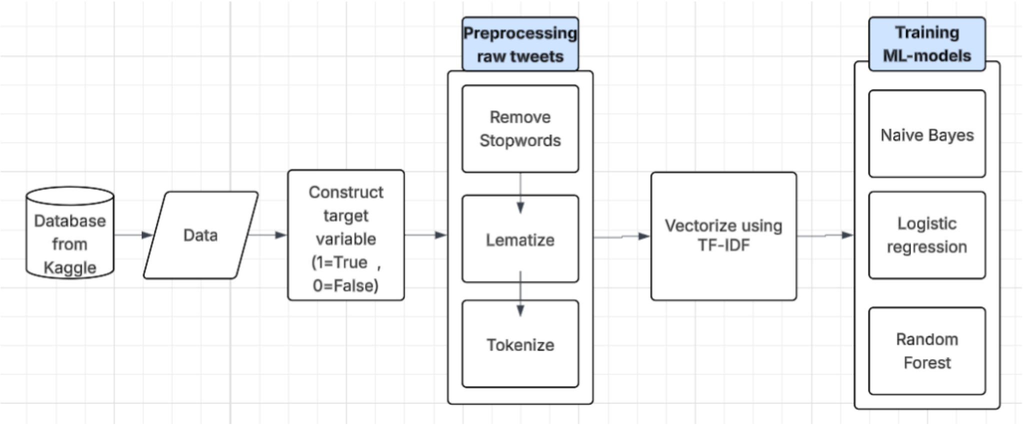
	Tweet being labelled true	Tweet being labelled false
Condition 1	BinaryNumTarget == 1 and 3_label_majority_answer== 'Agree'	BinaryNumTarget == 0 and 3_label_majority_answer== 'Agree'
Condition 2	BinaryNumTarget == 0 and 3_label_majority_answer== 'Disagree'	BinaryNumTarget == 1 and 3_label_majority_answer== 'Disagree'

3.2. Data preprocessing

Two new columns have been constructed, one that contains a tokenized version of the tweets with # and @ and one that contains a tokenized version of the tweets where # and @ have been removed. The tweets were converted into lowercase and lemmatized to get the root form of the tokens/words. Both columns have been preprocessed using the spacy library since this is better at capturing contexts than NLTK, which is important for lemmatization and the POS analysis conducted during the EDA.

Preprocessing for EDA
The column where @ and # has been removed were created in order to see the differences in the EDA between tweets that have been completely cleaned and tweets that still keep the @ and # to get a better understanding of the role that these special characters play in the tweets.

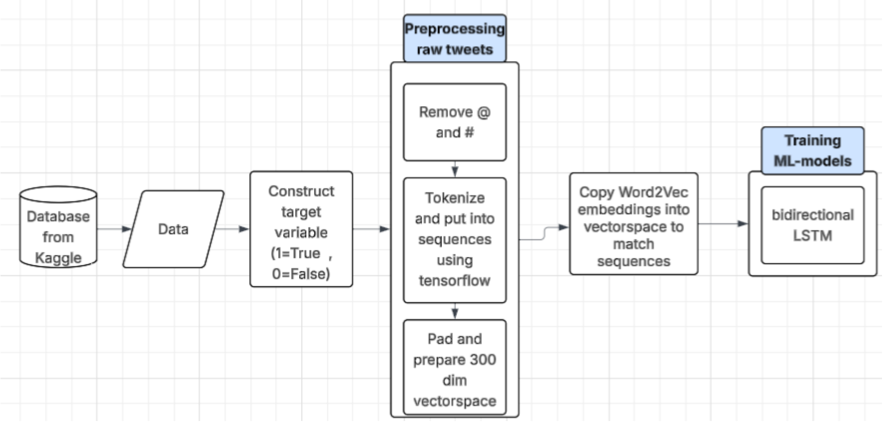
Preprocessing for TF-IDF
For the TF-IDF, the column where the @ and # has been kept is used since it was found that @ and # are being used in tweets to show usernames, write numbers and make subject tags. The conditions for keeping a @ or # were that the character had to be connected to a string, by assuming that those tokens are usernames or subject tags. If a @ or # was followed by a space, the character was not kept.



Preprocessing for Word2Vec
We start out by removing @ and # from the tweet column that has not been preprocessed and save it in a column called 'cleaned_tweet'. We do this since we are using pre-trained word embeddings that we want our data to match, and tokens containing @ and # would not

match the pre-trained word embeddings and therefore be overlooked by the model and only contribute with noise.

Next, the tweets are tokenized using the Keras tokenizer object, and takes the tweets from 'cleaned_tweet' in the format of a list of strings. The sequencer then maps each token to an integer value, consisting of an ID for the tokenized word in the vocabulary of Word2Vec, whereafter the IDs are arranged in a list for each tweet. Afterwards, length of the longest sequence is being saved as a variable for length of padding, and padding is then applied to ensure that each tweet has the same sequence length by adding zeros after each sequence up until the maximum length for the sequences.
A matrix with 300 dimensions is created to match the dimensionality of the pre-trained Word2Vec embeddings, where each word is represented by a 300-dimensional vector. Then, for each word (token) in our dataset's vocabulary, we check if it exists in the Word2Vec model. If it does, we copy the corresponding pre-trained vector into the matrix. This ensures that every word in our data has a matching 300-dimensional embedding, based on the representations learned from the Word2Vec model.


Database from Kaggle
Data
Construct target variable (1=True , 0=False)
Preprocessing raw tweets
Remove @ and #
Tokenize and put into sequences using tensorflow
Pad and prepare 300 dim vectorspace
Copy Word2Vec embeddings into vectorspace to match sequences
Training ML-models
bidirectional LSTM

3.3. Metrics used for evaluation

The primary evaluation metric used in this analysis is the F1-score, which combines precision and recall into a single measure. While our dataset is fairly balanced, there is still a slight imbalance between the target classes. For that reason, relying solely on accuracy could be misleading, as it may obscure model performance differences between the classes.

The F1-score is a particularly useful evaluation metric because it accounts for both false positives and false negatives. The F1-score, as the harmonic mean of precision and recall, takes into account both these metrics and minimizes their trade off, making it a great metric even when the class distribution is not perfectly equal.

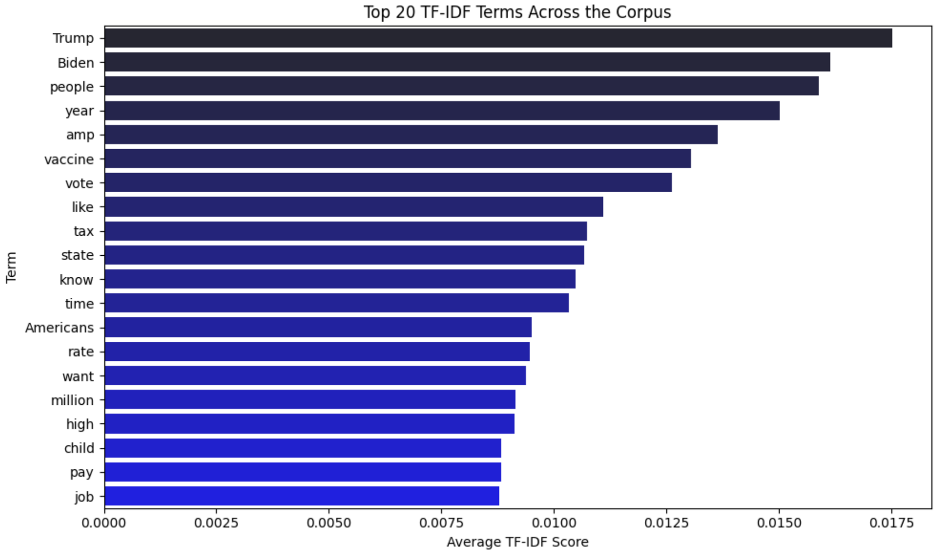
While accuracy is still being reported as a standard benchmark, the F1-score is prioritized due to its ability to capture performance across both classes more evenly. This is particularly important in the context of the analysis in this paper, since it must be ensured that the model performs well for both the fake- and true news classes.

The macro-average is used, which calculates the F1-score for each class independently and then averages them, giving equal weight to each class regardless of their frequency since the aim of the evaluation is to treat both target classes equally, the macro-average is considered to be the more appropriate choice than the micro-average for evaluation.

3.4. Word Embeddings

TF-IDF

The primary method that has been used as foundation for training the traditional machine learning models is Term Frequency Inverse-Document Frequency (TF-IDF) which is a statistic embedding that reflects the importance of a given term word (feature) in a document relative to the corpus (collection of documents/tweets). It consists of two elements, the first being the term frequency which measures how often a word appears in a document, and the second being the inverse document frequency which measures the importance of a word. The score is then calculated by multiplying the TF and IDF values, and the higher the TF-IDF score is the more important a word will be for that document/tweet, considering the context of the entire corpus/collection of tweets (GeeksforGeeks, 2024). Below is a visual showing the most relevant words based on their average TF-IDF scores across all the documents in the corpus:



Word2Vec

Word2Vec is a neural network approach to embedding the language data into vectors, since the embedding comes from training a recurrent neural network. This paper utilizes the google news 300 model which contains 300 dimensional vectors for 3 million words and phrases, and this was trained using Google's skip-gram architecture. The skip-gram architecture attempts to predict the surrounding context words (nearest neighbors) in a given target word, and the weights in the neural network that are being learned are actually constituting the vector that represents the given word.

Because the model is based on skip-train it allows the vector embeddings to capture the meaning of the words and capture contextual semantic meanings. However, the model is limited in the sense that it still struggles to capture complex language contexts such as sarcasm. The context of Word2Vec is then happening in the vector space where words appearing in similar contexts are placed near each other, enabling the model to reflect general word meaning and even analogical relationships.

3.5. Models

3.5.1. Naive Bayes

Naive Bayes is a probabilistic classifier that, as the name suggests, makes a naive assumption of complete conditional independence between all features and their probability of each class. This means that the presence of one word in a document is assumed to be independent of the presence of any other word, given the class (Jurafsky & Martin, 2023a, p. 4). In this case, it means that if the words 'hoax' and 'media' both appear in a tweet, the effect that 'hoax' has on the probability of the class is independent of the presence of "media".

In this project, Multinomial Naive Bayes is applied which is a version of Naive Bayes that handles discrete count data and is therefore suitable for text classification where a feature is represented by its frequency (Jurafsky & Martin, 2023a, p. 20). Due to the naive assumption of independence between features, the method uses the frequency of words in the vocabulary of the data, which is why Bag-of-Words (BoW) is an appropriate text representation (Jurafsky & Martin, 2023a, pp. 3-4). In this project, Naive Bayes is, however, used in combination with TF-IDF because it is used as an embedding-pipeline for the use of Logistic Regression and Random Forest. As described earlier, TF-IDF does look at frequency of words, but it does so in the relation of frequency within each document, hence not creating a text representation of feature independence. Another reason for applying TF-IDF in combination with Naive Bayes is the reduction of noise in terms of frequently used words that might appear in both classes.

3.5.2. Logistic Regression

Logistic Regression is also a probabilistic classifier but is a model that is being trained to optimize the parameters by using a loss-function that reduces the errors. The loss function contains a gradient descent algorithm that minimizes loss (Jurafsky & Martin, 2023b, p. 2).

Logistic Regression captures correlations between features in contrast to Naive Bayes and therefore assigns more accurate probabilities to features (Jurafsky & Martin, 2023b, p. 8). Logistic Regression is applied to get a better result than Naive Bayes due to the training of the model by adjusting weights for reducing loss. It is also applied to enhance results by using a fast and cost effective model compared to bigger models containing neural networks.

3.5.3. Random Forest

This model is essentially made up of multiple smaller models (decision trees that each use their own subset of the data and a different number of features) where the predictions of each tree are combined, taking the most popular predictions amongst the trees.

The Random Forest model has 300 trees in the forest and there is no max_dept. It uses sampling with replacement (bootstrapping) and the splits are performed based on the gini impurity criterion. The minimum leafs (minimum number of samples required to be in a leaf node) are set to 20 which is relatively cautious but deemed necessary to prevent overfitting and help the regularization of the model.

It is first used to extract the most important features for classifying real and fake news, and here the 100 most important features are extracted to get an overview of the most important features. Based on this dimensionality reduction the model is then retrained on the 100 most important features, as well as trained on the full dataset to see if a dimensionality reduced Random Forest can outperform a Random Forest that has been trained on the full dataset. A balanced weight parameter is also used since there is a mild unbalance in the proportion of fake news and true news, and this prevents the model from overemphasizing the importance of the most prominent class.

3.5.4. biLSTM based on Word2Vec

A Bidirectional LSTM (BiLSTM) is a type of recurrent neural network (RNN) that extends the standard Long Short-Term Memory (LSTM) architecture. Unlike traditional RNNs, which process sequences in one direction, BiLSTM processes input sequences in both forward and backward directions, which makes the model able to process the data in relation to past and future context. This is particularly useful for short, context-sensitive text such as tweets.

The model uses pre-trained Word2Vec embeddings, with the embedding layer's input dimension set to the vocabulary size plus one for padding and the output matching Word2Vec's dimensionality of 300. To retain the semantic knowledge from the large Word2Vec corpus, the embeddings are frozen by setting trainable=False, reducing the risk of overfitting.

The BiLSTM layer has 64 hidden units and applies a 30% dropout rate for regularization. The output layer consists of a single neuron with a sigmoid activation for binary classification (fake vs. non-fake). The dataset is split using stratified sampling to maintain class balance, and Early Stopping stops training after 5 epochs without improvement in validation loss, restoring the best weights.

4. Exploratory Data Analysis (EDA)

4.1. Named Entity Recognition (NER)

For the NER, the data was split into two subsets: one for tweets containing fake news and one for non-fake news tweets. When performing the NER on the preprocessed text containing usernames, usernames (words starting with @) were added to a category called Usernames. See appendix 8 for visuals.

Across all four plots, "PERSON", "ORG", and "GPE" (Geopolitical Entity) are the most frequent NER tags. Fake news tweets with hashtags and usernames show a higher overall entity count, particularly in the PERSON category, suggesting that fake news may more frequently mention individuals to potentially spread targeted misinformation. We also see that mentions of Nationalities or religious or political groups (NORP) is higher in non-fake news than in fake news.

Additionally, usernames appear prominently in both fake and non-fake tweets when punctuation is retained. This reinforces the idea that user mentions are a valuable feature for analysis. However, when user mentions are removed, the total NER-tag-count significantly drops in all categories. This indicates that punctuation plays a key role in the NER system's ability to correctly recognize named entities. Overall, the NER shows some linguistic patterns, but they do not differ as much as assumed between the two categories.

4.2. Part of Speech (POS)

POS tagging was applied to the same subsets as in the NER analysis. Across all four subsets, nouns (NOUN) are by far the most frequent tag, followed by proper nouns (PROPN), verbs (VERB), and adjectives (ADJ). The results are very similar between fake news and non-fake news. However, some subtle differences emerge when we see a slightly bigger use of verbs in fake news than non-fake news. This may reflect more use of assertiveness or allegations. In contrast, non-fake tweets contain slightly more nouns, reinforcing their likely use of more formal phrasing. These results hint more to the use of more advanced methods due to the big similarities in their pattern of speech (see appendix 8 for visual).

4.3. Topic modelling

For topic modelling we apply Latent Semantic Analysis (LSA) which is a statistical approach to capturing relations between words and documents. It works by collecting the vocabulary of the documents and putting it into a matrix to then reduce the dimensions of the matrix (Dumais, 2004, pp. 191-192). We mainly apply LSA rather than Latent Dirichlet Allocation (LDA) due to our use of TF-IDF as well as the semantic focus of using matrix that takes latent structures into account from the vector space. When looking at the results from LSA, we interpret the topics for the subset of fake news to be about themes like election misinformation, vaccine and COVID denial, income cut due to the Biden administration as well as celebrities. When interpreting the non-fake news, we see themes like minimum wage, inflation, taxation, gun control, racial justice, and public health or drug policy. We see that the non-fake news are more consistently policy focused.

5. Analysis results

5.1. Naive Bayes

The results from the Naive Bayes gave an accuracy of 89% with a recall score of 88% for tweets containing fake news and 91% for tweets with non-fake news. This indicates that the model is slightly better at identifying tweets with non-fake news by correctly identifying 91%. The f1-score is 89% for fake news tweets and 90% for non-fake news tweets, meaning that the model has a balanced ability to identify both classes. See appendix 6 for visuals.

We see a difference between the features and usernames with the highest correlation with fake news and non-fake news. The features with the highest correlation with fake news are DMX, Bruno, Burisma, and Ashli as well as the usernames @WendyRogersAZ, @CocaCola, @joe_exotic, @CawthornforNC and @SidneyPowell1. For non-fake news it is the features opioid, uninsured, heroin, evenly and birthright and the usernames @whitehouse, @Ask_Spectrum, @RyanLizza, @SenGillibrand and @NickKristof. This can be interpreted as fake tweets having a higher correlation with topics such as conspiracy theories and people involved in those topics as well as controversies. At the same time tweets with non-fake news have a higher correlation with topics from legitimate political topics as well as people and institutions.

We do see slightly more non-fake news tweets being classified as fake news than the other way around, however, looking at the f1-scores, we see that the difference is minimal. Also, the lack of context in Naive Bayes could also result in more fake news tweets being classified as non-fake news.

5.2. Logistic Regression

The Logistic Regression model achieved an accuracy of 92%, with a precision and recall of 92% for fake news tweets and a recall of 93% for non-fake news tweets. The F1-scores are balanced at 92% for both classes, indicating that the model performs equally well in identifying both fake and non-fake tweets. The confusion matrix shows that only 1,324 non-fake tweets were misclassified as fake, and 1,291 fake tweets were misclassified as true, reflecting a low and balanced number of false positives and false negatives. These results support the assumption that Logistic Regression, by capturing correlations between features, can outperform Naive Bayes in this context. See appendix 5 for visuals.

5.3. Random Forest

We showed feature importance based on the Naive Nayes above and this shows what the model weighs as being important. However, Naive Bayes also assumes feature independence which will most likely not be generalizable with complex data, and a more advanced model like Random Forest that can capture these complex relations and will be better at extracting the most important features that can be used to make generalized statements about feature importance (see appendix 1 for visual representation). Words such as Biden, Trump, Fauci, Vaccine, Americans, Covid and Obama play an important role in predicting fake news. This implies that recent themes from covid and American politics seem to be large factors in the model.

The Random Forest model trained on the full dimensionality dataset showed an F1-score of 90% and the macro average is also at 90%, whereas the RF-model trained on reduced dimensionality **under**performs this significantly. The confusion matrix showed that the model is equally efficient at predicting true false news and true, true news.

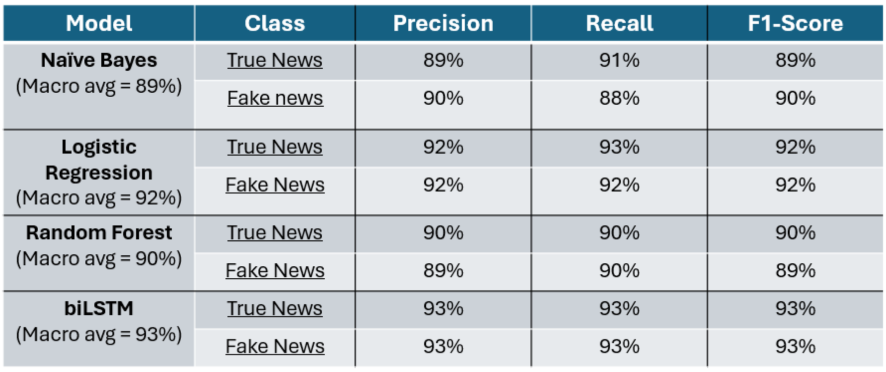
5.4. biLSTM

The biLSTM reached 12 epochs before the early stopping criteria was activated and ended the training. After training for 12 epochs, the model arrived at a training accuracy of 0.95%

and a validation accuracy of 0.92% (see appendix 2 for visualization of training). When tested on the test-split the model achieved an F1-score of 0.93% and a macro average of 0.93%. Furthermore, the model is also almost equally as good at predicting true negatives as it is at predicting true positives, and the confusion matrix showed that it predicted 1169 false negatives and 1192 false positives.

5.5. Comparison of the models

Below is an overview of the results of the models, considering precision, recall, F1 and the macro average:



Model	Class	Precision	Recall	F1-Score
Naïve Bayes (Macro avg = 89%)	True News	89%	91%	89%
	Fake news	90%	88%	90%
Logistic Regression (Macro avg = 92%)	True News	92%	93%	92%
	Fake News	92%	92%	92%
Random Forest (Macro avg = 90%)	True News	90%	90%	90%
	Fake News	89%	90%	89%
biLSTM (Macro avg = 93%)	True News	93%	93%	93%
	Fake News	93%	93%	93%

We see that the biLSTM is the best performing model, both in terms of predicting true positives but also in terms of predicting true negatives, and lands at a macro average of 93%. However, the biLSTM is not a significant improvement of the Logistic Regression which achieves a macro average of 92% and seems to be almost as good as the advanced biLSTM model. Naive Bayes and Random Forest are performing significantly worse than the LS, achieving a macro average of 89% and 90%, respectively.

6. Discussion, limitations and future work

From an ethical point of view, it is important to be aware of the potential unforeseen consequences of using a deep neural to predict something as important as true and fake news/tweets. Large deep neural networks function as a black box where it is not really possible to see how each layer learns and what factors/information the model emphasizes. Implementing a deep neural network for prediction of real and fake news might therefore be problematic since the model might contain unwanted biases that could potentially end up discriminating against users or classify satire as fake news. Using a model to classify something that ultimately impacts free speech, where it is not clear what the model bases its classifications on, calls for continuous human supervision.

It would be highly relevant to incorporate external labelled twitter and- or hashtag contextual data (HCFND) into the model, as proposed by Kumar et al. (2025), to nuance the data and make the model more generalizable. External data can also be used as a way to fact check tweets with legitimate sources, which could help ensure that the scaled classification is not only based on linguistic patterns but also on the factual accuracy of the content being evaluated. Using external data this way could potentially also help the model be scalable to other areas where news are being posted (e.g. Facebook, LinkedIn, etc.) since it would not only fit the “twitter”-lingo but would be adapted to a nuanced and context rich setting.

Language is also not static but constantly developing and changing. Therefore, a model that has been trained on data from a given period might not be generalizable for future applications since the model's training will be influenced by how the jargon and state of speech was at the time of collecting the language data for the model training. The speech and subjects of focus will also often be highly influenced by events happening around the world, and a dataset collected during the Corona crisis would, for example, be impacted by this. Therefore, a model needs to be continuously retrained to follow language developments in society if it should have any real life application.

Another potentially interesting area of investigation could have been to perform a sentiment analysis, if we for example had comments on the tweets, we could use these to analyze if there were generally negative sentiments in comments on fake news and investigate different sentiment relevant patterns in the comments pertaining to fake news and true news respectively. This could provide a better understanding of how fake news are received on twitter, and in case there is a systematic difference compared to true news it could be used to supplement the model by being incorporated as a feature of the model.

Lastly, an attempt to apply a BERT model for the deep neural network was made, since related work shows that this is by far the best embedding model when it comes to making classifications of fake news, but it was outside the capabilities of the authors and was therefore not implemented.

7. Conclusion

This study investigated the classification of fake and real news on Twitter using the TruthSeeker2023 dataset, leveraging a combination of traditional machine learning models, Naive Bayes, Logistic Regression, and Random Forest and a deep learning-based biLSTM architecture. The goal was to evaluate the performance of these models using both TF-IDF and Word2Vec embeddings, with a particular focus on maximizing the F1-score to ensure balanced performance across slightly imbalanced target classes.

The results showed that while the biLSTM model achieved the highest macro F1-score of 93%, Logistic Regression delivered nearly equivalent performance (92%) with significantly lower computational requirements. Naive Bayes and Random Forest followed closely, with F1-scores of 89% and 90%, respectively. These findings align with prior research showing that simpler models, when coupled with appropriate text representation and preprocessing, can perform competitively in binary classification tasks involving short-form content like tweets.

The exploratory data analysis (EDA) revealed notable linguistic and topical patterns between fake and real tweets. Named Entity Recognition and Part-of-Speech tagging suggested subtle differences, while topic modeling showed that fake news content is more focused on conspiracy theories and misinformation, whereas real news tweets lean toward policy and public welfare themes. These findings helped validate that certain language patterns and themes are systematically associated with fake or real content, supporting the use of such features in predictive models.

Ethical considerations were central to the analysis. Deep learning models such as biLSTM function as black boxes, making it difficult to interpret what drives their predictions. This opacity is particularly problematic in applications that affect freedom of expression, such as content moderation and misinformation detection. Furthermore, the subjectivity of crowd-sourced tweet labeling and the dynamic nature of online language introduce additional limitations regarding model fairness and generalizability.

Despite these limitations, the study highlights the feasibility of developing robust, scalable models for fake news detection. The ability to automatically flag problematic posts can enable faster platform-level interventions and support the trustworthiness of public discourse, especially in high-impact domains like public health and politics. Future work could include evaluating model performance across time periods, integrating external sources such as hashtags and contextual user data, or incorporating sentiments from user interactions to improve predictive accuracy and generalizability across platforms.

Ultimately, while no model can fully replace human judgment, the findings presented here offer a data-driven foundation for building decision-support tools that can help combat the growing challenge of misinformation on social media.

8. Literature

- Dadkhah, S., Zhang, X., Weismann, A. G., Firouzi, A., & Ghorbani, A. A. (2024). *The largest social media ground-truth dataset for real/fake content: TruthSeeker*. IEEE Transactions on Computational Social Systems, 11(3), 3376–3390.

- Dumais, S. T. (2004). Latent semantic analysis. *Annual Review of Information Science and Technology*, *38*(1), 188–230. https://doi.org/10.1002/aris.1440380105

- GeeksforGeeks. (2024, January 5). Word embeddings in NLP. GeeksforGeeks. https://www.geeksforgeeks.org/word-embeddings-in-nlp/

- Harika, J., & Hussain, S. J. (2024). Two-way Truth Seeker: a hybrid method using LSTM and BiLSTM to recognize and classify fake news.

- Iqubal, A., Kant Tiwari, S., Kumar Pasawan, M., & Asad, S. (2024). Machine Learning Methodologies for Predicting Fake News on Social Media X: A Comparative Investigation over TruthSeeker Dataset. 2024 International Conference on Computing, Sciences and Communications (ICCSC).

- Jurafsky, D., & Martin, J. H. (2023a). Speech and language processing (3rd ed., draft, Chapter 4). Retrieved from https://web.stanford.edu/~jurafsky/slp3/

- Jurafsky, D., & Martin, J. H. (2023b). Speech and language processing (3rd ed., draft, Chapter 5). Retrieved from https://web.stanford.edu/~jurafsky/slp3/

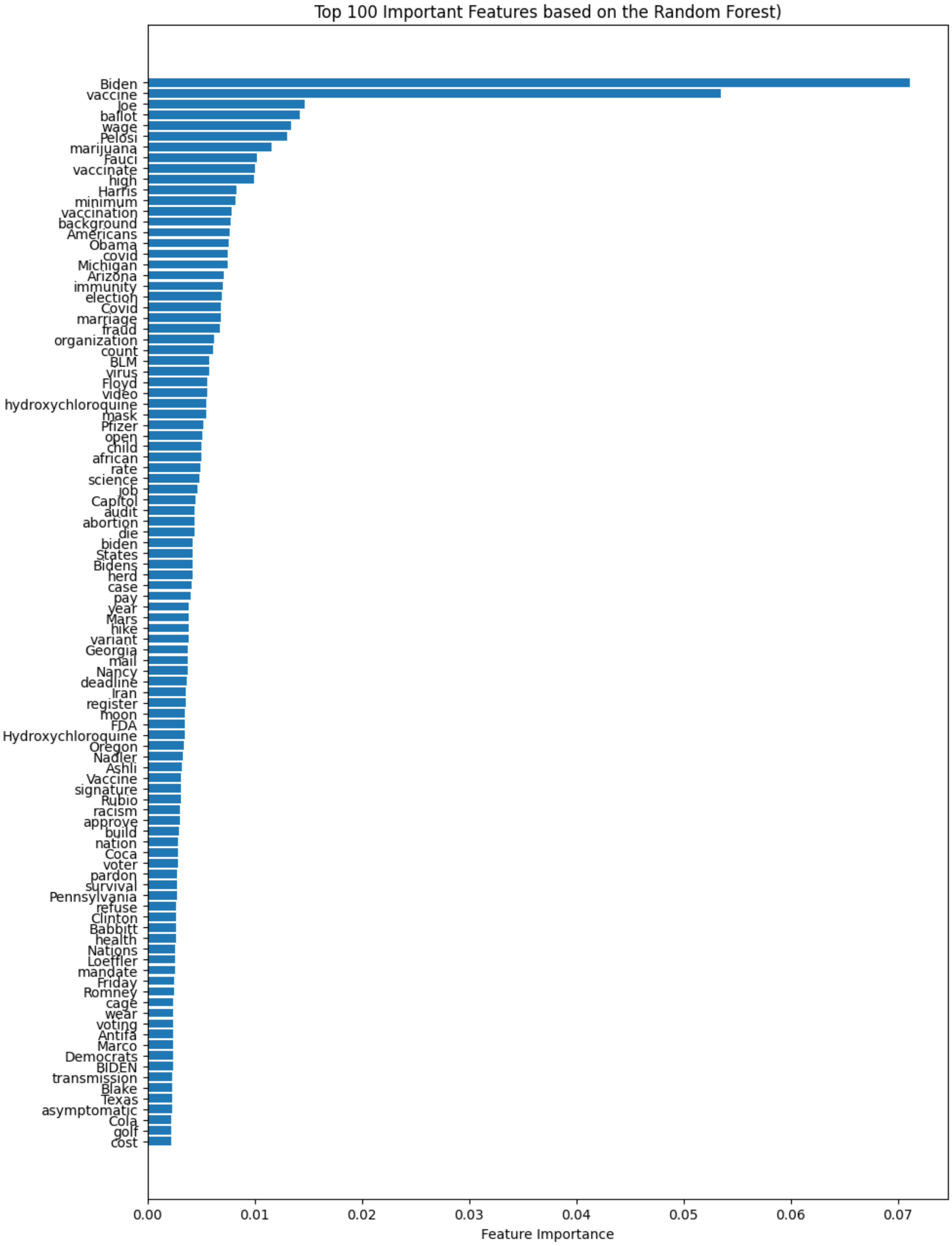
- Khalil, M., & Azzeh, M. (2024). *Fake news detection models using the largest social media ground-truth dataset (TruthSeeker)*. International Journal of Speech Technology, 27, 389–404.

- Kumar, S., Agrahari, S., Soni, P., Sachdeva, A., & Singh, S. (2025). Fake news detection using hastag content. Elsevier.

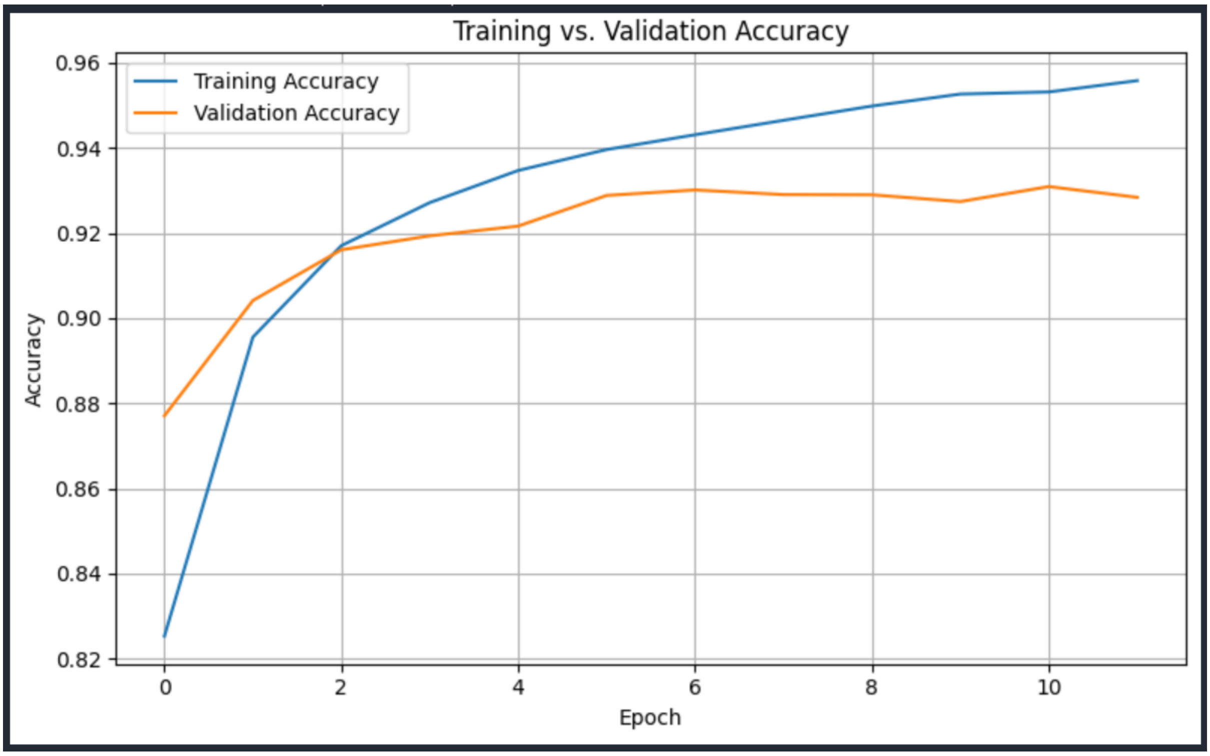
- University of California Berkeley Library. (n.d.). *Evaluate sources: Fact-checking websites*. UC Berkeley Library. https://guides.lib.berkeley.edu/c.php?g=620677&p=4333407

9. Appendix

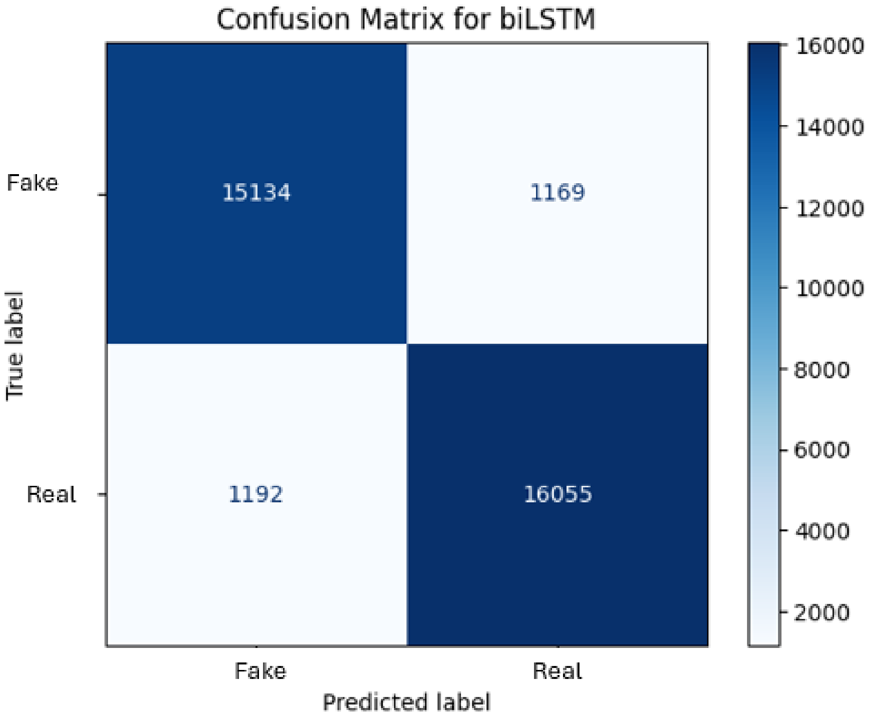
Appendix 1 - RF feature importance



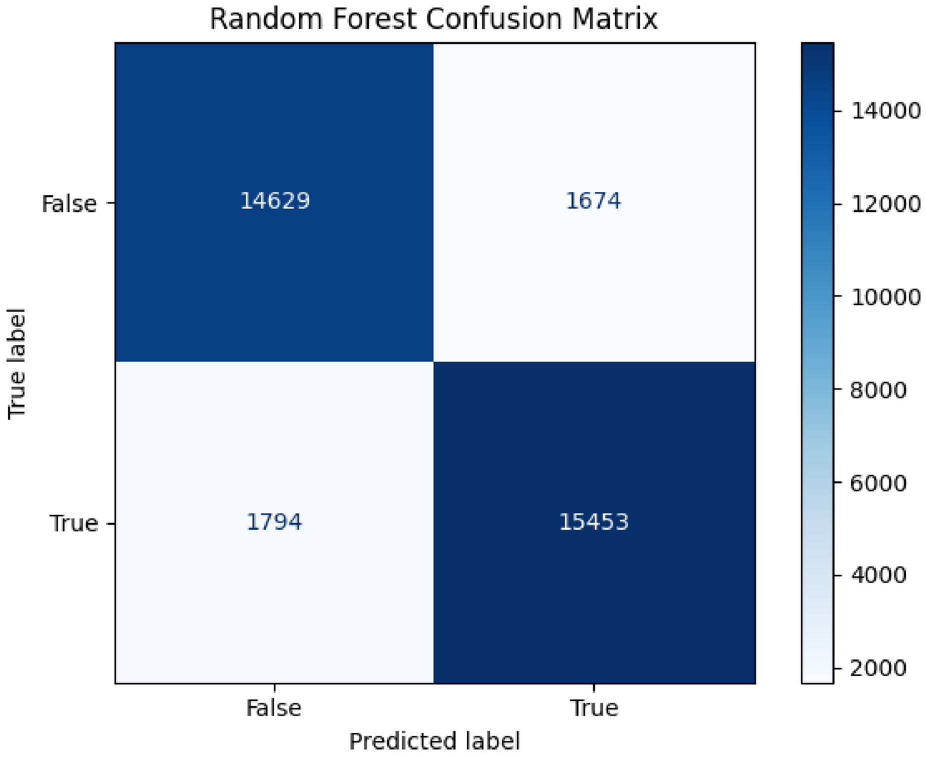
Appendix 2 - traning vs. val accuracy



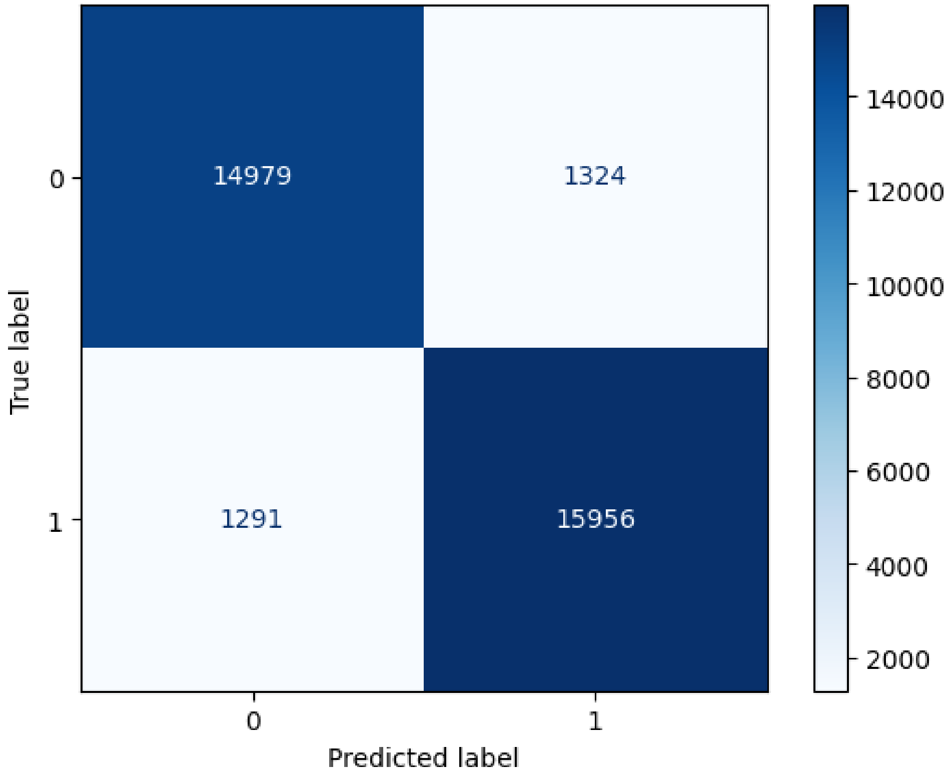
Appendix 3 - biLSTM confusion matrix



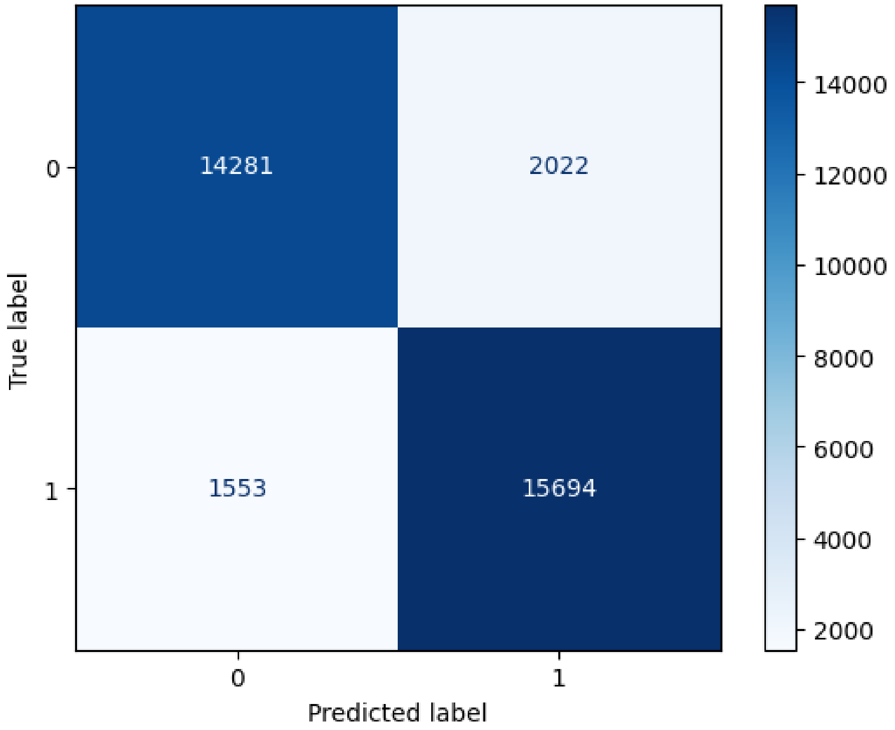
Appendix 4 - RF confusion matrix



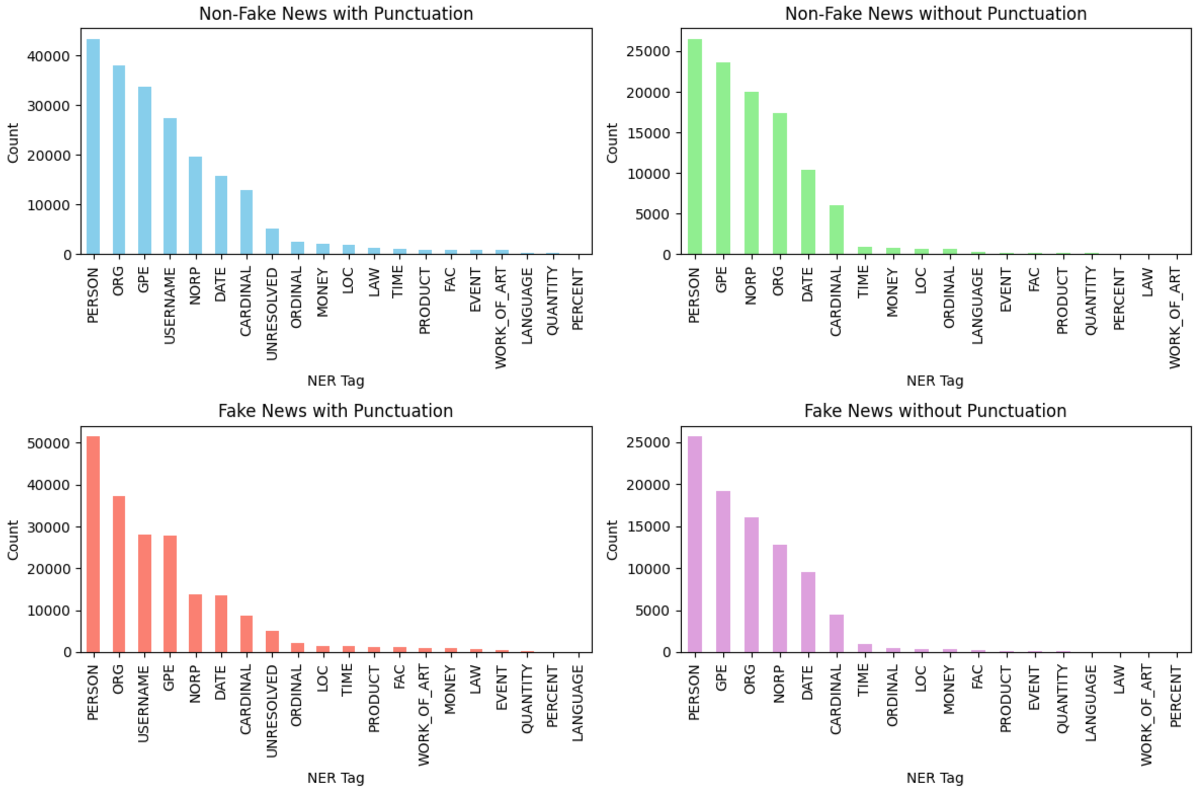
Appendix 5 - Logistic Regression confusion matrix



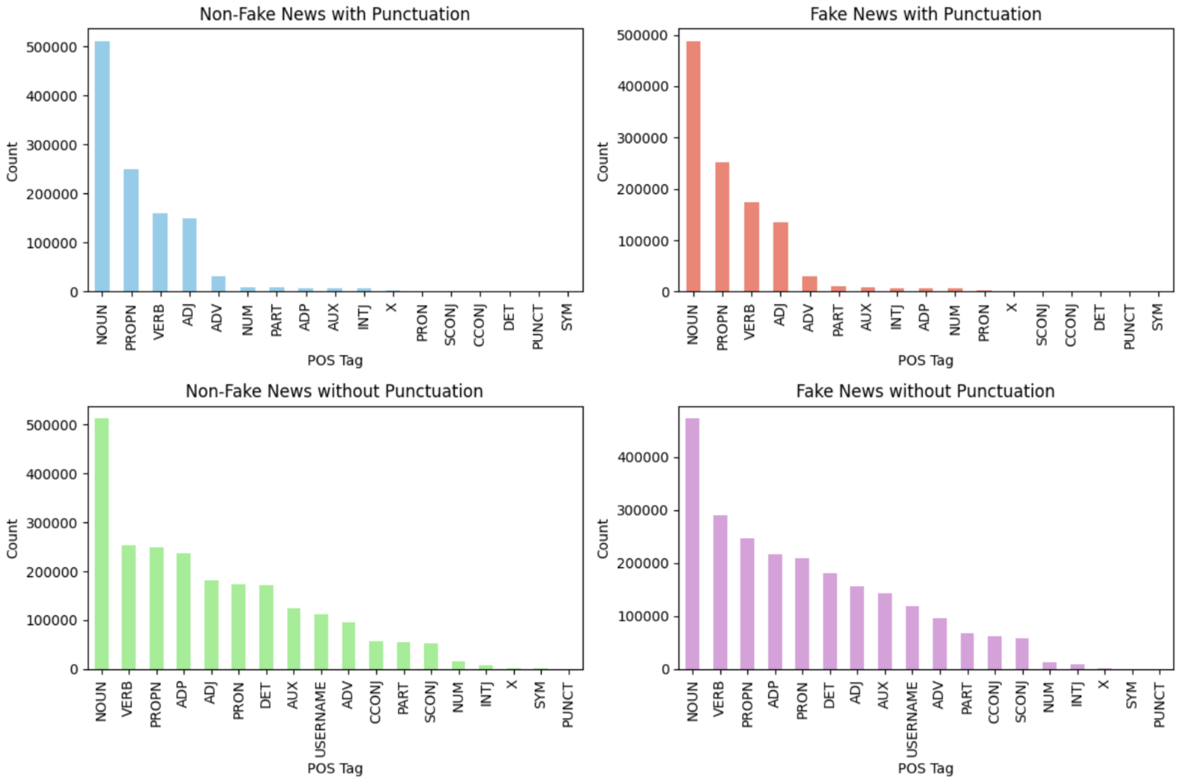
Appendix 6 - Naive Bayes confusion matrix



Appendix 7 - NER



Appendix 8 - POS


Non-Fake News with Punctuation
Count
NOUN
PROPN
VERB
ADJ
ADV
NUM
PART
ADP
AUX
INTJ
X
PRON
SCONJ
CCONJ
DET
PUNCT
SYM
POS Tag
Fake News with Punctuation
Count
NOUN
PROPN
VERB
ADJ
ADV
PART
AUX
INTJ
ADP
NUM
PRON
X
SCONJ
CCONJ
DET
PUNCT
SYM
POS Tag
Non-Fake News without Punctuation
Count
NOUN
VERB
PROPN
ADP
ADJ
PRON
DET
AUX
USERNAME
ADV
CCONJ
PART
SCONJ
NUM
INTJ
X
SYM
PUNCT
POS Tag
Fake News without Punctuation
Count
NOUN
VERB
PROPN
ADP
PRON
DET
ADJ
AUX
USERNAME
ADV
PART
CCONJ
SCONJ
NUM
INTJ
X
SYM
PUNCT
POS Tag