Fake News Detection in Hindi Language Using Ensemble Learning

Parth Kapil
Indiana University Bloomington
pkapil@iu.edu

Abstract

Due to the exponential growth in online information, it has become easier to consume news on online platforms. Unfortunately, this quick and easy access to the news has also made it convenient to spread fake and unreliable information. Fake news has the potential to cause extreme harm to individuals or society. Hence, fake news detection is one of the emerging research topic in today's era. Spreading fake news is a common issue in countries with lower literacy rates like India. Therefore, a system is required to flag the news as *fake* or *not-fake* in order to inform people about the news they are consuming. This paper suggests an automatic classification method based on ensemble learning techniques for news items. This project investigates many textual characteristics that can be utilized to distinguish between authentic and fraudulent content where various machine learning algorithms are trained using ensemble approaches based on those characteristics and assess their performance on two real-world datasets.

1 Introduction

The rise of fake news has become a major problem in today's society, with its ability to spread quickly and widely through social media and other online platforms. This has led to the spread of misinformation and a general erosion of trust in the media. The spread of fake news in India is a serious issue with negative consequences for the country and its population. False information can cause confusion and mistrust among the public, undermining the credibility of legitimate news sources and even leading to violence and riots.

Fake news is not a new phenomenon, but the rise of social media has made it easier than ever to spread false information quickly and widely. In India, fake news has been used to incite violence, including lynchings and other incidents of mob violence. This dangerous development threatens the safety and well-being of the individuals and communities.

A number of measures have been taken by the Indian government to halt the spread of false information, including the implementation of laws and regulations making it unlawful to disseminate false information, the creation of an independent fact-checking organization, and the provision of funding and support for media literacy initiatives. Additionally, it has worked with digital companies to create tools and processes for spotting and deleting bogus news Rodrigues and Xu (2020). Even though these initiatives may have had some effectiveness in preventing fake news, a better strategy is needed to automatically flag news as fake or authentic.

In order to address this problem, a machine learning solution for detecting fake news in Hindi in introduced in this paper. Machine learning algorithms can be trained to identify false information by examining various factors, such as the material's source, the language used, and the article's content. This system can quickly and correctly identify false or misleading information by leveraging techniques like natural language processing and predictive modeling. This contributes to reducing the spread of false information online. The ability to be updated and refined over time makes machine learning algorithms more effective at identifying and thwarting bogus information.

The main objective of the study was to develop and evaluate a machine learning model for detecting fake news in Hindi. The dataset of Hindi news articles was used (both real and fake) in order to train and test the model. The performance of the model was compared for different machine learning models (Naive Bayes, K Nearest Neighbors, Logistic Regression, Support Vector Machine, Random Forest, Stochastic Gradient Descent, Gradient

Boosting Classifier, XGBoost) to evaluate their effectiveness.

The remainder of this paper is organized as follows. In the next section, the related literature on fake news detection with machine learning is reviewed followed by methodology and evaluation of the proposed model. Finally, the result and implications are discussed in the last section.

2 Related Work

In recent years, there has been a growing interest in applying machine learning algorithms to identify fake news. In the past, researchers have concentrated on applying machine learning approaches to identify false information in a range of languages, including English Wang et al. (2020); Shu et al. (2019); Wang et al. (2018), Spanish Alonso García et al. (2020), and Chinese Alonso García et al. (2020).

A few studies have looked into the use of machine learning algorithms to identify bogus news in Hindi. They presented a framework for spotting fake news in Hindi that combines machine learning and natural language processing (NLP) approaches Sahoo and Gupta (2021). In their method, the text was first pre-processed to extract variables like word frequency and sentiment before being used to train a support vector machine (SVM) classifier. On a dataset of phony news stories in Hindi, the classifier was able to attain an accuracy of 88%.

In a different study Chakraborty et al. (2020), a deep learning model was used in place of SVM. Their algorithm achieved a 92% accuracy rate on a test set of falsified Hindi news stories.

Overall, these earlier investigations have shown how well machine learning algorithms can identify bogus news in Hindi. In this paper, the main goal is to compare the performance of different machine learning models on the task of detecting Hindi fake news and evaluate the effectiveness of various feature engineering approaches in this context.

3 Methodology

The Kaggle datasets were gathered for the study. The dataset was then cleaned and and feature engineering was performed on it. The text data was vectorized using the bag of words (BOW) and term frequency-inverse document frequency (tf-idf) techniques. The best models were utilized to generate an ensemble model after performing hyperparameter tuning on a variety of machine learning models.

3.1 Dataset

The lack of a high-quality dataset presents a serious obstacle to false news detection. For the English language, there were many datasets available. However, there were a very few datasets in Hindi language. So, quality dataset on Kaggle was gathered and the rest was collected from scrapping the Hindi news websites. The final dataset used for this study had 4000 fake news articles and 3500 genuine news articles, as shown in table 1.

News Type	News Count
Fake	4000
True	3500

Table 1: Count of true and fake news in the dataset.

3.2 Data Cleaning

Cleaning the dataset is an important step in the machine learning process because the quality of the data has a direct impact on the accuracy of the model. The model is probably going to be less accurate if the data is of poor quality. A model can learn more effectively and perform better if it is trained on a huge volume of clean, preprocessed data. For cleaning our dataset we performed following actions:

- Remove special characters
- Tokenization
- Remove stop words
- Stemming

3.2.1 Remove special characters

Removing special characters from text data is often required in machine learning because it helps the algorithms to process the data correctly and produce more accurate results. Special characters can interfere with the algorithms and cause them to produce incorrect results. By removing these characters, we can ensure that the data is clean and ready for analysis. For removing special characters from the dataset in this project, regular expression was used, and any character which is not alphanumeric or a Hindi character was removed.

3.2.2 Tokenization

Tokenization is the process of breaking a piece of text into smaller pieces called tokens. Tokenization is used to convert text into a numerical form

that can be used as input to a machine learning model. This is necessary because most machine learning algorithms cannot work with raw text data directly. By converting the text into tokens, one can represent the text in a way that the algorithm can understand and use it for training. For tokenizing Hindi text in the dataset, the NLTK python package was used in this project.

3.2.3 Remove stop words

Stop words are common words in a language that are typically filtered out before NLP tasks are performed on text data. This is because stop words do not contain useful information and can actually hinder the performance of some NLP algorithms. Classifier trained on a dataset with stop words included might be less accurate than a classifier trained on a dataset with stop words removed. This is because the inclusion of stop words can increase the size of the dataset and make it more difficult for the algorithm to identify the underlying patterns in the data. By removing stop words, one can reduce the size of the dataset and improve the performance of the NLP algorithms.

A predetermined list of Hind stop words Singh and Siddiqui (2012) was used to eliminate stop words from the dataset. The entire corpus of Hindi texts was examined and any instances of these stop words were deleted.

3.2.4 Stemming

Stemming is a technique used in NLP to reduce words to their base or stem form. This is typically done by removing suffixes. The main reason for using stemming in a machine learning problem is to reduce the dimensionality of the data. Since many words in the Hindi language have similar meanings, reducing them to their base form can help reduce the number of unique words that the model needs to learn. This can make the model more efficient and improve its performance.

A list of frequently used Hindi suffixes was compiled to conduct stemming. After that, a list of these suffixes was created and looped over each sentence. The sentences were broken into tokens to examine if there is any match from the list. If the match was found, the suffix portion of that token was taken away and replaced with a updated token.

3.3 Feature Engineering

Some of the useful feature engineering techniques that were used for the dataset are:

- Number of words in an news article.
- Number of unique words in an news article.
- Bag-of-words
- Term frequency-inverse document frequency

3.3.1 Number of words in an news article

The number of words in a sentence is an important metric for NLP analysis since it can give information on the length of the phrase, which can be helpful for a variety of tasks. Most NLP tasks such as text classification or sentiment analysis use this method frequently while feature engineering. For the purpose of this project, the length function was applied to each news article and a new column "word_count" was added to the dataset.

3.3.2 Number of unique words in an news article

This is another very useful metric used for NLP problems. It can provide information about the complexity and diversity of the words used in the sentence. A sentence with a higher number of unique words is likely to be more complex and diverse than a sentence with a lower number of unique words. This information can be useful for a variety of NLP tasks, such as text classification, sentiment analysis, and summarization. For this project, each news article was broken into tokens and every unique token was counted. This metric was stored in a new column as "unique_words" for the dataset.

3.3.3 Bag-of-words

Bag-of-words is a way of representing text data for use in NLP and machine learning algorithms. It is called a bag-of-words because any information about the order or structure of the words in the text is discarded, and only the words themselves are used.

To create a bag-of-words representation, the text is first split into individual words, and a vocabulary is created by collecting all of the unique words in the text. Then, for each document (or piece of text), a vector is created where each element represents the frequency of a specific word in the vocabulary. This vector can then be used as input to a machine learning algorithm, or as a feature in other NLP tasks.

For this project, sklearn package's CountVectorizer function was used and preprocessed clean news article data was converted into vector form.

3.3.4 Term frequency-inverse document frequency

Term frequency-inverse document frequency (TFIDF) is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. The basic idea behind TF-IDF is that the more times a word appears in a document, the more important it is to that document. However, if a word appears many times in a document but also appears many times in other documents, then it may not be as important to that document.

To calculate the TF-IDF of a word in a document, one has to first calculate the term frequency (TF) of the word, which is simply the number of times the word appears in the document. It is defined as,

$$TF = \frac{\text{\# of times word appears in document}}{\text{Total \# of words in document}}$$

Then, one needs to calculate the inverse document frequency (IDF) of the word, which is a measure of how common the word is across all documents in the collection. It is defined as,

$$IDF = \log\left(\frac{\text{Total \# of documents in collection}}{\text{\# of documents with the word in them}}\right)$$

Finally, to calculate the TF-IDF of a word in a document, multiply the TF and IDF values together:

$$TF - IDF = TF * IDF$$

TF-IDF is used in many NLP tasks, such as document classification and information retrieval. It is often used as a weighting factor in these tasks, with words that have higher TF-IDF values being considered more important than words with lower TF-IDF values.

For this dataset, sklearn package's TfidfVectorizer function is used and preprocessed clean news article data was converted into vector form.

4 Modeling

After cleaning and performing feature engineering on the data, the dataset was divided into two different types of datasets.

1. Bag-of-words dataset: contains vector generated through BOW technique.
2. TF-IDF dataset: contains vector generated through TF-IDF technique.

For performing analysis of the machine learning models, these two datasets were divided into three sub-categories of datasets:

1. Train set: This set was used to train the machine learning models.
2. Cross validation set: This set was used to validate the models using hyperparameter tuning.
3. Test set: This set was used to test the performance of the final trained model.

For deciding the base classifiers for the ensemble machine learning models, hyperparameter tuning was performed on the following machine learning classifiers:

- Naive Bayes
- K Nearest Neighbors
- Logistic Regression
- Support Vector Machine
- Random Forest
- Stochastic Gradient Descent
- Gradient Boosting Classifier
- XGBoost

The scikit-learn library's GridSearchCV function was used to perform hyperparameter tuning. GridSearchCV function trains a model using each combination of hyperparameters and evaluate each model using a scoring function given at the time of initializing it. A 10 fold cross validation was performed for each machine learning classifiers mentioned above. F1 score metric was used as a scoring measure. The hyperparameter tuning was performed on both the BOW and the TF-IDF datasets. The best hyperparameters and their corresponding F1 score for each model on the BOW dataset can be viewed in Table 2.

The findings of the BOW datasets' hyperparameter tuning suggest that all machine learning classifiers performed similarly. But, support vector machine, random forrest and XGBoost performed the best.

Classifier	F1 Score	Best Paramters
Naive Bayes	0.80	prior=[0.75, 0.25] var_smooting=0.01
K-nearest Neighbors	0.81	metric= euclidean n_neighbors=5 weights=distance
Logistic Regression	0.83	C=1 max_iter=100 penalty=l2 solver=liblinear
Support Vector Machine	0.86	C=1 gamma=0.1
Random Forest	0.85	criterion=gini max_depth=20 n_estimator=30
Stochastic Gradient Descent	0.83	alpha=0.001 max_iter=300 penalty=l2
Gradient Boosting Classifier	0.82	min_samples_split=10 max_depth=5 n_estimator=50
XGBoost	0.87	gamma=0.01 max_depth=5 n_estimator=100

Table 2: F1 score and best parameters for each machine learning classifier on BOW dataset.

Classifier	F1 Score	Best Paramters
Naive Bayes	0.82	prior=[0.75, 0.25] var_smooting=0.1
K-nearest Neighbors	0.65	metric= chebyshev n_neighbors=7 weights=distance
Logistic Regression	0.84	C=1 max_iter=100 penalty=l2 solver=saga
Support Vector Machine	0.85	C=10 gamma=1
Random Forest	0.85	criterion=gini max_depth=25 n_estimator=50
Stochastic Gradient Descent	0.79	alpha=0.001 max_iter=500 penalty=l2
Gradient Boosting Classifier	0.84	min_samples_split=10 max_depth=5 n_estimator=50
XGBoost	0.87	gamma=0.1 max_depth=5 n_estimator=50

Table 3: F1 score and best parameters for each machine learning classifier on TFIDF dataset.

The best hyperparameters and their corresponding F1 score for each model on the TFIDF dataset can be viewed in Table 3.

Similar to the findigs of BOW dataset, TFIDF dataset' hyperparameter tuning suggest that all machine learning classifiers performed similarly. But, support vector machine, random forrest and XGBoost performed the best.

Based on the findings of hyperparameter tuning on both the BOW and TF-IDF datasets, support vector machine, random forrest and XGBoost were chosen as the base classifiers for the ensemble model. For building the ensemble model VotingClassifier was used.

VotingClassifier is an ensemble machine learning model that combines the predictions of multiple individual models in order to make a final prediction. It works by training multiple models on the same dataset, then allowing each model to make predictions on new data. The final prediction is made by taking the average of the predictions of all the models, weighted by their accuracy. This can improve the performance of the model because it allows the model to combine the strengths of each individual model, potentially leading to more accurate predictions.

For this study VotingClassifier was setup with the best hyperparameters for support vector machine, random forrest and the XGBoost classifiers and was trained on the BOW and TF-IDF training set.

5 Results

The ensemble model was evaluated on three different metrics:

- Accuracy: It is the proportion of correct predictions that the model makes, with respect to all the predictions that it makes. It is calculated by:

$$Accuracy = \frac{\text{number of correct predictions}}{\text{total number of predictions}}$$

- F1 score: is defined as the harmonic mean of the precision and recall of the model, where precision is the ratio of true positive predictions to all positive predictions, and recall is the ratio of true positive predictions to all actual positive cases. The formula for the F1 score is:

$$F1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

- Log Loss: It measures the accuracy of a model's predictions, but unlike the accuracy metric, which only considers correct predictions, log loss takes into account the probability of each prediction. It is calculated by taking the logarithm of the predicted probability for each sample and multiplying it by the true label, then taking the average of all the individual log loss values. The formula for log loss is:

$$LogLoss = -\frac{1}{N}\sum_{i=1}^{N} y_i \log(p_i) - \frac{1}{N}\sum_{i=1}^{N}(1-y_i)\log(1-p_i)$$

where N is the total number of samples, y_i is the true label for the ith sample (0 or 1)and p_i is the predicted probability that the ith sample belongs to the positive class.

The result metrics for the proposed ensemble model on the BOW and the TF-IDF model can be viewed in table 4.

Dataset	Accuracy	F1 score	Log Loss
BOW	0.85	0.87	3.9
TF-IDF	0.87	0.89	3.6

Table 4: Ensemble model performance on the BOW and TF-IDF datasets.

The ensemble model's test results demonstrate that both models performed similarly. However, the model performed slightly better on the TF-IDF dataset than it did on the BOW dataset.

6 Conclusion

In conclusion, the results of this study show that the use of an ensemble model can improve the performance of a base classifiers in the case of Hindi language fake news detection when the Hindi text is vectorized using TF-IDF technique. It was found that the support vector machine, random forests and XGBoost classifiers performed the best on both the bag-of-words and TF-IDF representation of hindi news corpus. When all of these machine learning models were used as base learners for the ensemble model, it significantly improved the performance of fake news detection. However, the results can be further improved in future work by using part of speech tagging techniques and using deep learning models like Long Short-Term Memory (LSTM) which are designed to process and predict sequential data.

References

Santiago Alonso García, Gerardo Gómez García, Mariano Sanz Prieto, Antonio José Moreno Guerrero, and Carmen Rodríguez Jiménez. 2020. The impact of term fake news on the scientific community. scientific performance and mapping in web of science. *Social Sciences*, 9(5):73.

Koyel Chakraborty, Surbhi Bhatia, Siddhartha Bhattacharyya, Jan Platos, Rajib Bag, and Aboul Ella Hassanien. 2020. Sentiment analysis of covid-19 tweets by deep learning classifiers—a study to show how popularity is affecting accuracy in social media. *Applied Soft Computing*, 97:106754.

Usha M Rodrigues and Jian Xu. 2020. ¡? covid19?¿ regulation of covid-19 fake news infodemic in china and india. *Media International Australia*, 177(1):125–131.

Somya Ranjan Sahoo and Brij B Gupta. 2021. Multiple features based approach for automatic fake news detection on social networks using deep learning. *Applied Soft Computing*, 100:106983.

Kai Shu, Limeng Cui, Suhang Wang, Dongwon Lee, and Huan Liu. 2019. defend: Explainable fake news detection. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 395–405.

Satyendr Singh and Tanveer J Siddiqui. 2012. Evaluating effect of context window size, stemming and stop word removal on hindi word sense disambiguation. In *2012 International Conference on Information Retrieval & Knowledge Management*, pages 1–5. IEEE.

Yaqing Wang, Fenglong Ma, Zhiwei Jin, Ye Yuan, Guangxu Xun, Kishlay Jha, Lu Su, and Jing Gao. 2018. Eann: Event adversarial neural networks for multi-modal fake news detection. In *Proceedings of the 24th acm sigkdd international conference on knowledge discovery & data mining*, pages 849–857.

Yaqing Wang, Weifeng Yang, Fenglong Ma, Jin Xu, Bin Zhong, Qiang Deng, and Jing Gao. 2020. Weak supervision for fake news detection via reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 516–523.