FAKE NEWS DETECTION

Ashika Srinivasan 5/18/23 Applied Artificial Intelligence

Table of Contents

Table of Figure

INTRODUCTION

The amount of information in the digital age has provided us with unparalleled access to news and opinions worldwide. However, this democratization of data has its own set of issues, the most alarming of which is the proliferation of false news. Fake news is a systemic menace to our society, undermining confidence, influencing public opinion, and spreading strife. The growth of inaccurate and misleading information on the internet is the problem that an AI prototype for fake news identification tries to remedy. Fake news may have significant implications, such as distributing incorrect information, causing reputational harm, influencing public opinion, and even influencing political decisions. This implies training an AI model on a dataset of labeled news items that include instances of genuine and misleading news.

LITERATURE REVIEW

Most U.S. people (62%) get their news largely from social media. For many individuals all across the world, social media is becoming their primary source of information. The structure of current social media platforms differs significantly from that of traditional news channels such as newspapers. Steps taken by generic news platforms, such as third-party filtering, fact-checking, and editorial judgment, are not accepted by social media networks. Other methods for spotting false news include social network analysis, web mining, and data visualization. A study published in the Proceedings of the 27th ACM Conference on Hypertext and Social Media, for example, used social network analysis to identify sources of fake news on Twitter, while another study published in the Journal of Computational Science extracted features from news articles for classification. Overall, the research indicates that AI-based approaches for detecting fake news have promise and that machine learning algorithms and natural language processing techniques can be useful for identifying patterns and traits associated with false information.

DATASET

The dataset is titled "Fake and Real News Dataset" and is available on Kaggle. It is a compilation of news items classified as "fake" or "real." The dataset is made up of two CSV files. The first file, "fake.csv," contains roughly 23,000 news stories labeled "fake." These stories were gathered from many sources, including well-known fake news websites and social media platforms. The second file, "true.csv," contains roughly 21,000 news stories classified as "real." These stories were compiled by credible news organizations such as Reuters and the Associated Press. Each article in the dataset is labeled with a title, text content, publication date, and publisher name. The articles were also preprocessed to eliminate any extraneous material and formatting, such as HTML elements and punctuation marks.

SELECTED AI TECHNIQUE

Natural language processing (NLP) is used to examine the language used to identify patterns that are common in misleading news stories. Researchers emphasize the challenges involved in fake news identification in a paper released in 2018. They employed NLP approaches to extract linguistic elements from news articles and achieved an accuracy of 0.92 in recognizing false news.

Logistic regression is a statistical model for binary classification problems. It predicts if a certain news item is fraudulent or not based on a set of attributes extracted from the article. According to the 2018 research, the goal is to classify tweets by comparing logistic regression with naive Bayes for text classification. When compared to the naive Bayes classifier, the logistic regression-based classifier had the greatest accuracy of 91.1%.

Decision trees are used to construct a model that predicts whether a news report is true or fake based on a collection of attributes extracted from the article. In this study (2021), the mean accuracy and standard deviation for recognizing untrue political news items using decision tree algorithms were 99.6990 and 0.10577, respectively, when compared to naive Bayes. The significance rate is 0.022, indicating that the prediction is correct.

Naive Bayes is a text classification method that has been used in various research projects to detect false news. In this study, Mykhailo Granik et al. demonstrate a naive Bayes classifier-based approach for recognizing misleading propaganda. This method was turned into a development platform and tested with a variety of Facebook new articles with a classification precision of around 74.05%.

Bi-LSTM is an LSTM extension that combines two separate RNNs. Bi-LSTM will predict future word usage because it receives backward information from the other RNN layer. This 2019 study provides a false-news detection approach based on a bidirectional LSTM recurrent neural network. The recommended approach discovered false news by determining whether the article's content is correct, evaluating the bias of a published news story, and analyzing the relationship between the title and the article body.

All of these evidence-based AI techniques may be used to detect bogus news. Because the data reveals that the naive Bayes technique has lower accuracy than other algorithms and models, using it for computations is not an excellent choice.

a) Advantages

- NLP enables the rapid analysis of massive volumes of text data and is used to extract a wide range of characteristics. NLP models may learn from data and improve over time, making them increasingly successful at detecting fake news as they are trained on more data.

- Logistic regression is a simple and efficient technique that gives a probabilistic output that may be used to evaluate the likelihood of a piece of news being false or not.

- Decision trees are simple to read and explain, which can help to grasp exactly how the model produces predictions. It can handle both category and numerical data.

- BI-LSTM can capture both the past and future context of a word in a sequence, making it ideal for tasks such as language modeling and text categorization. It can also be taught end-to-end via backpropagation. It enables it to evaluate both past and future context, perhaps making it more successful at capturing long-term dependencies in data.

b) Disadvantage

- NLP models can struggle to recognize sarcasm, irony, and other kinds of figurative language, which are frequently utilized in false news, which can lead to incorrect projections.

- Logistic regression may struggle to capture linguistic subtleties and the complexities of social media platforms, which are frequently used to spread fake news.

- Decision trees are sensitive to slight changes in data and feature selection, making them less resilient and dependable than other methods.

- To prevent overfitting, hyperparameter adjustments are sometimes required. It might be difficult to identify which characteristics are driving the model's predictions when interpreting the judgments made by a BI-LSTM model.

IMPLEMENTATION

a) Preprocessing techniques

1. Concatenating the title and text of rows for the news to be categorized based on the title and text together.
2. Removed links and news headlines from the date column, which would cause problems when converting to date and time format.
3. Preprocessing the text by eliminating punctuation and using stop words to exclude regularly used terms like "the,” "a,” "an,” and so on.
4. Creating the story from the news by counting both the news and the news subject on whether they are accurate or false.
5. Taking the qualities of the news and extracting the features such as polarity, length of the review, size of the news, and word count.
6. N-gram analysis(bigram and trigram) is valuable in identifying false news because it captures not just individual words but also the associations between nearby words, which can help comprehend the context and meaning of the text.
7. WordCloud was used to visualize data for both fake and authentic news.

8. Stemming is utilized to derive the root word from the impacted term since the root words do not need semantic significance.

b) Initialization of hyperparameters

Lower C values, such as 0.1, indicate more regularisation. 'l2' refers to L2 (ridge) regularisation, which introduces a penalty term into the loss function to promote lower coefficients. The optimizer option specifies the optimization process used for getting the best logistic regression coefficients. By setting *max_iter* to 1000, the solver has enough iterations to converge. The random seed value of 42 guarantees that the model is randomly initialized.

The term *gini* refers to the Gini impurity criteria, which calculate a node's impurity or disorder according to the distribution of class labels. When set to 3, the tree can only have three levels of nodes, from the root to the leaves. A node with a value of 2 must contain at least two samples to be evaluated for splitting. When *max_features* is set to none, all features are considered for each split. The random processes involved in the decision tree's training and splitting have a value of 42. Developing a binary classification model with a 32-unit bidirectional LSTM layer, a 0.2-rate dropout layer, and a dense layer containing a sigmoid activation function.

The model's input is intended to be a series of 5000 integer-encoded words, each represented by a 128-dimensional embedding vector. Input sequences with more than 25 words will be terminated. The binary cross-entropy loss and the Adam optimizer are used to construct the model, and accuracy is employed as the evaluation metric. To discover the optimum model, these parameters must be tuned using techniques such as grid search and cross-validation.

c) Brief Explanation

Using the train_test_split function, divide the dataset X and its labels y for training and testing. The test_size option defines the percentage of data that should be utilized for testing, which in this example is 0.25. Then cross-validating models using ten folds, logistic regression, and decision trees. The classifier, the feature matrix, the target vector, the number of cross-validation folds, and the scoring metric (scoring = "accuracy") are all sent into the cross_val_score function. The mean accuracy got acquired tenfold.

The embedding layer is used to transform integer-encoded input sequences into fixed-sized vectors by specifying the vocabulary size, the dimensionality of the dense embedding, and the input sequence length. The model has acquired two bidirectional LSTM layers. By analyzing the input sequence in both directions, bidirectional LSTM integrates forward and backward information. The first LSTM layer has return_sequences=true, which causes it to return the whole series of outputs; however, the second LSTM layer does not, so it produces the last output.

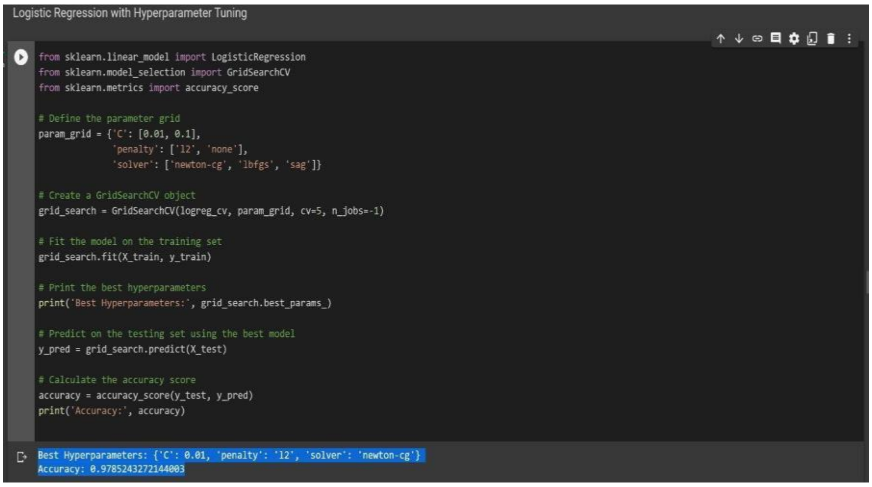
Dropout prevents overfitting by randomly setting a percentage of input units to 0 during training. The model includes a dense output layer with a sigmoid activation function and is constructed using the binary cross-entropy loss and the Adam optimizer. The model's performance is evaluated using the accuracy measure.

The model architecture is summarised, including each layer's input and output shapes and the total number of trainable parameters.

FINE-TUNING TECHNIQUES

Combining these methods improves the efficiency and accuracy of decision trees, logistic regression, and Bi-LSTM for detecting fake news. The performance and accuracy were improved by oversampling the minority class and undersampling the majority class to balance the dataset.

Using a weighted average ensemble technique, combine the predictions of a decision tree classifier with those of a logistic regression classifier. It sums together each forecast and divides the total by two to get the ensemble predictions, which are rounded. A 5-fold cross-validation approach is specified to test each hyperparameter combination's performance. It then predicts the target variable from the testing data using the best model found through the grid search. The logistic regression model's accuracy with tuned hyperparameters is roughly 97.85%. These results show that the logistic regression model accurately predicted the target variable using the hyperparameters supplied. The best alternatives were the 'l2' penalty and the 'newton-cg' solver, with a regularisation value of C = 0.01.



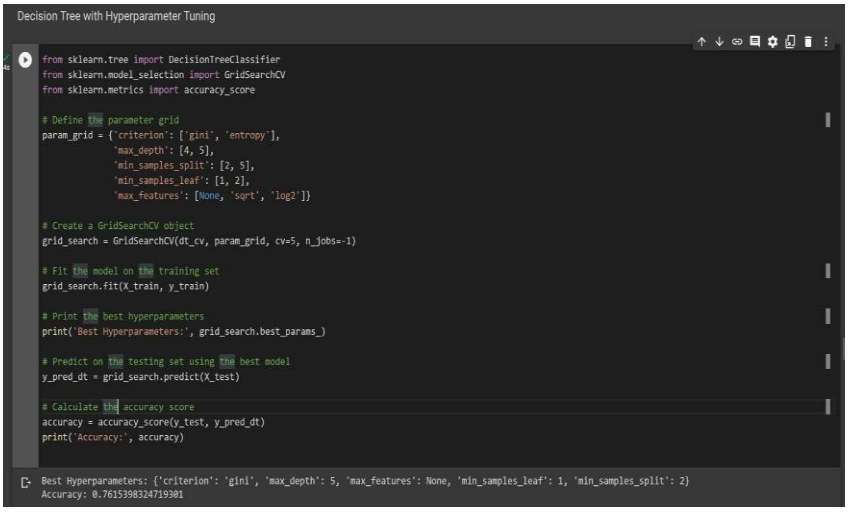
Logistic Regression with Hyperparameter Tuning

```
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import accuracy_score

# Define the parameter grid
param_grid = {'C': [0.01, 0.1],
              'penalty': ['l2', 'none'],
              'solver': ['newton-cg', 'lbfgs', 'sag']}

# Create a GridSearchCV object
grid_search = GridSearchCV(logreg_cv, param_grid, cv=5, n_jobs=-1)

# Fit the model on the training set
grid_search.fit(X_train, y_train)

# Print the best hyperparameters
print('Best Hyperparameters:', grid_search.best_params_)

# Predict on the testing set using the best model
y_pred = grid_search.predict(X_test)

# Calculate the accuracy score
accuracy = accuracy_score(y_test, y_pred)
print('Accuracy:', accuracy)
```

```
Best Hyperparameters: {'C': 0.01, 'penalty': 'l2', 'solver': 'newton-cg'}
Accuracy: 0.9785243272144003
```

Figure 1 - Logistic Regression

For the Decision Tree Classifier, Several factors need to consider while determining the best split. When the grid search is finished, the optimal hyperparameter combination is identified. The decision tree classifier's accuracy with configured hyperparameters is roughly 76.15%, which is a moderate level of accuracy. To assess the quality of the split, the *gini* criteria were used. The *max_depth* was set to 5, and the minimum number of samples necessary to be at a leaf node was set to 1. Furthermore, because *max_features* was set to None, the overall amount of features to analyze for the optimal split was not limited.



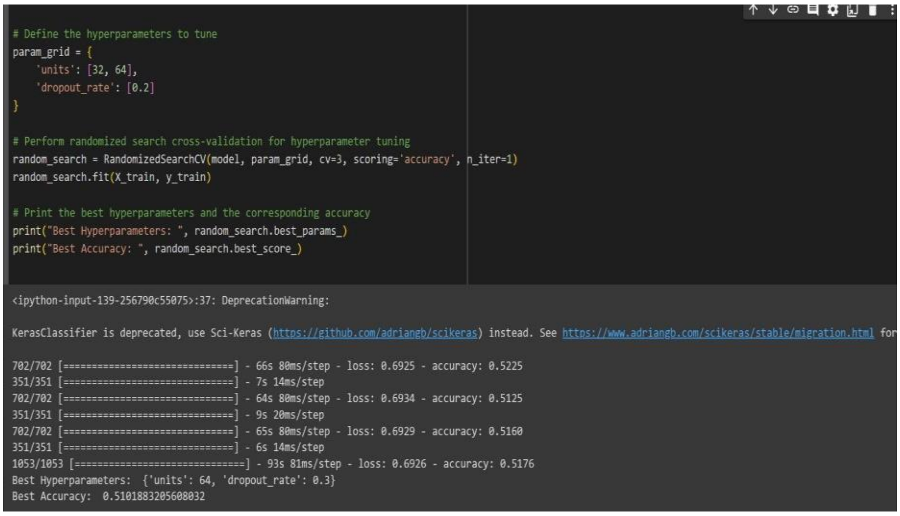
Decision Tree with Hyperparameter Tuning

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import accuracy_score

# Define the parameter grid
param_grid = {'criterion': ['gini', 'entropy'],
              'max_depth': [4, 5],
              'min_samples_split': [2, 5],
              'min_samples_leaf': [1, 2],
              'max_features': [None, 'sqrt', 'log2']}

# Create a GridSearchCV object
grid_search = GridSearchCV(dt_cv, param_grid, cv=5, n_jobs=-1)

# Fit the model on the training set
grid_search.fit(X_train, y_train)

# Print the best hyperparameters
print('Best Hyperparameters:', grid_search.best_params_)

# Predict on the testing set using the best model
y_pred_dt = grid_search.predict(X_test)

# Calculate the accuracy score
accuracy = accuracy_score(y_test, y_pred_dt)
print('Accuracy:', accuracy)
```

```
Best Hyperparameters: {'criterion': 'gini', 'max_depth': 5, 'max_features': None, 'min_samples_leaf': 1, 'min_samples_split': 2}
Accuracy: 0.7615398324719301
```

Figure 2 - Decision Tree

The hyperparameters for Bi-LSTM are units and *dropout_rate*. The parameter grid holds the to-be-tuned hyperparameters. Experimented with the different dropout rates and adjusted the embedding sizes. A 3-fold cross-validation approach is specified to test each hyperparameter combination's performance. Accuracy (scoring) specifies the metric to be optimized during the hyperparameter search. The Bi-LSTM model's highest accuracy with tweaked hyperparameters is roughly 51.02%. The model design includes two bidirectional LSTM layers, each with 64 units or neurons. To reduce overfitting, dropout regularisation at a rate of 0.3 is used.



```
# Define the hyperparameters to tune
param_grid = {
    'units': [32, 64],
    'dropout_rate': [0.2]
}

# Perform randomized search cross-validation for hyperparameter tuning
random_search = RandomizedSearchCV(model, param_grid, cv=3, scoring='accuracy', n_iter=1)
random_search.fit(X_train, y_train)

# Print the best hyperparameters and the corresponding accuracy
print("Best Hyperparameters: ", random_search.best_params_)
print("Best Accuracy: ", random_search.best_score_)
```

```
<ipython-input-139-256790c55075>:37: DeprecationWarning:

KerasClassifier is deprecated, use Sci-Keras (https://github.com/adriangb/scikeras) instead. See https://www.adriangb.com/scikeras/stable/migration.html for

702/702 [==============================] - 66s 80ms/step - loss: 0.6925 - accuracy: 0.5225
351/351 [==============================] - 7s 14ms/step
702/702 [==============================] - 64s 80ms/step - loss: 0.6934 - accuracy: 0.5125
351/351 [==============================] - 9s 20ms/step
702/702 [==============================] - 65s 80ms/step - loss: 0.6929 - accuracy: 0.5160
351/351 [==============================] - 6s 14ms/step
1053/1053 [==============================] - 93s 81ms/step - loss: 0.6926 - accuracy: 0.5176
Best Hyperparameters:  {'units': 64, 'dropout_rate': 0.3}
Best Accuracy:  0.5101883205608032
```

Figure 3 - Bi Lstm

Handling missing data by detecting and then calling the resultant DataFrames as fake_news and true_news to calculate the total missing values for each column. Also, calculate the sum of all missing values across all columns to determine the overall number of missing values.

Here, the top 5000 features with the highest TF-IDF scores are selected. The decision is based on the corpus value. Developed a TF-IDF feature matrix from the text data by specifying the number of rows and columns in the matrix. It enables capturing the value of terms in each text while taking their overall relevance in the corpus into account.

EVALUATION AND ANALYSIS

Logistic regression

The model's accuracy is 0.9604, which implies that it accurately predicts class labels for around 96.04% of the data. Also, the recall score is 0.9604. It shows that the model catches a large fraction of real positive samples (class 1) out of all true positive samples. The F1 score is 0.9604, which gives a fair model's performance by accounting for both false positives and false negatives. For each class, the classification report offers a full summary. Precision, recall, and F1 scores in both classes are about 0.96, indicating balanced performance. The curve visually represents the trade-off between the TPR and FPR, and the AUC score summarises with a higher value of 0.96, indicating better performance with higher TPR and lower FPR across various threshold values.

Decision Tree

The accuracy is 0.8148, which indicates that the model has a low false-positive rate. The recall is 0.7615, which means that the model accurately recognized 76.15% of the actual positive events. The F1 score is 0.7475, showing a reasonable mix of precision and recall. The categorization report presents data for both classes and also gives weighted averages that account for class distribution imbalances. The decision tree model, which provides a graphical depiction of the tree-based classifier's decision-making process, was plotted. It specifies the plot proportions to guarantee that the decision tree visualization is clear and understandable. The AUC value of 0.82 indicates that the model has good discriminating power, with a greater probability of determining higher scores for positive cases than adverse instances. The ROC curve would be considered appropriately good. This shows that the model can discriminate between positive and negative cases, with a low false positive rate and a high true positive rate.

Bi-LSTM

The model successfully categorized 53.36% of the occurrences. The accuracy is 0.6131, indicating the model has a low false-positive rate. In this case, the recall is 0.5336, suggesting that the model successfully detected 53.36% of the actual positive events. The F1 score is 0.3904, offering that the overall performance in balancing accuracy and recall could be better. The ROC curve would not be regarded as successful in discriminating between positive and negative cases since the AUC is 0.51. The TPR and FPR of the model are comparable in this case, showing that the model is unable to categorize positive occurrences while minimizing false positives efficiently. The AUC of 0.51 indicates the model's discriminating power is relatively poor.

For each class, the resultant confusion matrix plot depicts the distribution.

By comparing and analyzing each model's accuracy, classification report, and AUC-ROC values, logistic regression outperformed the other two classifiers.

ETHICAL IMPLICATIONS

AI systems rely on training data to recognize patterns and make predictions. If the input data used for training the models are biased or contain misinformation, biased predictions may follow. Fraudulent news detection algorithms may falsely identify or suppress genuine content that has been tagged as fraudulent. This could violate free speech and expression, restricting people's freedom to disseminate information and views. These algorithms typically evaluate massive amounts of data to detect fake news effectively. Users' information may be collected, stored, and even used without their explicit consent, posing privacy concerns. Many AI models' fundamental workings, particularly sophisticated models like bidirectional LSTM, can be opaque and difficult to comprehend. Transparency may diminish trust in technology and raise concerns about accountability. Fake news sources might use it to target model faults.

CONCLUSIONS

To summarise, false news identification is an essential application of AI approaches that can aid in mitigating the spread of disinformation and protecting individuals and society from its potential consequences. However, using AI algorithms for false news identification raises ethical concerns and possible effects that must be carefully considered. Building and implementing AI approaches for false news detection that are fair, trustworthy, and contribute to a healthy information environment if we get acknowledged and address the ethical issues.

REFERENCES

Aborisade, O., & Anwar, M. (2018, July). Classification for authorship of tweets by comparing logistic regression and naive Bayes classifiers. In *2018 IEEE International Conference on Information Reuse and Integration (IRI)* (pp. 269-276). IEEE.

Bahad, P., Saxena, P., & Kamal, R. (2019). Fake news detection using bi-directional LSTM-recurrent neural network. *Procedia Computer Science*, *165*, 74-82.

Blustein, J., Herder, E., Rubart, J., & Ashman, H. (2016). 27th ACM International Conference on Hypertext and Social Media. *ACM SIGWEB Newsletter*, (Winter), 1-4.

Dinesh, T., & Rajendran, T. (2021). The higher classification of fake political news using decision tree algorithm over naive Bayes algorithm. *Revista Geintec-Gestao Inovacao E Tecnologias*, *11*(2), 1084-1096.

Gottfried, J., & Shearer, E. (2016). News use across social media platforms 2016.

Granik, M., & Mesyura, V. (2017, May). Fake news detection using naive Bayes classifier. In *2017 IEEE first Ukraine conference on electrical and computer engineering (UKRCON)* (pp. 900-903). IEEE.

Oshikawa, R., Qian, J., & Wang, W. Y. (2018). A survey on natural language processing for processing for fake news detection. *arXiv preprint arXiv:1811.00770*.