جامعة أم القرى

UMM AL-QURA UNIVERSITY

Fake News Detection

TEXT ANALYSIS

Data analysis

By:

Rawan Alharthi	444001029
Mashael Abdali	444001062
Shaimaa Alghamdi	444000746

Objective

The objective is to build a reliable model to accurately classify news articles as fake or real. Below, the steps taken for text preprocessing, model performance evaluation, and insights drawn from the analysis are detailed.

Data Processing

Loading the Dataset:

The dataset was loaded from the (Fake-News) dataset.

Cleaning Data:

Unnecessary columns were removed after merging relevant text into a single column to simplify the data and improve analysis efficiency.

Missing values in the dataset were filled with empty strings, effectively dealing with the absence of data and allowing consistent text processing.

```
id           0
title      558
author    1957
text        39
label        0
dtype: int64
label
1    10413
0    10387
Name: count, dtype: int64
```

```
title     0
author    0
text      0
label     0
dtype: int64
label
1    10413
0    10387
Name: count, dtype: int64
```

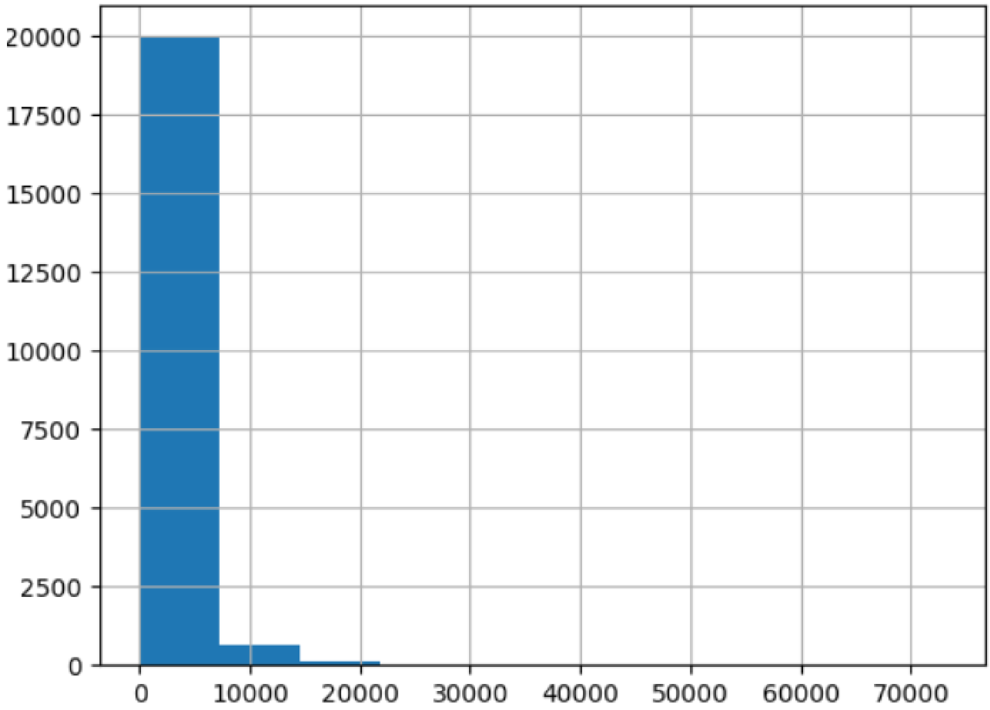
Text Preprocessing

The text data underwent several preprocessing steps to ensure it was suitable for classification.

- Stemming: is the process of reducing a word to its root or base form.
- Stop words: are common words (like "the," "is," "and") that are often removed from text because they do not add significant meaning during text analysis.
- Remove URLs: The function deletes any URLs starting with "http," "www," or "https" from the text.
- Remove non-alphanumeric characters: It filters out all characters except letters and spaces.
- Convert to lowercase: The text is converted to lowercase to standardize it.
- Tokenize: The text is split into individual words (tokens).

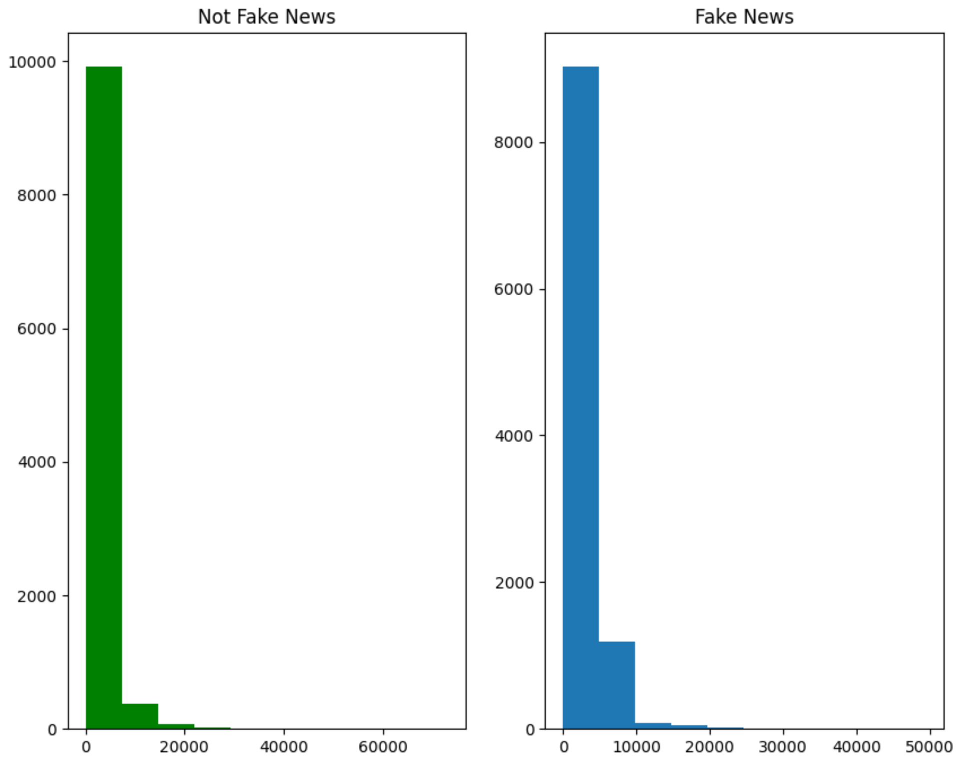
Length of Text:

This calculates the length of each text entry in the (news) column.



Length Fake vs. Not Fake:

It creates two subplots displaying the distribution of text lengths in the dataset based on their classification (fake and non-fake).



Word Cloud

Creates a word cloud representing the (fake and Not fake) texts from the dataset. The most frequently used words appear larger, helping to understand the common topics or words frequently used in positive news.

Fake News:

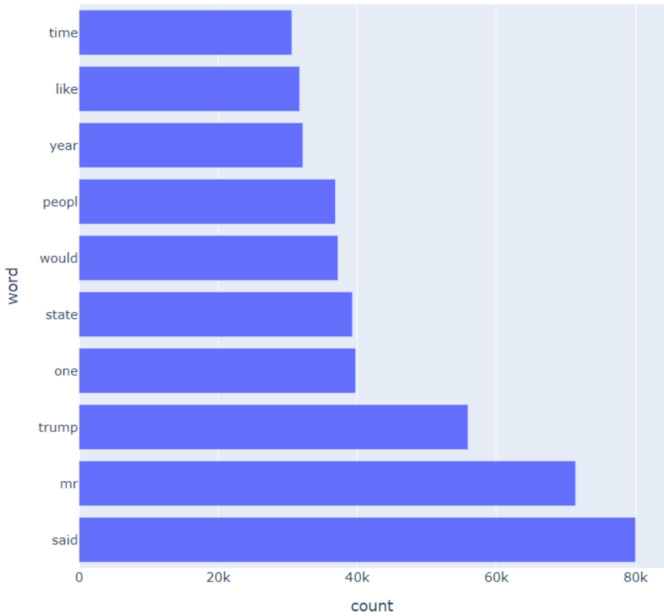
Not Fake News:

10 most common words:

Counts the frequency of each word in the news column and then displays the top 10 most common words in the form of a DataFrame.

```
[95] # Top 10 Most Common Words in News Texts
     cnt = Counter()
     for text in df["news"].values:
         for word in text.split():
             cnt[word] += 1
     cnt.most_common(10)
     temp = pd.DataFrame(cnt.most_common(10))
     temp.columns=['word', 'count']
     temp
```

	word	count
0	said	80012
1	mr	71401
2	trump	55932
3	one	39784
4	state	39322
5	would	37237
6	peopl	36882
7	year	32196
8	like	31721
9	time	30606



TF-IDF

We creates a (TF-IDF) matrix from the texts in the DataFrame, where it analyzes the texts to extract important words and transforms them into a numerical representation that can be used in machine learning models.

- **20800: Represents the number of rows in the matrix.**
- **133559: Represents the number of columns in the matrix.**
- **5151357: This number represents the total number of stored elements in the matrix (that are not zero). It means that there are 5,151,357 values representing the importance of words in the texts.**

```
[17] # TF-IDF vectors
     x=df['news'].values
     y=df['label'].values

     word_vector=TfidfVectorizer()
     x=word_vector.fit_transform(x)

[18] x

<20800x133559 sparse matrix of type '<class 'numpy.float64'>'
        with 5151357 stored elements in Compressed Sparse Row format>
```

Complement Naive Bayes

Model Accuracy:

- **The Complement Naive Bayes model achieved an accuracy of 86.08%, indicating that it successfully classified the majority of the data correctly.**

Confusion Matrix:

- **The confusion matrix shows that the model misclassified 20 texts as "fake" and 559 texts as "not fake," indicating that the model performed better at classifying "not fake" texts (label 0) compared to "fake" texts (label 1).**

```
[21] # Complement Naive Bayes
     CNB = ComplementNB()
     CNB.fit(x_train, y_train)

     predicted = CNB.predict(x_test)
     accuracy_score = metrics.accuracy_score(predicted, y_test)

     print('ComplementNB model accuracy is',str('{:04.2f}'.format(accuracy_score*100))+'%')
     print('------------------------------------------------')
     print('Confusion Matrix:')
     print(pd.DataFrame(confusion_matrix(y_test, predicted)))
     print('------------------------------------------------')
     print('Classification Report:')
     print(classification_report(y_test, predicted))
```

```
ComplementNB model accuracy is 86.08%
------------------------------------------------
Confusion Matrix:
      0     1
0  2112    20
1   559  1469
------------------------------------------------
Classification Report:
              precision    recall  f1-score   support

           0       0.79      0.99      0.88      2132
           1       0.99      0.72      0.84      2028

    accuracy                           0.86      4160
   macro avg       0.89      0.86      0.86      4160
weighted avg       0.89      0.86      0.86      4160
```

Logistic Regression:

The logistic regression model achieved a high accuracy of (95%) in classifying the data. The classification report shows that the model has balanced performance between the two classes, with precision and recall for "not fake" (class 0) and "fake" (class 1) around 95%.

```
[22] # Logistic Regression
     model = LogisticRegression()
     model.fit(x_train, y_train)

     # Make predictions
     y_pred = model.predict(x_test)

     # Evaluate our model
     print(f'Accuracy: {accuracy_score(y_test, y_pred)}')
     print(classification_report(y_test, y_pred))
```

```
Accuracy: 0.9526442307692308
              precision    recall  f1-score   support

           0       0.96      0.95      0.95      2132
           1       0.95      0.96      0.95      2028

    accuracy                           0.95      4160
   macro avg       0.95      0.95      0.95      4160
weighted avg       0.95      0.95      0.95      4160
```

Conclusion

- **Logistic Regression was effective in handling high-dimensional, sparse data like TF-IDF matrices, while Complement Naive Bayes provided an alternative approach, particularly suited for handling imbalanced datasets. Both models performed reasonably well.**