Big Data Analysis and Project

Assignment 1 – Part C

To analyze the impact of the figure of speech, sentiment, and social media on the detection and propagation of fake news over social media.

In the earlier stage of the project, we gathered data to analyze the effects of figures of speech, mood, and social media on the identification and spread of false information online. At the end of phase B, we effectively differentiated between fake and genuine news. Now in part C, we aim to refine further our model developed earlier in part B, as well as Interpret the results of your models and how well they performed.

Part 1- Problem Description

The project examines how figures of speech, sentiment, and social media impact fake news detection and propagation. It uses two datasets: tweets and fake/true news. Preprocessing involves converting news data to word vectors and cleaning tweets by removing emojis, URLs, etc. Two models, NB, and KNN are trained and evaluated for news classification. The results show that machine learning algorithms effectively distinguish between fake and genuine news. The analysis underscores ML's potential in detecting fake news, but more research is needed to explore the influence of tweets on fake news detection. [2]

Part 2- Data Pre-processing

We prepared the news dataset for machine learning by preprocessing, imputing, removing features, and scaling the data. The Spacy en_core_web_lg model created word embeddings for the text data, which were then scaled with MinMaxScaler for uniform feature scaling. The text from news articles was converted to lists using the to_list() function, combined, and tokenized with the Tokenizer() function. Tokenized sequences helped calculate article lengths and generate a Kernel Density Estimate (KDE) plot, comparing article length distribution between real and fake articles. These steps enable the numerical representation of text data, handle missing values, and ensure uniform feature scales for machine learning algorithms. [4][3]

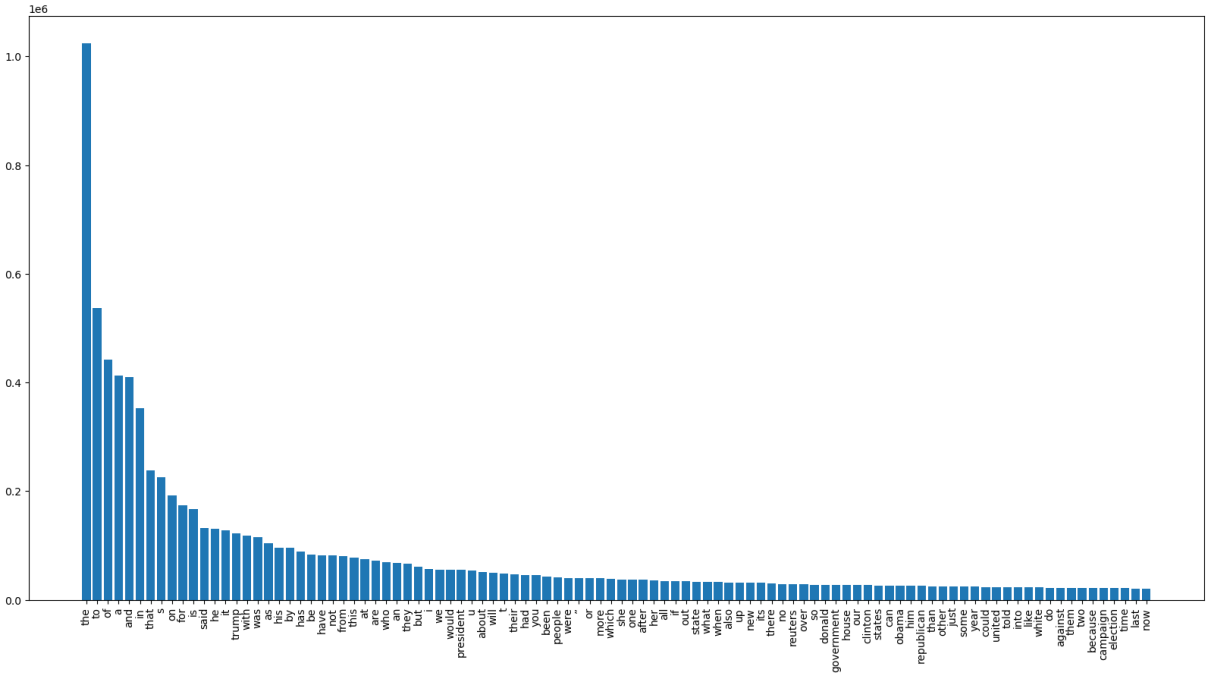

Figure 1- The graph illustrates the top 100 most frequently occurring words in a text news article, presented along with their respective frequencies.

The resulting tokenized sequences were used to calculate the length of each article and plot a distribution of article lengths as shown below:

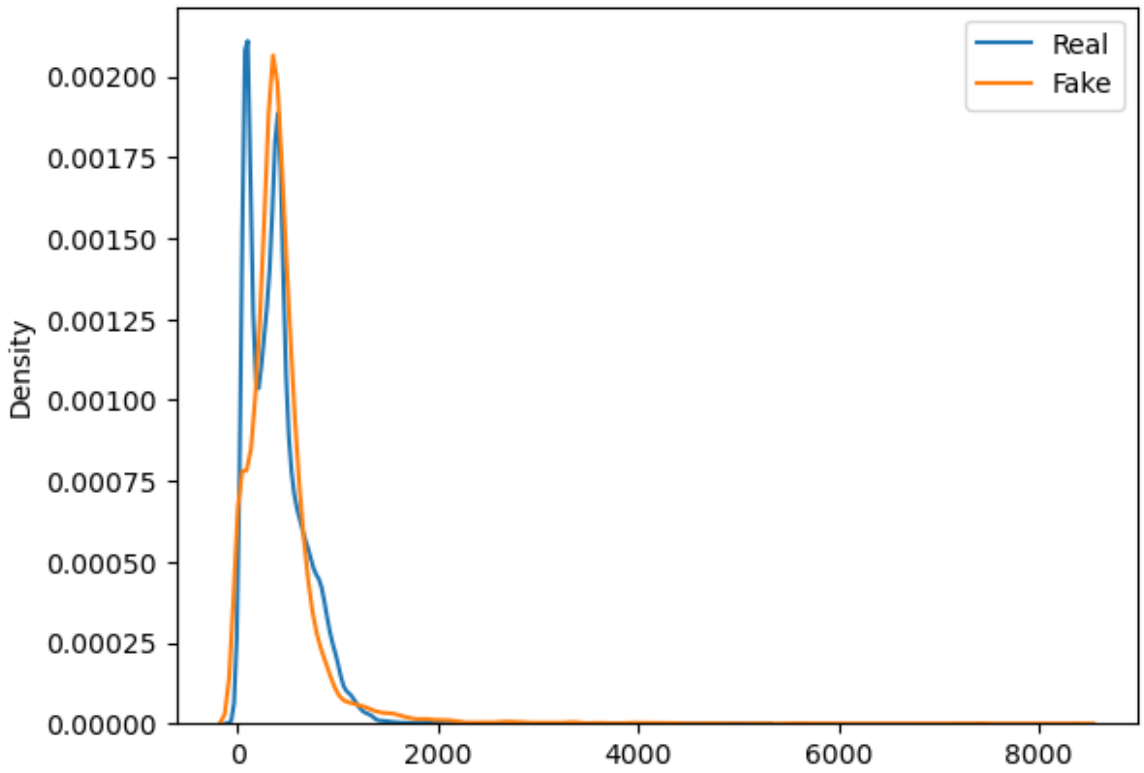

Figure 2-The graph visualizes the distribution of article lengths for real and fake articles using Kernel Density Estimate (KDE) plots.

```python
lengths = [len(item) for item in tokenized]

max_length = max(lengths)
min_length = min(lengths)

max_length_index = lengths.index(max_length)
min_length_index = lengths.index(min_length)

true_tokenized = tokenizer.texts_to_sequences(true_titles)

fake_tokenized = tokenizer.texts_to_sequences(fake_titles)

true_lengths = [len(item) for item in true_tokenized]

fake_lengths = [len(item) for item in fake_tokenized]

print("Distribution of article lengths")

#sns.kdeplot(lengths, label='Overall')
sns.kdeplot(true_lengths, label='Real')
sns.kdeplot(fake_lengths, label='Fake')

plt.legend()
plt.show()
```

Figure 3- Code used for generating a plot of lengths of fake vs true news.

This code generated a KDE plot to compare the distribution of article lengths between real and fake articles. It visualizes the probability density function of continuous random variables, providing insights into potential differences or similarities in length patterns between the two categories. [1]

Part 3- Model Selection

Two machine learning models, NB and KNN were chosen for text classification tasks, particularly distinguishing fake, and genuine news articles. Implementing pipelines allows efficient chaining of preprocessing and modelling steps, demonstrating a thoughtful approach to detecting and propagating fake news on social media.

```
print(classification_report(y_test, clf1.predict(X_test)))
[24]  ✓ 0.0s
...                  precision    recall  f1-score   support

               0       0.98      0.98      0.98      7136
               1       0.98      0.97      0.98      6334

        accuracy                           0.98     13470
       macro avg       0.98      0.98      0.98     13470
    weighted avg       0.98      0.98      0.98     13470
```

Figure 4- The Classification report for the fake and true news data sets. (Note- 0 stands for the fake news while 1 stands for the true news dataset.)

Part 4- Model Refinement

In this project, we refined models to classify fake and true news using Multinomial NB and KNN algorithms. We utilized the Spacy en_core_web_lg model for word embeddings and applied MinMaxScaler to scale features. The refinement process included hyperparameter tuning with GridSearchCV for the best alpha value of Multinomial NB. To ensure fair testing,

we employed 5-fold cross-validation. The results were highly satisfactory, achieving excellent performance in classifying news articles based on titles. This approach effectively differentiated between fake and genuine news, showcasing the refined models' potential to detect fake news accurately.[5]

```
# Print best hyperparameters and results
print("Best Hyperparameters:", grid_search.best_params_)
print("Best Cross-Validation Score:", grid_search.best_score_)

# Evaluate the model with best hyperparameters on the test set
best_model = grid_search.best_estimator_
print("Classification Report for Test Set:")
print(classification_report(y_test, best_model.predict(X_test)))
```

✓ 4m 39.6s

```
Best Hyperparameters: {'multinomialnb__alpha': 0.1}
Best Cross-Validation Score: 0.9773132045021313
Classification Report for Test Set:
              precision    recall  f1-score   support

           0       0.98      0.98      0.98      7136
           1       0.98      0.97      0.98      6334

    accuracy                           0.98     13470
   macro avg       0.98      0.98      0.98     13470
weighted avg       0.98      0.98      0.98     13470
```

Figure 5- A final overview of precision and recall of our model, after refinement. (Performed on the test set)

Part 5- Performance Description

In this project, we select performance measures to evaluate model effectiveness. The classification report function calculates essential metrics like precision, recall, etc. These measures assess accuracy, the models' ability to distinguish fake and genuine news, and the balance between precision and recall. Comparing Multinomial NB and KNN models using these metrics provides valuable insights into their strengths and weaknesses.

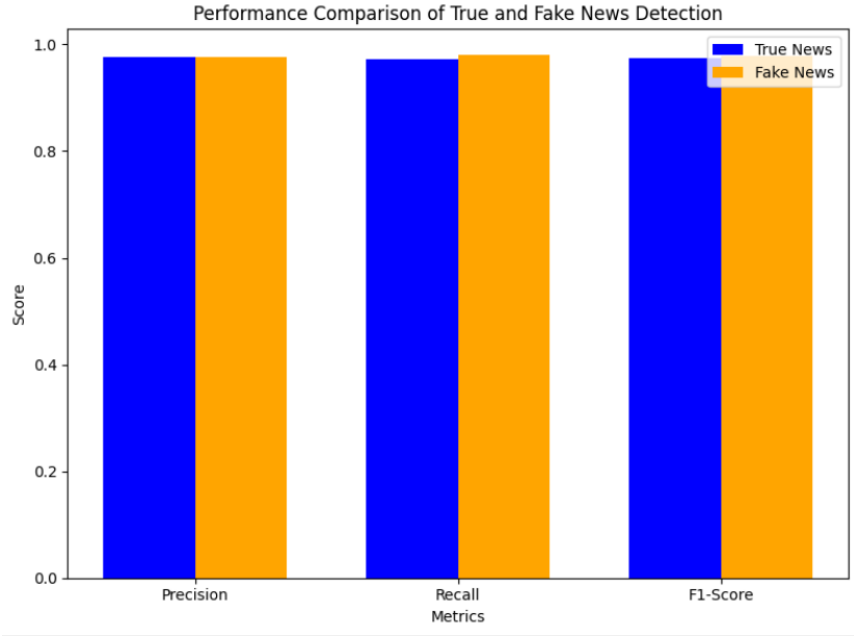

Figure 6- The grouped bar chart compares the precision, recall, and F1-score of the best model for detecting true and fake news.

Part 6- Results Interpretation

With all the modelling completed, we were able to populate our confusion matrix, which turned out like below:

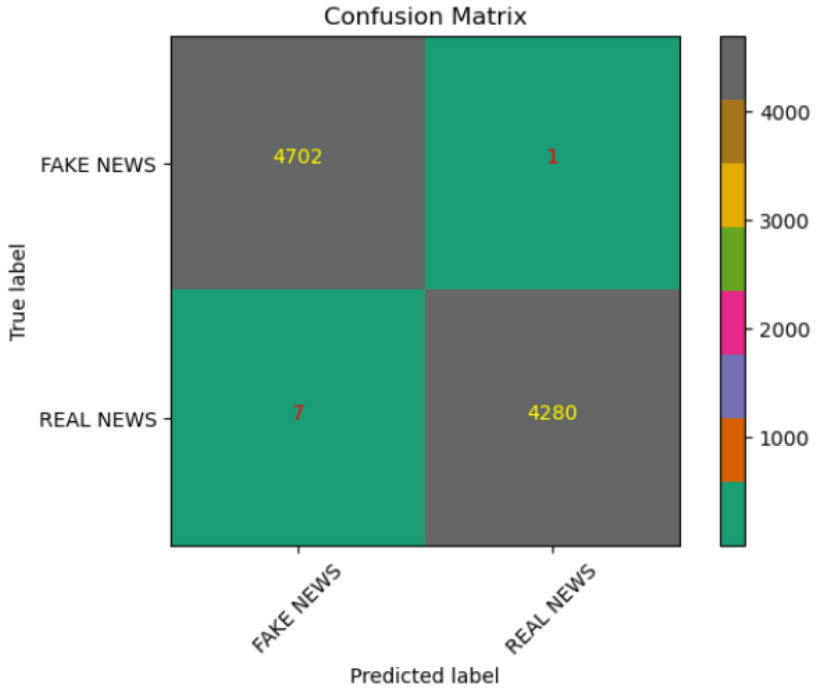

Figure 7- Final confusion matrix of distinguishing fake and real news.

The confusion matrix summarized model performance for fake and true news, explaining model selection, hyperparameters, and metrics. The accuracy of the chosen model and the confusion matrix visualizations demonstrated actual versus predicted labels. The analysis showcased the model's strengths and weaknesses, well-supported by evaluation metrics and visualization, indicating successful discrimination between fake and true news.

Addressing the questions asked in the previous phases:

Question 1 - How can we effectively detect and deal with the proliferation of fake news on social media platforms?

NLP and machine learning are utilized to detect and handle fake news on social media. News articles are transformed into word vectors and classified with NB and KNN. Refinement, like hyperparameter tuning, improves model performance against fake news proliferation.

Question 2- Can machine learning algorithms effectively differentiate between fake news and genuine news articles before they get published to the public?

Yes, machine learning algorithms can effectively differentiate between fake and genuine news pre-publication. Utilizing word embeddings and classifiers like NB and KNN achieves

promising precision, recall, and F1-scores. Additional research is needed, but this demonstrates ML's potential in pre-publication fake news detection.

References:

1- Pydata.org. (2012). *seaborn.kdeplot — seaborn 0.9.0 documentation*. [online] Available at: https://seaborn.pydata.org/generated/seaborn.kdeplot.html.

2- Bostonia. (n.d.). *Fake News Influences Real News*. [online] Available at: https://www.bu.edu/bostonia/2017/fake-news-influences-real-news/ [Accessed 25 July 2023].

3- (Singh, 2019) www.kaggle.com. (n.d.). *News Headlines Dataset for Sarcasm Detection*. [online] Available at: https://www.kaggle.com/datasets/rmisra/news-headlines-dataset-for-sarcasm-detection.

4- Singh, S. (2019). *What is Tokenization | Methods to Perform Tokenization*. [online] Analytics Vidhya. Available at: https://www.analyticsvidhya.com/blog/2019/07/how-get-started-nlp-6-unique-ways-perform-tokenization/.

5- Rosebrock, A. (2021). *Introduction to hyperparameter tuning with scikit-learn and Python*. [online] PyImageSearch. Available at: https://pyimagesearch.com/2021/05/17/introduction-to-hyperparameter-tuning-with-scikit-learn-and-python/.