Project Title: Fake News Detection

Project Members: Saurabh Chaudhary, Charan Raju M.

Cs-675 Data Science, Fall 2023

**INTRODUCTION:**

Fake news is a widespread issue in the news industry and has become a global problem. In the United States, the term and concept grew in popularity during the 2016 election but has since manifested itself in areas outside the realm of politics. A recent example of this is the COVID-19 pandemic – almost 80 percent of consumers in the United States reported having seen fake news on the coronavirus outbreak, highlighting the extent of the issue and the reach fake news can achieve. The Fake News Prediction model aims to leverage the power of ML algorithms & data science to discern between genuine and deceptive information, offering a proactive approach to mitigate the impact of misinformation. By analyzing textual features within news articles, headlines, or social media content, the model can learn patterns and associations indicative of false narratives. This introduction sets the stage for understanding the fundamental purpose of a Fake News Prediction model. As we delve into the intricacies of data collection, preprocessing, feature engineering, and model selection, the documentation provides a comprehensive guide to building a robust and effective tool in the fight against the dissemination of fake news. The following sections will detail the step-by-step process, emphasizing the significance of each stage in creating a reliable and scalable solution to identify and combat misinformation.

Objective:

The primary objective is to create a machine learning model that can analyze textual information and accurately classify news articles as real or fake. The model aims to contribute to efforts in combating the spread of misinformation and promoting media integrity.

**DATA COLLECTION:**

We took the datasets from **https://www.kaggle.com/datasets/clmentbisaillon/fake-and-real-news-dataset. T**t is an open-source website for data sets. We have taken two data sets of actual and fake news. We classified the text classification by numeric. 0 for real news and 1 for fake news. By adding both the datasets into one which has 44898 rows and 6 columns.

**DATA PROCESSING:**

To get the effective machine learning analysis we have cleaned the datasets by removing the irrelevant characters, punctuation, and 'stop words' from the text. We Create a method called "clean\_and\_lower" to Clean and process the textual data by removing irrelevant characters (Special Characters & Numbers, White spaces), stop words using nltk.corpus library, and performing tasks such as stemming with the help of nltk.stem.porter library.

A screenshot of a computer screen

Description automatically generated

For Recurrent Neural Network we converted the text and sentences into tokens using Tokenizer from tensorflow.keras.preprocessing.text . For consistent input size we used pad sequences from tensorflow.keras.preprocessing.sequence

A screenshot of a computer screen

Description automatically generated

**In our approach, we leveraged the content column values as our Feature matrix X and the Classification column values as our Target Vector Y. To ensure a robust evaluation of our model, we employed the train\_test\_split method from the sklearn.model\_selection library to split our datasets into training and testing datasets. By utilizing the stratify parameter in the train\_test\_split method, we ensured an equal distribution of the Target Vector, contributing to a balanced representation in both training and testing sets.**

**To prepare our Feature matrix for machine learning algorithms, we utilized the TfidfVectorizer from the sklearn.feature\_extraction.text module. This transformation converted the textual content into numerical vectors, a format suitable for training and evaluating machine learning models. This vectorization process using TfidfVectorizer is crucial for capturing the importance of terms within the content, allowing our model to learn and generalize effectively from the textual data.**

**MODEL SELECTION, TRAINING & EVALUATION:**

For the model selection, training, and evaluation phase, we opted for a diverse set of three machine learning models. Firstly, we chose Logistic Regression from the sklearn.linear\_model module, a versatile and widely used algorithm for binary classification tasks. Secondly, we employed DecisionTreeClassifier from the sklearn.tree module, which can capture complex relationships in the data through a tree-like structure. Lastly, we introduced a Recurrent Neural Network (RNN) from the tensorflow.keras.models module, showcasing the utilization of deep learning for sequence-based data.

This ensemble of models was selected to explore a range of techniques, from traditional linear models to tree-based approaches and the more sophisticated capabilities offered by neural networks. Each model was trained on our preprocessed datasets and evaluated for its performance in capturing the underlying patterns in the content for classification purposes. This diverse approach enables us to assess the strengths and weaknesses of each model type and choose the one that best suits the characteristics of our specific task.

Logistic Regression vs Decision tree.

A screenshot of a computer

Description automatically generated

Recurrent Neural Network (RNN)

A screenshot of a computer

Description automatically generated

Challenges:

The project encountered several challenges primarily associated with the size of the dataset. The data processing phase proved to be time-consuming, particularly during tasks such as removing English language stop words, handling special characters, and implementing stemming. These preprocessing steps are crucial for refining the textual data and preparing it for effective model training.

Another challenge revolved around the selection of parameters for the Neural Network (NN) model, specifically the decision-making process for choosing the appropriate number of Long Short-Term Memory (LSTM) layers. This decision is critical as it directly impacts the model's ability to capture sequential dependencies in the data, influencing the overall performance in sentiment analysis.

Additionally, determining the optimal embedding dimensions for the NN model posed a challenge. The choice of embedding dimensions is pivotal as it defines the vector space in which words are represented, influencing the model's capacity to learn nuanced relationships within the textual content. These challenges underscore the intricacies involved in handling large datasets and fine-tuning neural network architectures for optimal performance in sentiment analysis tasks.

Conclusion:

In conclusion, the decision tree model emerged as the top-performing model, surpassing both the RNN and logistic regression models with an impressive accuracy rate of 99%. This outcome underscores the effectiveness of decision trees in capturing the underlying patterns within the dataset and making accurate predictions for the sentiment analysis task. Moreover, it highlights the importance of considering the computational resources required by different models. Despite the sophistication of Recurrent Neural Networks (RNNs), the decision tree's simplicity and efficiency proved advantageous in achieving superior accuracy for this specific task. The acknowledgement that employing RNNs might be considered overkill in situations where substantial computational resources are demanded provides valuable insights into the practical considerations involved in choosing the appropriate model for sentiment analysis tasks.