**FAKE NEWS DETECTION USING NPL**

**Methods** :

1. \*\*Dataset Collection\*\*:

Start by acquiring a dataset of news articles, where some are labeled as “fake” and others as “real.” Common datasets for this task include the “Fake News Dataset” or “LIAR-PLUS” dataset. You may also need a dataset for pre-trained word embeddings (like GloVe or Word2Vec) to enhance your NLP model.

2. \*\*Data Preprocessing\*\*:

- Clean the text data by removing any irrelevant characters, punctuation, and numbers.

- Tokenize the text into words or subword units.

- Remove stop words, which are common words that do not carry significant information.

- Apply stemming or lemmatization to reduce words to their base form.

3. \*\*Feature Extraction\*\*:

- Convert the text data into numerical vectors. You can use techniques like TF-IDF (Term Frequency-Inverse Document Frequency) or Word Embeddings (e.g., Word2Vec, GloVe).

4. \*\*Split the Dataset\*\*:

- Divide your dataset into training and testing sets to evaluate your model’s performance.

5. \*\*Model Selection\*\*:

- Choose an NLP model for fake news detection. Common choices include:

- Recurrent Neural Networks (RNNs)

- Convolutional Neural Networks (CNNs)

- Transformer-based models (e.g., BERT, GPT) if you have a large dataset.

6. \*\*Model Training\*\*:

- Train your chosen model on the training data. You’ll use the labeled data to teach the model to distinguish between fake and real news.

7. \*\*Model Evaluation\*\*:

- Use the testing dataset to evaluate the model’s performance. Common evaluation metrics include accuracy, precision, recall, F1-score, and ROC-AUC.

8. \*\*Fine-Tuning\*\*:

- Depending on the model’s performance, you may need to fine-tune hyperparameters or consider different NLP architectures.

9. \*\*Deployment\*\*:

- Once you’re satisfied with your model’s performance, you can deploy it to make predictions on new news articles.

10. \*\*Continual Improvement\*\*:

- Stay up-to-date with the latest NLP techniques and continue improving your model as new data becomes available.

**Example**

In this example, you’ll need to replace ‘fake\_news\_dataset.csv’ with the path to your dataset file. Make sure your dataset contains two columns: ‘text’ for news articles and ‘label’ for their corresponding labels (real or fake).

This code loads the dataset, preprocesses the text data, converts it to a Bag of Words representation, trains a Multinomial Naïve Bayes classifier, makes predictions, and evaluates the model’s performance.

Please note that this is a basic example. To improve accuracy, you can explore more advanced NLP techniques, use larger datasets, and experiment with different machine learning algorithms, such as logistic regression, random forests, or even deep learning models like BERT. Additionally, you should consider ethical aspects and bias mitigation techniques in real-world applications.

**Program**

# Import necessary libraries

**Import pandas as pd**

**From sklearn.model\_selection import train\_test\_split**

**From sklearn.feature\_extraction.text import CountVectorizer**

**From sklearn.naive\_bayes import MultinomialNB**

**From sklearn.metrics import accuracy\_score, classification\_report**

**# Load your fake news dataset**

**Data = pd.read\_csv(‘fake\_news\_dataset.csv’) # Replace ‘fake\_news\_dataset.csv’ with your dataset file**

**# Data preprocessing**

**X = data[‘text’] # ‘text’ is the column containing news articles**

**Y = data[‘label’] # ‘label’ is the column containing labels (real or fake)**

**# Split the data into training and testing sets**

**X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)**

**# Text vectorization using Bag of Words**

**Vectorizer = CountVectorizer()**

**X\_train\_bow = vectorizer.fit\_transform(X\_train)**

**X\_test\_bow = vectorizer.transform(X\_test)**

**# Train a Multinomial Naïve Bayes classifier**

**Classifier = MultinomialNB()**

**Classifier.fit(X\_train\_bow, y\_train)**

**# Make predictions**

**Y\_pred = classifier.predict(X\_test\_bow)**

**# Evaluate the model**

**Accuracy = accuracy\_score(y\_test, y\_pred)**

**Print(f’Accuracy: {accuracy:.2f}’)**

**# Generate a classification report**

**Report = classification\_report(y\_test, y\_pred)**

**Print(report).**

**Name: K Hari Gokul**

**Reg no: 621521104053**