**AI\_Phase3**

**DEVELOPMENT PART 1**

-To load, preprocess the dataset and perform different analysis.

**1. Download the Dataset:**

   - Download the dataset from the provided Kaggle link.

<https://www.kaggle.com/datasets/clmentbisaillon/fake-and-real-news-dataset>

**2. Import Necessary Libraries:**

   - Use Python and import libraries such as pandas, numpy, and matplotlib for data manipulation and visualization.

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

**3. Load the Dataset:**

   - Read the dataset into a Pandas DataFrame.

# Assuming the file is downloaded as 'fake\_and\_real\_news.csv'

file\_path = 'path/to/fake\_and\_real\_news.csv'

data = pd.read\_csv(file\_path)

**4. Explore the Dataset:**

   - Check the first few rows of the dataset to understand its structure.

   - Examine column names and data types.

print(data.head())

print([data.info](http://data.info/" \t "_blank)())

**5. Handle Missing Values:**

   - Check for missing values and decide on an approach to handle them (drop or impute).

# Example: Drop rows with missing values

data = data.dropna()

# Or: Impute missing values

# data['column\_name'].fillna(value, inplace=True)

**6. Text Preprocessing:**

   - Clean and preprocess the text data using NLP techniques.

   - Convert text to lowercase, remove stop words, punctuation, etc.

import nltk

from nltk.corpus import stopwords

from nltk.tokenize import word\_tokenize

nltk.download('stopwords')

nltk.download('punkt')

stop\_words = set(stopwords.words('english'))

def preprocess\_text(text):

    # Convert to lowercase

    text = text.lower()

    # Remove punctuation, numbers, and special characters

    text = ''.join([char for char in text if char.isalnum() or char.isspace()])

    # Tokenize and remove stop words

    tokens = word\_tokenize(text)

    tokens = [word for word in tokens if word not in stop\_words]

    return ' '.join(tokens)

data['processed\_text'] = data['text'].apply(preprocess\_text)

**7. Save Preprocessed Data:**

   - Save the preprocessed data to a new CSV file.

data.to\_csv('preprocessed\_data.csv', index=False)

Before building your fake news detection model, it's essential to conduct some exploratory data analysis (EDA) to gain insights into the dataset.

**1.Class distribution:**

Check the distribution of genuine and fake news in the dataset. This helps in understanding the balance or imbalance between classes

class\_distribution = data['label'].value\_counts()

print(class\_distribution)

**2. Text Length Analysis:**

Analyze the distribution of text lengths in both genuine and fake news articles.

data['text\_length'] = data['text'].apply(len)

plt.figure(figsize=(10, 6))

plt.hist(data[data['label'] == 'fake']['text\_length'], bins=50, alpha=0.5, label='Fake')

plt.hist(data[data['label'] == 'real']['text\_length'], bins=50, alpha=0.5, label='Real')

plt.title('Text Length Distribution')

plt.xlabel('Text Length')

plt.ylabel('Frequency')

plt.legend()

plt.show()

**3.Word Frequency Analysis:**

Analyze the most common words in both genuine and fake news articles.

from wordcloud import WordCloud

def generate\_wordcloud(text):

wordcloud = WordCloud(width=800, height=400, random\_state=42, max\_words=150).generate(text)

plt.figure(figsize=(10, 6))

plt.imshow(wordcloud, interpolation='bilinear')

plt.axis('off')

plt.show()

# Generate word clouds for both classes

generate\_wordcloud(' '.join(data[data['label'] == 'fake']['processed\_text']))

generate\_wordcloud(' '.join(data[data['label'] == 'real']['processed\_text']))