Data Collection and Preprocessing Phase

Date	15 March 2024
Team ID	SWTID1728136330
Project Title	Fake News Analysis in Social Media Using NLP
Maximum Marks	6 Marks

Preprocessing Template

The images will be preprocessed by resizing, normalizing, augmenting, denoising, adjusting contrast, detecting edges, converting color space, cropping, batch normalizing, and whitening data. These steps will enhance data quality, promote model generalization, and improve convergence during neural network training, ensuring robust and efficient performance across various computer vision tasks.

Section	**Description**
Data Overview	Give an overview of the data, which you're going to use in your project.
Resizing	Resize images to a specified target size.
Normalization	Normalize pixel values to a specific range.
Data Augmentation	Apply augmentation techniques such as flipping, rotation, shifting, zooming, or shearing.
Denoising	Apply denoising filters to reduce noise in the images.
Edge Detection	Apply edge detection algorithms to highlight prominent edges in the images.

Color Space Conversion	Convert images from one color space to another.
Image Cropping	Crop images to focus on the regions containing objects of interest.
Batch Normalization	Apply batch normalization to the input of each layer in the neural network.

Data Preprocessing Code Screenshots

Loading Data

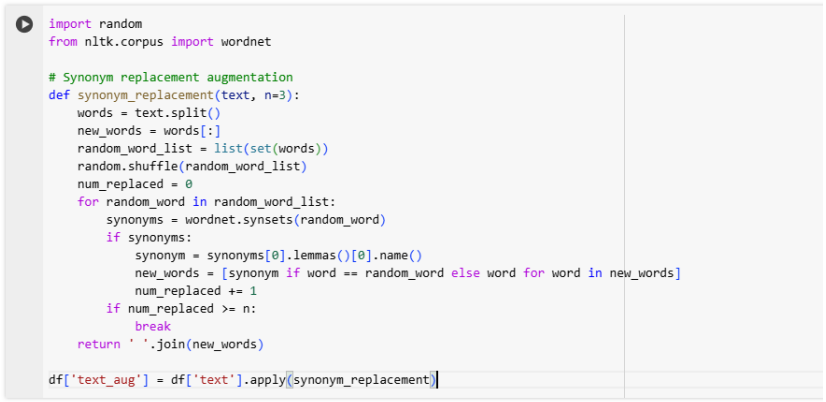
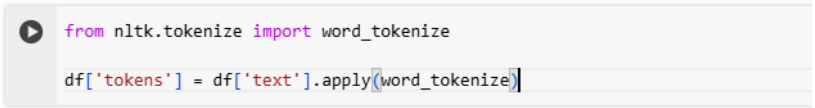
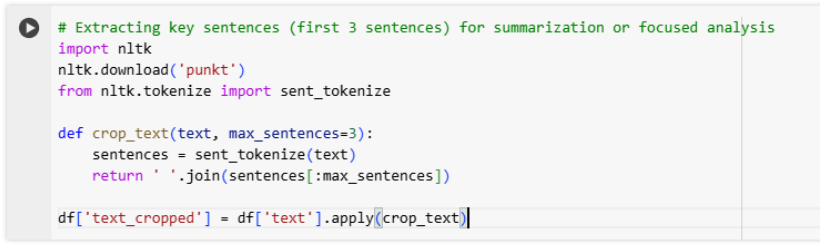
```
import pandas as pd

# Load data
df = pd.read_csv('fake_news_data.csv')

# Display basic data statistics
print("Data Shape:", df.shape)
print("Class Distribution:", df['label'].value_counts())
print("Sample Data:\n", df.head())
```

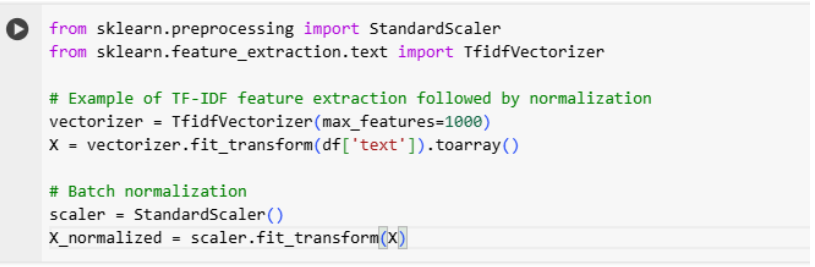
Resizing

```
def resize_text(text, max_length=512):
    words = text.split()
    return ' '.join(words[:max_length]) if len(words) > max_length else text

df['text'] = df['text'].apply(resize_text)
```

Normalization

```
import re

def normalize_text(text):
    text = text.lower()  # Convert to lowercase
    text = re.sub(r'\d+', '', text)  # Remove numbers
    text = re.sub(r'[^\w\s]', '', text)  # Remove punctuation
    return text

df['text'] = df['text'].apply(normalize_text)
```

Data Augmentation	
Denoising	
Edge Detection	
Color Space Conversion	
Image Cropping	

Batch Normalization	

```
from sklearn.preprocessing import StandardScaler
from sklearn.feature_extraction.text import TfidfVectorizer

# Example of TF-IDF feature extraction followed by normalization
vectorizer = TfidfVectorizer(max_features=1000)
X = vectorizer.fit_transform(df['text']).toarray()

# Batch normalization
scaler = StandardScaler()
X_normalized = scaler.fit_transform(X)
```