#=== Packages used in this research ===#

import pandas as FNPDS

import seaborn as FNSBN

import matplotlib.pyplot as FNMTP

import warnings as FNWRN

FNWRN.filterwarnings("ignore")

from sklearn import preprocessing as FNPREP

#=== Import the Fake News Data using Pandas ===#

FN\_Classification = FNPDS.read\_csv('WELFake\_Dataset.csv')

FN\_Classification

FN\_Classification.info()  #=== Fake News Data Types and their information ===#

#=== Data Cleaning Process of handling Missed Values and Duplicates ===#

print("Number of Missed Values in the Fake News = ",FN\_Classification.isnull().values.sum())

print(FN\_Classification.isnull().sum())

print("Number of Duplicate Rows in the Fake News = ",FN\_Classification.duplicated().values.sum())

#=== Drop the Unnamed, title column and missed values ===#

del FN\_Classification['title']

del FN\_Classification['Unnamed: 0']

FN\_Classification = FN\_Classification.dropna()

FN\_Classification.shape

FN\_Classification['label'].value\_counts()

#=== Visualizing Count Plot to check the amount of Real and Fake News ===#

FN\_Classification['label'].value\_counts().plot(kind='bar', color='rosybrown', edgecolor='brown')

FNMTP.xlabel('Real and Fake News')

FNMTP.ylabel('Count')

FNMTP.show()

#=== The above Count Plot proved the data is balanced and 1 indicates 'Real News', 0 indicates 'Fake News'.

#=== Text Preprocessing for Fake News Classification ===#

import regex as FNREGX  #=== used for String operations ===#

import nltk as EffPhishNL

EffPhishNL.download('stopwords')

from nltk.corpus import stopwords as FNSWRD

EFNW = FNSWRD.words('english') #=== store the english stopwords ===#

def FNews(impN):

    clnFN = impN.lower()  #=== convert the input text to lowercase ===#

    clnFN = FNREGX.sub("[^a-z]+", " ", clnFN)  #=== remove non-alphabetic characters ===#

    clnFN = FNREGX.sub(r'http://www+', " ", clnFN) #=== remove the urls ===#

    clnFN = FNREGX.sub(r'www+', " ", clnFN)

    clnFN = " ".join([FNwrd for FNwrd in clnFN.split() if FNwrd not in EFNW])  #=== split the text into words ===#

    return clnFN  #=== return the cleaned text ===#

lstFN = []  #=== Empty list to store cleaned text ===#

for clnFN in FN\_Classification['text']:  #=== iterate ech text entry into 'text' column ===#

    lstFN.append(FNews(clnFN))

FN\_Classification['text'] = lstFN

#=== Save the preprocessed data and this data is used for next level of process ===#

FN\_Classification.to\_csv('FNews\_Classification.csv', index=False)

FN\_Classification