```
#      Install necessary libraries
!pip install -q nltk gradio

#      Import libraries
import zipfile, io, re, string
import pandas as pd
import numpy as np
import nltk
import gradio as gr

from google.colab import files
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import PassiveAggressiveClassifier
from sklearn.utils import resample

#      Download stopwords
nltk.download('stopwords')
from nltk.corpus import stopwords
stop_words = set(stopwords.words('english'))

#      File upload
uploaded = files.upload()  # Upload the zip file containing Fake.csv and True.csv

#      Extract uploaded zip
zip_file = list(uploaded.keys())[0]
with zipfile.ZipFile(io.BytesIO(uploaded[zip_file]), 'r') as zip_ref:
    zip_ref.extractall()

#      Load datasets
fake = pd.read_csv("Fake.csv")
real = pd.read_csv("True.csv")
fake['label'] = 0
real['label'] = 1

#      Combine and clean
df = pd.concat([fake, real]).reset_index(drop=True)
df['combined'] = df['title'] + " " + df['text']

def clean_text(text):
    text = text.lower()
    text = re.sub(r"http\S+|www.\S+", "", text)
    text = re.sub(r"<.*?>", "", text)
    text = re.sub(r"[%s]" % re.escape(string.punctuation), "", text)
    text = re.sub(r"\d+", "", text)
    text = re.sub(r"\s+", " ", text)
    return " ".join([word for word in text.split() if word not in stop_words])
```

```
df['clean_text'] = df['combined'].apply(clean_text)

#      Balance dataset
df_fake = df[df['label'] == 0]
df_real = df[df['label'] == 1]
df_real_downsampled = resample(df_real, replace=True, n_samples=len(df_fake),
random_state=42)
df_balanced = pd.concat([df_fake, df_real_downsampled]).sample(frac=1,
random_state=42)

#      Train/Test split
X = df_balanced['clean_text']
y = df_balanced['label']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

#      TF-IDF Vectorization
vectorizer = TfidfVectorizer(max_df=0.7, min_df=5, max_features=5000)
X_train_tfidf = vectorizer.fit_transform(X_train)
X_test_tfidf = vectorizer.transform(X_test)

#      Model training
model = PassiveAggressiveClassifier(max_iter=1000)
model.fit(X_train_tfidf, y_train)

#      Gradio prediction function
def predict_news(text):
    cleaned = clean_text(text)
    vec = vectorizer.transform([cleaned])
    pred = model.predict(vec)[0]
    return "      REAL News" if pred == 1 else "      FAKE News"

#      Launch Gradio interface
demo = gr.Interface(
    fn=predict_news,
    inputs=gr.Textbox(lines=10, placeholder="Paste news content here..."),
    outputs=gr.Label(label="Prediction"),
    title="      Fake News Detector",
    description="Paste a news article to check if it's REAL or FAKE using NLP + ML.",
)
```