```
from google.colab import files
uploaded = files.upload()
```

```
Choose Files No file chosen  Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving Fake.csv to Fake.csv
Saving True.csv to True.csv
```

```
# SECTION 0: SETUP
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import nltk
import string
import re
from nltk.corpus import stopwords, wordnet
from nltk.tokenize import sent_tokenize, word_tokenize
from nltk.stem import PorterStemmer, WordNetLemmatizer
from nltk import pos_tag, RegexpParser
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
from sklearn.metrics.pairwise import cosine_similarity
import spacy

# Downloads for NLTK
nltk.download('punkt')
nltk.download('averaged_perceptron_tagger')
nltk.download('stopwords')
nltk.download('wordnet')
nltk.download('punkt_tab') # Download the missing 'punkt_tab' data
nltk.download('tagsets') # Download tagsets for pos_tag
nltk.download('averaged_perceptron_tagger_eng') # Download the missing 'averaged_perceptron_tagger_eng' data

# Load spaCy model
!python -m spacy download en_core_web_sm
nlp = spacy.load("en_core_web_sm")
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
[nltk_data] Downloading package punkt_tab to /root/nltk_data...
[nltk_data]   Package punkt_tab is already up-to-date!
[nltk_data] Downloading package tagsets to /root/nltk_data...
[nltk_data]   Package tagsets is already up-to-date!
[nltk_data] Downloading package averaged_perceptron_tagger_eng to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger_eng.zip.
Collecting en-core-web-sm==3.8.0
  Using cached https://github.com/explosion/spacy-models/releases/download/en_core_web_sm-3.8.0/en_core_web_sm-3.8.0-py3-none-any.whl (1
✓ Download and installation successful
You can now load the package via spacy.load('en_core_web_sm')
⚠ Restart to reload dependencies
If you are in a Jupyter or Colab notebook, you may need to restart Python in
order to load all the package's dependencies. You can do this by selecting the
'Restart kernel' or 'Restart runtime' option.
```

```
# SECTION 1: LOAD DATASET
fake = pd.read_csv("Fake.csv")
true = pd.read_csv("True.csv")

fake['label'] = 0
true['label'] = 1

df = pd.concat([fake, true]).sample(frac=1, random_state=42).reset_index(drop=True)
df['content'] = df['title'] + " " + df['text']

print("Dataset shape:", df.shape)
df.head()
```

```
Dataset shape: (44898, 6)
```

	title	text	subject	date	label	content
0	Ben Stein Calls Out 9th Circuit Court: Committ...	21st Century Wire says Ben Stein, reputable pr...	US_News	February 13, 2017	0	Ben Stein Calls Out 9th Circuit Court: Committ...
1	Trump drops Steve Bannon from National Securit...	WASHINGTON (Reuters) - U.S. President Donald T...	politicsNews	April 5, 2017	1	Trump drops Steve Bannon from National Securit...
2	Puerto Rico expects U.S. to lift Jones Act shi...	(Reuters) - Puerto Rico Governor Ricardo Rosse...	politicsNews	September 27, 2017	1	Puerto Rico expects U.S. to lift Jones Act shi...
3	OOPS: Trump Just Accidentally Confirmed He Le...	On Monday, Donald Trump once again embarrassed...	News	May 22, 2017	0	OOPS: Trump Just Accidentally Confirmed He Le...
4	Donald Trump heads for Scotland to reopen a go...	GLASGOW, Scotland (Reuters) - Most U.S. presid...	politicsNews	June 24, 2016	1	Donald Trump heads for Scotland to reopen a go...

```
# SECTION 2: TOKENIZATION & NORMALIZATION
sample_text = df['content'][0]

# Sentence & word tokenization
sentences = sent_tokenize(sample_text)
words = word_tokenize(sample_text)

# Normalize
stop_words = set(stopwords.words('english'))
words_cleaned = [w.lower() for w in words if w.isalpha() and w.lower() not in stop_words]

print("Cleaned words:", words_cleaned[:10])
```

```
Cleaned words: ['ben', 'stein', 'calls', 'circuit', 'court', 'committed', 'coup', 'état', 'constitution', 'century']
```

```
# SECTION 3: STEMMING AND LEMMATIZATION
stemmer = PorterStemmer()
lemmatizer = WordNetLemmatizer()

stemmed = [stemmer.stem(w) for w in words_cleaned]
lemmatized = [lemmatizer.lemmatize(w) for w in words_cleaned]

print("Stemmed:", stemmed[:10])
print("Lemmatized:", lemmatized[:10])
```

```
Stemmed: ['ben', 'stein', 'call', 'circuit', 'court', 'commit', 'coup', 'état', 'constitut', 'centuri']
Lemmatized: ['ben', 'stein', 'call', 'circuit', 'court', 'committed', 'coup', 'état', 'constitution', 'century']
```

```
# SECTION 4: POS TAGGING & CHUNKING
tags = pos_tag(words_cleaned)

# Chunking for noun phrases
grammar = "NP: {<DT>?<JJ>*<NN>}"
chunk_parser = RegexpParser(grammar)
tree = chunk_parser.parse(tags)
tree.pretty_print()  # Tree in text form
```

```
calls/VBZ committed/VBD état/NNP says/VBZ also/RB hollywood/VBD fame/JJ appearing/VBG shows/NNS films/VBP made/VBD provocative/JJ statem
```

```
# SECTION 5: NAMED ENTITY RECOGNITION (NER)
doc = nlp(sample_text)
for ent in doc.ents:
    print(ent.text, "->", ent.label_)
```

```
Ben Stein Calls -> PERSON
9th Circuit Court -> ORG
the Constitution 21st Century Wire -> LAW
Ben Stein -> PERSON
Pepperdine University -> ORG
```

```
Hollywood -> GPE
Ferris Bueller -> PERSON
Jeanine Pirro -> PERSON
Trump s Executive -> PERSON
Stein -> PERSON
the 9th Circuit Court -> ORG
Washington -> GPE
Stein -> PERSON
Seattle -> GPE
the Executive Order -> ORG
21st Century -> DATE
```

```
# SECTION 6: WORDNET - SYNONYMS & DEFINITIONS
from nltk.corpus import wordnet as wn

syns = wn.synsets('news')
for s in syns[:3]:
    print(f"Definition: {s.definition()}")
    print(f"Synonyms: {[l.name() for l in s.lemmas()]}")
```

```
Definition: information about recent and important events
Synonyms: ['news', 'intelligence', 'tidings', 'word']
Definition: information reported in a newspaper or news magazine
Synonyms: ['news']
Definition: a program devoted to current events, often using interviews and commentary
Synonyms: ['news_program', 'news_show', 'news']
```

```
# SECTION 7: TF-IDF FEATURE EXTRACTION
tfidf = TfidfVectorizer(stop_words='english', max_features=1000)
tfidf_matrix = tfidf.fit_transform(df['content'])

# Top features
tfidf_scores = pd.Series(tfidf_matrix.sum(axis=0).A1, index=tfidf.get_feature_names_out())
top_tfidf = tfidf_scores.sort_values(ascending=False).head(10)
print(top_tfidf)
```

```
trump        4328.001867
said         2932.720491
president    1676.056139
clinton      1338.231280
obama        1332.439136
people       1315.318830
video        1200.399509
state        1197.823620
house        1170.183656
new          1064.276416
dtype: float64
```

```
# SECTION 8: DOCUMENT SIMILARITY
# Compute similarity between 2 random articles
doc1 = tfidf_matrix[0]
doc2 = tfidf_matrix[1]
cos_sim = cosine_similarity(doc1, doc2)
print("Cosine Similarity:", cos_sim[0][0])
```

```
Cosine Similarity: 0.08319180549267273
```

```
# SECTION 9: WORD FREQUENCY PLOT (Fake News)
fake_texts = fake['text']
cv = CountVectorizer(stop_words='english', max_features=20)
word_counts = cv.fit_transform(fake_texts)

sum_words = word_counts.sum(axis=0)
words_freq = [(word, sum_words[0, idx]) for word, idx in cv.vocabulary_.items()]
words_freq = sorted(words_freq, key=lambda x: x[1], reverse=True)

# Plotting
words, freqs = zip(*words_freq)
plt.figure(figsize=(10, 6))
plt.bar(words, freqs)
plt.xticks(rotation=45)
plt.title("Top 20 Words in Fake News")
plt.show()
```