```
# Core libraries
import numpy as np
import pandas as pd

# Text preprocessing
import re
import nltk
from nltk.corpus import stopwords, wordnet
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer

# Feature extraction and similarity
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
nltk.download('punkt')
nltk.download('stopwords')
nltk.download('wordnet')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
[nltk_data] Downloading package wordnet to /root/nltk_data...
True
```

```
documents = [
    # Sports
    "The football team won the championship match",
    "Cricket players trained hard for the tournament",
    "The athlete broke the world record",

    # Politics
    "The government announced a new policy",
    "The election results were declared today",
    "Parliament passed the new bill",

    # Health
    "Doctors recommend regular exercise for good health",
    "The patient received medical treatment",
    "Healthy diet improves immunity",

    # Technology
    "Artificial intelligence is transforming technology",
    "The smartphone uses advanced processors",
    "Cybersecurity is important in the digital age",

    # Science / Mixed
    "Vaccines help prevent diseases",
    "Machine learning improves data analysis",
    "The physician treated the sick patient",
    "The match was exciting and competitive",
    "New software improves computer performance",
    "Political leaders discussed healthcare reforms",
    "Exercise helps prevent heart disease",
    "The team trained daily for the competition",
    "Advanced technology improves medical diagnosis",
    "Doctors and physicians save lives",
    "The government focuses on digital transformation"
]

df = pd.DataFrame({"Text": documents})
df.head()
```

	Text
0	The football team won the championship match
1	Cricket players trained hard for the tournament
2	The athlete broke the world record
3	The government announced a new policy
4	The election results were declared today

Next steps: Generate code with df New interactive sheet

```
nltk.download('punkt_tab')
stop_words = set(stopwords.words('english'))
lemmatizer = WordNetLemmatizer()
```

```
def preprocess_text(text):
    text = text.lower()                           # Lowercase
    text = re.sub(r'[^a-z\s]', '', text)          # Remove punctuation & numbers
    tokens = word_tokenize(text)                  # Tokenize
    tokens = [t for t in tokens if t not in stop_words]  # Remove stopwords
    tokens = [lemmatizer.lemmatize(t) for t in tokens]   # Lemmatization
    return " ".join(tokens)

df["Cleaned_Text"] = df["Text"].apply(preprocess_text)
df.head()
```

```
[nltk_data] Downloading package punkt_tab to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt_tab.zip.
```

	Text	Cleaned_Text
0	The football team won the championship match	football team championship match
1	Cricket players trained hard for the tournament	cricket player trained hard tournament
2	The athlete broke the world record	athlete broke world record
3	The government announced a new policy	government announced new policy
4	The election results were declared today	election result declared today

Next steps: Generate code with df New interactive sheet

```
vectorizer = TfidfVectorizer()
tfidf_matrix = vectorizer.fit_transform(df["Cleaned_Text"])
```

```
cosine_sim = cosine_similarity(tfidf_matrix)

cosine_df = pd.DataFrame(cosine_sim, index=df["Text"], columns=df["Text"])
cosine_df.iloc[:5, :5]
```

Text	The football team won the championship match	Cricket players trained hard for the tournament	The athlete broke the world record	The government announced a new policy	The election results were declared today
Text					
The football team won the championship match	1.0	0.0	0.0	0.0	0.0
Cricket players trained hard for the tournament	0.0	1.0	0.0	0.0	0.0
The athlete broke the world record	0.0	0.0	1.0	0.0	0.0
The government announced a new	0.0	0.0	0.0	1.0	0.0

```
def jaccard_similarity(doc1, doc2):
    set1 = set(doc1.split())
    set2 = set(doc2.split())
    return len(set1 & set2) / len(set1 | set2)

jaccard_scores = []
for i in range(len(df)):
    for j in range(i+1, len(df)):
        score = jaccard_similarity(df["Cleaned_Text"][i], df["Cleaned_Text"][j])
        jaccard_scores.append((df["Text"][i], df["Text"][j], score))

jaccard_df = pd.DataFrame(jaccard_scores, columns=["Doc1", "Doc2", "Jaccard"])
jaccard_df.head()
```

	Doc1	Doc2	Jaccard
0	The football team won the championship match	Cricket players trained hard for the tournament	0.0
1	The football team won the championship match	The athlete broke the world record	0.0
2	The football team won the championship match	The government announced a new policy	0.0
3	The football team won the championship match	The election results were declared today	0.0
4	The football team won the championship match	Parliament passed the new bill	0.0

Next steps: Generate code with jaccard_df New interactive sheet

```
def wordnet_similarity(word1, word2):
    syn1 = wordnet.synsets(word1)
    syn2 = wordnet.synsets(word2)
    if syn1 and syn2:
        return syn1[0].wup_similarity(syn2[0])
    return None

pairs = [("doctor", "physician"), ("football", "cricket"),
         ("government", "parliament"), ("exercise", "fitness"),
         ("technology", "software")]

for w1, w2 in pairs:
    print(w1, "-", w2, ":", wordnet_similarity(w1, w2))
```

```
doctor - physician : 1.0
football - cricket : 0.09090909090909091
government - parliament : 0.5333333333333333
exercise - fitness : 0.25
technology - software : 0.23529411764705882
```