Krisha Botadara 64070503484

```
import pandas as pd

df = pd.read_pickle('consumer_complaint_dataset.data', compression='gzip')
```

```
df
```

	topic	input
0	Debt collection	transworld systems inc. \nis trying to collect...
1	Credit reporting, credit repair services, or o...	I would like to request the suppression of the...
2	Debt collection	Over the past 2 weeks, I have been receiving e...
3	Credit reporting, credit repair services, or o...	I HAD FILED WITH CFPB ON XX/XX/XXXX19 TO HAVE ...
4	Credit reporting, credit repair services, or o...	I have several accounts that the balance is in...
...	...	...
492250	Consumer Loan	I was on automatic payment for my car loan. In...
492251	Debt collection	I recieved a collections call from an unknown ...
492252	Mortgage	On XXXX XXXX, 2015, I contacted XXXX XXXX, who...
492253	Mortgage	I can not get from chase who services my mortg...
492254	Credit card	I made a payment to CITI XXXX Credit Card on X...

492255 rows × 2 columns

```
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer
import string

nltk.download('punkt')
nltk.download('stopwords')
nltk.download('wordnet')

stop_words = set(stopwords.words('english'))
lemmatizer = WordNetLemmatizer()

def preprocess_text(text):
    # Convert to lowercase
    text = text.lower()
    # Tokenize the text
    words = word_tokenize(text)
    # Remove stopwords and punctuation, and lemmatize
    words = [lemmatizer.lemmatize(w) for w in words if w not in stop_words and w not in string.punctuation]
    return words

# Apply the pre-processing to the 'input' column
df['processed_input'] = df['input'].apply(preprocess_text)
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-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!
```

```
df
```

	topic	input	processed_input
0	Debt collection	transworld systems inc. \nis trying to collect...	[transworld, system, inc., trying, collect, de...
1	Credit reporting, credit repair services, or o...	I would like to request the suppression of the...	[would, like, request, suppression, following,...
2	Debt collection	Over the past 2 weeks, I have been receiving e...	[past, 2, week, receiving, excessive, amount, ...
3	Credit reporting, credit repair services, or o...	I HAD FILED WITH CFPB ON XX/XX/XXXX19 TO HAVE ...	[filed, cfpb, xx/xx/xxxx19, listed, account, d...
4	Credit reporting, credit repair services, or o...	I have several accounts that the balance is in...	[several, account, balance, incorrect, couple,...
...	...	...	...
492250	Consumer Loan	I was on automatic payment for my car loan. In...	[automatic, payment, car, loan, fine, print, s...
492251	Debt collection	I recieved a collections call from an unknown ...	[recieved, collection, call, unknown, company,...
492252	Mortgage	On XXXX XXXX, 2015, I contacted XXXX XXXX, who...	[xxxx, xxxx, 2015, contacted, xxxx, xxxx, bran...
492253	Mortgage	I can not get from chase who services my mortg...	[get, chase, service, mortgage, owns, original...
492254	Credit card	I made a payment to CITI XXXX Credit Card on X...	[made, payment, citi, xxxx, credit, card, xxxx...

492255 rows × 3 columns

```
from gensim.models import Word2Vec

# Train Word2Vec on the processed text
model = Word2Vec(sentences=df['processed_input'], vector_size=300, window=10, min_count=2, workers=4)
```

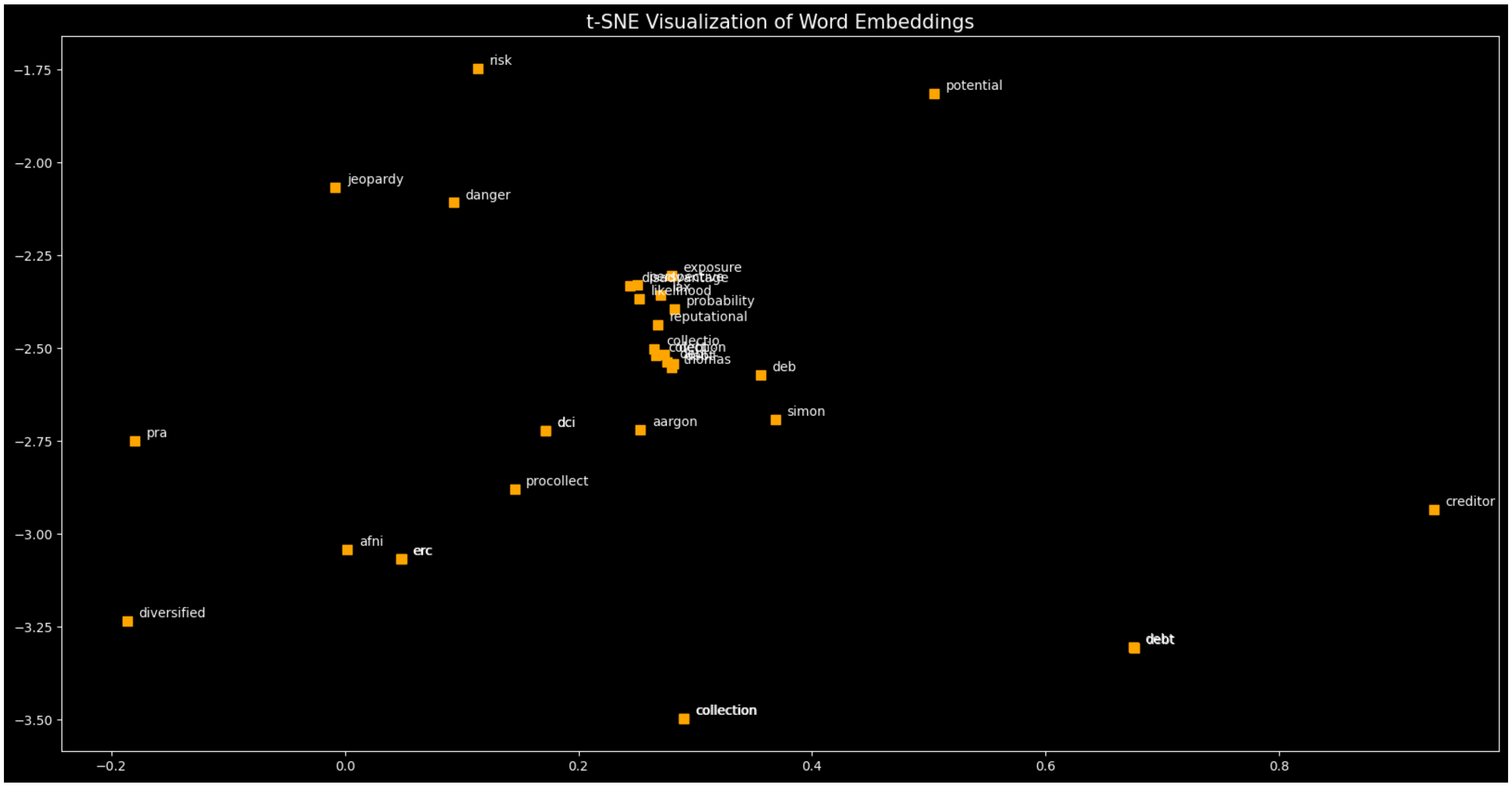
```
import pandas as pd

# Get the top 10 similar words for 'debt', 'collection', and 'risk'
similar_words_debt = model.wv.most_similar('debt', topn=10)
similar_words_collection = model.wv.most_similar('collection', topn=10)
similar_words_risk = model.wv.most_similar('risk', topn=10)

# Extract the words only (ignore similarity scores)
debt_words = [word for word, _ in similar_words_debt]
collection_words = [word for word, _ in similar_words_collection]
risk_words = [word for word, _ in similar_words_risk]

# Create a DataFrame with 3 columns (debt, collection, risk)
df_similar_words = pd.DataFrame({
    'Debt': debt_words,
    'Collection': collection_words,
    'Risk': risk_words
})

# Display the DataFrame
df_similar_words
```

	Debt	Collection	Risk
0	debt-	aargon	danger
1	deb	colection	exposure
2	debt.i	simon	jeopardy
3	'debt	thomas	disadvantage
4	erc	erc	likelihood
...	...	...	...
95	ecmc	allied	population
96	bill	owed	ineptitude
97	trident	vance	whim
98	allege	amca	ramification
99	grantor/beneficiary	comcast	jeopardize

100 rows × 3 columns

Next steps: Generate code with `df_similar_words` View recommended plots New interactive sheet

```
from sklearn.manifold import TSNE
import matplotlib.pyplot as plt
import numpy as np

# Get the vectors of the similar words
words = ['debt', 'collection', 'risk'] + [word for word, _ in similar_words_debt + similar_words_collection + similar_words_risk]
word_vectors = [model.wv[word] for word in words]

word_vectors = np.array(word_vectors)
tsne = TSNE(n_components=2, random_state=42)
word_vec_tsne = tsne.fit_transform(word_vectors)

# Plotting
plt.figure(figsize=(20, 10))
plt.scatter(word_vec_tsne[:, 0], word_vec_tsne[:, 1], color='orange', marker='s', s=50)

for i, word in enumerate(words):
    plt.annotate(word, (word_vec_tsne[i, 0] + 0.01, word_vec_tsne[i, 1] + 0.01), fontsize=10, color='white')

plt.title("t-SNE Visualization of Word Embeddings", fontsize=15, color='white')
plt.show()
```


t-SNE Visualization of Word Embeddings
risk
potential
jeopardy
danger
exposure
likelihood
probability
reputational
collectio
thomas
deb
simon
pra
dci
aargon
procollect
creditor
afni
erc
diversified
debt
collection