FUTURESKILLS AI BOOTCAMP ASSIGNMENT 5

Problem Statement -

Preprocess product review text data for sentiment analysis by performing text cleaning, tokenization, stopword removal, stemming, and lemmatization to create a structured dataset for further analysis.

```
# Install necessary libraries
!pip install pandas nltk --quiet

# Import required libraries
import pandas as pd
import nltk
import re
import string
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer, WordNetLemmatizer
from nltk.tokenize import RegexpTokenizer
from collections import Counter

# Download required NLTK resources
nltk.download("stopwords")
nltk.download("wordnet")

# Load dataset with proper encoding handling
file_path = "product_reviews.csv"
df = pd.read_csv(file_path, encoding="ISO-8859-1", on_bad_lines="skip")

# Upload the file manually
uploaded = files.upload()

# Get the filename
file_name = list(uploaded.keys())[0]
```

```
# Load the dataset into a DataFrame with error handling
try:
    df = pd.read_csv(io.BytesIO(uploaded[file_name]), encoding='utf-8')  # Try UTF-8 encoding first
except UnicodeDecodeError:
    df = pd.read_csv(io.BytesIO(uploaded[file_name]), encoding='latin1')  # Use Latin-1 encoding as fallback

# Display the first few rows
df.head()
```

Choose Files product_reviews.csv

- **product_reviews.csv**(text/csv) - 1410 bytes, last modified: 3/25/2025 - 100% done

```
Saving product_reviews.csv to product_reviews (4).csv
```

	Review_ID	Review_Text
0	1	"The product is GREAT! Loved it, but it□s a bi...
1	2	"Worst product ever!! Wouldn□t recommend to an...
2	3	"Satisfactory quality, works as expected, no m...
3	4	"Amazing product, I would buy it again and aga...
4	5	"The delivery was slow, but the product is good."

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
# Function to fix encoding issues (remove non-ASCII and normalize quotes)
def clean_encoding_issues(text):
    if isinstance(text, str):
        text = text.encode("ascii", "ignore").decode("ascii")
        text = text.replace("’", "'").replace("“", '"').replace("”", '"')
    return text
```

```
df["Review_Text"] = df["Review_Text"].apply(clean_encoding_issues)

print("Original Dataset:")
print(df.head())

# Initialize preprocessing tools
stop_words = set(stopwords.words("english"))
stemmer = PorterStemmer()
lemmatizer = WordNetLemmatizer()
# Use RegexpTokenizer to tokenize words without relying on 'punkt'
tokenizer = RegexpTokenizer(r'\w+')

# Define text preprocessing function
def preprocess_text(text):
    if not isinstance(text, str):
        return {"tokens": [], "stemmed": [], "lemmatized": [], "tf": {}}

    # 1. Convert to lowercase
    text = text.lower()

    # 2. Remove punctuation, numbers, and special characters using regex
    text = re.sub(r"[^a-z\s]", " ", text)

    # 3. Tokenize into words
    tokens = tokenizer.tokenize(text)

    # 4. Remove stopwords
    tokens_clean = [word for word in tokens if word not in stop_words]

    # 5a. Apply stemming
    stemmed_tokens = [stemmer.stem(word) for word in tokens_clean]

    # 5b. Apply lemmatization
    lemmatized_tokens = [lemmatizer.lemmatize(word) for word in tokens_clean]

    # 6. Calculate Term Frequency (TF) using Counter from collections
    term_freq = dict(Counter(tokens_clean))
```

```
    return {
        "tokens": tokens_clean,
        "stemmed": stemmed_tokens,
        "lemmatized": lemmatized_tokens,
        "tf": term_freq
    }

# Apply preprocessing to each review and expand results into separate columns
preprocessed = df["Review_Text"].apply(preprocess_text)
df["Tokens"] = preprocessed.apply(lambda x: x["tokens"])
df["Stemmed"] = preprocessed.apply(lambda x: x["stemmed"])
df["Lemmatized"] = preprocessed.apply(lambda x: x["lemmatized"])
df["Term_Frequency"] = preprocessed.apply(lambda x: x["tf"])

# Display processed data
print("\nProcessed Data:")
print(df[["Review_Text", "Tokens", "Stemmed", "Lemmatized", "Term_Frequency"]].head())

# Optionally, save the cleaned dataset for further analysis
df.to_csv("cleaned_product_reviews.csv", index=False)
print("\nCleaned dataset saved as 'cleaned_product_reviews.csv'.")
```

```
Original Dataset:
   Review_ID                                        Review_Text
0          1  "The product is GREAT! Loved it, but its a bit...
1          2  "Worst product ever!! Wouldnt recommend to any...
2          3  "Satisfactory quality, works as expected, no m...
3          4  "Amazing product, I would buy it again and aga...
4          5  "The delivery was slow, but the product is good."

Processed Data:
                                        Review_Text  \
0  "The product is GREAT! Loved it, but its a bit...
1  "Worst product ever!! Wouldnt recommend to any...
2  "Satisfactory quality, works as expected, no m...
3  "Amazing product, I would buy it again and aga...
4  "The delivery was slow, but the product is good."
```

```
                                              Tokens  \
0              [product, great, loved, bit, pricey]
1  [worst, product, ever, wouldnt, recommend, any...
2  [satisfactory, quality, works, expected, major...
3                     [amazing, product, would, buy]
4                   [delivery, slow, product, good]

                                             Stemmed  \
0               [product, great, love, bit, pricey]
1  [worst, product, ever, wouldnt, recommend, anyon]
2  [satisfactori, qualiti, work, expect, major, i...
3                       [amaz, product, would, buy]
4                   [deliveri, slow, product, good]

                                           Lemmatized  \
0              [product, great, loved, bit, pricey]
1  [worst, product, ever, wouldnt, recommend, any...
2  [satisfactory, quality, work, expected, major,...
3                     [amazing, product, would, buy]
4                   [delivery, slow, product, good]

                                      Term_Frequency
0  {'product': 1, 'great': 1, 'loved': 1, 'bit': ...
1  {'worst': 1, 'product': 1, 'ever': 1, 'wouldnt...
2  {'satisfactory': 1, 'quality': 1, 'works': 1, ...
3  {'amazing': 1, 'product': 1, 'would': 1, 'buy'...
4  {'delivery': 1, 'slow': 1, 'product': 1, 'good...

Cleaned dataset saved as 'cleaned_product_reviews.csv'.
```