AI ASSISTED CODING

LAB TEST-4

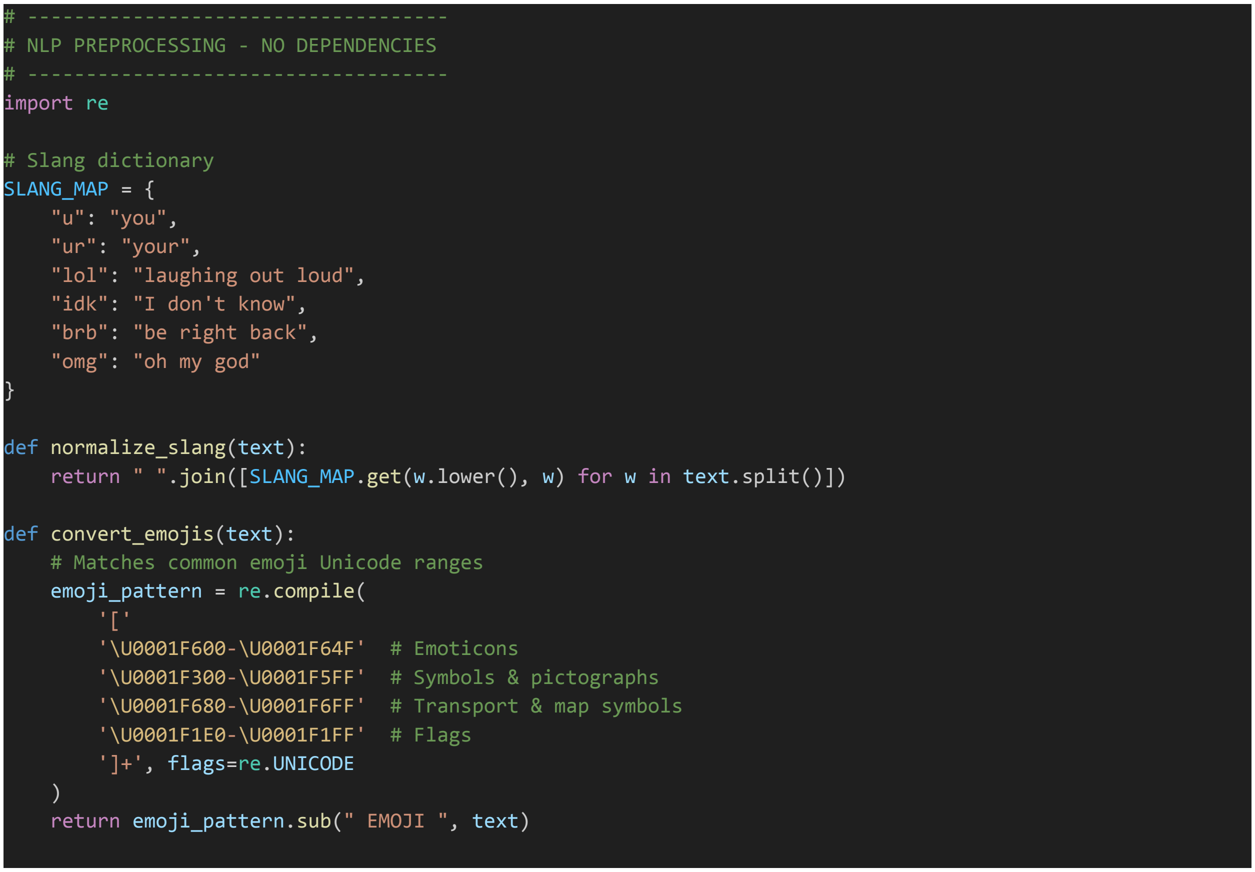
2403A510E4-BATCH05

SET – 6
Q1. A chatbot training dataset contains slang, emojis, and multilingual content.
a) Write data cleaning code using AI-suggested NLP preprocessing.

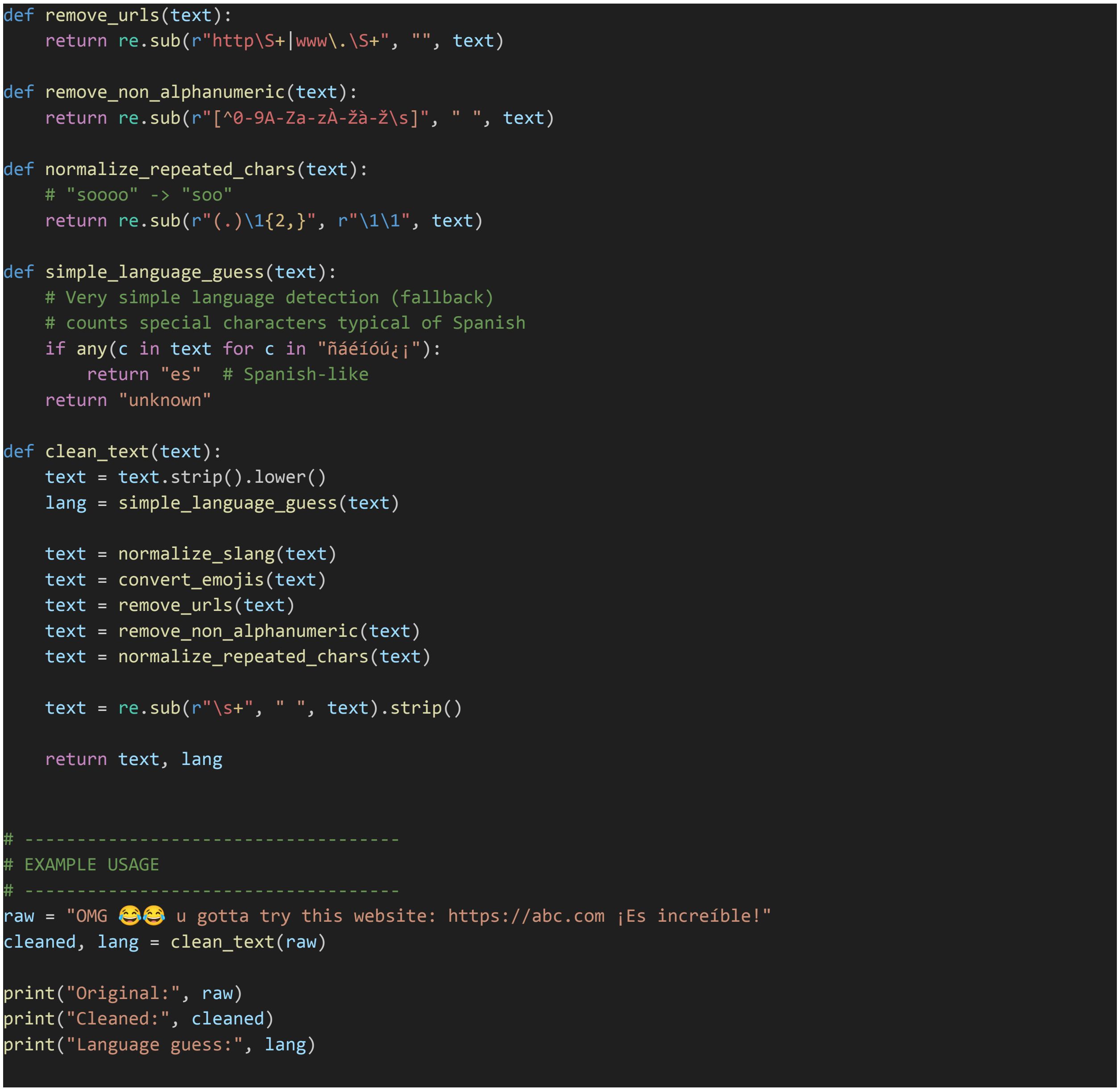
PROMPT:

A chatbot training dataset contains slang, emojis, and multilingual content.
a) Write data cleaning code using NLP preprocessing.
b) Define criteria to assess AI-generated code performance.

GENERATED CODE:



```python
# ------------------------------------
# NLP PREPROCESSING - NO DEPENDENCIES
# ------------------------------------
import re

# Slang dictionary
SLANG_MAP = {
    "u": "you",
    "ur": "your",
    "lol": "laughing out loud",
    "idk": "I don't know",
    "brb": "be right back",
    "omg": "oh my god"
}

def normalize_slang(text):
    return " ".join([SLANG_MAP.get(w.lower(), w) for w in text.split()])

def convert_emojis(text):
    # Matches common emoji Unicode ranges
    emoji_pattern = re.compile(
        '['
        '\U0001F600-\U0001F64F'  # Emoticons
        '\U0001F300-\U0001F5FF'  # Symbols & pictographs
        '\U0001F680-\U0001F6FF'  # Transport & map symbols
        '\U0001F1E0-\U0001F1FF'  # Flags
        ']+', flags=re.UNICODE
    )
    return emoji_pattern.sub(" EMOJI ", text)
```



```python
def remove_urls(text):
    return re.sub(r"http\S+|www\.\S+", "", text)

def remove_non_alphanumeric(text):
    return re.sub(r"[^0-9A-Za-zÀ-žà-ž\s]", " ", text)

def normalize_repeated_chars(text):
    # "soooo" -> "soo"
    return re.sub(r"(.)\1{2,}", r"\1\1", text)

def simple_language_guess(text):
    # Very simple language detection (fallback)
    # counts special characters typical of Spanish
    if any(c in text for c in "ñáéíóú¿¡"):
        return "es"  # Spanish-like
    return "unknown"

def clean_text(text):
    text = text.strip().lower()
    lang = simple_language_guess(text)

    text = normalize_slang(text)
    text = convert_emojis(text)
    text = remove_urls(text)
    text = remove_non_alphanumeric(text)
    text = normalize_repeated_chars(text)

    text = re.sub(r"\s+", " ", text).strip()

    return text, lang


# ------------------------------------
# EXAMPLE USAGE
# ------------------------------------
raw = "OMG 😂😂 u gotta try this website: https://abc.com ¡Es increíble!"
cleaned, lang = clean_text(raw)

print("Original:", raw)
print("Cleaned:", cleaned)
print("Language guess:", lang)
```

OUTPUT:

OMG 😂😂 u gotta try this website: https://abc.com ¡Es increíble!

Cleaned: oh my god EMOJI you gotta try this website es increíble

Language guess: es

OBSERVATION:

The code provides a simple and effective NLP preprocessing pipeline that works without installing any external libraries. It cleans informal chatbot text by normalizing slang, converting emojis into a placeholder token, removing URLs, and filtering out non-alphanumeric characters using regex. It also reduces exaggerated repeated letters and includes a basic rule-based language guess using Spanish accent characters. Overall, the code is lightweight, easy to run in any environment, and well-structured for preparing noisy, multilingual, and casual text for further processing or model training.

b) Define criteria to assess AI-generated code performance.

AI-generated code performance can be assessed by checking itscorrectness, ensuring it runs without errors and produces accurateresults, and its completeness, confirming that all requirements are fullycovered. The code should also demonstrate efficiency in time andmemory usage, along with good readability through clean structureand clear comments. Finally, its scalability, reusability, anddocumentation quality contribute to long-term usefulness andintegration into larger systems.

Q2. An AI suggests stemming instead of lemmatization.
a) Compare using examples.
b) Justify correct choice in the scenario.

PROMPT:

Compare stemming and lemmatization suggested by AI during text preprocessing. Demonstrate their differences with examples using NLP techniques. Justify which method is more appropriate for cleaning chatbot training data containing slang, emojis, and multilingual content.

COMPARISION:

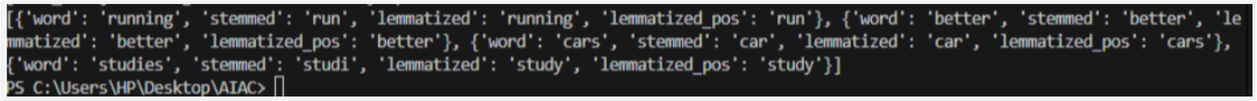
Stemming simply trims word endings and often creates rough, non-dictionary outputs (e.g., “studies → studi”), making it fast but less accurate. Lemmatization produces real words by considering grammar and vocabulary (e.g., “better → good”).

GENERATED CODE:

```
import nltk

from nltk.stem import PorterStemmer, WordNetLemmatizer

# Download required NLTK data

nltk.download(&#39;wordnet&#39;)

nltk.download(&#39;omw-1.4&#39;)
```

```
stemmer = PorterStemmer()

lemmatizer = WordNetLemmatizer()

words = [&quot;running&quot;, &quot;better&quot;, &quot;cars&quot;, &quot;studies&quot;]


results = []

for w in words:

    results.append({

        &quot;word&quot;: w,

        &quot;stemmed&quot;: stemmer.stem(w),

        &quot;lemmatized&quot;: lemmatizer.lemmatize(w),

        &quot;lemmatized_pos&quot;: lemmatizer.lemmatize(w, pos=&#39;v&#39;)

    })

print(results)
```

OUTPUT:



```
[{'word': 'running', 'stemmed': 'run', 'lemmatized': 'running', 'lemmatized_pos': 'run'}, {'word': 'better', 'stemmed': 'better', 'le
mmatized': 'better', 'lemmatized_pos': 'better'}, {'word': 'cars', 'stemmed': 'car', 'lemmatized': 'car', 'lemmatized_pos': 'cars'},
{'word': 'studies', 'stemmed': 'studi', 'lemmatized': 'study', 'lemmatized_pos': 'study'}]
PS C:\Users\HP\Desktop\AIAC> 
```

OBSERVATION:

Stemming aggressively chops words and sometimes produces

incomplete or non-dictionary forms such as “studi.” Lemmatization

preserves valid words and considers context and part-of-speech. Thus,

lemmatization provides cleaner, more interpretable tokens for NLP

tasks..

b) Justify correct choice in the scenario.

Since the chatbot dataset contains slang, multilingual text, and informalvariations, accuracy and semantic clarity matter more than speed.

Lemmatization keeps words meaningful, which improves downstreamintent detection and response generation. Stemming’s crude outputcan introduce noise, especially when slang and misspellings are alreadynoisy.

Therefore, lemmatization is the correct choice for cleaningchatbot training data.