**IMPLEMENT N GRAM LANGUAGE MODEL:**

import nltk

from nltk.util import ngrams

from collections import Counter, defaultdict

# Download necessary resources (only first time)

nltk.download("punkt")

# Sample text corpus

text = """I love natural language processing.

I love deep learning.

I love machine learning.

Machine learning is fun and powerful."""

# Tokenize the text into words

tokens = nltk.word\_tokenize(text.lower())

# Function to build an N-gram model

def build\_ngram\_model(tokens, n):

ngrams\_list = list(ngrams(tokens, n, pad\_left=True, pad\_right=True,

left\_pad\_symbol="<s>", right\_pad\_symbol="</s>"))

model = defaultdict(Counter)

for gram in ngrams\_list:

prefix, word = gram[:-1], gram[-1]

model[prefix][word] += 1

return model

# Build a Bigram (2-gram) model

bigram\_model = build\_ngram\_model(tokens, 2)

# Print the bigram counts

print("=== Bigram Model ===")

for context, counter in bigram\_model.items():

print(f"{context} -> {dict(counter)}")

# Generate a sentence using the bigram model

import random

def generate\_sentence(model, n, max\_words=15):

context = tuple(["<s>"]\*(n-1))

result = []

for \_ in range(max\_words):

if context not in model:

break

possible\_words = list(model[context].elements())

next\_word = random.choice(possible\_words)

if next\_word == "</s>":

break

result.append(next\_word)

context = tuple((list(context) + [next\_word])[1:])

return " ".join(result)

print("\n=== Generated Sentence ===")

print(generate\_sentence(bigram\_model, 2))