**Lab 9 task**

Url: <https://www.geeksforgeeks.org/natural-language-processing-nlp-tutorial/>

**Text Summarization:**

import gradio as gr

from transformers import T5Tokenizer, T5ForConditionalGeneration

import torch

# --- Load model and tokenizer ---

model\_name = 't5-small'

tokenizer = T5Tokenizer.from\_pretrained(model\_name)

model = T5ForConditionalGeneration.from\_pretrained(model\_name)

# --- Define summarization function ---

def summarize(text):

    inputs = tokenizer.encode("summarize: " + text, return\_tensors="pt", max\_length=1024, truncation=True)

    summary\_ids = model.generate(inputs, max\_length=150, min\_length=40, length\_penalty=2.0, num\_beams=4, early\_stopping=True)

    summary = tokenizer.decode(summary\_ids[0], skip\_special\_tokens=True)

    return summary

# --- Example test ---

text = """ The Hugging Face library has revolutionized the field of natural language processing with its transformers library.

This library provides state-of-the-art models for various NLP tasks including text summarization, text classification, question answering, and more.

With easy-to-use APIs and pre-trained models, developers can quickly integrate advanced NLP capabilities into their applications.

The community-driven approach ensures continuous improvement and innovation in the library, making it a valuable resource for both researchers and practitioners."""

print("Summary:\n", summarize(text))

# --- Gradio Interface ---

# Gradio interface

iface = gr.Interface(fn=summarize, inputs="text", outputs="text", title="Text Summarization with T5", description="Enter text to get a summarized version using the T5 model.")

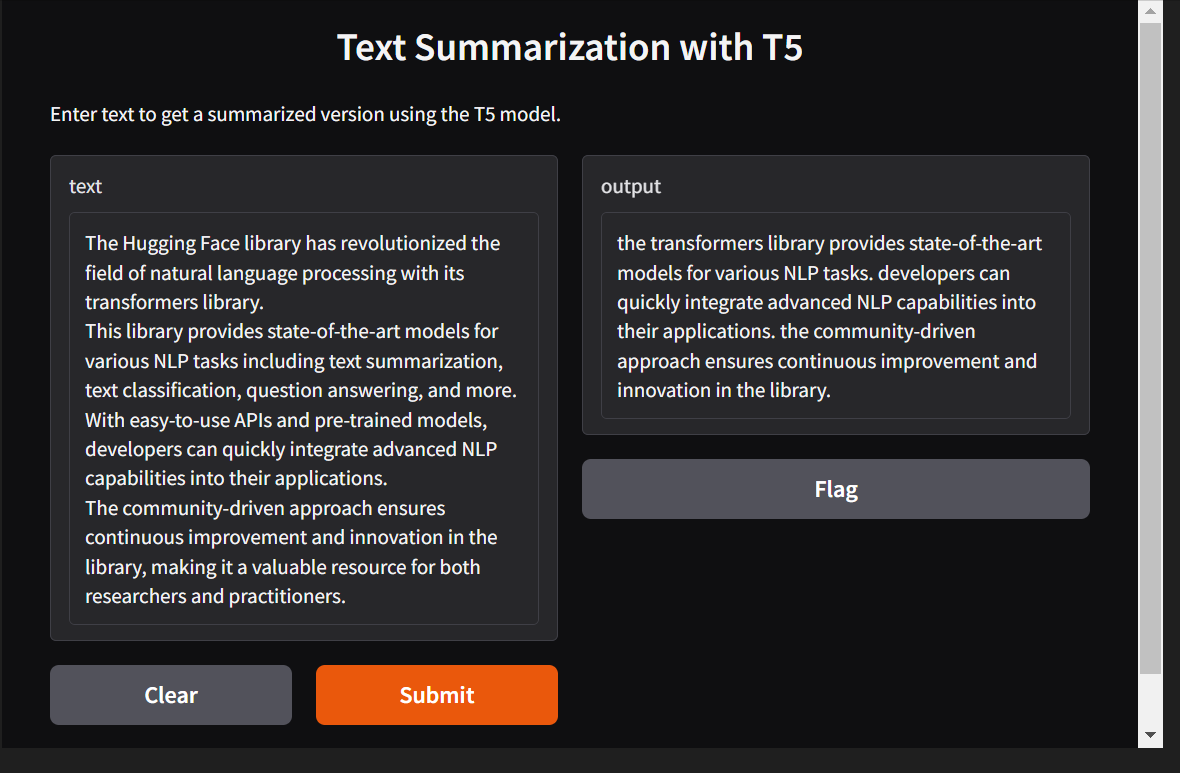
# Launch the interface

iface.launch()

**Explanation:**

This code uses the T5 model from Hugging Face to create a text summarization application. It loads a pre-trained model and tokenizer, which convert input text into a format the model understands. The model generates a concise summary of the text based on specified parameters like length and beam search. The summarization function is integrated into a user-friendly web interface created with Gradio, allowing users to input text and receive summarized output in real-time. The Gradio interface makes it easy to interact with the model through a simple web page.

**Output:**



**Code:**

from sumy.nlp.tokenizers import Tokenizer

import nltk

# Download punkt tokenizer if not already present

nltk.download('punkt')

# Initialize tokenizer

tokenizer = Tokenizer("en")

# Sample text

text = """Hello, this is GeeksForGeeks! We are a computer science portal for geeks, offering a wide range of articles, tutorials, and resources on various topics in computer science and programming. Our mission is to provide quality education and knowledge sharing to help you excel in your career and academic pursuits. Whether you're a beginner looking to learn the basics of coding or an experienced developer seeking advanced concepts, GeeksForGeeks has something for everyone."""

# Tokenize text into sentences

sentences = tokenizer.to\_sentences(text)

# Tokenize sentences into words and print them

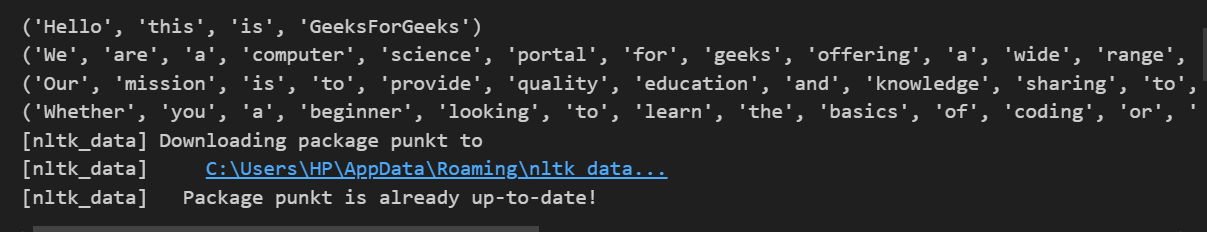
for sentence in sentences:

    print(tokenizer.to\_words(sentence))

**Explanation:**

This code uses the sumy library’s Tokenizer and NLTK to split a paragraph into sentences and then into words. It first ensures the punkt tokenizer is downloaded using NLTK, which is needed for sentence splitting. After initializing the tokenizer for English, it processes a sample text by breaking it into individual sentences. Then, each sentence is further tokenized into words, and the words are printed one by one. This setup helps in basic text preprocessing, useful for tasks like text analysis or summarization.

**Output:**



**Code:**

from sumy.nlp.stemmers import Stemmer

stemmer = Stemmer("en")

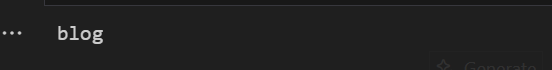
stem = stemmer("Blogging")

print(stem)

**Explanation:**

This code initializes a **Stemmer** from the sumy library for the English language and applies it to the word **"Blogging"**. Stemming reduces a word to its root form, so **"Blogging"** would be stemmed to **"blog"**.

**Output:**



**Code:**

from sumy.parsers.plaintext import PlaintextParser

from sumy.nlp.tokenizers import Tokenizer

from sumy.summarizers.luhn import LuhnSummarizer

from sumy.nlp.stemmers import Stemmer

from sumy.utils import get\_stop\_words

import nltk

nltk.download('punkt')

def summarize\_paragraph(paragraph, sentences\_count=2):

    parser = PlaintextParser.from\_string(paragraph, Tokenizer("english"))

    summarizer = LuhnSummarizer(Stemmer("english"))

    summarizer.stop\_words = get\_stop\_words("english")

    summary = summarizer(parser.document, sentences\_count)

    return summary

if \_\_name\_\_ == "\_\_main\_\_":

    paragraph = """Artificial intelligence (AI) is intelligence demonstrated by machines, in contrast

                   to the natural intelligence displayed by humans and animals. Leading AI textbooks define

                   the field as the study of "intelligent agents": any device that perceives its environment

                   and takes actions that maximize its chance of successfully achieving its goals. Colloquially,

                   the term "artificial intelligence" is often used to describe machines (or computers) that mimic

                   "cognitive" functions that humans associate with the human mind, such as "learning" and "problem solving"."""

    sentences\_count = 2

    summary = summarize\_paragraph(paragraph, sentences\_count)

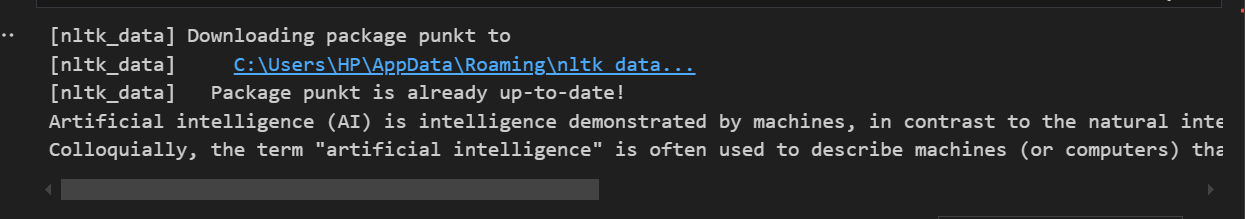
    for sentence in summary:

        print(sentence)

**Explanation:**

This code performs text summarization using the **Luhn algorithm** from the sumy library. It defines a function summarize\_paragraph() that takes a paragraph and the number of summary sentences as input. It parses the paragraph, tokenizes it into sentences, and then uses the **LuhnSummarizer** with stemming and English stopwords to generate a concise summary. In the main block, a sample paragraph about Artificial Intelligence is summarized into two sentences, and the result is printed.

**Output:**



**Code:**

from sumy.parsers.plaintext import PlaintextParser

from sumy.nlp.tokenizers import Tokenizer

from sumy.summarizers.edmundson import EdmundsonSummarizer

from sumy.nlp.stemmers import Stemmer

from sumy.utils import get\_stop\_words

import nltk

nltk.download('punkt')

def summarize\_paragraph(paragraph, sentences\_count=2, bonus\_words=None, stigma\_words=None, null\_words=None):

    parser = PlaintextParser.from\_string(paragraph, Tokenizer("english"))

    summarizer = EdmundsonSummarizer(Stemmer("english"))

    summarizer.stop\_words = get\_stop\_words("english")

    if bonus\_words:

        summarizer.bonus\_words = bonus\_words

    if stigma\_words:

        summarizer.stigma\_words = stigma\_words

    if null\_words:

        summarizer.null\_words = null\_words

    summary = summarizer(parser.document, sentences\_count)

    return summary

if \_\_name\_\_ == "\_\_main\_\_":

    paragraph = """Artificial intelligence (AI) is intelligence demonstrated by machines, in contrast

                   to the natural intelligence displayed by humans and animals. Leading AI textbooks define

                   the field as the study of "intelligent agents": any device that perceives its environment

                   and takes actions that maximize its chance of successfully achieving its goals. Colloquially,

                   the term "artificial intelligence" is often used to describe machines (or computers) that mimic

                   "cognitive" functions that humans associate with the human mind, such as "learning" and "problem solving"."""

    sentences\_count = 2

    bonus\_words = ["intelligence", "AI"]

    stigma\_words = ["contrast"]

    null\_words = ["the", "of", "and", "to", "in"]

    summary = summarize\_paragraph(paragraph, sentences\_count, bonus\_words, stigma\_words, null\_words)

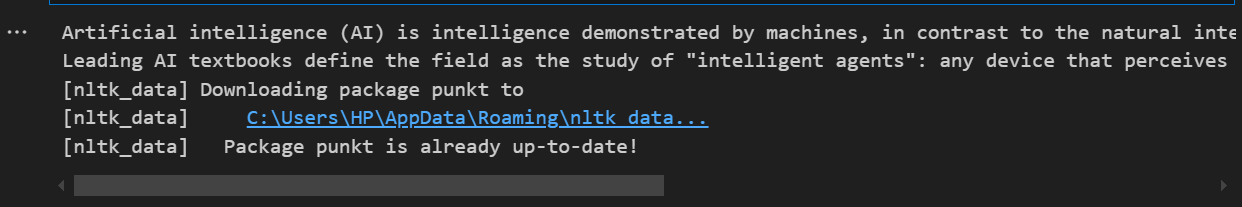
    for sentence in summary:

        print(sentence)

**Explanation:**

This code uses the **Edmundson Summarizer** from the sumy library to generate a summary of a given paragraph. It defines a function summarize\_paragraph() that allows customization using **bonus words** (which are preferred), **stigma words** (which are penalized), and **null words** (which are ignored). The paragraph is parsed and tokenized, and the summarizer highlights important sentences based on these word settings. In the main block, a sample text about Artificial Intelligence is summarized into two sentences, and the selected sentences are printed.

**Output:**



**Code:**

from sumy.parsers.plaintext import PlaintextParser

from sumy.nlp.tokenizers import Tokenizer

from sumy.summarizers.lsa import LsaSummarizer

from sumy.nlp.stemmers import Stemmer

from sumy.utils import get\_stop\_words

import nltk

nltk.download('punkt')

def summarize\_paragraph(paragraph, sentences\_count=2):

    parser = PlaintextParser.from\_string(paragraph, Tokenizer("english"))

    summarizer = LsaSummarizer(Stemmer("english"))

    summarizer.stop\_words = get\_stop\_words("english")

    summary = summarizer(parser.document, sentences\_count)

    return summary

if \_\_name\_\_ == "\_\_main\_\_":

    paragraph = """Artificial intelligence (AI) is intelligence demonstrated by machines, in contrast

                   to the natural intelligence displayed by humans and animals. Leading AI textbooks define

                   the field as the study of "intelligent agents": any device that perceives its environment

                   and takes actions that maximize its chance of successfully achieving its goals. Colloquially,

                   the term "artificial intelligence" is often used to describe machines (or computers) that mimic

                   "cognitive" functions that humans associate with the human mind, such as "learning" and "problem solving"."""

    sentences\_count = 2

    summary = summarize\_paragraph(paragraph, sentences\_count)

    for sentence in summary:

        print(sentence)

**Explanation:**

This code uses the **Latent Semantic Analysis (LSA) Summarizer** from the sumy library to create a short summary of a given paragraph. The summarize\_paragraph() function parses and tokenizes the input text, removes common stop words, and applies LSA to select the most meaningful sentences based on underlying topics and patterns. In the main section, a paragraph about Artificial Intelligence is summarized into two key sentences, which are then printed.

**Output:**

