



Code:

import ipywidgets as widgets

from IPython.display import display, Audio, clear\_output, Image as IPyImage

import base64

from io import BytesIO

from transformers import pipeline

import speech\_recognition as sr

from PIL import Image

from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer

import time

# Initialize models

text\_summarizer = pipeline("summarization")

image\_classifier = pipeline("image-classification")

speech\_recognizer = sr.Recognizer()

sentiment\_analyzer = SentimentIntensityAnalyzer()

# Text Summarization Tab

text\_input = widgets.Textarea(placeholder='Enter text to summarize', layout={'width': '50%'})

summarize\_button = widgets.Button(description="Summarize")

text\_output = widgets.Output()

def summarize\_text(b):

with text\_output:

clear\_output()

try:

summary = text\_summarizer(text\_input.value)[0]['summary\_text']

print(summary)

time.sleep(10) # Display output for 10 seconds

except Exception as e:

print(f"Error: {e}")

summarize\_button.on\_click(summarize\_text)

text\_tab = widgets.VBox([text\_input, summarize\_button, text\_output])

# Image Classification Tab

upload\_image\_button = widgets.FileUpload(description="Upload Image")

classify\_image\_button = widgets.Button(description="Classify")

image\_output = widgets.Output()

def classify\_image(b):

with image\_output:

clear\_output()

try:

image\_data = next(iter(upload\_image\_button.value.values()))['content']

# Convert image data to base64

image\_base64 = base64.b64encode(image\_data).decode("utf-8")

result = image\_classifier(image\_base64)

print("Predicted label:", result[0]['label'])

print("Confidence score:", result[0]['score'])

except Exception as e:

print(f"Error: {e}")

classify\_image\_button.on\_click(classify\_image)

image\_tab = widgets.VBox([upload\_image\_button, classify\_image\_button, image\_output])

# Speech Recognition Tab

record\_button = widgets.Button(description="Record")

audio\_output = widgets.Output()

def record\_audio(b):

with audio\_output:

clear\_output()

try:

with sr.Microphone() as source:

print("Say something...")

audio\_text = speech\_recognizer.listen(source)

print("Recognizing...")

text = speech\_recognizer.recognize\_google(audio\_text)

print(f"Text: {text}")

display(Audio(data=audio\_text, autoplay=True))

except Exception as e:

print(f"Error: {e}")

record\_button.on\_click(record\_audio)

audio\_tab = widgets.VBox([record\_button, audio\_output])

# Audio Sentiment Analysis Tab

upload\_audio\_button = widgets.FileUpload(description="Upload Audio")

analyze\_audio\_button = widgets.Button(description="Analyze Sentiment")

audio\_sentiment\_output = widgets.Output()

def analyze\_audio\_sentiment(b):

with audio\_sentiment\_output:

clear\_output()

try:

audio\_data = next(iter(upload\_audio\_button.value.values()))['content']

# Perform sentiment analysis

sentiment\_scores = sentiment\_analyzer.polarity\_scores(str(audio\_data))

print("Sentiment Scores:", sentiment\_scores)

# Determine sentiment label

if sentiment\_scores['compound'] >= 0.05:

sentiment\_label = "Positive"

elif sentiment\_scores['compound'] <= -0.05:

sentiment\_label = "Negative"

else:

sentiment\_label = "Neutral"

print("Sentiment Label:", sentiment\_label)

except Exception as e:

print(f"Error: {e}")

analyze\_audio\_button.on\_click(analyze\_audio\_sentiment)

audio\_sentiment\_tab = widgets.VBox([upload\_audio\_button, analyze\_audio\_button, audio\_sentiment\_output])

# Display tabs

tabs = widgets.Tab(children=[text\_tab, image\_tab, audio\_tab, audio\_sentiment\_tab])

tabs.set\_title(0, 'Text Summarization')

tabs.set\_title(1, 'Image Classification')

tabs.set\_title(2, 'Speech Recognition')

tabs.set\_title(3, 'Audio Sentiment Analysis')

display(tabs)