Roll no – 2300290120271

Name – Vaishnavi Agrahari

Branch – Computer Science

Sec – D

Year – 2nd

Domain – Generative AI

ABOUT PROJECT-

Project - Create a comprehensive NLP solution.

Description - Use Azure AI Services to build and deploy an NLP solution for a real-world application.

This project Detect the Language and Analyse the Sentiments and Extract the Key Phrases and also Recognise the Entity.

Detailed Report on Code

Overview:

The provided Python code utilizes Azure's Text Analytics client to analyze text files within a specified directory. It covers several steps such as detecting language, analyzing sentiment, extracting key phrases, recognizing named entities, and identifying linked entities. The necessary credentials and endpoint for the Azure service are loaded from environment variables using the dotenv package.

Key Functions:

- **Environment Setup**:
    - Loading environment variables (AI_SERVICE_ENDPOINT and AI_SERVICE_KEY) using dotenv.
    - Setting up the Azure Text Analytics client using the endpoint and key.
- **Text File Analysis**:
    - Looping through text files in the specified folder (reviews).

- For each text file, performing language detection, sentiment analysis, key phrase extraction, entity recognition, and linked entity identification.

Table Summary:

Step	Description
Environment Setup	**Uses dotenv to load environment variables (AI_SERVICE_ENDPOINT, AI_SERVICE_KEY). Sets up TextAnalyticsClient.**
Text File Processing	**Reads text files from the reviews folder.**
Language Detection	**Detects the primary language of each text file using detect_language.**
Sentiment Analysis	**Analyzes the sentiment (positive, negative, neutral) of each text file using analyze_sentiment.**
Key Phrase Extraction	**Extracts key phrases from each text file using extract_key_phrases.**
Named Entity Recognition	**Identifies named entities (people, places, organizations) using recognize_entities.**

Linked Entity Recognition	**Identifies linked entities (like Wikipedia articles) using recognize_linked_entities.**

Code Walkthrough:

1. **Imports and Setup**:

```
from dotenv import load_dotenv
import os
from azure.core.credentials import AzureKeyCredential
from azure.ai.textanalytics import TextAnalyticsClient

load_dotenv()
ai_endpoint = os.getenv('AI_SERVICE_ENDPOINT')
ai_key = os.getenv('AI_SERVICE_KEY')
credential = AzureKeyCredential(ai_key)
ai_client = TextAnalyticsClient(endpoint=ai_endpoint, credential=credential)
```

2. **Processing Files**:

```
reviews_folder = 'reviews'
for file_name in os.listdir(reviews_folder):
    print('\n-------------------------------------------------\n' + file_name)
    text = open(os.path.join(reviews_folder, file_name),
encoding='utf8').read()
    print('\n' + text)
```

3. **Language Detection**:

```
detectedLanguage = ai_client.detect_language(documents=[text])[0]
print('\nLanguage: {}'.format(detectedLanguage.primary_language.name))
```

4. **Sentiment Analysis**:

```
sentimentAnalysis = ai_client.analyze_sentiment(documents=[text])[0]
print("\nSentiment: {}".format(sentimentAnalysis.sentiment))
```

5. **Key Phrase Extraction**:

```
phrases =
ai_client.extract_key_phrases(documents=[text])[0].key_phrases
if len(phrases) > 0:
    print("\nKey Phrases:")
    for phrase in phrases:
        print('\t{}'.format(phrase))
```

6. **Named Entity Recognition**:

```
entities = ai_client.recognize_entities(documents=[text])[0].entities
if len(entities) > 0:
    print("\nEntities")
    for entity in entities:
        print('\t{} ({})'.format(entity.text, entity.category))
```

7. **Linked Entity Recognition**:

```
entities =
ai_client.recognize_linked_entities(documents=[text])[0].entities
```

```
if len(entities) > 0:
    print("\nLinks")
    for linked_entity in entities:
        print('\t{} ({})'.format(linked_entity.name, linked_entity.url))
```

Exception Handling: If any exceptions occur during the execution, they are caught and printed:

```
except Exception as ex:
    print(ex)
```