Name : Aryan Deepak Saraf
Div : D20B
Roll No. : 49

AI&DS2 Experiment 02

Aim : To build a Cognitive text based application to understand context for a Customer service application/ Insurance / Healthcare Application / Smarter Cities / Government etc.

Theory :

What is a Cognitive Text-Based Application?
A cognitive text-based application is a smart software system that understands and responds to human language in a way that feels natural. It uses Artificial Intelligence (AI) techniques to analyze text input from users, identify their intent, and provide appropriate responses. For example, a chatbot for customer service can answer questions about product details or troubleshoot issues by understanding the user's query.

What is Cognitive Computing?
Cognitive computing is a part of AI that mimics how humans think, learn, and make decisions. It enables computers to:

- Understand: Interpret human language, including text or speech.
- Learn: Improve responses over time by learning from data and user interactions.
- Reason: Make decisions or predictions based on patterns in data.

Cognitive systems are designed to handle large amounts of information, recognize patterns, and provide helpful, human-like responses. They are widely used in applications like chatbots, virtual assistants, and automated customer support.

What is Natural Language Processing (NLP)?
Natural Language Processing (NLP) is a field of AI that allows computers to understand, interpret, and respond to human language. It involves several key processes:

- Tokenization: Breaking text into smaller parts, like words or sentences, for analysis.
- Stopword Removal: Ignoring common words (e.g., "the", "is", "and") that don't add significant meaning.
- Lemmatization/Stemming: Reducing words to their base or root form (e.g., "running" to "run").
- Entity Recognition: Identifying specific terms like names, places, or numbers in the text.

In this experiment, libraries like spaCy and NLTK are used for NLP tasks to preprocess and analyze user input effectively.

Keyword-Based Analysis
Keyword-based analysis is a simple yet effective technique for understanding user queries. The system looks for specific words or phrases (keywords) in the user's input to determine the intent or topic. For example, if a user asks, "How to water plants?", the system detects keywords like "water" and "plants" and responds with relevant advice. This method is useful for building basic chatbots that match user queries to predefined responses.

Applications in Different Sectors

Cognitive text-based applications are transforming various industries by automating tasks and improving user interaction:

- Customer Service: Chatbots answer common queries, provide product information, or resolve issues, reducing human workload.
- Insurance: Applications help users understand policy details, file claims, or get premium information.
- Healthcare: Virtual assistants offer advice, monitor symptoms, or provide medical information based on user input.
- Smarter Cities: Systems analyze citizen queries to manage resources, provide traffic updates, or improve urban planning.
- Government: Cognitive applications assist with policy analysis, citizen engagement, or crisis management by processing large datasets.

Steps in Building the Application

The process of creating a cognitive text-based application involves:

1. Data Preparation: Create a list of sample user queries and their corresponding responses specific to the chosen domain.
2. Text Preprocessing: Clean and prepare user input using NLP techniques like lowercase conversion, tokenization, and stopword removal.
3. Keyword Matching: Identify keywords in the user's input to classify the query and select an appropriate response.
4. Response Generation: Provide a relevant response based on the classified query or a default message for unrecognized inputs.
5. User Interaction: Enable continuous interaction with the user through a chatbot interface until the user chooses to exit.

Libraries Used

- spaCy: A powerful library for advanced NLP tasks like tokenization, lemmatization, and entity recognition.
- NLTK (Natural Language Toolkit): A Python library for text analysis, including tokenization, stopword removal, and part-of-speech tagging.

Code :

STEP 1: Install required libraries

```
!pip install spacy --quiet
!python -m spacy download en_core_web_sm --quiet
import nltk
nltk.download('punkt')
nltk.download('stopwords')
nltk.download('punkt_tab')
```

STEP 2: Import libraries and load spaCy model

```
import spacy
import nltk
```

```
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords

# Load spaCy English model
nlp = spacy.load("en_core_web_sm")
```

STEP 3: Define sample queries and responses for smart gardening

```
queries_and_responses = [
    ("water plants", "Most houseplants need watering once a week. Check soil moisture before watering to avoid overwatering."),
    ("plant sunlight", "Place sun-loving plants like succulents in bright, indirect light for 6-8 hours daily."),
    ("pest control", "Use neem oil spray to manage common pests like aphids or spider mites safely."),
    ("fertilizer use", "Apply a balanced liquid fertilizer every 4-6 weeks during the growing season (spring/summer)."),
    ("pruning tips", "Prune dead or yellow leaves with clean scissors to encourage new growth."),
    ("soil type", "Well-draining soil with perlite or sand works best for most indoor plants."),
    ("repotting plants", "Repot every 1-2 years or when roots are crowded, using a pot 1-2 inches larger."),
    ("yellow leaves", "Yellow leaves may indicate overwatering, low light, or nutrient deficiency. Check conditions."),
    ("indoor plants", "Low-maintenance indoor plants include pothos, snake plant, and ZZ plant.")
]

# Default responses for greetings, farewells, and unrecognized queries
default_responses = {
    "greeting": "Hello! I'm your Smart Gardening Assistant. Ask me about plant care, watering, or pests!",
    "farewell": "Happy gardening! Come back with more plant questions!",
    "default": "Sorry, I didn't catch that. Try asking about watering, sunlight, or pest control."
}
```

STEP 4: Preprocess text function

```
def preprocess_text(text):
    # Convert to lowercase
    text = text.lower()
    # Tokenize using NLTK
    tokens = word_tokenize(text)
    # Remove stopwords
    stop_words = set(stopwords.words('english'))
    tokens = [token for token in tokens if token not in stop_words]
    # Process with spaCy for lemmatization
    doc = nlp(" ".join(tokens))
    lemmas = [token.lemma_ for token in doc]
    return set(lemmas)
```

STEP 5: Classify query function

```
def classify_query(user_query):
    # Preprocess user query
    user_tokens = preprocess_text(user_query)

    # Check for greetings or farewells
    if user_tokens.intersection({"hi", "hello", "hey"}):
        return "greeting"
    elif user_tokens.intersection({"bye", "goodbye", "exit"}):
        return "farewell"

    # Check for keyword matches in queries_and_responses
    for keywords, response in queries_and_responses:
        keyword_tokens = set(preprocess_text(keywords))
        if user_tokens.intersection(keyword_tokens):
            return response

    return "default"
```

STEP 6: Main chatbot loop

```
def chatbot():
    print("Smart Gardening Assistant: Type your question or 'bye' to exit.")
    while True:
        user_query = input("You: ")
        response_type = classify_query(user_query)

        if response_type == "greeting":
            print("Assistant:", default_responses["greeting"])
        elif response_type == "farewell":
            print("Assistant:", default_responses["farewell"])
            break
        elif response_type == "default":
            print("Assistant:", default_responses["default"])
        else:
            print("Assistant:", response_type)
```

STEP 7: Run the chatbot

```
if __name__ == "__main__":
    chatbot()
```

```
Smart Gardening Assistant: Type your question or 'bye' to exit.
You: hello
Assistant: Hello! I'm your Smart Gardening Assistant. Ask me about plant care, watering, or pests!
You: how to water plants
Assistant: Most houseplants need watering once a week. Check soil moisture before watering to avoid overwatering.
You: pest control
Assistant: Use neem oil spray to manage common pests like aphids or spider mites safely.
You: something random
Assistant: Sorry, I didn't catch that. Try asking about watering, sunlight, or pest control.
You: bye
Assistant: Happy gardening! Come back with more plant questions!
```

Conclusion : This experiment successfully demonstrates the development of a cognitive text-based application that uses NLP and keyword-based analysis to understand and respond to user queries in a context-aware manner. By leveraging spaCy and NLTK libraries, the chatbot effectively processes input and provides relevant responses, showcasing the potential of AI in automating tasks for domains like customer service, healthcare, or smarter cities. The approach highlights the importance of text preprocessing and intent classification in building intelligent, user-friendly applications.

49_Prac_02