SURYA GROUPS OF INSTITUTIONS

Naan Mudhalvan

IBM - Artificial Intelligence

GNANAVEL B

422221104012

Team - 03

Create A Chatbot In Python

Problem Definition :

Creating a chatbot in Python is a fun and rewarding project that can help you learn more about natural language processing and artificial intelligence. There are many libraries and frameworks that you can use to build a chatbot, but one of the most popular and easy to use is ChatterBot. ChatterBot is a Python library that allows you to create a chatbot that can learn from existing conversations and generate responses based on the context and logic of the dialogue. You can also customize the chatbot's personality, language, and knowledge base to suit your needs.

Data Source :

To create a chatbot in Python using ChatterBot, you need to follow these steps:

- Install the ChatterBot and chatterbot_corpus modules using pip1.
- Import the ChatBot class from chatterbot and the ListTrainer class from chatterbot.trainers2.
- Create an instance of the ChatBot class and give it a name2.
-
- Create an instance of the ListTrainer class and pass it the chatbot object2.

- **Train the chatbot on some sample conversations using the train method of the trainer object23.**
- **Start a conversation with the chatbot using a while loop and the get_response method of the chatbot object2.**

Sample Program :

```
 # Import the modules

from chatterbot import ChatBot

from chatterbot.trainers import ListTrainer



# Create a chatbot

chatbot = ChatBot("MyChatBot")



# Create a trainer

trainer = ListTrainer(chatbot)



# Train the chatbot on some sample conversations

trainer.train([

    "Hi",

    "Hello",

    "How are you?",

    "I'm fine, thank you.",

    "What is your name?",

    "My name is MyChatBot.",

    "Nice to meet you.",

    "Nice to meet you too."
])


# Start a conversation with the chatbot

while True:

    user_input = input("You: ")

    if user_input.lower() == "quit":

        break

    response = chatbot.get_response(user_input)

    print("ChatBot: ", response)
```

Sample Output :

You can run this code in your Python interpreter or save it as a script and execute it. You should see something like this :

```
You: Hi
ChatBot:  Hello
You: How are you?
ChatBot:  I'm fine, thank you.
You: What is your hobby?
ChatBot:  I don't have any hobbies.
You: quit
```

Data Preprocessor :

Step 1 — Setting Up Your Environment

Make sure you are in the directory where you set up your environment and then run the following command:

```
source my_env/bin/activate
```

Now install spaCy :

```
pip install -U spacy
```

Run the following command :

```
python -m spacy download en_core_web_md
```

If you run into an error like the following :

Output :

```
ERROR: Failed building wheel for en-core-web-md
```

You need to install wheel:

```
pip install -U wheel
```

Then download the English-language model again.

To confirm that you have spaCy installed properly, open the Python interpreter :

```
Python
```

Next, import spaCy and load the English-language model :

```
>>> import spacy
>>> nlp = spacy.load("en_core_web_md")
```

If those two statements execute without any errors, then you have spaCy installed.

Now close the Python interpreter :

```
>>> exit()
```

Step 2 — Creating the City Weather Program

First, create and open a Python file called weather_bot.py with your preferred editor :

```
nano weather_bot.py
```

Add the following code into your weather_bot.py file:

```
 import requests
api_key = "your_api_key"

def get_weather(city_name):
        api_url =
"http://api.openweathermap.org/data/2.5/weather?q={}&appid={}".
format(city_name, api_key)

        response = requests.get(api_url)
        response_dict = response.json()

    weather = response_dict["weather"][0]["description"]

        if response.status_code == 200:
                return weather
        else:
                print('[!] HTTP {0} calling
[{1}]'.format(response.status_code, api_url))
                return None
```

Step 3 — Creating the Chatbot

To begin, open the script:

```
nano weather_bot.py
```

Then, import spaCy and load the English language model:

```
import spacy
import requests

nlp = spacy.load("en_core_web_md")
```

Following your definition, add the highlighted code to create tokens for the two statements you'll be comparing. Tokens are the different meaningful segments of a statement, like words and punctuation. This is necessary to allow spaCy to compute the semantic similarity:. . .

```
import spacy


 . . .

def chatbot(statement):
weather = nlp("Current weather in a city")
statement = nlp(statement)
```