AI_Phase4

October 20, 2023

1 Create a chatbot in python

Phase 4: Development Part 2

In this part you will continue building your project.

Continue building the chatbot by integrating it into a web app using Flask.

Continuation of Phase 3 starts from the topic "Integrating it into a web app using Flask" (Phase 4)

1.1 Importing required libraries

[1]:
```
import numpy as np
import string
from nltk.corpus import stopwords
import pandas as pd
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.tree import DecisionTreeClassifier
from sklearn.feature_extraction.text import TfidfTransformer,TfidfVectorizer
from sklearn.pipeline import Pipeline
```

1.2 Loading Dataset

[2]:
```
df = pd.read_csv('dataset/dialogs.txt',sep='\t')
```

[3]:
```
a = pd.Series(df.columns)
```

[4]:
```
df
```

[4]:
```
                                 hi, how are you doing?  \
0                        i'm fine. how about yourself?
1                  i'm pretty good. thanks for asking.
2                    no problem. so how have you been?
3                     i've been great. what about you?
4              i've been good. i'm in school right now.
…                                                    …
3719    that's a good question. maybe it's not old age.
3720                              are you right-handed?
```

```
3721                                 yes. all my life.
3722  you're wearing out your right hand. stop using…
3723        but i do all my writing with my right hand.

                          i'm fine. how about yourself?
0                   i'm pretty good. thanks for asking.
1                     no problem. so how have you been?
2                      i've been great. what about you?
3                i've been good. i'm in school right now.
4                               what school do you go to?
…                                                     …
3719                               are you right-handed?
3720                                   yes. all my life.
3721  you're wearing out your right hand. stop using…
3722        but i do all my writing with my right hand.
3723  start typing instead. that way your left hand …

[3724 rows x 2 columns]
```

1.3 Pre Processing the dataset

[5]:
```
a = a.rename({0: df.columns[0],1: df.columns[1]})
```

[6]:
```
b = {'Questions':'Hi','Answers':'hello'}
```

[7]:
```
c = {'Questions':'Hello','Answers':'hi'}
```

[8]:
```
d= {'Questions':'how are you','Answers':"i'm fine. how about yourself?"}
```

[9]:
```
e= {'Questions':'how are you doing','Answers':"i'm fine. how about yourself?"}
```

[10]:
```
df = df.append(a,ignore_index=True)
```

```
/tmp/ipykernel_21803/3772295606.py:1: FutureWarning: The frame.append method is
deprecated and will be removed from pandas in a future version. Use
pandas.concat instead.
  df = df.append(a,ignore_index=True)
```

[11]:
```
df.columns=['Questions','Answers']
```

[12]:
```
df = df.append([b,c,d,e],ignore_index=True)
```

```
/tmp/ipykernel_21803/10559575.py:1: FutureWarning: The frame.append method is
deprecated and will be removed from pandas in a future version. Use
pandas.concat instead.
  df = df.append([b,c,d,e],ignore_index=True)
```

[13]:
```
df
```

```
[13]:                                         Questions  \
      0                   i'm fine. how about yourself?
      1             i'm pretty good. thanks for asking.
      2               no problem. so how have you been?
      3                i've been great. what about you?
      4        i've been good. i'm in school right now.
      …                                             …
      3724                     hi, how are you doing?
      3725                                         Hi
      3726                                      Hello
      3727                                how are you
      3728                          how are you doing

                                                Answers
      0             i'm pretty good. thanks for asking.
      1               no problem. so how have you been?
      2                i've been great. what about you?
      3        i've been good. i'm in school right now.
      4                       what school do you go to?
      …                                             …
      3724                  i'm fine. how about yourself?
      3725                                      hello
      3726                                         hi
      3727                  i'm fine. how about yourself?
      3728                  i'm fine. how about yourself?

      [3729 rows x 2 columns]
```

```
[14]: df = df.append(c,ignore_index=True)
```

```
/tmp/ipykernel_21803/921143614.py:1: FutureWarning: The frame.append method is
deprecated and will be removed from pandas in a future version. Use
pandas.concat instead.
  df = df.append(c,ignore_index=True)
```

```
[15]: df = df.append(d,ignore_index=True)
```

```
/tmp/ipykernel_21803/1512068047.py:1: FutureWarning: The frame.append method is
deprecated and will be removed from pandas in a future version. Use
pandas.concat instead.
  df = df.append(d,ignore_index=True)
```

```
[16]: df = df.append(d,ignore_index=True)
```

```
/tmp/ipykernel_21803/1512068047.py:1: FutureWarning: The frame.append method is
deprecated and will be removed from pandas in a future version. Use
pandas.concat instead.
  df = df.append(d,ignore_index=True)
```

```
[17]: df
```

```
[17]:                                    Questions  \
0                i'm fine. how about yourself?
1          i'm pretty good. thanks for asking.
2            no problem. so how have you been?
3             i've been great. what about you?
4     i've been good. i'm in school right now.
…                                          …
3727                                how are you
3728                          how are you doing
3729                                      Hello
3730                                how are you
3731                                how are you

                                       Answers
0          i'm pretty good. thanks for asking.
1            no problem. so how have you been?
2             i've been great. what about you?
3     i've been good. i'm in school right now.
4                    what school do you go to?
…                                          …
3727             i'm fine. how about yourself?
3728             i'm fine. how about yourself?
3729                                        hi
3730             i'm fine. how about yourself?
3731             i'm fine. how about yourself?

[3732 rows x 2 columns]
```

```
[18]: def cleaner(x):
          return [a for a in (''.join([a for a in x if a not in string.punctuation])).
       ↪lower().split()]
```

```
[19]: df.head()
```

```
[19]:                                Questions  \
0             i'm fine. how about yourself?
1       i'm pretty good. thanks for asking.
2         no problem. so how have you been?
3          i've been great. what about you?
4  i've been good. i'm in school right now.

                                    Answers
0       i'm pretty good. thanks for asking.
1         no problem. so how have you been?
2          i've been great. what about you?
```

```
3  i've been good. i'm in school right now.
4                what school do you go to?
```

1.4 Training model

[20]:
```python
model = Pipeline([
    ('bow',CountVectorizer(analyzer=cleaner)),
    ('tfidf',TfidfTransformer()),
    ('classifier',DecisionTreeClassifier())
])
```

[22]:
```python
model.fit(df['Questions'],df['Answers'])
```

[22]:
```
Pipeline(steps=[('bow',
                  CountVectorizer(analyzer=<function cleaner at
0x7f9dde5317e0>)),
                 ('tfidf', TfidfTransformer()),
                 ('classifier', DecisionTreeClassifier())])
```

1.5 Testing model

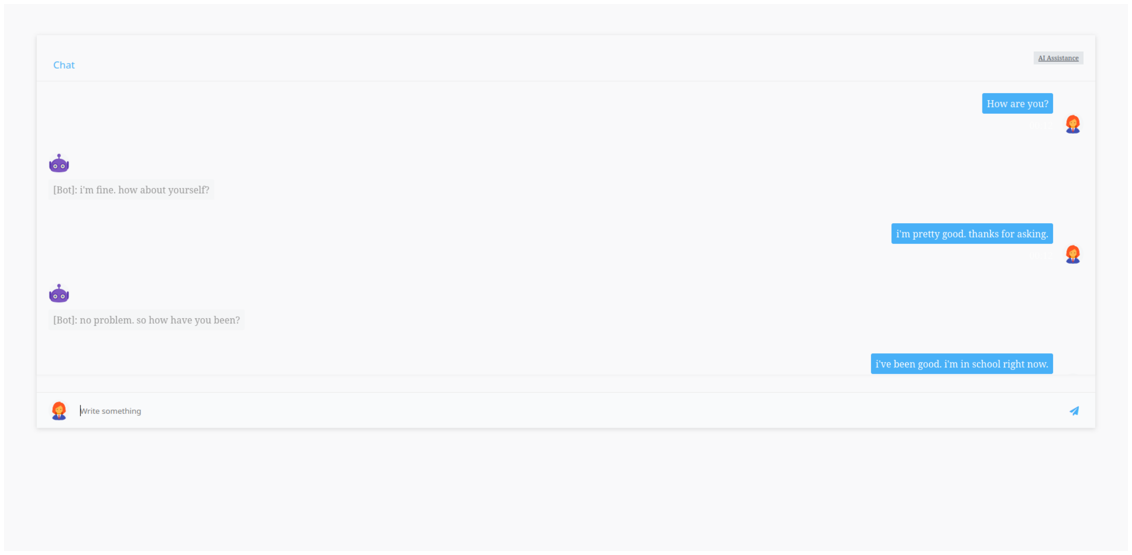
[23]:
```python
model.predict(['how are you'])[0]
```

[23]:
```
"i'm fine. how about yourself?"
```

1.6 Integrating it into a web app using Flask

1.7 Environment and Implementing basic user interactions

https://github.com/kimpoobi/Create_chatbot_in_python/blob/main/app/templates/index.html



How are you?

[Bot]: i'm fine. how about yourself?

i'm pretty good. thanks for asking.

[Bot]: no problem. so how have you been?

i've been good. i'm in school right now.

Explanation of about HTML and Javascript for user interface:

HTML Part:

1. The HTML code defines a basic webpage structure for the chatbot interface.
2. It includes external script and style libraries, such as jQuery for JavaScript functionality, Socket.IO for real-time communication, and Bootstrap for styling.
3. It also includes Font Awesome icons and a custom CSS file (style.css) for styling.
4. Within the `body` element, there's a chat interface with a message display area and an input field for sending messages.

JavaScript Part:

1. JavaScript code is used to make the chatbot interface interactive and handle real-time communication with the server.
2. It connects to the Socket.IO server running at "http://localhost:5000" to establish a real-time connection with the server.
3. It listens for the 'connect' event and sends a "User Connected!" message to the server when the user initially connects.
4. It listens for incoming messages from the server and displays them in the chat interface. Messages are displayed in different styles depending on whether they are from the user or the chatbot.
5. The 'Send' button and the Enter key press are both set to trigger a function that sends the user's input message to the server when clicked or when the Enter key is pressed.
6. The function also appends the user's message to the chat interface and clears the input field after sending.
7. Overall, the JavaScript code provides real-time chat functionality and a user-friendly interface for interacting with the chatbot.

This code combines HTML and JavaScript to create a functional and visually appealing chatbot interface that communicates with the Python server running your chatbot logic. Users can send messages, and the chatbot responds in real-time, making it an interactive and engaging experience.

1.8 integration and flask for web app development

app.py https://github.com/kimpoobi/Create_chatbot_in_python/blob/main/app/app.py

[ ]:
```
from flask import Flask, render_template
from flask_socketio import SocketIO, send
import pickle
import sklearn
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.tree import DecisionTreeClassifier
from sklearn.feature_extraction.text import TfidfTransformer,TfidfVectorizer
from sklearn.pipeline import Pipeline
import string
```

```python
app = Flask(__name__)
app.config['SECRET'] = 'somethingelse'
socketio = SocketIO(app, cors_allowed_origins="*")

def cleaner(x):
    return [a for a in (''.join([a for a in x if a not in string.punctuation])).
  ↪lower().split()]
def bot_replay(message):
    pred_msg = ""
    with open('static/models/dclf.pkl', 'rb') as f:
        clf2 = pickle.load(f)
        pred_msg = clf2.predict([message])[0]

    return pred_msg

@socketio.on('message')
def handle_message(message):
    print("Received message: " + message)
    if message != "User Connected!":
        send(message, broadcast=True)
        send("[Bot]: " + bot_replay(message), broadcast=True)


@app.route('/')
def index():
    return render_template("index.html")


if __name__ == "__main__":
    socketio.run(app, debug=True, host="localhost")
```

Explanation of above code:

1. **Importing Libraries:**
   - The code starts by importing the necessary libraries. Flask is used for creating the web application, and Flask-SocketIO is used for handling real-time communication between the server and the client. Additionally, the code imports libraries related to machine learning, including scikit-learn (sklearn), which is used for text classification, and pickle for model serialization.
2. **Flask Application Setup:**
   - The Flask application is created and configured. A Flask `app` instance is created, and the `app.config` is set with a 'SECRET' key.
3. **SocketIO Setup:**
   - The Flask-SocketIO extension is used to set up a WebSocket for real-time communication. A `socketio` instance is created, and it is associated with the Flask app.
4. **Text Cleaning Function:**
   - The `cleaner` function is defined to preprocess text data. It removes punctuation, con-

verts text to lowercase, and splits the text into words. This function is used to prepare user input for the machine learning model.

5. **Bot Response Function:**
    - The `bot_replay` function is responsible for generating responses from the chatbot. It uses a pre-trained machine learning model to classify user messages and return a corresponding response. The model is loaded from a file called 'dclf.pkl'.
6. **SocketIO Event Handler:**
    - The `@socketio.on('message')` decorator defines an event handler for messages received from the client. When a message is received, it is printed to the console. If the message is not "User Connected!", it is broadcasted to all connected clients, and the chatbot response (generated by the `bot_replay` function) is also sent to all clients with the "[Bot]" prefix.
7. **Flask Route:**
    - A Flask route is defined for the root URL '/'. When a user accesses the root URL, the 'index.html' template is rendered. This template presumably contains the chat interface.
8. **Main Application Execution:**
    - The application is configured to run only if the script is the main entry point. It uses the `socketio.run()` method to start the Flask-SocketIO server with debugging enabled on 'localhost'.

Project Components: - The project consists of a Flask web application that serves an HTML interface (index.html) for users to interact with the chatbot. - The chatbot uses a pre-trained decision tree classifier (loaded from 'dclf.pkl') to classify user messages and generate responses. - Real-time communication is established using Flask-SocketIO, allowing for seamless user-bot interactions.

To Run the Project: - Ensure you have the required libraries installed (Flask, Flask-SocketIO, scikit-learn). - Create the 'dclf.pkl' file with a trained decision tree classifier. - Run the Python script, and access the chatbot via a web browser.

[ ]:

```
```