chatbot

May 19, 2024

[1]:

```
import numpy as np
import nltk
import string
import random
```

[2]:

```
f=open('chatbot.txt','r',errors='ignore')
raw_doc=f.read()
raw_doc=raw_doc.lower()
nltk.download('punkt')#tokenizer
nltk.download('wordnet')#wordnet
sent_tokens=nltk.sent_tokenize(raw_doc)
word_tokens=nltk.word_tokenize(raw_doc)
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     /home/4e79f932-cc0a-4092-9ae1-
[nltk_data]     e5c5a6784f57/nltk_data…
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package wordnet to
[nltk_data]     /home/4e79f932-cc0a-4092-9ae1-
[nltk_data]     e5c5a6784f57/nltk_data…
[nltk_data]   Package wordnet is already up-to-date!
```

[3]:

```
sent_tokens[:2]
```

[3]:

```
['nobody likes to be alone always, but sometimes loneliness could be a better
medicine to hunch the thirst for a peaceful environment.',
 'even during such lonely quarantines, we may ignore humans but not humanoids.']
```

[4]:

```
word_tokens[:2]
```

[4]:

```
['nobody', 'likes']
```

[5]:

```
 lemmer=nltk.stem.WordNetLemmatizer()
def LemTokens(tokens):
    return[lemmer.lemmatize(token)for token in tokens]
remove_punct_dict=dict((ord(punct),None) for punct in string.punctuation)
def LemNormalize(text):
    return LemTokens(nltk.word_tokenize(text.lower().
 ↪translate(remove_punct_dict)))
```

[6]:

```
GREET_INPUTS=("hello","hi","greetings","sup","whats'up","hey")
GREET_RESPONSES=["hi","hey","nods","i am glad to talk u","hello"]
def greet(sentence):
    for word in sentence.split():
        if word.lower() in GREET_INPUTS:
            return random.choice(GREET_RESPONSES)
```

[7]:

```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
```

[8]:

```
def response(user_response):
    robo1_response=''
    TfidfVec=TfidfVectorizer(tokenizer=LemNormalize,stop_words='english')
    tfidf=TfidfVec.fit_transform(sent_tokens)
    vals=cosine_similarity(tfidf[-1],tfidf)
    idx=vals.argsort()[0][-2]
    flat=vals.flatten()
    flat.sort()
    req_tfidf=flat[-2]
    if(req_tfidf==0):
        robo1_response=robo1_response+"i am sorry! i dont understand you"
        return robo1_response
    else:
        robo1_response=robo1_response+sent_tokens[idx]
        return robo1_response
```

[ ]:

```
flag=True
print("Bot:i m star.do u wanna have conversation let begin and if want exit␣
 ↪just type bye")
while(flag==True):
    user_response=input("you:")
    user_response=user_response.lower()
    if(user_response!='bye'):
        if(user_response=='thanks' or user_response=='thank you'):
           flag=False
           print("bot:u r welcome")
        else:
            if(greet(user_response)!=None):
                print("bot: "+greet(user_response))
            else:
                sent_tokens.append(user_response)
                word_tokens=word_tokens+nltk.word_tokenize(user_response)
                final_words=list(set(word_tokens))
                print("bot: ",end="")
                print(response(user_response))
                sent_tokens.remove(user_response)
    else:
        flag=False
        print("bot: goodbye")
```

```
Bot:i m star.do u wanna have conversation let begin and if want exit just type
bye

you: helli

bot:

/opt/conda/envs/anaconda-panel-2023.05-py310/lib/python3.11/site-
packages/sklearn/feature_extraction/text.py:525: UserWarning: The parameter
'token_pattern' will not be used since 'tokenizer' is not None'
  warnings.warn(
/opt/conda/envs/anaconda-panel-2023.05-py310/lib/python3.11/site-
packages/sklearn/feature_extraction/text.py:408: UserWarning: Your stop_words
may be inconsistent with your preprocessing. Tokenizing the stop words generated
tokens ['ha', 'le', 'u', 'wa'] not in stop_words.
  warnings.warn(

i am sorry! i dont understand you

you: hello

bot: hello

you: chatbot

bot: yes, if you have guessed this article for a chatbot, then you have cracked
it right.

you: python

bot: let us have a quick glance at python’s chatterbot to create our bot.
```

[ ]:

```

```

[ ]:

```

```