|  |
| --- |
| CREATE CHATBOT USING PYTHON |

#Proprietary content. © Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited.

Importing the required libraries

import numpy as np

import nltk

import string

import random

**Importing and reading the corpus**

f=open("/content/dialogs (1).txt" ,'r',errors = 'ignore')

raw\_doc=f.read()

raw\_doc=raw\_doc.lower() #Converts text to lowercase

nltk.download('punkt') #Using the Punkt tokenizer

nltk.download('wordnet') #Using the WordNet dictionary

sent\_tokens = nltk.sent\_tokenize(raw\_doc) #Converts doc to list of sentences

word\_tokens = nltk.word\_tokenize(raw\_doc) #Converts doc to list of words

[nltk\_data] Downloading package punkt to /root/nltk\_data...

[nltk\_data] Unzipping tokenizers/punkt.zip.

[nltk\_data] Downloading package wordnet to /root/nltk\_data...

**Example of sentance tokens**

sent\_tokens[:2]

['hi, how are you doing?', "i'm fine."]

**Example of word tokens**

word\_tokens[:2]

['hi', ',']

**Text preprocessing**

lemmer = nltk.stem.WordNetLemmatizer()

#WordNet is a semantically-oriented dictionary of English included in NLTK.

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)))

**Defining the greeting function**

GREET\_INPUTS = ("hello", "hi", "greetings", "sup", "what's up","hey")

GREET\_RESPONSES = ["hi", "hey", "\*nods\*", "hi there", "hello", "I am glad! You are talking to me"]

def greet(sentence):

    for word in sentence.split():

        if word.lower() in GREET\_INPUTS:

            return random.choice(GREET\_RESPONSES)

**Response generation**

from sklearn.feature\_extraction.text import TfidfVectorizer

from sklearn.metrics.pairwise import cosine\_similarity

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 don't understand you"

    return robo1\_response

  else:

    robo1\_response = robo1\_response+sent\_tokens[idx]

    return robo1\_response

**Defining conversation start/end protocols**

flag=True

print("BOT: My name is Stark. Let's have a conversation! Also, if you want to exit any time, just type Bye!")

while(flag==True):

    user\_response = input()

    user\_response=user\_response.lower()

    if(user\_response!='bye'):

        if(user\_response=='thanks' or user\_response=='thank you' ):

            flag=False

            print("BOT: You are 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! Take care <3 ")

BOT: My name is Stark. Let's have a conversation! Also, if you want to exit any time, just type Bye!

hi

BOT: hello

how are you doing?

BOT: what have you been doing?

i am fine how about yourself?

BOT: fine.

bye

BOT: Goodbye! Take care <3