In [1]:

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

import reading corpus

In [2]:

```
f = open('chatbot.txt.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 sentences
word_tokens = nltk.word_tokenize(raw_doc) # converts doc to list of words
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\sajid\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\sajid\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
```

In [3]:

```
sent_tokens[:2]
```

Out[3]:

```
['data science is an interdisciplinary field that uses scientific methods, processes, algorithms and systems to e
xtract knowledge and insights from noisy, structured and unstructured data,[1][2] and apply knowledge from data a
cross a broad range of application domains.',
 'data science is related to data mining, machine learning and big data.']
```

In [4]:

```
word_tokens[:2]
```

Out[4]:

```
['data', 'science']
```

text preprocessing

In [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)))
```

defining the greeting function

In [6]:

```
GREET_INPUTS = ("hello","hi","greetings", "sup", "what's up ", "hey")
GREET_INPUTS = ["hi","hey","*nods*","hi there", "hello","Iam 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

In [7]:

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

In [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):
        robol_response=robo1_response+"I am sorry ! I don't understsnd you"
        return robo1_response
    else:
        robo1_response = robo1_response+sent_tokens[idx]
        return robo1_response
```

Defining conversation start/end protocals

In [ ]:

```
flag=True
print("Bot: My name is Sajid.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:
        flat=False
        print("BOT: Goodbye! Take care <3")
```

In [ ]:

In [ ]:

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js