In [1]:

```
# Libraries needed for NLP
import nltk
nltk.download('punkt')
from nltk.stem.lancaster import LancasterStemmer
stemmer = LancasterStemmer()

# Libraries needed for Tensorflow processing
import tensorflow as tf
import numpy as np
import tflearn
import random
import json
```

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

```
WARNING:tensorflow:From C:\Users\LENOVO\anaconda3\lib\site-packages\tensorflow\python\com
pat\v2_compat.py:107: disable_resource_variables (from tensorflow.python.ops.variable_sco
pe) is deprecated and will be removed in a future version.
Instructions for updating:
non-resource variables are not supported in the long term
curses is not supported on this machine (please install/reinstall curses for an optimal e
xperience)
```

In [2]:

```
import pandas as pd
with open('intents.json') as json_data:
    intents = json.load(json_data)
```

In [3]:

```
intents
```

Out[3]:

```
{'intents': [{'tag': 'greeting',
   'patterns': ['Hi', 'Hello Amizone', 'Hello', 'Start'],
   'responses': ['Hello, This is Amizone, How can we help you?'],
   'context_set': ''},
  {'tag': 'goodbye',
   'patterns': ['Bye', 'See you later', 'Goodbye'],
   'responses': ['See you later, thanks for visiting Amizone',
    'Have a nice day',
    'Bye! Come back again soon.']},
  {'tag': 'thanks',
   'patterns': ['Thanks', 'Thank you', "That's helpful"],
   'responses': ['Happy to help!',
    'Any time!',
    'Our pleasure.',
    'Glad we were useful.']},
  {'tag': 'amizone',
   'patterns': ['What is amizone?', 'What are you?'],
   'responses': ['Amizone is a unique comprehensive Cloud-based Learning Management Syste
m. Students and Faculty continuously interact through Amizone one of the most hi-tech uni
versity intranet portals in the world that brings a world-class online experience for Ami
ty students.']},
  {'tag': 'attendance',
   'patterns': ['What is the minimum attendance required for each subject?',
    'How to check attendance?'],
   'responses': ['You need atleast 75% attendance in every subject to avoid getting debar
red. You can check your attendance for every course in the pie chart on the home section.
']},
  {'tag': 'courses',
   'patterns': ['What are my courses in this semester?',
    'How many courses do i have this semester?',
    'Where can i find the list of all the courses for this semester?']
```

```
    'Where can I find the list of all the courses for this semester? '],
   'responses': ["You can find all the details about your subjects/courses in 'My Courses
' section in the side menu on the left."]},
  {'tag': 'faculty',
   'patterns': ['Which faculties will teach me this semester?',
    'Which faculties are alloted to me?',
    'Faculty',
    'Faculties alloted for this semester'],
   'responses': ["You can find all the details about the faculty in 'My Faculty' section
in the side menu on the left."]},
  {'tag': 'todaystimetable',
   'patterns': ['timetable'],
   'responses': ['You can find the timetable on the home section of Amizone',
    'You can check the timetable for today in the side menu on the left.']},
  {'tag': 'classes',
   'patterns': ['How many classes do i have to attend every day?',
    'How many classes will there be in total?',
    'Classes'],
   'responses': ['There will be Classes from 9:15 A.M. till 5:10 P.M. with lunch break fr
om 1:10 P.M. to 2:15 P.M., there will classes 5 days a week i.e. monday to friday.',
    'You can check the timetable for today in drop down menu in the left side.']},
  {'tag': 'fees',
   'patterns': ['Fees'],
   'responses': ['You can find the fees section in side menu on the left. Here you can fi
nd every related detail regarding fees.']},
  {'tag': 'LMS',
   'patterns': ['How can i access the LMS?',
    'How can i access the lms?',
    'Where is LMS?',
    'Where is lms?',
    'LMS',
    'lms'],
   'responses': ['Scroll down the home section of to access the LMS.']},
  {'tag': 'registration',
   'patterns': ['Registration for next semester',
    'How to register for next semester?'],
   'responses': ['You can register for next semester in side menu on the left.']},
  {'tag': 'profile',
   'patterns': ['profile'],
   'responses': ['Go to the home section of Amizone and then on the navigation bar, then
click on the right side of the bar to access your profile.']},
  {'tag': 'hostel',
   'patterns': ['whom should i contact to queries regarding hostel?',
    'hostel'],
   'responses': ['For Hostel related information click on the hostel section in the side
menu on the left and register in it. You can also contact the class coordinator for more
details.']},
  {'tag': 'hostelhours',
   'patterns': ['what are hostel hours?', 'Hostel hours'],
   'responses': ['Hostellers are not allowed to go outside the university campus before 1
0:00 A.M. and after 8:00 A.M., to go outside the university campus the student must carry
his/her ID card and fill up the entries at the university gate.']},
  {'tag': 'notices',
   'patterns': ['Notice', 'Is there any new notice?'],
   'responses': ['You an check the bell icon on the navigation bar on the home secction o
f amizone for new notices.',
    'Scroll down the home section for notices.']},
  {'tag': 'wifi',
   'patterns': ['How to enroll for wifi services in amity university?',
    'wifi',
    'wifi services'],
   'responses': ['Contact the IT department about wifi registration at A-Block in the cam
pus.']},
  {'tag': 'examination',
   'patterns': ['How to register for End semester exams?',
    'Exam registration'],
   'responses': ['Open the side menu on the left, click on Examination and then in the dr
op down menu click on Examination Form for registration.']},
  {'tag': 'results',
   'patterns': ['How to check the exam result?', 'result', 'Exam result'],
   'responses': ['Open the side menu on the left, click on Examination and then in the dr
op down menu click on Examination Form for exam results.']},
  {'tag': 'calendar'
```

```
  {'tag': 'calendar',
   'patterns': ['calendar'],
   'responses': ['The calendar shows the daily classes and forthcoming happenings. You ca
n access the calender section in the side menu on the left or click on the calendar icon
on the navigation bar at the home section.']},
  {'tag': 'Scholarship',
   'patterns': ['scholarship'],
   'responses': ['Contact your class coordinator.']},
  {'tag': 'aboutus',
   'patterns': ['about amity', 'about'],
   'responses': ['https://www.amity.edu/gwalior/']},
  {'tag': 'placement',
   'patterns': ["what was last year's placement record?"],
   'responses': ['https://www.amity.edu/gwalior/about-placements']},
  {'tag': 'amityfaculty',
   'patterns': ['Faculties of other departments', 'amity faculty'],
   'responses': ['https://www.amity.edu/gwalior/about-faculty']},
  {'tag': 'location',
   'patterns': ['Where is amity university madhya pradesh?',
    'amity university location',
    'Where is amity university madhya pradesh situated? '],
   'responses': ['https://www.google.com/maps/place/Amity+University/@26.2727152,78.22620
1,17z/data=!3m1!4b1!4m5!3m4!1s0x3976c0e647bece07:0xd848338a0d6a5393!8m2!3d26.2727152!4d78
.2283897']}]}
```

In [4]:

```
words = []
classes = []
documents = []
ignore = ['?']
# loop through each sentence in the intent's patterns
for intent in intents['intents']:
    for pattern in intent['patterns']:
        # tokenize each and every word in the sentence
        w = nltk.word_tokenize(pattern)
        # add word to the words list
        words.extend(w)
        # add word(s) to documents
        documents.append((w, intent['tag']))
        # add tags to our classes list
        if intent['tag'] not in classes:
            classes.append(intent['tag'])
```

In [5]:

```
# Perform stemming and lower each word as well as remove duplicates
words = [stemmer.stem(w.lower()) for w in words if w not in ignore]
words = sorted(list(set(words)))

# remove duplicate classes
classes = sorted(list(set(classes)))

print (len(documents), "documents")
print (len(classes), "classes", classes)
print (len(words), "unique stemmed words", words)
```

```
59 documents
25 classes ['LMS', 'Scholarship', 'aboutus', 'amityfaculty', 'amizone', 'attendance', 'ca
lendar', 'classes', 'courses', 'examination', 'faculty', 'fees', 'goodbye', 'greeting', '
hostel', 'hostelhours', 'location', 'notices', 'placement', 'profile', 'registration', 'r
esults', 'thanks', 'todaystimetable', 'wifi']
92 unique stemmed words ["'s", 'about', 'access', 'al', 'allot', 'am', 'amizon', 'any', '
ar', 'attend', 'be', 'bye', 'calend', 'can', 'check', 'class', 'contact', 'cours', 'day',
'depart', 'do', 'each', 'end', 'enrol', 'every', 'exam', 'facul', 'fee', 'find', 'for', '
goodby', 'hav', 'hello', 'help', 'hi', 'hostel', 'hour', 'how', 'i', 'in', 'is', 'last',
'lat', 'list', 'lms', 'loc', 'madhy', 'many', 'me', 'minim', 'my', 'new', 'next', 'not',
'of', 'oth', 'plac', 'pradesh', 'profil', 'query', 'record', 'reg', 'regard', 'requir', '
result', 'scholarship', 'see', 'semest', 'serv', 'should', 'situ', 'start', 'subject', 't
each', 'thank', 'that', 'the', 'ther', 'thi', 'timet', 'to', 'tot', 'univers', 'was', 'wh
at', 'wher', 'which', 'whom', 'wif', 'wil', 'year', 'you']
```

In [6]:

```
# create training data
training = []
output = []
# create an empty array for output
output_empty = [0] * len(classes)

# create training set, bag of words for each sentence
for doc in documents:
    # initialize bag of words
    bag = []
    # list of tokenized words for the pattern
    pattern_words = doc[0]
    # stemming each word
    pattern_words = [stemmer.stem(word.lower()) for word in pattern_words]
    # create bag of words array
    for w in words:
        bag.append(1) if w in pattern_words else bag.append(0)

    # output is '1' for current tag and '0' for rest of other tags
    output_row = list(output_empty)
    output_row[classes.index(doc[1])] = 1

    training.append([bag, output_row])

# shuffling features and turning it into np.array
random.shuffle(training)
training = np.array(training)

# creating training lists
train_x = list(training[:,0])
train_y = list(training[:,1])
```

```
C:\Users\LENOVO\AppData\Local\Temp\ipykernel_42876\1242873455.py:27: VisibleDeprecationWa
rning: Creating an ndarray from ragged nested sequences (which is a list-or-tuple of list
s-or-tuples-or ndarrays with different lengths or shapes) is deprecated. If you meant to
do this, you must specify 'dtype=object' when creating the ndarray.
  training = np.array(training)
```

In [7]:

```
# resetting underlying graph data
tf.compat.v1.reset_default_graph()

# Building neural network
net = tflearn.input_data(shape=[None, len(train_x[0])])
net = tflearn.fully_connected(net, 10)
net = tflearn.fully_connected(net, 10)
net = tflearn.fully_connected(net, len(train_y[0]), activation='softmax')
net = tflearn.regression(net)

# Defining model and setting up tensorboard
model = tflearn.DNN(net, tensorboard_dir='tflearn_logs')

# Start training
model.fit(train_x, train_y, n_epoch=1000, batch_size=8, show_metric=True)
model.save('model.tflearn')
```

```
Training Step: 7999  | total loss: 0.04330 | time: 0.050s
| Adam | epoch: 1000 | loss: 0.04330 - acc: 0.9983 -- iter: 56/59
Training Step: 8000  | total loss: 0.04007 | time: 0.066s
| Adam | epoch: 1000 | loss: 0.04007 - acc: 0.9985 -- iter: 59/59
--
INFO:tensorflow:C:\Users\LENOVO\SIP\model.tflearn is not in all_model_checkpoint_paths. M
anually adding it.
INFO:tensorflow:C:\Users\LENOVO\SIP\model.tflearn.data-00000-of-00001
INFO:tensorflow:0
INFO:tensorflow:C:\Users\LENOVO\SIP\model.tflearn.index
INFO:tensorflow:0
INFO:tensorflow:C:\Users\LENOVO\SIP\model.tflearn.meta
INFO:tensorflow:100
```

In [8]:

```
import pickle
pickle.dump( {'words':words, 'classes':classes, 'train_x':train_x, 'train_y':train_y}, o
pen( "training_data", "wb" ) )
```

In [9]:

```
# restoring all the data structures
data = pickle.load( open( "training_data", "rb" ) )
words = data['words']
classes = data['classes']
train_x = data['train_x']
train_y = data['train_y']
```

In [10]:

```
with open('intents.json') as json_data:
    intents = json.load(json_data)
```

In [11]:

```
# load the saved model
model.load('./model.tflearn')
```

```
INFO:tensorflow:Restoring parameters from C:\Users\LENOVO\SIP\model.tflearn
```

In [12]:

```
def clean_up_sentence(sentence):
    # tokenizing the pattern
    sentence_words = nltk.word_tokenize(sentence)
    # stemming each word
    sentence_words = [stemmer.stem(word.lower()) for word in sentence_words]
    return sentence_words

# returning bag of words array: 0 or 1 for each word in the bag that exists in the senten
ce
def bow(sentence, words, show_details=False):
    # tokenizing the pattern
    sentence_words = clean_up_sentence(sentence)
    # generating bag of words
    bag = [0]*len(words)
    for s in sentence_words:
        for i,w in enumerate(words):
            if w == s:
                bag[i] = 1
                if show_details:
                    print ("found in bag: %s" % w)

    return(np.array(bag))
```

In [13]:

```
ERROR_THRESHOLD = 0.30
def classify(sentence):
    # generate probabilities from the model
    results = model.predict([bow(sentence, words)])[0]
    # filter out predictions below a threshold
    results = [[i,r] for i,r in enumerate(results) if r>ERROR_THRESHOLD]
    # sort by strength of probability
    results.sort(key=lambda x: x[1], reverse=True)
    return_list = []
    for r in results:
        return_list.append((classes[r[0]], r[1]))
    # return tuple of intent and probability
    return return_list

def response(sentence, userID='123', show_details=False):
    results = classify(sentence)
```

```
    # if we have a classification then find the matching intent tag
    if results:
        # loop as long as there are matches to process
        while results:
            for i in intents['intents']:
                # find a tag matching the first result
                if i['tag'] == results[0][0]:
                    # a random response from the intent
                    return print(random.choice(i['responses']))

            results.pop(0)
```

In [14]:

```
classify('Hello')
```

Out[14]:

```
[('greeting', 0.98105097)]
```

In [15]:

```
response('What is amizone?')
```

```
Amizone is a unique comprehensive Cloud-based Learning Management System. Students and Fa
culty continuously interact through Amizone one of the most hi-tech university intranet p
ortals in the world that brings a world-class online experience for Amity students.
```

In [16]:

```
response('timetable')
```

```
You can find the timetable on the home section of Amizone
```

In [17]:

```
#Some of other context free responses.
response('classes')
```

```
You can check the timetable for today in drop down menu in the left side.
```

In [18]:

```
response('Notice')
```

```
You an check the bell icon on the navigation bar on the home secction of amizone for new
notices.
```

In [19]:

```
response('attendance')
```

```
You need atleast 75% attendance in every subject to avoid getting debarred. You can check
your attendance for every course in the pie chart on the home section.
```

In [20]:

```
response('Bye')
```

```
See you later, thanks for visiting Amizone
```

In [21]:

```
#Adding some context to the conversation i.e. Contexualization for altering question and
intents etc.
# create a data structure to hold user context
context = {}

ERROR_THRESHOLD = 0.25
def classify(sentence):
    # generate probabilities from the model
    results = model.predict([bow(sentence, words)])[0]
```

```
    # filter out predictions below a threshold
    results = [[i,r] for i,r in enumerate(results) if r>ERROR_THRESHOLD]
    # sort by strength of probability
    results.sort(key=lambda x: x[1], reverse=True)
    return_list = []
    for r in results:
        return_list.append((classes[r[0]], r[1]))
    # return tuple of intent and probability
    return return_list

def response(sentence, userID='123', show_details=False):
    results = classify(sentence)
    # if we have a classification then find the matching intent tag
    if results:
        # loop as long as there are matches to process
        while results:
            for i in intents['intents']:
                # find a tag matching the first result
                if i['tag'] == results[0][0]:
                    # set context for this intent if necessary
                    if 'context_set' in i:
                        if show_details: print ('context:', i['context_set'])
                        context[userID] = i['context_set']

                    # check if this intent is contextual and applies to this user's conv
ersation
                    if not 'context_filter' in i or \
                        (userID in context and 'context_filter' in i and i['context_filt
er'] == context[userID]):
                        if show_details: print ('tag:', i['tag'])
                        # a random response from the intent
                        return print(random.choice(i['responses']))

            results.pop(0)
```

In [22]:

```
response('How to register for next semester?')
```

```
You can register for next semester in side menu on the left.
```

In [23]:

```
response('Exam registration')
```

```
Open the side menu on the left, click on Examination and then in the drop down menu click
on Examination Form for registration.
```

In [24]:

```
response('What are my courses in this semester?')
```

```
You can find all the details about your subjects/courses in 'My Courses' section in the s
ide menu on the left.
```

In [25]:

```
response("what was last year's placement record?")
```

```
https://www.amity.edu/gwalior/about-placements
```

In [26]:

```
response('Which faculties are alloted to me?')
```

```
You can find all the details about the faculty in 'My Faculty' section in the side menu or
the left.
```

In [27]:

```
response('amity faculty')
```

https://www.amity.edu/gwalior/about-faculty

In [28]:

```
context
```

Out[28]:

```
{}
```

In [29]:

```
response("Hello", show_details=True)
```

```
context:
tag: greeting
Hello, This is Amizone, How can we help you?
```