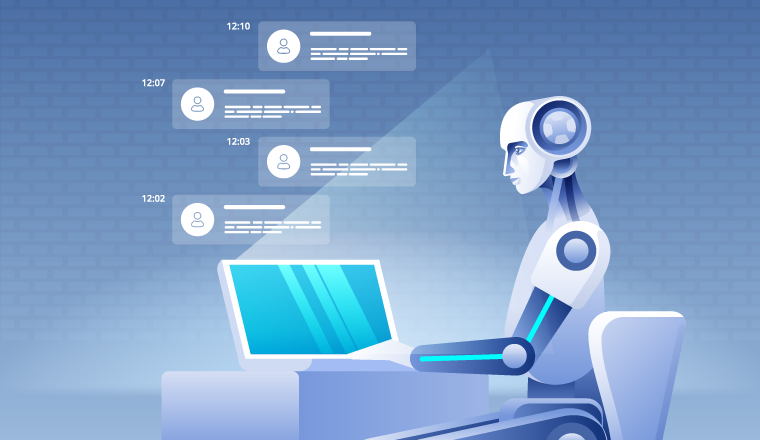
*CHATBOT USING PYTHON*

*PHASE-3*

*INTRODUCTION:*

**

Creating a chatbot using Python can be a fun and educational project. In this introduction, I'll provide an overview of how to create a simple text-based chatbot using Python. We'll use the **nltk** library for natural language processing anCreating a chatbot using Python can be a fun and educational project. In this introduction, I'll provide an overview of how to create a simple text-based chatbot using Python. We'll use the **nltk** library for natural language processing and a basic rule-based approach. Please note that more advanced chatbots often utilize machine learning and deep learning techniques, but this example is a good starting point.d a basic rule-based approach. Please note that more advanced chatbots often utilize machine learning and deep learning techniques, but this example is a good starting point.

DATASET LINK:

https://kaggle.com/datasets/88e7dbb2d897840f0e0b25c2dbd487e233911337eca197eba4a3ac21c9b3c22c

This notebook is pre trained model for creating chatbot with good accuracy for this dataset. However, Accuracy range will be different.

You can add dataset with .csv file. But, you need to transfer your data json file into csv file format using python script. I tried it but don't seem to have converted successfully.

**Import and load the data file**

We import the necessary packages for our chatbot and initialize the variables we will use in our Python project. The data file is in JSON format so we used the json package to parse the JSON file into Python.

In [1]:

import nltk

nltk.download('punkt')*#Sentence tokenizer*

[nltk\_data] Downloading package punkt to /usr/share/nltk\_data...

[nltk\_data] Package punkt is already up-to-date!

Out[1]:

True

In [2]:

import nltk

from nltk.stem import WordNetLemmatizer

lemmatizer = WordNetLemmatizer()

import json

import pickle

import warnings

warnings.filterwarnings('ignore')

In [3]:

import numpy as np

import tensorflow as tf

from keras.models import Sequential

from keras.layers import Dense, Activation, Dropout

from tensorflow.keras.optimizers import SGD

import random

**Preprocessing**

In [4]:

words=[]

classes = []

documents = []

ignore\_words = ['?', '!']

data\_file = open('/kaggle/input/chatbot-dataset/intents.json').read() *# read json file*

intents = json.loads(data\_file) *# load json file*

When working with text data, we need to perform various preprocessing on the data before we make a machine learning or a deep learning model. Based on the requirements we need to apply various operations to preprocess the data.

* Tokenizing is the most basic and first thing you can do on text data.
* Tokenizing is the process of breaking the whole text into small parts like words.
* Here we iterate through the patterns and tokenize the sentence using nltk.word\_tokenize() function and append each word in the words list. We also create a list of classes for our tags.

In [5]:

for intent **in** intents['intents']:

for pattern **in** intent['patterns']:

*#tokenize each word*

w = nltk.word\_tokenize(pattern)

words.extend(w)*# add each elements into list*

*#combination between patterns and intents*

documents.append((w, intent['tag']))*#add single element into end of list*

*# add to tag in our classes list*

if intent['tag'] **not** **in** classes:

classes.append(intent['tag'])

In [6]:

nltk.download('wordnet') *#lexical database for the English language*

[nltk\_data] Downloading package wordnet to /usr/share/nltk\_data...

[nltk\_data] Package wordnet is already up-to-date!

Out[6]:

True

In [7]:

nltk.download('omw-1.4')

[nltk\_data] Downloading package omw-1.4 to /usr/share/nltk\_data...

Out[7]:

True

Now we will lemmatize each word and remove duplicate words from the list.

* Lemmatizing is the process of converting a word into its lemma form and then creating a pickle file to store the Python objects which we will use while predicting.

In [8]:

*# lemmatize, lower each word and remove duplicates*

words = [lemmatizer.lemmatize(w.lower()) for w **in** words if w **not** **in** ignore\_words]

words = sorted(list(set(words)))

*# sort classes*

classes = sorted(list(set(classes)))

*# documents = combination between patterns and intents*

print (len(documents), "documents**\n**", documents, "**\n**")

*# classes = intents[tag]*

print (len(classes), "classes**\n**", classes, "**\n**")

*# words = all words, vocabulary*

print (len(words), "unique lemmatized words**\n**", words, "**\n**")

pickle.dump(words,open('words.pkl','wb'))

pickle.dump(classes,open('classes.pkl','wb'))

[(['Hi'], 'greeting'), (['How', 'are', 'you', '?'], 'greeting'), (['Is', 'anyone', 'there', '?'], 'greeting'), (['Hello'], 'greeting'), (['Good', 'day'], 'greeting'), (['What', "'s", 'up'], 'greeting'), (['how', 'are', 'ya'], 'greeting'), (['heyy'], 'greeting'), (['whatsup'], 'greeting'), (['?', '?', '?', '?', '?', '?', '?', '?'], 'greeting'), (['cya'], 'goodbye'), (['see', 'you'], 'goodbye'), (['bye', 'bye'], 'goodbye'), (['See', 'you', 'later'], 'goodbye'), (['Goodbye'], 'goodbye'), (['I', 'am', 'Leaving'], 'goodbye'), (['Bye'], 'goodbye'), (['Have', 'a', 'Good', 'day'], 'goodbye'), (['talk', 'to', 'you', 'later'], 'goodbye'), (['ttyl'], 'goodbye'), (['i', 'got', 'to', 'go'], 'goodbye'), (['gtg'], 'goodbye'), (['what', 'is', 'the', 'name', 'of', 'your', 'developers'], 'creator'), (['what', 'is', 'the', 'name', 'of', 'your', 'creators'], 'creator'), (['what', 'is', 'the', 'name', 'of', 'the', 'developers'], 'creator'), (['what', 'is', 'the', 'name', 'of', 'the', 'creators'], 'creator'), (['who', 'created', 'you'], 'creator'), (['your', 'developers'], 'creator'), (['your', 'creators'], 'creator'), (['who', 'are', 'your', 'developers'], 'creator'), (['developers'], 'creator'), (['you', 'are', 'made', 'by'], 'creator'), (['you', 'are', 'made', 'by', 'whom'], 'creator'), (['who', 'created', 'you'], 'creator'), (['who', 'create', 'you'], 'creator'), (['creators'], 'creator'), (['who', 'made', 'you'], 'creator'), (['who', 'designed', 'you'], 'creator'), (['name'], 'name'), (['your', 'name'], 'name'), (['do', 'you', 'have', 'a', 'name'], 'name'), (['what', 'are', 'you', 'called'], 'name'), (['what', 'is', 'your', 'name'], 'name'), (['what', 'should', 'I', 'call', 'you'], 'name'), (['whats', 'your', 'name', '?'], 'name'), (['what', 'are', 'you'], 'name'), (['who', 'are', 'you'], 'name'), (['who', 'is', 'this'], 'name'), (['what', 'am', 'i', 'chatting', 'to'], 'name'), (['who', 'am', 'i', 'taking', 'to'], 'name'), (['what', 'are', 'you'], 'name'), (['timing', 'of', 'college'], 'hours'), (['what', 'is', 'college', 'timing'], 'hours'), (['working', 'days'], 'hours'), (['when', 'are', 'you', 'guys', 'open'], 'hours'), (['what', 'are', 'your', 'hours'], 'hours'), (['hours', 'of', 'operation'], 'hours'), (['when', 'is', 'the', 'college', 'open'], 'hours'), (['college', 'timing'], 'hours'), (['what', 'about', 'college', 'timing'], 'hours'), (['is', 'college', 'open', 'on', 'saturday'], 'hours'), (['tell', 'something', 'about', 'college', 'timing'], 'hours'), (['what', 'is', 'the', 'college', 'hours'], 'hours'), (['when', 'should', 'i', 'come', 'to', 'college'], 'hours'), (['when', 'should', 'i', 'attend', 'college'], 'hours'), (['what', 'is', 'my', 'college', 'time'], 'hours'), (['college', 'timing'], 'hours'), (['timing', 'college'], 'hours'), (['more', 'info'], 'number'), (['contact', 'info'], 'number'), (['how', 'to', 'contact', 'college'], 'number'), (['college', 'telephone', 'number'], 'number'), (['college', 'number'], 'number'), (['What', 'is', 'your', 'contact', 'no'], 'number'), (['Contact', 'number', '?'], 'number'), (['how', 'to', 'call', 'you'], 'number'), (['College', 'phone', 'no', '?'], 'number'), (['how', 'can', 'i', 'contact', 'you'], 'number'), (['Can', 'i', 'get', 'your', 'phone', 'number'], 'number'), (['how', 'can', 'i', 'call', 'you'], 'number'), (['phone', 'number'], 'number'), (['phone', 'no'], 'number'), (['call'], 'number'), (['list', 'of', 'courses'], 'course'), (['list', 'of', 'courses', 'offered'], 'course'), (['list', 'of', 'courses', 'offered', 'in'], 'course'), (['what', 'are', 'the', 'courses', 'offered', 'in', 'your', 'college', '?'], 'course'), (['courses', '?'], 'course'), (['courses', 'offered'], 'course'), (['courses', 'offered', 'in', '(', 'your', 'univrsity', '(', 'UNI', ')', 'name', ')'], 'course'), (['courses', 'you', 'offer'], 'course'), (['branches', '?'], 'course'), (['courses', 'available', 'at', 'UNI', '?'], 'course'), (['branches', 'available', 'at', 'your', 'college', '?'], 'course'), (['what', 'are', 'the', 'courses', 'in', 'UNI', '?'], 'course'), (['what', 'are', 'branches', 'in', 'UNI', '?'], 'course'), (['what', 'are', 'courses', 'in', 'UNI', '?'], 'course'), (['branches', 'available', 'in', 'UNI', '?'], 'course'), (['can', 'you', 'tell', 'me', 'the', 'courses', 'available', 'in', 'UNI', '?'], 'course'), (['can', 'you', 'tell', 'me', 'the', 'branches', 'available', 'in', 'UNI', '?'], 'course'), (['computer', 'engineering', '?'], 'course'), (['computer'], 'course'), (['Computer', 'engineering', '?'], 'course'), (['it'], 'course'), (['IT'], 'course'), (['Information', 'Technology'], 'course'), (['AI/Ml'], 'course'), (['Mechanical', 'engineering'], 'course'), (['Chemical', 'engineering'], 'course'), (['Civil', 'engineering'], 'course'), (['information', 'about', 'fee'], 'fees'), (['information', 'on', 'fee'], 'fees'), (['tell', 'me', 'the', 'fee'], 'fees'), (['college', 'fee'], 'fees'), (['fee', 'per', 'semester'], 'fees'), (['what', 'is', 'the', 'fee', 'of', 'each', 'semester'], 'fees'), (['what', 'is', 'the', 'fees', 'of', 'each', 'year'], 'fees'), (['what', 'is', 'fee'], 'fees'), (['what', 'is', 'the', 'fees'], 'fees'), (['how', 'much', 'is', 'the', 'fees'], 'fees'), (['fees', 'for', 'first', 'year'], 'fees'), (['fees'], 'fees'), (['about', 'the', 'fees'], 'fees'), (['tell', 'me', 'something', 'about', 'the', 'fees'], 'fees'), (['What', 'is', 'the', 'fees', 'of', 'hostel'], 'fees'), (['how', 'much', 'is', 'the', 'fees'], 'fees'), (['hostel', 'fees'], 'fees'), (['fees', 'for', 'AC', 'room'], 'fees'), (['fees', 'for', 'non-AC', 'room'], 'fees'), (['fees', 'for', 'Ac', 'room', 'for', 'girls'], 'fees'), (['fees', 'for', 'non-Ac', 'room', 'for', 'girls'], 'fees'), (['fees', 'for', 'Ac', 'room', 'for', 'boys'], 'fees'), (['fees', 'for', 'non-Ac', 'room', 'for', 'boys'], 'fees'), (['where', 'is', 'the', 'college', 'located'], 'location'), (['college', 'is', 'located', 'at'], 'location'), (['where', 'is', 'college'], 'location'), (['where', 'is', 'college', 'located'], 'location'), (['address', 'of', 'college'], 'location'), (['how', 'to', 'reach', 'college'], 'location'), (['college', 'location'], 'location'), (['college', 'address'], 'location'), (['wheres', 'the', 'college'], 'location'), (['how', 'can', 'I', 'reach', 'college'], 'location'), (['whats', 'is', 'the', 'college', 'address'], 'location'), (['what', 'is', 'the', 'address', 'of', 'college'], 'location'), (['address'], 'location'), (['location'], 'location'), (['hostel', 'facility'], 'hostel'), (['hostel', 'servive'], 'hostel'), (['hostel', 'location'], 'hostel'), (['hostel', 'address'], 'hostel'), (['hostel', 'facilities'], 'hostel'), (['hostel', 'fees'], 'hostel'), (['Does', 'college', 'provide', 'hostel'], 'hostel'), (['Is', 'there', 'any', 'hostel'], 'hostel'), (['Where', 'is', 'hostel'], 'hostel'), (['do', 'you', 'have', 'hostel'], 'hostel'), (['do', 'you', 'guys', 'have', 'hostel'], 'hostel'), (['hostel'], 'hostel'), (['hostel', 'capacity'], 'hostel'), (['what', 'is', 'the', 'hostel', 'fee'], 'hostel'), (['how', 'to', 'get', 'in', 'hostel'], 'hostel'), (['what', 'is', 'the', 'hostel', 'address'], 'hostel'), (['how', 'far', 'is', 'hostel', 'from', 'college'], 'hostel'), (['hostel', 'college', 'distance'], 'hostel'), (['where', 'is', 'the', 'hostel'], 'hostel'), (['how', 'big', 'is', 'the', 'hostel'], 'hostel'), (['distance', 'between', 'college', 'and', 'hostel'], 'hostel'), (['distance', 'between', 'hostel', 'and', 'college'], 'hostel'), (['events', 'organised'], 'event'), (['list', 'of', 'events'], 'event'), (['list', 'of', 'events', 'organised', 'in', 'college'], 'event'), (['list', 'of', 'events', 'conducted', 'in', 'college'], 'event'), (['What', 'events', 'are', 'conducted', 'in', 'college'], 'event'), (['Are', 'there', 'any', 'event', 'held', 'at', 'college'], 'event'), (['Events', '?'], 'event'), (['functions'], 'event'), (['what', 'are', 'the', 'events'], 'event'), (['tell', 'me', 'about', 'events'], 'event'), (['what', 'about', 'events'], 'event'), (['document', 'to', 'bring'], 'document'), (['documents', 'needed', 'for', 'admision'], 'document'), (['documents', 'needed', 'at', 'the', 'time', 'of', 'admission'], 'document'), (['documents', 'needed', 'during', 'admission'], 'document'), (['documents', 'required', 'for', 'admision'], 'document'), (['documents', 'required', 'at', 'the', 'time', 'of', 'admission'], 'document'), (['documents', 'required', 'during', 'admission'], 'document'), (['What', 'document', 'are', 'required', 'for', 'admission'], 'document'), (['Which', 'document', 'to', 'bring', 'for', 'admission'], 'document'), (['documents'], 'document'), (['what', 'documents', 'do', 'i', 'need'], 'document'), (['what', 'documents', 'do', 'I', 'need', 'for', 'admission'], 'document'), (['documents', 'needed'], 'document'), (['size', 'of', 'campus'], 'floors'), (['building', 'size'], 'floors'), (['How', 'many', 'floors', 'does', 'college', 'have'], 'floors'), (['floors', 'in', 'college'], 'floors'), (['floors', 'in', 'college'], 'floors'), (['how', 'tall', 'is', 'UNI', "'s", 'College', 'of', 'Engineering', 'college', 'building'], 'floors'), (['floors'], 'floors'), (['Syllabus', 'for', 'IT'], 'syllabus'), (['what', 'is', 'the', 'Information', 'Technology', 'syllabus'], 'syllabus'), (['syllabus'], 'syllabus'), (['timetable'], 'syllabus'), (['what', 'is', 'IT', 'syllabus'], 'syllabus'), (['syllabus'], 'syllabus'), (['What', 'is', 'next', 'lecture'], 'syllabus'), (['is', 'there', 'any', 'library'], 'library'), (['library', 'facility'], 'library'), (['library', 'facilities'], 'library'), (['do', 'you', 'have', 'library'], 'library'), (['does', 'the', 'college', 'have', 'library', 'facility'], 'library'), (['college', 'library'], 'library'), (['where', 'can', 'i', 'get', 'books'], 'library'), (['book', 'facility'], 'library'), (['Where', 'is', 'library'], 'library'), (['Library'], 'library'), (['Library', 'information'], 'library'), (['Library', 'books', 'information'], 'library'), (['Tell', 'me', 'about', 'library'], 'library'), (['how', 'many', 'libraries'], 'library'), (['how', 'is', 'college', 'infrastructure'], 'infrastructure'), (['infrastructure'], 'infrastructure'), (['college', 'infrastructure'], 'infrastructure'), (['food', 'facilities'], 'canteen'), (['canteen', 'facilities'], 'canteen'), (['canteen', 'facility'], 'canteen'), (['is', 'there', 'any', 'canteen'], 'canteen'), (['Is', 'there', 'a', 'cafetaria', 'in', 'college'], 'canteen'), (['Does', 'college', 'have', 'canteen'], 'canteen'), (['Where', 'is', 'canteen'], 'canteen'), (['where', 'is', 'cafetaria'], 'canteen'), (['canteen'], 'canteen'), (['Food'], 'canteen'), (['Cafetaria'], 'canteen'), (['food', 'menu'], 'menu'), (['food', 'in', 'canteen'], 'menu'), (['Whats', 'there', 'on', 'menu'], 'menu'), (['what', 'is', 'available', 'in', 'college', 'canteen'], 'menu'), (['what', 'foods', 'can', 'we', 'get', 'in', 'college', 'canteen'], 'menu'), (['food', 'variety'], 'menu'), (['What', 'is', 'there', 'to', 'eat', '?'], 'menu'), (['What', 'is', 'college', 'placement'], 'placement'), (['Which', 'companies', 'visit', 'in', 'college'], 'placement'), (['What', 'is', 'average', 'package'], 'placement'), (['companies', 'visit'], 'placement'), (['package'], 'placement'), (['About', 'placement'], 'placement'), (['placement'], 'placement'), (['recruitment'], 'placement'), (['companies'], 'placement'), (['Who', 'is', 'HOD'], 'ithod'), (['Where', 'is', 'HOD'], 'ithod'), (['it', 'hod'], 'ithod'), (['name', 'of', 'it', 'hod'], 'ithod'), (['Who', 'is', 'computer', 'HOD'], 'computerhod'), (['Where', 'is', 'computer', 'HOD'], 'computerhod'), (['computer', 'hod'], 'computerhod'), (['name', 'of', 'computer', 'hod'], 'computerhod'), (['Who', 'is', 'extc', 'HOD'], 'extchod'), (['Where', 'is', 'extc', 'HOD'], 'extchod'), (['extc', 'hod'], 'extchod'), (['name', 'of', 'extc', 'hod'], 'extchod'), (['what', 'is', 'the', 'name', 'of', 'principal'], 'principal'), (['whatv', 'is', 'the', 'principal', 'name'], 'principal'), (['principal', 'name'], 'principal'), (['Who', 'is', 'college', 'principal'], 'principal'), (['Where', 'is', 'principal', "'s", 'office'], 'principal'), (['principal'], 'principal'), (['name', 'of', 'principal'], 'principal'), (['exam', 'dates'], 'sem'), (['exam', 'schedule'], 'sem'), (['When', 'is', 'semester', 'exam'], 'sem'), (['Semester', 'exam', 'timetable'], 'sem'), (['sem'], 'sem'), (['semester'], 'sem'), (['exam'], 'sem'), (['when', 'is', 'exam'], 'sem'), (['exam', 'timetable'], 'sem'), (['exam', 'dates'], 'sem'), (['when', 'is', 'semester'], 'sem'), (['what', 'is', 'the', 'process', 'of', 'admission'], 'admission'), (['what', 'is', 'the', 'admission', 'process'], 'admission'), (['How', 'to', 'take', 'admission', 'in', 'your', 'college'], 'admission'), (['What', 'is', 'the', 'process', 'for', 'admission'], 'admission'), (['admission'], 'admission'), (['admission', 'process'], 'admission'), (['scholarship'], 'scholarship'), (['Is', 'scholarship', 'available'], 'scholarship'), (['scholarship', 'engineering'], 'scholarship'), (['scholarship', 'it'], 'scholarship'), (['scholarship', 'ce'], 'scholarship'), (['scholarship', 'mechanical'], 'scholarship'), (['scholarship', 'civil'], 'scholarship'), (['scholarship', 'chemical'], 'scholarship'), (['scholarship', 'for', 'AI/ML'], 'scholarship'), (['available', 'scholarships'], 'scholarship'), (['scholarship', 'for', 'computer', 'engineering'], 'scholarship'), (['scholarship', 'for', 'IT', 'engineering'], 'scholarship'), (['scholarship', 'for', 'mechanical', 'engineering'], 'scholarship'), (['scholarship', 'for', 'civil', 'engineering'], 'scholarship'), (['scholarship', 'for', 'chemical', 'engineering'], 'scholarship'), (['list', 'of', 'scholarship'], 'scholarship'), (['comps', 'scholarship'], 'scholarship'), (['IT', 'scholarship'], 'scholarship'), (['mechanical', 'scholarship'], 'scholarship'), (['civil', 'scholarship'], 'scholarship'), (['chemical', 'scholarship'], 'scholarship'), (['automobile', 'scholarship'], 'scholarship'), (['first', 'year', 'scholarship'], 'scholarship'), (['second', 'year', 'scholarship'], 'scholarship'), (['third', 'year', 'scholarship'], 'scholarship'), (['fourth', 'year', 'scholarship'], 'scholarship'), (['What', 'facilities', 'college', 'provide'], 'facilities'), (['College', 'facility'], 'facilities'), (['What', 'are', 'college', 'facilities'], 'facilities'), (['facilities'], 'facilities'), (['facilities', 'provided'], 'facilities'), (['max', 'number', 'of', 'students'], 'college intake'), (['number', 'of', 'seats', 'per', 'branch'], 'college intake'), (['number', 'of', 'seats', 'in', 'each', 'branch'], 'college intake'), (['maximum', 'number', 'of', 'seats'], 'college intake'), (['maximum', 'students', 'intake'], 'college intake'), (['What', 'is', 'college', 'intake'], 'college intake'), (['how', 'many', 'stundent', 'are', 'taken', 'in', 'each', 'branch'], 'college intake'), (['seat', 'allotment'], 'college intake'), (['seats'], 'college intake'), (['college', 'dress', 'code'], 'uniform'), (['college', 'dresscode'], 'uniform'), (['what', 'is', 'the', 'uniform'], 'uniform'), (['can', 'we', 'wear', 'casuals'], 'uniform'), (['Does', 'college', 'have', 'an', 'uniform'], 'uniform'), (['Is', 'there', 'any', 'uniform'], 'uniform'), (['uniform'], 'uniform'), (['what', 'about', 'uniform'], 'uniform'), (['do', 'we', 'have', 'to', 'wear', 'uniform'], 'uniform'), (['what', 'are', 'the', 'different', 'committe', 'in', 'college'], 'committee'), (['different', 'committee', 'in', 'college'], 'committee'), (['Are', 'there', 'any', 'committee', 'in', 'college'], 'committee'), (['Give', 'me', 'committee', 'details'], 'committee'), (['committee'], 'committee'), (['how', 'many', 'committee', 'are', 'there', 'in', 'college'], 'committee'), (['I', 'love', 'you'], 'random'), (['Will', 'you', 'marry', 'me'], 'random'), (['Do', 'you', 'love', 'me'], 'random'), (['fuck'], 'swear'), (['bitch'], 'swear'), (['shut', 'up'], 'swear'), (['hell'], 'swear'), (['stupid'], 'swear'), (['idiot'], 'swear'), (['dumb', 'ass'], 'swear'), (['asshole'], 'swear'), (['fucker'], 'swear'), (['holidays'], 'vacation'), (['when', 'will', 'semester', 'starts'], 'vacation'), (['when', 'will', 'semester', 'end'], 'vacation'), (['when', 'is', 'the', 'holidays'], 'vacation'), (['list', 'of', 'holidays'], 'vacation'), (['Holiday', 'in', 'these', 'year'], 'vacation'), (['holiday', 'list'], 'vacation'), (['about', 'vacations'], 'vacation'), (['about', 'holidays'], 'vacation'), (['When', 'is', 'vacation'], 'vacation'), (['When', 'is', 'holidays'], 'vacation'), (['how', 'long', 'will', 'be', 'the', 'vacation'], 'vacation'), (['sports', 'and', 'games'], 'sports'), (['give', 'sports', 'details'], 'sports'), (['sports', 'infrastructure'], 'sports'), (['sports', 'facilities'], 'sports'), (['information', 'about', 'sports'], 'sports'), (['Sports', 'activities'], 'sports'), (['please', 'provide', 'sports', 'and', 'games', 'information'], 'sports'), (['okk'], 'salutaion'), (['okie'], 'salutaion'), (['nice', 'work'], 'salutaion'), (['well', 'done'], 'salutaion'), (['good', 'job'], 'salutaion'), (['thanks', 'for', 'the', 'help'], 'salutaion'), (['Thank', 'You'], 'salutaion'), (['its', 'ok'], 'salutaion'), (['Thanks'], 'salutaion'), (['Good', 'work'], 'salutaion'), (['k'], 'salutaion'), (['ok'], 'salutaion'), (['okay'], 'salutaion'), (['what', 'can', 'you', 'do'], 'task'), (['what', 'are', 'the', 'thing', 'you', 'can', 'do'], 'task'), (['things', 'you', 'can', 'do'], 'task'), (['what', 'can', 'u', 'do', 'for', 'me'], 'task'), (['how', 'u', 'can', 'help', 'me'], 'task'), (['why', 'i', 'should', 'use', 'you'], 'task'), (['ragging'], 'ragging'), (['is', 'ragging', 'practice', 'active', 'in', 'college'], 'ragging'), (['does', 'college', 'have', 'any', 'antiragging', 'facility'], 'ragging'), (['is', 'there', 'any', 'ragging', 'cases'], 'ragging'), (['is', 'ragging', 'done', 'here'], 'ragging'), (['ragging', 'against'], 'ragging'), (['antiragging', 'facility'], 'ragging'), (['ragging', 'juniors'], 'ragging'), (['ragging', 'history'], 'ragging'), (['ragging', 'incidents'], 'ragging'), (['hod'], 'hod'), (['hod', 'name'], 'hod'), (['who', 'is', 'the', 'hod'], 'hod')]

38 classes

['admission', 'canteen', 'college intake', 'committee', 'computerhod', 'course', 'creator', 'document', 'event', 'extchod', 'facilities', 'fees', 'floors', 'goodbye', 'greeting', 'hod', 'hostel', 'hours', 'infrastructure', 'ithod', 'library', 'location', 'menu', 'name', 'number', 'placement', 'principal', 'ragging', 'random', 'salutaion', 'scholarship', 'sem', 'sports', 'swear', 'syllabus', 'task', 'uniform', 'vacation']

263 unique lemmatized words

["'s", '(', ')', 'a', 'about', 'ac', 'active', 'activity', 'address', 'admision', 'admission', 'against', 'ai/ml', 'allotment', 'am', 'an', 'and', 'antiragging', 'any', 'anyone', 'are', 'as', 'asshole', 'at', 'attend', 'automobile', 'available', 'average', 'be', 'between', 'big', 'bitch', 'book', 'boy', 'branch', 'bring', 'building', 'by', 'bye', 'cafetaria', 'call', 'called', 'campus', 'can', 'canteen', 'capacity', 'case', 'casuals', 'ce', 'chatting', 'chemical', 'civil', 'code', 'college', 'come', 'committe', 'committee', 'comp', 'company', 'computer', 'conducted', 'contact', 'course', 'create', 'created', 'creator', 'cya', 'date', 'day', 'designed', 'detail', 'developer', 'different', 'distance', 'do', 'document', 'doe', 'done', 'dress', 'dresscode', 'dumb', 'during', 'each', 'eat', 'end', 'engineering', 'event', 'exam', 'extc', 'facility', 'far', 'fee', 'first', 'floor', 'food', 'for', 'fourth', 'from', 'fuck', 'fucker', 'function', 'game', 'get', 'girl', 'give', 'go', 'good', 'goodbye', 'got', 'gtg', 'guy', 'have', 'held', 'hell', 'hello', 'help', 'here', 'heyy', 'hi', 'history', 'hod', 'holiday', 'hostel', 'hour', 'how', 'i', 'idiot', 'in', 'incident', 'info', 'information', 'infrastructure', 'intake', 'is', 'it', 'job', 'junior', 'k', 'later', 'leaving', 'lecture', 'library', 'list', 'located', 'location', 'long', 'love', 'made', 'many', 'marry', 'max', 'maximum', 'me', 'mechanical', 'menu', 'more', 'much', 'my', 'name', 'need', 'needed', 'next', 'nice', 'no', 'non-ac', 'number', 'of', 'offer', 'offered', 'office', 'ok', 'okay', 'okie', 'okk', 'on', 'open', 'operation', 'organised', 'package', 'per', 'phone', 'placement', 'please', 'practice', 'principal', 'process', 'provide', 'provided', 'ragging', 'reach', 'recruitment', 'required', 'room', 'saturday', 'schedule', 'scholarship', 'seat', 'second', 'see', 'sem', 'semester', 'servive', 'should', 'shut', 'size', 'something', 'sport', 'start', 'student', 'stundent', 'stupid', 'syllabus', 'take', 'taken', 'taking', 'talk', 'tall', 'technology', 'telephone', 'tell', 'thank', 'thanks', 'the', 'there', 'these', 'thing', 'third', 'this', 'time', 'timetable', 'timing', 'to', 'ttyl', 'u', 'uni', 'uniform', 'univrsity', 'up', 'use', 'vacation', 'variety', 'visit', 'we', 'wear', 'well', 'what', 'whats', 'whatsup', 'whatv', 'when', 'where', 'wheres', 'which', 'who', 'whom', 'why', 'will', 'work', 'working', 'ya', 'year', 'you', 'your']

**Training Model**

Now, we will create the training data in which we will provide the input and the output.

* Our input will be the pattern and output will be the class our input pattern belongs to. But the computer doesn’t understand text so we will convert text into numbers

In [9]:

*# create our training data*

training = []

*# create an empty array for our output*

output\_empty = [0] \* len(classes)

*# training set, bag of words for each sentence*

for doc **in** documents:

*# initialize our bag of words*

bag = []

*# list of tokenized words*

pattern\_words = doc[0]

*# convert pattern\_words in lower case*

pattern\_words = [lemmatizer.lemmatize(word.lower()) for word **in** pattern\_words]

*# create bag of words array,if word match found in current pattern then put 1 otherwise 0.[row \* colm(263)]*

for w **in** words:

bag.append(1) if w **in** pattern\_words else bag.append(0)

*# in output array 0 value for each tag ang 1 value for matched tag.[row \* colm(8)]*

output\_row = list(output\_empty)

output\_row[classes.index(doc[1])] = 1

training.append([bag, output\_row])

*# shuffle training and turn into np.array*

random.shuffle(training)

training = np.array(training)

*# create train and test. X - patterns(words), Y - intents(tags)*

train\_x = list(training[:,0])

train\_y = list(training[:,1])

print("Training data created")

Training data created

In [10]:

from tensorflow.python.framework import ops

ops.reset\_default\_graph()

**Build the model**

We have our training data ready, now we will build a deep neural network that has 3 layers. We use the Keras sequential API for this. After training the model for 200 epochs, we achieved 100% accuracy on our model. Let us save the model as ‘chatbot\_model.h5'.

In [11]:

*# Create model - 3 layers. First layer 128 neurons, second layer 64 neurons and 3rd output layer contains number of neurons*

*# equal to number of intents to predict output intent with softmax*

model = Sequential()

model.add(Dense(128, input\_shape=(len(train\_x[0]),), activation='relu'))

model.add(Dropout(0.5))

model.add(Dense(64, activation='relu'))

model.add(Dropout(0.5))

model.add(Dense(len(train\_y[0]), activation='softmax'))

print("First layer:",model.layers[0].get\_weights()[0])

First layer: [[ 0.08108504 -0.06599443 -0.10388638 ... -0.01234975 0.02568085

0.00633688]

[-0.02540757 -0.0221673 -0.0489299 ... 0.10772091 0.00711305

0.03869867]

[-0.06639696 -0.05009066 -0.03959011 ... -0.0571945 -0.11444904

-0.06228179]

...

[ 0.02686372 0.0873628 0.12299983 ... -0.07360662 0.05407895

-0.01691054]

[-0.08417445 -0.10581411 -0.07542053 ... -0.06181952 -0.12180413

-0.08388676]

[-0.07259022 0.11421812 -0.04386763 ... 0.00979565 0.05784626

0.09121044]]

In [12]:

*# Compile model. Stochastic gradient descent with Nesterov accelerated gradient gives good results for this model*

*# sgd = SGD(lr=0.01, decay=1e-6, momentum=0.9, nesterov=True)*

model.compile(loss='categorical\_crossentropy', optimizer='adam', metrics=['accuracy'])

In [13]:

*#fitting and saving the model*

hist = model.fit(np.array(train\_x), np.array(train\_y), epochs=200, batch\_size=5, verbose=1)

model.save('chatbot\_model.h5', hist)

print("model created")

Epoch 1/200

81/81 [==============================] - 1s 2ms/step - loss: 3.6136 - accuracy: 0.0543

Epoch 2/200

81/81 [==============================] - 0s 2ms/step - loss: 3.4736 - accuracy: 0.1259

Epoch 3/200

81/81 [==============================] - 0s 2ms/step - loss: 3.2848 - accuracy: 0.1753

Epoch 4/200

81/81 [==============================] - 0s 2ms/step - loss: 3.0604 - accuracy: 0.2346

Epoch 5/200

81/81 [==============================] - 0s 2ms/step - loss: 2.8305 - accuracy: 0.2716

Epoch 6/200

81/81 [==============================] - 0s 2ms/step - loss: 2.5375 - accuracy: 0.3432

Epoch 7/200

81/81 [==============================] - 0s 2ms/step - loss: 2.3111 - accuracy: 0.4025

Epoch 8/200

81/81 [==============================] - 0s 2ms/step - loss: 2.1470 - accuracy: 0.4568

Epoch 9/200

81/81 [==============================] - 0s 2ms/step - loss: 1.9539 - accuracy: 0.4864

Epoch 10/200

81/81 [==============================] - 0s 2ms/step - loss: 1.7000 - accuracy: 0.6025

Epoch 11/200

81/81 [==============================] - 0s 2ms/step - loss: 1.5961 - accuracy: 0.6148

Epoch 12/200

81/81 [==============================] - 0s 2ms/step - loss: 1.4055 - accuracy: 0.6593

Epoch 13/200

81/81 [==============================] - 0s 2ms/step - loss: 1.3002 - accuracy: 0.6963

Epoch 14/200

81/81 [==============================] - 0s 2ms/step - loss: 1.1978 - accuracy: 0.6963

Epoch 15/200

81/81 [==============================] - 0s 2ms/step - loss: 1.0640 - accuracy: 0.7407

Epoch 16/200

81/81 [==============================] - 0s 2ms/step - loss: 1.0210 - accuracy: 0.7506

Epoch 17/200

81/81 [==============================] - 0s 2ms/step - loss: 0.9202 - accuracy: 0.7679

Epoch 18/200

81/81 [==============================] - 0s 2ms/step - loss: 0.8287 - accuracy: 0.8099

Epoch 19/200

81/81 [==============================] - 0s 2ms/step - loss: 0.7831 - accuracy: 0.8198

Epoch 20/200

81/81 [==============================] - 0s 2ms/step - loss: 0.7525 - accuracy: 0.8148

Epoch 21/200

81/81 [==============================] - 0s 2ms/step - loss: 0.7355 - accuracy: 0.8123

Epoch 22/200

81/81 [==============================] - 0s 2ms/step - loss: 0.6728 - accuracy: 0.8272

Epoch 23/200

81/81 [==============================] - 0s 2ms/step - loss: 0.6377 - accuracy: 0.8321

Epoch 24/200

81/81 [==============================] - 0s 2ms/step - loss: 0.5440 - accuracy: 0.8741

Epoch 25/200

81/81 [==============================] - 0s 2ms/step - loss: 0.4673 - accuracy: 0.8889

Epoch 26/200

81/81 [==============================] - 0s 2ms/step - loss: 0.5191 - accuracy: 0.8469

Epoch 27/200

81/81 [==============================] - 0s 2ms/step - loss: 0.5168 - accuracy: 0.8840

Epoch 28/200

81/81 [==============================] - 0s 2ms/step - loss: 0.4686 - accuracy: 0.8864

Epoch 29/200

81/81 [==============================] - 0s 2ms/step - loss: 0.4586 - accuracy: 0.8790

Epoch 30/200

81/81 [==============================] - 0s 2ms/step - loss: 0.4126 - accuracy: 0.8963

Epoch 31/200

81/81 [==============================] - 0s 2ms/step - loss: 0.4247 - accuracy: 0.8889

Epoch 32/200

81/81 [==============================] - 0s 2ms/step - loss: 0.4080 - accuracy: 0.8840

Epoch 33/200

81/81 [==============================] - 0s 2ms/step - loss: 0.3659 - accuracy: 0.8988

Epoch 34/200

81/81 [==============================] - 0s 2ms/step - loss: 0.4184 - accuracy: 0.8889

Epoch 35/200

81/81 [==============================] - 0s 2ms/step - loss: 0.3590 - accuracy: 0.9062

Epoch 36/200

81/81 [==============================] - 0s 2ms/step - loss: 0.3597 - accuracy: 0.9185

Epoch 37/200

81/81 [==============================] - 0s 2ms/step - loss: 0.3258 - accuracy: 0.9111

Epoch 38/200

81/81 [==============================] - 0s 2ms/step - loss: 0.3448 - accuracy: 0.9111

Epoch 39/200

81/81 [==============================] - 0s 2ms/step - loss: 0.2794 - accuracy: 0.9259

Epoch 40/200

81/81 [==============================] - 0s 2ms/step - loss: 0.3334 - accuracy: 0.9012

Epoch 41/200

81/81 [==============================] - 0s 2ms/step - loss: 0.3310 - accuracy: 0.9037

Epoch 42/200

81/81 [==============================] - 0s 2ms/step - loss: 0.2302 - accuracy: 0.9407

Epoch 43/200

81/81 [==============================] - 0s 2ms/step - loss: 0.2965 - accuracy: 0.9185

Epoch 44/200

81/81 [==============================] - 0s 2ms/step - loss: 0.2444 - accuracy: 0.9333

Epoch 45/200

81/81 [==============================] - 0s 2ms/step - loss: 0.2701 - accuracy: 0.9210

Epoch 46/200

81/81 [==============================] - 0s 2ms/step - loss: 0.3027 - accuracy: 0.9309

Epoch 47/200

81/81 [==============================] - 0s 3ms/step - loss: 0.2240 - accuracy: 0.9531

Epoch 48/200

81/81 [==============================] - 0s 2ms/step - loss: 0.2129 - accuracy: 0.9432

Epoch 49/200

81/81 [==============================] - 0s 2ms/step - loss: 0.2348 - accuracy: 0.9407

Epoch 50/200

81/81 [==============================] - 0s 2ms/step - loss: 0.2572 - accuracy: 0.9358

Epoch 51/200

81/81 [==============================] - 0s 2ms/step - loss: 0.2377 - accuracy: 0.9259

Epoch 52/200

81/81 [==============================] - 0s 2ms/step - loss: 0.2324 - accuracy: 0.9358

Epoch 53/200

81/81 [==============================] - 0s 2ms/step - loss: 0.2190 - accuracy: 0.9407

Epoch 54/200

81/81 [==============================] - 0s 2ms/step - loss: 0.2175 - accuracy: 0.9432

Epoch 55/200

81/81 [==============================] - 0s 2ms/step - loss: 0.2259 - accuracy: 0.9160

Epoch 56/200

81/81 [==============================] - 0s 2ms/step - loss: 0.2127 - accuracy: 0.9481

Epoch 57/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1997 - accuracy: 0.9457

Epoch 58/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1975 - accuracy: 0.9407

Epoch 59/200

81/81 [==============================] - 0s 2ms/step - loss: 0.2083 - accuracy: 0.9333

Epoch 60/200

81/81 [==============================] - 0s 2ms/step - loss: 0.2078 - accuracy: 0.9407

Epoch 61/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1838 - accuracy: 0.9432

Epoch 62/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1736 - accuracy: 0.9506

Epoch 63/200

81/81 [==============================] - 0s 2ms/step - loss: 0.2022 - accuracy: 0.9407

Epoch 64/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1883 - accuracy: 0.9481

Epoch 65/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1416 - accuracy: 0.9654

Epoch 66/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1826 - accuracy: 0.9531

Epoch 67/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1590 - accuracy: 0.9605

Epoch 68/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1992 - accuracy: 0.9481

Epoch 69/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1426 - accuracy: 0.9556

Epoch 70/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1475 - accuracy: 0.9506

Epoch 71/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1653 - accuracy: 0.9506

Epoch 72/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1344 - accuracy: 0.9580

Epoch 73/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1578 - accuracy: 0.9481

Epoch 74/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1274 - accuracy: 0.9679

Epoch 75/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1598 - accuracy: 0.9457

Epoch 76/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1740 - accuracy: 0.9309

Epoch 77/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1240 - accuracy: 0.9605

Epoch 78/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1306 - accuracy: 0.9580

Epoch 79/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1419 - accuracy: 0.9605

Epoch 80/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1218 - accuracy: 0.9679

Epoch 81/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1187 - accuracy: 0.9630

Epoch 82/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1383 - accuracy: 0.9506

Epoch 83/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1569 - accuracy: 0.9481

Epoch 84/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1289 - accuracy: 0.9605

Epoch 85/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1383 - accuracy: 0.9531

Epoch 86/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0971 - accuracy: 0.9679

Epoch 87/200

81/81 [==============================] - 0s 3ms/step - loss: 0.1353 - accuracy: 0.9605

Epoch 88/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1068 - accuracy: 0.9654

Epoch 89/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1142 - accuracy: 0.9654

Epoch 90/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1210 - accuracy: 0.9630

Epoch 91/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1283 - accuracy: 0.9679

Epoch 92/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1069 - accuracy: 0.9728

Epoch 93/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1280 - accuracy: 0.9654

Epoch 94/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0919 - accuracy: 0.9728

Epoch 95/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1054 - accuracy: 0.9728

Epoch 96/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0958 - accuracy: 0.9704

Epoch 97/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1400 - accuracy: 0.9630

Epoch 98/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1409 - accuracy: 0.9531

Epoch 99/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1149 - accuracy: 0.9630

Epoch 100/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1143 - accuracy: 0.9704

Epoch 101/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0963 - accuracy: 0.9728

Epoch 102/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1397 - accuracy: 0.9556

Epoch 103/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1061 - accuracy: 0.9605

Epoch 104/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0933 - accuracy: 0.9753

Epoch 105/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1105 - accuracy: 0.9704

Epoch 106/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0838 - accuracy: 0.9753

Epoch 107/200

81/81 [==============================] - 0s 3ms/step - loss: 0.0982 - accuracy: 0.9679

Epoch 108/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0808 - accuracy: 0.9679

Epoch 109/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1242 - accuracy: 0.9605

Epoch 110/200

81/81 [==============================] - 0s 3ms/step - loss: 0.1259 - accuracy: 0.9630

Epoch 111/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1027 - accuracy: 0.9704

Epoch 112/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1112 - accuracy: 0.9679

Epoch 113/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0989 - accuracy: 0.9679

Epoch 114/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0944 - accuracy: 0.9778

Epoch 115/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1029 - accuracy: 0.9605

Epoch 116/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0962 - accuracy: 0.9679

Epoch 117/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1131 - accuracy: 0.9654

Epoch 118/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0941 - accuracy: 0.9827

Epoch 119/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0847 - accuracy: 0.9753

Epoch 120/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1190 - accuracy: 0.9654

Epoch 121/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0636 - accuracy: 0.9802

Epoch 122/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1253 - accuracy: 0.9704

Epoch 123/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0991 - accuracy: 0.9679

Epoch 124/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1255 - accuracy: 0.9580

Epoch 125/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0781 - accuracy: 0.9728

Epoch 126/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0828 - accuracy: 0.9704

Epoch 127/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0826 - accuracy: 0.9753

Epoch 128/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1005 - accuracy: 0.9852

Epoch 129/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0535 - accuracy: 0.9852

Epoch 130/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0874 - accuracy: 0.9654

Epoch 131/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0816 - accuracy: 0.9728

Epoch 132/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1143 - accuracy: 0.9630

Epoch 133/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1165 - accuracy: 0.9679

Epoch 134/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0836 - accuracy: 0.9753

Epoch 135/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0975 - accuracy: 0.9753

Epoch 136/200

81/81 [==============================] - 0s 3ms/step - loss: 0.1253 - accuracy: 0.9605

Epoch 137/200

81/81 [==============================] - 0s 3ms/step - loss: 0.0874 - accuracy: 0.9704

Epoch 138/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0721 - accuracy: 0.9802

Epoch 139/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0993 - accuracy: 0.9753

Epoch 140/200

81/81 [==============================] - 0s 3ms/step - loss: 0.0795 - accuracy: 0.9728

Epoch 141/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0957 - accuracy: 0.9728

Epoch 142/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0914 - accuracy: 0.9728

Epoch 143/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0679 - accuracy: 0.9802

Epoch 144/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0654 - accuracy: 0.9802

Epoch 145/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0734 - accuracy: 0.9852

Epoch 146/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1020 - accuracy: 0.9630

Epoch 147/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1036 - accuracy: 0.9654

Epoch 148/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0893 - accuracy: 0.9704

Epoch 149/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0591 - accuracy: 0.9877

Epoch 150/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0726 - accuracy: 0.9778

Epoch 151/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0931 - accuracy: 0.9630

Epoch 152/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1536 - accuracy: 0.9506

Epoch 153/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0930 - accuracy: 0.9778

Epoch 154/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0995 - accuracy: 0.9630

Epoch 155/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0607 - accuracy: 0.9802

Epoch 156/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0754 - accuracy: 0.9778

Epoch 157/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0520 - accuracy: 0.9852

Epoch 158/200

81/81 [==============================] - 0s 3ms/step - loss: 0.0712 - accuracy: 0.9778

Epoch 159/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0817 - accuracy: 0.9728

Epoch 160/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1164 - accuracy: 0.9728

Epoch 161/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0821 - accuracy: 0.9753

Epoch 162/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0605 - accuracy: 0.9802

Epoch 163/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0524 - accuracy: 0.9802

Epoch 164/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0452 - accuracy: 0.9852

Epoch 165/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0972 - accuracy: 0.9679

Epoch 166/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0902 - accuracy: 0.9679

Epoch 167/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1203 - accuracy: 0.9630

Epoch 168/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0579 - accuracy: 0.9827

Epoch 169/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0879 - accuracy: 0.9728

Epoch 170/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0604 - accuracy: 0.9802

Epoch 171/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0945 - accuracy: 0.9728

Epoch 172/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0775 - accuracy: 0.9728

Epoch 173/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0583 - accuracy: 0.9802

Epoch 174/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0602 - accuracy: 0.9877

Epoch 175/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0573 - accuracy: 0.9802

Epoch 176/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0717 - accuracy: 0.9753

Epoch 177/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1128 - accuracy: 0.9630

Epoch 178/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0673 - accuracy: 0.9654

Epoch 179/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0716 - accuracy: 0.9728

Epoch 180/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0659 - accuracy: 0.9778

Epoch 181/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0671 - accuracy: 0.9827

Epoch 182/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0840 - accuracy: 0.9778

Epoch 183/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0628 - accuracy: 0.9827

Epoch 184/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0781 - accuracy: 0.9802

Epoch 185/200

81/81 [==============================] - 0s 2ms/step - loss: 0.1037 - accuracy: 0.9753

Epoch 186/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0830 - accuracy: 0.9753

Epoch 187/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0463 - accuracy: 0.9802

Epoch 188/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0528 - accuracy: 0.9802

Epoch 189/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0570 - accuracy: 0.9802

Epoch 190/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0593 - accuracy: 0.9778

Epoch 191/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0761 - accuracy: 0.9778

Epoch 192/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0523 - accuracy: 0.9827

Epoch 193/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0898 - accuracy: 0.9778

Epoch 194/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0674 - accuracy: 0.9728

Epoch 195/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0676 - accuracy: 0.9753

Epoch 196/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0754 - accuracy: 0.9728

Epoch 197/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0806 - accuracy: 0.9753

Epoch 198/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0497 - accuracy: 0.98270

Epoch 199/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0485 - accuracy: 0.9852

Epoch 200/200

81/81 [==============================] - 0s 2ms/step - loss: 0.0705 - accuracy: 0.9753

model created

FOR PREDICTING RESPONSE

**You have to add or run this below script by loading model**

linkcode

**For loading saved model**

// from keras.models import load\_model

model = load\_model('chatbot\_model.h5') //

**Predict the response**

To predict the sentences and get a response from the user to let us create a new file ‘chatapp.py’.

* We will load the trained model and then use a graphical user interface that will predict the response from the bot. The model will only tell us the class it belongs to, so we will implement some functions which will identify the class and then retrieve us a random response from the list of responses.
* Again we import the necessary packages and load the ‘words.pkl’ and ‘classes.pkl’ pickle files which we have created when we trained our model.

//---

intents = json.loads(open('/kaggle/input/chatbot-dataset/intents.json').read())

words = pickle.load(open('words.pkl','rb'))

classes = pickle.load(open('classes.pkl','rb'))

//---

**To predict the class, we will need to provide input in the same way as we did while training. So we will create some functions that will perform text preprocessing and then predict the class**

//----

Utility Methods

def clean\_up\_sentence(sentence):

# tokenize the pattern - split words into array

sentence\_words = nltk.word\_tokenize(sentence)

#print(sentence\_words)

# stem each word - create short form for word

sentence\_words = [lemmatizer.lemmatize(word.lower()) for word in sentence\_words]

#print(sentence\_words)

return sentence\_words

return bag of words array: 0 or 1 for each word in the bag that exists in the sentence

def bow(sentence, words, show\_details=True):

# tokenize the pattern

sentence\_words = clean\_up\_sentence(sentence)

#print(sentence\_words)

# bag of words - matrix of N words, vocabulary matrix

bag = [0]\*len(words)

#print(bag)

for s in sentence\_words:

for i,w in enumerate(words):

if w == s:

# assign 1 if current word is in the vocabulary position

bag[i] = 1

if show\_details:

print ("found in bag: %s" % w)

#print ("found in bag: %s" % w)

#print(bag)

return(np.array(bag))

def predict\_class(sentence, model):

# filter out predictions below a threshold

p = bow(sentence, words,show\_details=False)

#print(p)

res = model.predict(np.array([p]))[0]

#print(res)

ERROR\_THRESHOLD = 0.25

results = [[i,r] for i,r in enumerate(res) if r>ERROR\_THRESHOLD]

#print(results)

# sort by strength of probability

results.sort(key=lambda x: x[1], reverse=True)

#print(results)

return\_list = []

for r in results:

return\_list.append({"intent": classes[r[0]], "probability": str(r[1])})

return return\_list

#print(return\_list)

//----

**After predicting the class, we will get a random response from the list of intents:**

//----

def getResponse(ints, intents\_json):

tag = ints[0]['intent']

#print(tag)

list\_of\_intents = intents\_json['intents']

#print(list\_of\_intents)

for i in list\_of\_intents:

if(i['tag']== tag):

result = random.choice(i['responses'])

break

return result

def chatbot\_response(text): ints = predict\_class(text, model)

#print(ints)

res = getResponse(ints, intents)

#print(res)

return res

//---  
**Enter you queries**  
//----  
start = True

while start:

query = input('Enter Message:')

if query in ['quit','exit','bye']:

start = False

continue

try:

res = chatbot\_response(query)

print(res)

except:

print('You may need to rephrase your question.')

//-----