Chatbot

A chatbot is an intelligent software that can communicate and perform actions like a human. Chatbots are used a lot in customer interaction, marketing on social network sites, and instant messaging the client. There are two basic types of chatbot models based on how they are built; Retrieval based and Generative based models.

In this Python project with source code, we are going to build a chatbot using machine learning techniques. The chatbot will be trained on the dataset which contains categories (intents), patterns, and responses.

Here is my code…

In [ ]:

```python
import nltk
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
import json
import pickle

import numpy as np
from keras.models import Sequential
from keras.layers import Dense, Activation, Dropout
from keras.optimizers import SGD
import random
```

In [2]:

```python
words=[]
classes = []
documents = []
ignore_words = ['?', '!']
```

In [3]:

```python
data = {"intents": [
             {"tag": "greeting",
              "patterns": ["Hello", "How are you?", "Hi there", "Hi", "Whats up"],
              "responses": ["Howdy Partner!", "Hello", "How are you doing?", "Greetings!", "How do you do?"],
             },
             {"tag": "age",
              "patterns": ["how old are you?", "when is your birthday?", "when was you born?"],
              "responses": ["I am 24 years old", "I was born in 1996", "My birthday is July 3rd and I was born in 1996", "03/07/1996"]
             },
             {"tag": "date",
              "patterns": ["what are you doing this weekend?",
               "do you want to hang out some time?", "what are your plans for this week"],
              "responses": ["I am available all week", "I don't have any plans", "I am not busy"]
             },
             {"tag": "name",
              "patterns": ["what's your name?", "what are you called?", "who are you?"],
              "responses": ["My name is Kippi", "I'm Kippi", "Kippi"]
             },
             {"tag": "goodbye",
              "patterns": [ "bye", "g2g", "see ya", "adios", "cya"],
              "responses": ["It was nice speaking to you", "See you later", "Speak soon!"]
             },
            {'patterns':[],
             'response': ["Sorry, can't understand you",'Please give me more info','Not sure I understand'],
             'tag':'noanswer'}
]}
```

In [4]:

```python
for intent in data["intents"]:
    for pattern in intent['patterns']:

        #tokenize each word
        w = nltk.word_tokenize(pattern)
        words.extend(w)
        #add documents in the corpus
        documents.append((w, intent['tag']))

        # add to our classes list
        if intent['tag'] not in classes:
            classes.append(intent['tag'])
```

In [5]:

```python
# 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")
# classes = intents
print (len(classes), "classes", classes)
# words = all words, vocabulary
print (len(words), "unique lemmatized words", words)

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

```
19 documents
5 classes ['age', 'date', 'goodbye', 'greeting', 'name']
39 unique lemmatized words ["'s", 'adios', 'are', 'birthday', 'born', 'bye', 'called', 'cya', 'do', 'doing', 'for', 'g2g', 'hang', 'hello', 'hi', 'how', 'is', 'name', 'old', 'out', 'plan', 'see', 'some', 'there', 'this', 'time', 'to', 'up', 'wa', 'want', 'week', 'weekend', 'what', 'whats', 'w
hen', 'who', 'ya', 'you', 'your']
```

In [6]:

```python
# 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 for the pattern
    pattern_words = doc[0]
    # lemmatize each word - create base word, in attempt to represent related words
    pattern_words = [lemmatizer.lemmatize(word.lower()) for word in pattern_words]
    # create our bag of words array with 1, if word match found in current pattern
    for w in words:
        bag.append(1) if w in pattern_words else bag.append(0)

    # output is a '0' for each tag and '1' for current tag (for each pattern)
    output_row = list(output_empty)
    output_row[classes.index(doc[1])] = 1

    training.append([bag, output_row])
# shuffle our features and turn into np.array
random.shuffle(training)
training = np.array(training)
# create train and test lists. X - patterns, Y - intents
train_x = list(training[:,0])
train_y = list(training[:,1])
print("Training data created")
```

```
Training data created
C:\Users\Lenovo\AppData\Local\Temp/ipykernel_41572/2748295590.py:24: VisibleDeprecationWarning: Creating an ndarray from ragged nested sequences (w
hich is a list-or-tuple of lists-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]:

```python
# 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'))

# 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=sgd, metrics=['accuracy'])

#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")
```

```
C:\Users\Lenovo\anaconda3\lib\site-packages\keras\optimizers\optimizer_v2\gradient_descent.py:108: UserWarning: The `lr` argument is deprecated, us
e `learning_rate` instead.
  super(SGD, self).__init__(name, **kwargs)
Epoch 1/200
4/4 [==============================] - 1s 0s/step - loss: 1.6737 - accuracy: 0.1579
Epoch 2/200
4/4 [==============================] - 0s 8ms/step - loss: 1.5669 - accuracy: 0.3158
Epoch 3/200
4/4 [==============================] - 0s 10ms/step - loss: 1.5956 - accuracy: 0.2632
Epoch 4/200
4/4 [==============================] - 0s 6ms/step - loss: 1.5525 - accuracy: 0.4211
Epoch 5/200
4/4 [==============================] - 0s 5ms/step - loss: 1.5359 - accuracy: 0.3158
Epoch 6/200
4/4 [==============================] - 0s 5ms/step - loss: 1.4088 - accuracy: 0.6316
Epoch 7/200
4/4 [==============================] - 0s 7ms/step - loss: 1.4684 - accuracy: 0.3158
Epoch 8/200
4/4 [==============================] - 0s 4ms/step - loss: 1.3874 - accuracy: 0.4737
Epoch 9/200
4/4 [==============================] - 0s 4ms/step - loss: 1.2990 - accuracy: 0.6842
Epoch 10/200
4/4 [==============================] - 0s 3ms/step - loss: 1.2345 - accuracy: 0.6842
Epoch 11/200
4/4 [==============================] - 0s 3ms/step - loss: 1.1419 - accuracy: 0.6316
Epoch 12/200
4/4 [==============================] - 0s 872us/step - loss: 1.0982 - accuracy: 0.7368
Epoch 13/200
4/4 [==============================] - 0s 6ms/step - loss: 1.0918 - accuracy: 0.7368
Epoch 14/200
4/4 [==============================] - 0s 10ms/step - loss: 0.9246 - accuracy: 0.8421
Epoch 15/200
4/4 [==============================] - 0s 5ms/step - loss: 0.8932 - accuracy: 0.7895
Epoch 16/200
4/4 [==============================] - 0s 5ms/step - loss: 1.0543 - accuracy: 0.6316
Epoch 17/200
4/4 [==============================] - 0s 4ms/step - loss: 1.0747 - accuracy: 0.6842
Epoch 18/200
4/4 [==============================] - 0s 0s/step - loss: 0.9097 - accuracy: 0.7895
Epoch 19/200
4/4 [==============================] - 0s 5ms/step - loss: 0.7921 - accuracy: 0.8421
Epoch 20/200
4/4 [==============================] - 0s 6ms/step - loss: 0.7244 - accuracy: 0.8421
Epoch 21/200
4/4 [==============================] - 0s 3ms/step - loss: 0.8623 - accuracy: 0.6842
Epoch 22/200
4/4 [==============================] - 0s 5ms/step - loss: 0.6487 - accuracy: 0.7368
Epoch 23/200
4/4 [==============================] - 0s 9ms/step - loss: 0.6131 - accuracy: 0.8421
Epoch 24/200
4/4 [==============================] - 0s 6ms/step - loss: 0.4808 - accuracy: 0.9474
Epoch 25/200
4/4 [==============================] - 0s 6ms/step - loss: 0.4561 - accuracy: 0.8947
Epoch 26/200
4/4 [==============================] - 0s 4ms/step - loss: 0.5006 - accuracy: 0.8947
Epoch 27/200
4/4 [==============================] - 0s 5ms/step - loss: 0.5962 - accuracy: 0.6842
Epoch 28/200
4/4 [==============================] - 0s 7ms/step - loss: 0.6357 - accuracy: 0.8947
Epoch 29/200
4/4 [==============================] - 0s 3ms/step - loss: 0.3577 - accuracy: 0.9474
Epoch 30/200
4/4 [==============================] - 0s 5ms/step - loss: 0.3748 - accuracy: 0.8947
Epoch 31/200
4/4 [==============================] - 0s 7ms/step - loss: 0.5282 - accuracy: 0.8947
Epoch 32/200
4/4 [==============================] - 0s 6ms/step - loss: 0.3984 - accuracy: 0.8421
Epoch 33/200
4/4 [==============================] - 0s 3ms/step - loss: 0.4082 - accuracy: 0.8947
Epoch 34/200
4/4 [==============================] - 0s 5ms/step - loss: 0.4341 - accuracy: 0.8947
Epoch 35/200
4/4 [==============================] - 0s 7ms/step - loss: 0.3564 - accuracy: 0.9474
Epoch 36/200
4/4 [==============================] - 0s 3ms/step - loss: 0.3147 - accuracy: 0.9474
Epoch 37/200
4/4 [==============================] - 0s 5ms/step - loss: 0.1719 - accuracy: 0.9474
Epoch 38/200
4/4 [==============================] - 0s 5ms/step - loss: 0.1901 - accuracy: 1.0000
Epoch 39/200
4/4 [==============================] - 0s 6ms/step - loss: 0.3028 - accuracy: 0.8947
Epoch 40/200
4/4 [==============================] - 0s 7ms/step - loss: 0.2143 - accuracy: 0.9474
Epoch 41/200
4/4 [==============================] - 0s 8ms/step - loss: 0.1990 - accuracy: 0.9474
Epoch 42/200
4/4 [==============================] - 0s 6ms/step - loss: 0.1482 - accuracy: 0.9474
Epoch 43/200
4/4 [==============================] - 0s 6ms/step - loss: 0.1916 - accuracy: 0.8947
Epoch 44/200
4/4 [==============================] - 0s 4ms/step - loss: 0.2123 - accuracy: 0.9474
Epoch 45/200
4/4 [==============================] - 0s 5ms/step - loss: 0.2063 - accuracy: 0.9474
Epoch 46/200
4/4 [==============================] - 0s 5ms/step - loss: 0.1606 - accuracy: 0.9474
Epoch 47/200
4/4 [==============================] - 0s 6ms/step - loss: 0.2262 - accuracy: 0.9474
Epoch 48/200
4/4 [==============================] - 0s 5ms/step - loss: 0.2287 - accuracy: 1.0000
Epoch 49/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0878 - accuracy: 1.0000
Epoch 50/200
4/4 [==============================] - 0s 5ms/step - loss: 0.1788 - accuracy: 0.9474
Epoch 51/200
4/4 [==============================] - 0s 3ms/step - loss: 0.1331 - accuracy: 1.0000
Epoch 52/200
4/4 [==============================] - 0s 5ms/step - loss: 0.1926 - accuracy: 0.8947
Epoch 53/200
4/4 [==============================] - 0s 6ms/step - loss: 0.1362 - accuracy: 1.0000
Epoch 54/200
4/4 [==============================] - 0s 6ms/step - loss: 0.0652 - accuracy: 1.0000
Epoch 55/200
4/4 [==============================] - 0s 5ms/step - loss: 0.1922 - accuracy: 0.9474
Epoch 56/200
4/4 [==============================] - 0s 2ms/step - loss: 0.2603 - accuracy: 0.8947
Epoch 57/200
4/4 [==============================] - 0s 5ms/step - loss: 0.1405 - accuracy: 0.9474
Epoch 58/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0669 - accuracy: 1.0000
Epoch 59/200
4/4 [==============================] - 0s 5ms/step - loss: 0.2484 - accuracy: 0.8947
Epoch 60/200
4/4 [==============================] - 0s 6ms/step - loss: 0.0758 - accuracy: 1.0000
Epoch 61/200
4/4 [==============================] - 0s 5ms/step - loss: 0.1300 - accuracy: 0.9474
Epoch 62/200
4/4 [==============================] - 0s 6ms/step - loss: 0.0653 - accuracy: 0.9474
Epoch 63/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0825 - accuracy: 1.0000
Epoch 64/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0391 - accuracy: 1.0000
Epoch 65/200
4/4 [==============================] - 0s 7ms/step - loss: 0.0367 - accuracy: 1.0000
Epoch 66/200
4/4 [==============================] - 0s 5ms/step - loss: 0.1755 - accuracy: 0.9474
Epoch 67/200
4/4 [==============================] - 0s 4ms/step - loss: 0.1170 - accuracy: 1.0000
Epoch 68/200
4/4 [==============================] - 0s 3ms/step - loss: 0.2079 - accuracy: 0.8947
Epoch 69/200
4/4 [==============================] - 0s 6ms/step - loss: 0.1162 - accuracy: 1.0000
Epoch 70/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0723 - accuracy: 1.0000
Epoch 71/200
4/4 [==============================] - 0s 5ms/step - loss: 0.2095 - accuracy: 0.8421
Epoch 72/200
4/4 [==============================] - 0s 6ms/step - loss: 0.0666 - accuracy: 1.0000
Epoch 73/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0706 - accuracy: 1.0000
Epoch 74/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0870 - accuracy: 1.0000
Epoch 75/200
4/4 [==============================] - 0s 5ms/step - loss: 0.1114 - accuracy: 1.0000
Epoch 76/200
4/4 [==============================] - 0s 5ms/step - loss: 0.1353 - accuracy: 0.9474
Epoch 77/200
4/4 [==============================] - 0s 5ms/step - loss: 0.1409 - accuracy: 0.9474
Epoch 78/200
4/4 [==============================] - 0s 5ms/step - loss: 0.1614 - accuracy: 0.9474
Epoch 79/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0581 - accuracy: 1.0000
Epoch 80/200
4/4 [==============================] - 0s 3ms/step - loss: 0.1666 - accuracy: 0.8947
Epoch 81/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0346 - accuracy: 1.0000
Epoch 82/200
4/4 [==============================] - 0s 5ms/step - loss: 0.1413 - accuracy: 0.9474
Epoch 83/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0574 - accuracy: 1.0000
Epoch 84/200
4/4 [==============================] - 0s 8ms/step - loss: 0.0979 - accuracy: 1.0000
Epoch 85/200
4/4 [==============================] - 0s 0s/step - loss: 0.2436 - accuracy: 0.8947
Epoch 86/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0690 - accuracy: 1.0000
Epoch 87/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0789 - accuracy: 1.0000
Epoch 88/200
4/4 [==============================] - 0s 8ms/step - loss: 0.1167 - accuracy: 1.0000
Epoch 89/200
4/4 [==============================] - 0s 0s/step - loss: 0.1310 - accuracy: 0.9474
Epoch 90/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0510 - accuracy: 1.0000
Epoch 91/200
4/4 [==============================] - 0s 5ms/step - loss: 0.1434 - accuracy: 0.9474
Epoch 92/200
4/4 [==============================] - 0s 8ms/step - loss: 0.2276 - accuracy: 0.8947
Epoch 93/200
4/4 [==============================] - 0s 6ms/step - loss: 0.2008 - accuracy: 0.9474
Epoch 94/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0210 - accuracy: 1.0000
Epoch 95/200
4/4 [==============================] - 0s 6ms/step - loss: 0.0336 - accuracy: 1.0000
Epoch 96/200
4/4 [==============================] - 0s 0s/step - loss: 0.0753 - accuracy: 1.0000
Epoch 97/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0452 - accuracy: 1.0000
Epoch 98/200
4/4 [==============================] - 0s 6ms/step - loss: 0.0259 - accuracy: 1.0000
Epoch 99/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0758 - accuracy: 1.0000
Epoch 100/200
4/4 [==============================] - 0s 9ms/step - loss: 0.0688 - accuracy: 0.9474
Epoch 101/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0304 - accuracy: 1.0000
Epoch 102/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0440 - accuracy: 1.0000
Epoch 103/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0332 - accuracy: 1.0000
Epoch 104/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0226 - accuracy: 1.0000
Epoch 105/200
4/4 [==============================] - 0s 5ms/step - loss: 0.1161 - accuracy: 1.0000
Epoch 106/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0265 - accuracy: 1.0000
Epoch 107/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0977 - accuracy: 0.9474
Epoch 108/200
4/4 [==============================] - 0s 2ms/step - loss: 0.1081 - accuracy: 0.9474
Epoch 109/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0283 - accuracy: 1.0000
Epoch 110/200
4/4 [==============================] - 0s 11ms/step - loss: 0.0387 - accuracy: 1.0000
Epoch 111/200
4/4 [==============================] - 0s 5ms/step - loss: 0.1171 - accuracy: 0.9474
Epoch 112/200
4/4 [==============================] - 0s 6ms/step - loss: 0.0273 - accuracy: 1.0000
Epoch 113/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0360 - accuracy: 1.0000
Epoch 114/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0156 - accuracy: 1.0000
Epoch 115/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0330 - accuracy: 1.0000
Epoch 116/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0118 - accuracy: 1.0000
Epoch 117/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0878 - accuracy: 1.0000
Epoch 118/200
4/4 [==============================] - 0s 6ms/step - loss: 0.1120 - accuracy: 0.9474
Epoch 119/200
4/4 [==============================] - 0s 2ms/step - loss: 0.0910 - accuracy: 0.9474
Epoch 120/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0315 - accuracy: 1.0000
Epoch 121/200
4/4 [==============================] - 0s 0s/step - loss: 0.0078 - accuracy: 1.0000
Epoch 122/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0138 - accuracy: 1.0000
Epoch 123/200
4/4 [==============================] - 0s 9ms/step - loss: 0.0196 - accuracy: 1.0000
Epoch 124/200
4/4 [==============================] - 0s 6ms/step - loss: 0.0586 - accuracy: 1.0000
Epoch 125/200
4/4 [==============================] - 0s 6ms/step - loss: 0.1321 - accuracy: 0.8947
Epoch 126/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0456 - accuracy: 1.0000
Epoch 127/200
4/4 [==============================] - 0s 3ms/step - loss: 0.0584 - accuracy: 1.0000
Epoch 128/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0307 - accuracy: 1.0000
Epoch 129/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0416 - accuracy: 1.0000
Epoch 130/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0167 - accuracy: 1.0000
Epoch 131/200
4/4 [==============================] - 0s 6ms/step - loss: 0.0329 - accuracy: 1.0000
Epoch 132/200
4/4 [==============================] - 0s 6ms/step - loss: 0.0999 - accuracy: 0.9474
Epoch 133/200
4/4 [==============================] - 0s 2ms/step - loss: 0.0158 - accuracy: 1.0000
Epoch 134/200
4/4 [==============================] - 0s 3ms/step - loss: 0.0127 - accuracy: 1.0000
Epoch 135/200
4/4 [==============================] - 0s 4ms/step - loss: 0.0314 - accuracy: 1.0000
Epoch 136/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0304 - accuracy: 1.0000
Epoch 137/200
4/4 [==============================] - 0s 4ms/step - loss: 0.0148 - accuracy: 1.0000
Epoch 138/200
4/4 [==============================] - 0s 4ms/step - loss: 0.0122 - accuracy: 1.0000
Epoch 139/200
4/4 [==============================] - 0s 6ms/step - loss: 0.0167 - accuracy: 1.0000
Epoch 140/200
4/4 [==============================] - 0s 3ms/step - loss: 0.0111 - accuracy: 1.0000
Epoch 141/200
4/4 [==============================] - 0s 4ms/step - loss: 0.0267 - accuracy: 1.0000
Epoch 142/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0532 - accuracy: 1.0000
Epoch 143/200
4/4 [==============================] - 0s 3ms/step - loss: 0.0248 - accuracy: 1.0000
Epoch 144/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0105 - accuracy: 1.0000
Epoch 145/200
4/4 [==============================] - 0s 4ms/step - loss: 0.1026 - accuracy: 0.9474
Epoch 146/200
4/4 [==============================] - 0s 3ms/step - loss: 0.0697 - accuracy: 0.9474
Epoch 147/200
4/4 [==============================] - 0s 6ms/step - loss: 0.0471 - accuracy: 1.0000
Epoch 148/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0180 - accuracy: 1.0000
Epoch 149/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0143 - accuracy: 1.0000
Epoch 150/200
4/4 [==============================] - 0s 10ms/step - loss: 0.0212 - accuracy: 1.0000
Epoch 151/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0053 - accuracy: 1.0000
Epoch 152/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0398 - accuracy: 1.0000
Epoch 153/200
4/4 [==============================] - 0s 0s/step - loss: 0.0254 - accuracy: 1.0000
Epoch 154/200
4/4 [==============================] - 0s 6ms/step - loss: 0.0193 - accuracy: 1.0000
Epoch 155/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0319 - accuracy: 1.0000
Epoch 156/200
4/4 [==============================] - 0s 0s/step - loss: 0.0175 - accuracy: 1.0000
Epoch 157/200
4/4 [==============================] - 0s 9ms/step - loss: 0.0611 - accuracy: 1.0000
Epoch 158/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0166 - accuracy: 1.0000
Epoch 159/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0052 - accuracy: 1.0000
Epoch 160/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0082 - accuracy: 1.0000
Epoch 161/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0136 - accuracy: 1.0000
Epoch 162/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0816 - accuracy: 0.9474
Epoch 163/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0208 - accuracy: 1.0000
Epoch 164/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0309 - accuracy: 1.0000
Epoch 165/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0146 - accuracy: 1.0000
Epoch 166/200
4/4 [==============================] - 0s 4ms/step - loss: 0.0106 - accuracy: 1.0000
Epoch 167/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0176 - accuracy: 1.0000
Epoch 168/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0072 - accuracy: 1.0000
Epoch 169/200
4/4 [==============================] - 0s 7ms/step - loss: 0.0225 - accuracy: 1.0000
Epoch 170/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0243 - accuracy: 1.0000
Epoch 171/200
4/4 [==============================] - 0s 5ms/step - loss: 0.1171 - accuracy: 1.0000
Epoch 172/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0363 - accuracy: 1.0000
Epoch 173/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0133 - accuracy: 1.0000
Epoch 174/200
4/4 [==============================] - 0s 3ms/step - loss: 0.0192 - accuracy: 1.0000
Epoch 175/200
4/4 [==============================] - 0s 6ms/step - loss: 0.0154 - accuracy: 1.0000
Epoch 176/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0192 - accuracy: 1.0000
Epoch 177/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0359 - accuracy: 1.0000
Epoch 178/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0141 - accuracy: 1.0000
Epoch 179/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0076 - accuracy: 1.0000
Epoch 180/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0155 - accuracy: 1.0000
Epoch 181/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0031 - accuracy: 1.0000
Epoch 182/200
4/4 [==============================] - 0s 2ms/step - loss: 0.0200 - accuracy: 1.0000
Epoch 183/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0051 - accuracy: 1.0000
Epoch 184/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0228 - accuracy: 1.0000
Epoch 185/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0243 - accuracy: 1.0000
Epoch 186/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0134 - accuracy: 1.0000
Epoch 187/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0309 - accuracy: 1.0000
Epoch 188/200
4/4 [==============================] - 0s 3ms/step - loss: 0.0054 - accuracy: 1.0000
Epoch 189/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0027 - accuracy: 1.0000
Epoch 190/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0322 - accuracy: 1.0000
Epoch 191/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0065 - accuracy: 1.0000
Epoch 192/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0028 - accuracy: 1.0000
Epoch 193/200
4/4 [==============================] - 0s 4ms/step - loss: 0.0375 - accuracy: 1.0000
Epoch 194/200
4/4 [==============================] - 0s 9ms/step - loss: 0.0104 - accuracy: 1.0000
Epoch 195/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0047 - accuracy: 1.0000
Epoch 196/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0184 - accuracy: 1.0000
Epoch 197/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0272 - accuracy: 1.0000
Epoch 198/200
4/4 [==============================] - 0s 5ms/step - loss: 0.0054 - accuracy: 1.0000
Epoch 199/200
4/4 [==============================] - 0s 7ms/step - loss: 0.0039 - accuracy: 1.0000
Epoch 200/200
4/4 [==============================] - 0s 6ms/step - loss: 0.0024 - accuracy: 1.0000
model created
```

In [8]:

```python
import nltk
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
import pickle
import numpy as np

from keras.models import load_model
model = load_model('chatbot_model.h5')
import json
import random
words = pickle.load(open('words.pkl','rb'))
classes = pickle.load(open('classes.pkl','rb'))
```

In [18]:

```python
def clean_up_sentence(sentence):
    # tokenize the pattern - split words into array
    sentence_words = nltk.word_tokenize(sentence)
    # stem each word - create short form for word
    sentence_words = [lemmatizer.lemmatize(word.lower()) for word in 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)
    # bag of words - matrix of N words, vocabulary matrix
    bag = [0]*len(words)
    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)
    return(np.array(bag))

def predict_class(sentence, model):
    # filter out predictions below a threshold
    p = bow(sentence, words,show_details=False)
    res = model.predict(np.array([p]))[0]
    ERROR_THRESHOLD = 0.25
    results = [[i,r] for i,r in enumerate(res) 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({"data": classes[r[0]], "probability": str(r[1])})
    return return_list
```

In [25]:

```python
def getResponse(ints, intents_json):
    tag = ints[0]['data']
    list_of_intents = intents_json['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)
    res = getResponse(ints, data)
    return res
```

In [ ]:

```python
#Creating GUI with tkinter
import tkinter
from tkinter import *


def send():
    msg = EntryBox.get("1.0",'end-1c').strip()
    EntryBox.delete("0.0",END)

    if msg != '':
        ChatLog.config(state=NORMAL)
        ChatLog.insert(END, "You: " + msg + '\n\n')
        ChatLog.config(foreground="#442265", font=("Verdana", 12 ))

        res = chatbot_response(msg)
        ChatLog.insert(END, "Bot: " + res + '\n\n')

        ChatLog.config(state=DISABLED)
        ChatLog.yview(END)

base = Tk()
base.title("Hello")
base.geometry("400x500")
base.resizable(width=FALSE, height=FALSE)

#Create Chat window
ChatLog = Text(base, bd=0, bg="white", height="8", width="50", font="Arial",)

ChatLog.config(state=DISABLED)

#Bind scrollbar to Chat window
scrollbar = Scrollbar(base, command=ChatLog.yview, cursor="heart")
ChatLog['yscrollcommand'] = scrollbar.set

#Create Button to send message
SendButton = Button(base, font=("Verdana",12,'bold'), text="Send", width="12", height=5,
                    bd=0, bg="#32de97", activebackground="#3c9d9b",fg='#ffffff',
                    command= send )


#Create the box to enter message
EntryBox = Text(base, bd=0, bg="white",width="29", height="5", font="Arial")
#EntryBox.bind("<Return>", send)


#Place all components on the screen
scrollbar.place(x=376,y=6, height=386)
ChatLog.place(x=6,y=6, height=386, width=370)
EntryBox.place(x=128, y=401, height=90, width=265)
SendButton.place(x=6, y=401, height=90)

base.mainloop()
```

In [ ]: