Report on Chatbot to be Deployed to Team Lancers

The chatbot model was done using the implementation guideline found on <https://techwithtim.net/tutorials/ai-chatbot/part-1/>. The model was trained based on the data in a json file. Although, we can json or text file from from any source to train a chatbot model, for simplicity and code along with the author, the JSON file was used. It is a set of messages that the users are likely to type in and mapping them to a group of appropriate responses.

The tag on each dictionary in the file indicates the group that each message belongs too. With this data we will train a neural network to take a sentence of words and classify it as one of the tags in our file. Then we can simply take a response from those groups and display that to the user. The more tags, responses, and patterns you provide to the chatbot the better and more complex it will be.

Training the model begins with installing and importing the libraries on the requirements.txt file, after which the json file was loaded, opened and saved to a **data** variable

Next, I looped through the data file to extract the **patterns and tags**, afterwhich we also extracted a list of all of the unique words in our patterns using the nltk.word\_tokenizer,

Next, the words were stemmed and store as a list in a variable to reduce the vocabulary of our model and attempt to find the more general meaning behind sentences.

To format these stemmed words in inputs value that the model will understand, we used the process called **bag of words,** we represented each sentence with a list the length of the amount of words in our model’s vocabulary. Each position in the list will represent a word from our vocabulary. If the position in the list is a 1 then that will mean that the word exists in our sentence, if it is a 0 then the word is nor present. This was also done to both input and output variables.

Having converted, next is to build and train our model, the model was built so as to look at the bag of words function and give a class that they belong too (one of our tags from the JSON file). After which the model was trained by fitting it with the data.

Next is to make prediction by generating a response to any sentence the user types in, taking into cognizance that the model does not take string input, it takes a bag of words. We also need to realize that our model does not spit out sentences, it generates a list of probabilities for all of our classes. This makes the process to generate a response look like the following:  
– Get some input from the user  
– Convert it to a bag of words  
– Get a prediction from the model  
– Find the most probable class  
– Pick a response from that class