Packages used: -

|  |  |  |
| --- | --- | --- |
| Code | Using | link |
| nltk | work with human language data | https://www.nltk.org/ |
| nltk.stem.wordnet.WordNetLemmatizer | Lemmatize using WordNet’s built-in morphy function. Returns the input word unchanged if it cannot be found in WordNet. | https://www.nltk.org/api/nltk.stem.wordnet.html |
| JSON | The JSON module is mainly used to convert the python dictionary into a JSON string that can be written into a file. |  |
| pickle | to converts any kind of python objects (list, dict, etc.) into byte streams (0s and 1s) |  |
| numpy | provides an array, lists related operations |  |
| **Keras**  **(layers)** | is an Open Source Neural Network library  note: Keras doesn’t handle low-level computation. Instead, it uses another library to do it, called the “Backend.  Note: Tensorflow is the default “backend engine” |  |
| keras.models. Sequential | **one input tensor and one output tensor.** | https://keras.io/guides/sequential\_model/ |
| Dense | Layer is a widely used Keras layer for creating a deeply connected layer in the neural network where each of the neurons of the dense layers receives input from all neurons of the previous layer | https://keras.io/api/layers/core\_layers/dense/ |
| The dropout layer | for reducing over-fitting in neural network models. | https://keras.io/api/layers/regularization\_layers/dropout/ |
| SGD | Gradient descent (with momentum) optimizer. | https://keras.io/api/optimizers/sgd/ |

Functions used: -

|  |  |
| --- | --- |
| words.extend(w) | Add the elements of w to the words list |
| documents.append((w, intent['tag'])) | Add an element to the documens list |
| Lower( ) | lower case |
| pickle.dump( ) | Convert to 0 ,1 |

Functions used in nltk:

|  |  |
| --- | --- |
| nltk.word\_tokenize(pattern) | Tokenizers divide strings into lists of substrings  Divide the text into word,puctuation |

the compilation process.

|  |  |  |
| --- | --- | --- |
| Function | using | link |
| ***fit()*** | used to evaluate your model on training. This can be also used for graphing model performance.  \*data will be the tuple (x, y)  \*epochs :no of times the model is needed to be evaluated during training 200  \*batch\_size :training instances 5 | https://www.tutorialspoint.com/keras/keras\_model\_compilation.htm |
| Compile() | to compile the model   * Use lose function (categorical\_crossentropy) * Use optimizer sgd * Use metrix (accuracy) | https://www.tutorialspoint.com/keras/keras\_model\_compilation.htm |

Steps in on word: -

1-Import

2-Tokenization

3-Convert to stream

4-Training data

5-Train x to pattern

6-Train y to intents

7-model (Neural network model)

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 soft max

8-fit model

9-compile model

10-predict answers using model