

# Chatbots

In this project you will make a chatbot in Python that can answer questions about a topic of your choice.

```
(env) Dales-MBP:python dalelane$ python chatbot.py
What would you like to know about owls?
> How many types of owl are there?
There are over 200 species of owl. Some common ones include Barn Owls, Eagle Owls, Snowy Owls, Elf Owls, Great Horned Owls, and Tawny Owls.

> What sort of things do owls eat?
It depends on the species of owl. Small owls eat invertebrates (such as spiders, insects and worms). Larger owls eat animals like fish, birds, mice, shrews and voles.

> What is the capital city of France?
I don't understand. Ask me something else!

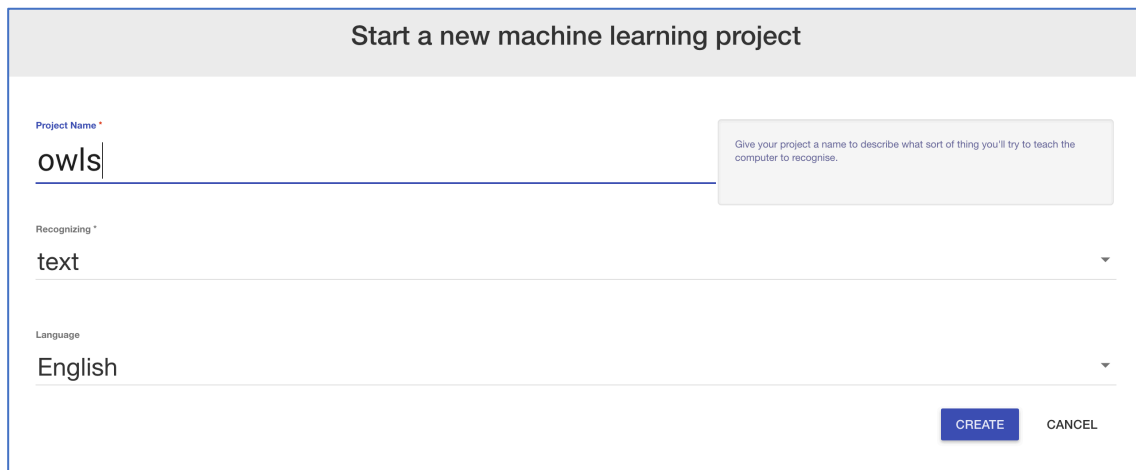
> █
```



This project worksheet is licensed under a Creative Commons Attribution Non-Commercial Share-Alike License  
<http://creativecommons.org/licenses/by-nc-sa/4.0/>

1. Decide on a **topic** for your chatbot  
Choose something that you know well enough to be able to answer questions about.  
*It could be a place (e.g. The town where you live?)*  
*It could be an animal (e.g. Tigers? Dinosaurs?)*  
*It could be an organisation (e.g. Your school)*  
*It could be something from history (e.g. Vikings? Romans?)*  
*For the rest of this worksheet, I'll be using **owls***
2. Think of **five things** someone might ask about your topic  
*e.g. for **owls**, this could be:*
  - \* *What do owls eat?*
  - \* *Where in the world do owls live?*
  - \* *How long do owls live?*
  - \* *What types of owls are there?*
  - \* *How big do owls grow?*
3. Go to <https://machinelearningforkids.co.uk/> in a web browser
4. Click on “**Get started**”
5. Click on “**Log In**” and type in your username and password  
*If you don't have a username, ask your teacher or group leader to create one for you.*  
*If you can't remember your username or password, ask your teacher or group leader to reset it for you.*
6. Click on “**Projects**” on the top menu bar
7. Click the “**+ Add a new project**” button.

8. Name your project and set it to learn how to recognise “text”.  
Click the “**Create**” button



The screenshot shows a form titled "Start a new machine learning project". It has three input fields: "Project Name" with the text "owls" entered, "Recognizing" with a dropdown menu set to "text", and "Language" with a dropdown menu set to "English". A blue "CREATE" button and a grey "CANCEL" button are at the bottom right. A tooltip for the Project Name field says: "Give your project a name to describe what sort of thing you'll try to teach the computer to recognise."

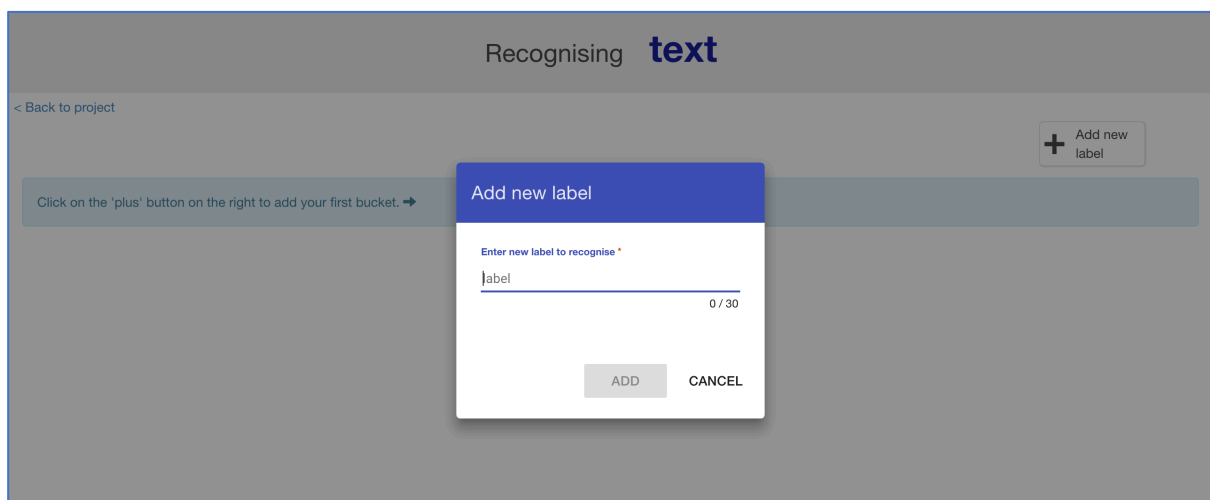
9. Click on your new project in the projects list

10. Click the **Train** button.



The screenshot shows the project dashboard for "owls". It has three main sections: "Train" (Collect examples of what you want the computer to recognise), "Learn & Test" (Use the examples to train the computer to recognise text), and "Make" (Use the machine learning model you've trained to make a game or app, in Scratch or in Python). Each section has a blue button: "Train", "Learn & Test", and "Make". A large blue arrow points from the top right towards the "Train" button.

11. Click the “+ Add new label” button



The screenshot shows the "Add new label" dialog box. It has a title bar "Add new label" and a text input field with the placeholder "label" and a character count "0 / 30". Below the input field are "ADD" and "CANCEL" buttons. The background shows the "Recognising text" interface with a "+ Add new label" button in the top right corner.

**12.** Type in **one word** that sums up the first of your things from Step 2, then click **Add**.

*I used “food” to sum up questions like “What do owls eat?”*

The screenshot shows the 'ml-for-kids' website interface. At the top is a navigation bar with links: ml-for-kids, Welcome, About, Projects, Worksheets, News, Help, and Log Out. Below this is a header area with the text 'Recognising **text** as **food**'. A link '< Back to project' is on the left. On the right is a button '+ Add new label'. The main area contains a large rectangular box labeled 'food' at the top. Inside this box, at the bottom, is a button '+ Add example'.

**13.** Do that again for all of the things in your list from Step 2  
*The words you choose don't really matter, as long as **you** understand what they mean.*

The screenshot shows the 'ml-for-kids' website interface with a project titled 'Recognising **text** as **food, countries or 3 other classes**'. The navigation bar is the same. Below the header, there is a link '< Back to project' and a button '+ Add new label'. The main area contains five rectangular boxes arranged in two rows. The top row has three boxes labeled 'food', 'countries', and 'lifespan'. The bottom row has two boxes labeled 'species' and 'size'. Each box has a '+ Add example' button at the bottom.

**14.** Click the “+ Add example” button in one of the buckets

15. Type in an example of how someone might ask that question

The screenshot shows a modal dialog box titled "Add new example" with a blue header. Inside, there is a text input field with the placeholder "Enter an example of 'food'". Below the input field, the text "What sort of foods do owls eat?" is entered. To the right of the input field, the character count "31 / 1000" is displayed. At the bottom of the dialog, there are two buttons: "ADD" (in blue) and "CANCEL". The background shows a grid of categories: "food", "lifespan", "species", and "size", each with an "Add example" button. There is also an "Add new label" button in the top right corner.

16. Click "Add"

17. Repeat until you've got **five examples** of how to ask that question.

The screenshot shows the project interface for "Recognising text as food, countries or 3 other classes". It features five vertical buckets labeled "food", "countries", "lifespan", "species", and "size". The "food" bucket contains five example questions: "What sort of foods do owls eat?", "What do owls eat?", "what do owls like to eat?", "what foods do owls like?", "what do they eat?", and "What can you eat?". Each bucket has an "Add example" button at the bottom. There is also an "Add new label" button in the top right corner.

18. Repeat until you've got at least five examples in every bucket

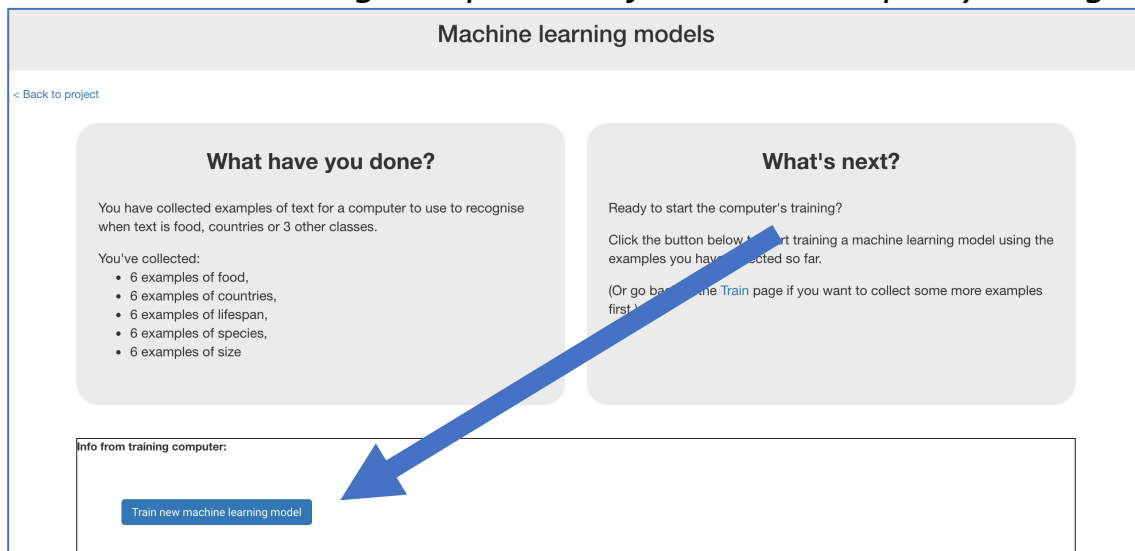
The screenshot shows the project interface for "Recognising text as food, countries or 3 other classes". It features five buckets labeled "food", "countries", "lifespan", "species", and "size". Each bucket contains multiple example questions and an "Add example" button at the bottom. The "food" bucket has 5 examples, "countries" has 3, "lifespan" has 3, "species" has 3, and "size" has 4. There is also an "Add new label" button in the top right corner.

**19.** Click on the “< Back to project” link

**20.** Click the “Learn & Test” button

**21.** Click the “Train new machine learning model” button

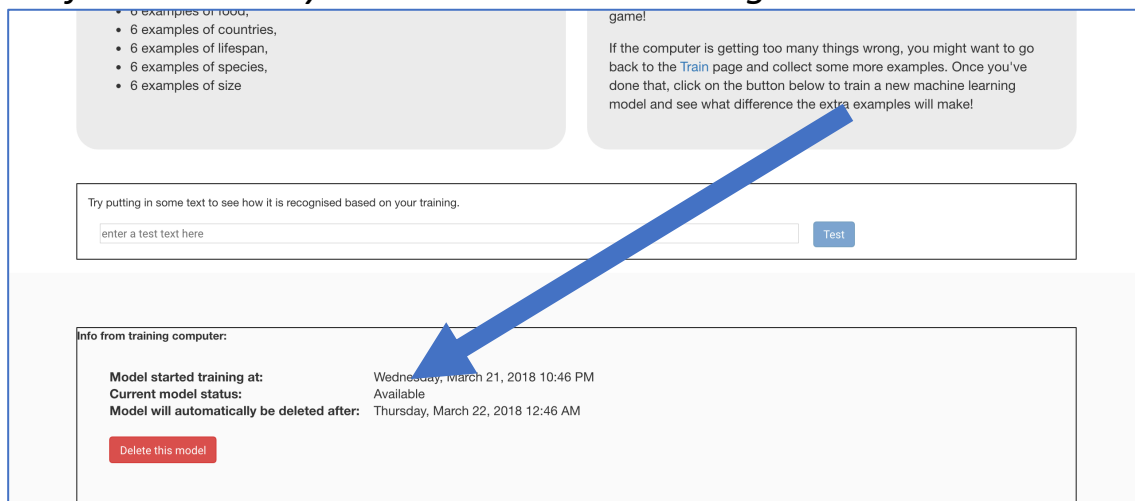
*As long as you’ve collected enough examples, the computer should start to learn how to recognise questions from the examples you’ve given to it.*



**22.** Wait for the training to complete.

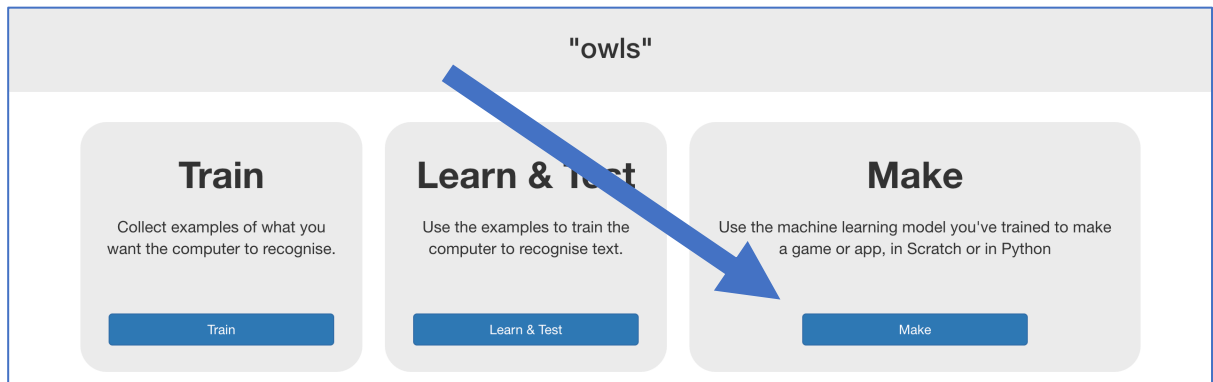
*This might take a couple of minutes.*

*It's finished once you see the “status” change to “Available”*

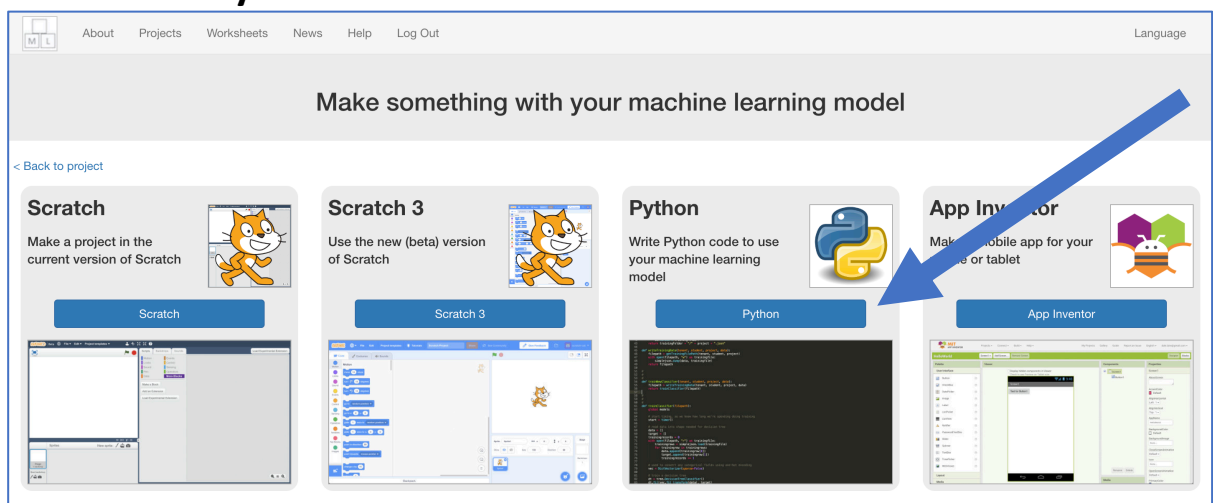


**23.** Click the “< Back to project” link

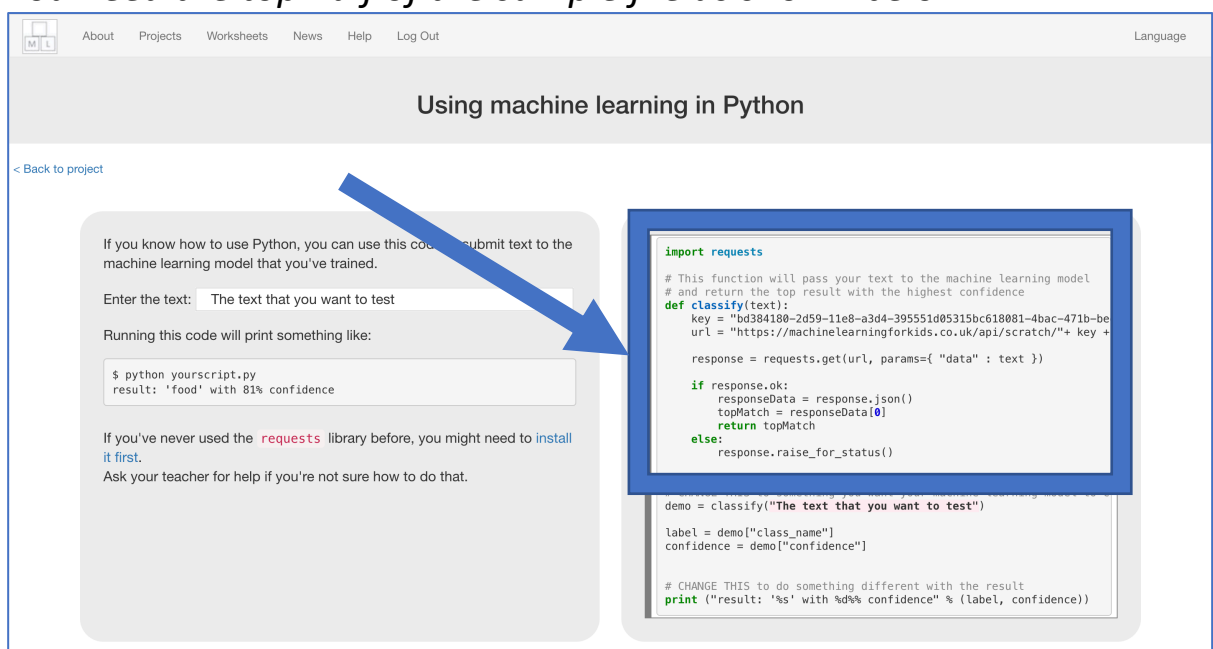
## 24. Click the “Make” button



## 25. Click “Python”



## 26. Copy the **import** line and the “**classify**” function *You need the top half of the sample file as shown below*



## 27. Paste that function into a new text file called **chatbot.py**

```
chatbot.py
1 import requests
2
3 # This function will pass your text to the machine learning model
4 # and return the top result with the highest confidence
5 def classify(text):
6     key = "bd384180-2d59-11e8-a3d4-395551d05315bc618081-4bac-471b-beca-16cbafc7372e"
7     url = "https://machinelearningforkids.co.uk/api/scratch/"+ key + "/classify"
8
9     response = requests.get(url, params={ "data" : text })
10
11     if response.ok:
12         responseData = response.json()
13         topMatch = responseData[0]
14         return topMatch
15     else:
16         response.raise_for_status()
17
```

## 28. Add a new function at the bottom, called **“answer\_question”** If you’re using Python 3, you can replace **“raw\_input”** with **“input”** *You need an if check for each of your answer types. Replace the “food”, “countries”, “lifespan”, etc. with the answer types that you used.*

```
16         response.raise_for_status()
17
18
19 def answer_question():
20     question = raw_input("> ")
21     answer = classify(question)
22     answerclass = answer["class_name"]
23     if answerclass == "food":
24         print ""
25     elif answerclass == "countries":
26         print ""
27     elif answerclass == "lifespan":
28         print ""
29     elif answerclass == "species":
30         print ""
31     elif answerclass == "size":
32         print ""
33
34
```

## 29. Put your answers to the questions into each of the **“print”** lines

```
18
19 def answer_question():
20     question = raw_input("> ")
21     answer = classify(question)
22     answerclass = answer["class_name"]
23     if answerclass == "food":
24         print "It depends on the species of owl. Small owls eat invertebrates (such as spiders,
25     elif answerclass == "countries":
26         print "Some owls live in deserts, some owls live in forests, some owls live in Arctic t
27     elif answerclass == "lifespan":
28         print "Different species of owls live for different lengths of time. The European Eagle
29     elif answerclass == "species":
30         print "There are over 200 species of owl. Some common ones include Barn Owls, Eagle Owl
31     elif answerclass == "size":
32         print "Different owl species can grow to different sizes. The Great Grey Owl can grow t
33
```



### 30. Add the following to the bottom of your script

*This will welcome you, and allow you to keep asking questions forever*

```
33
34
35 print "What would you like to know about owls?"
36
37 while True:
38     answer_question()
39
40
```

### 31. Install “requests”

*Line 1 in your Python script imports the “requests” library. If you’ve never used it before, you’ll probably need to install it now.*

*There are different ways to do this. One way is to type:*

**pip install requests**

*Ask your teacher for advice on how to do this if you’re not sure.*

```
(env) Dales-MBP:python dalelane$ pip install requests
Collecting requests
  Downloading https://files.pythonhosted.org/packages/ff/17/5cbb026005115301a8fb2f9b0e3e8d32313142fe8b617070e7baad20554f/requests-2.20.1-py2.py3-none-any.whl (57kB)
    100% |#####| 61kB 1.7MB/s
Collecting idna<2.8,>=2.5 (from requests)
  Using cached https://files.pythonhosted.org/packages/4b/2a/0276479a4b3caeb8a8c1af2f8e4355746a97fab05a372e4a2c6a6b876165/idna-2.7-py2.py3-none-any.whl
Collecting urllib3<1.25,>=1.21.1 (from requests)
  Downloading https://files.pythonhosted.org/packages/62/00/ee1d7de624db8ba7090d1226aebefab96a2c71cd5cfa7629d6ad3f61b79e/urllib3-1.24.1-py2.py3-none-any.whl (118kB)
    100% |#####| 122kB 3.5MB/s
Collecting certifi>=2017.4.17 (from requests)
  Using cached https://files.pythonhosted.org/packages/56/9d/1d02dd80bc4cd955f98980f28c5ee2200e1209292d5f9e9cc8d030d18655/certifi-2018.10.15-py2.py3-none-any.whl
Collecting chardet<3.1.0,>=3.0.2 (from requests)
  Using cached https://files.pythonhosted.org/packages/bc/a9/01ffebfb562e4274b6487b4bb1ddec7ca55ec7510b22e4c51f14098443b8/chardet-3.0.4-py2.py3-none-any.whl
Installing collected packages: idna, urllib3, certifi, chardet, requests
Successfully installed certifi-2018.10.15 chardet-3.0.4 idna-2.7 requests-2.20.1 urllib3-1.24.1
(env) Dales-MBP:python dalelane$
```

### 32. Test your chatbot!

*Run your Python script and try asking a question*

```
(env) Dales-MBP:python dalelane$ python chatbot.py
What would you like to know about owls?
> what sort of things do owls eat?
It depends on the species of owl. Small owls eat invertebrates (such as
spiders, insects and worms). Larger owls eat animals like fish, birds,
mice, shrews and voles.
> █
```

## What have you done so far?

You've started to train a computer to recognise questions on a topic. You did this by collecting examples. These examples were used to train a machine learning "model".

This is called "supervised learning" because of the way you are supervising the computer's training.

The computer will learn from patterns in the examples you've given it, such as the choice of words, and the way questions are structured. These will be used to be able to recognise new questions.

The biggest problem with this is that if you ask it something unexpected, it will still give you one of the answers you've written

### 33. Change the **answer\_question** function to look like this

The bit that you need to change are shown below in lines 23 – 27

This will help when someone asks a question that wasn't on your list.

*The confidence score is a percentage (from 0 to 100).*

*It will be lower if someone asks a question that isn't similar to any of the examples you used to train the machine learning model.*

*Use this to return a "I don't understand" message if the score is too low.*

*Experiment to work out the right confidence number to use.*

```
18
19 def answer_question():
20     question = raw_input("> ")
21     answer = classify(question)
22     answerclass = answer["class_name"]
23     confidence = answer["confidence"]
24
25     if confidence < 75:
26         print "I don't understand. Ask me something else!"
27     elif answerclass == "food":
28         print "It depends on the species of owl. Small owls eat invertebrates (such
29     elif answerclass == "countries":
30         print "Some owls live in deserts, some owls live in forests, some owls live
31     elif answerclass == "lifespan":
32         print "Different species of owls live for different lengths of time. The Eur
33     elif answerclass == "species":
34         print "There are over 200 species of owl. Some common ones include Barn Owls
35     elif answerclass == "size":
36         print "Different owl species can grow to different sizes. The Great Grey Owl
37
38
```

## Ideas and Extensions

Now that you've finished, why not give one of these ideas a try?

Or come up with one of your own?

### Try other chatbots

<http://talktothetrex.com> is a good example of the sort of thing you've made. Give it a try and see if you can get any ideas of how to improve your bot.

### Add more topics

Can you add more topics to your chatbot, so that there are more types of question that it can answer?

### Provide alternate answers

If someone asks the same question more than once, they'll get the exact same answer every time.

Can you update your Scratch script so that it varies the answers each time a little? Or just starts the answer with "You've asked me this before, but"

### Say goodbye

Add a "goodbye" bucket to your training examples and fill it with examples of ways to say goodbye. Can you update your Python script so it exits when it recognises someone saying that they've had enough?