



Smart Classroom

In this project you will make a virtual classroom that can react to what you say to it.

You'll be able to control the virtual devices in the classroom by saying what you want.

To start with, you'll program a list of rules for understanding commands, and learn why that approach isn't very good.

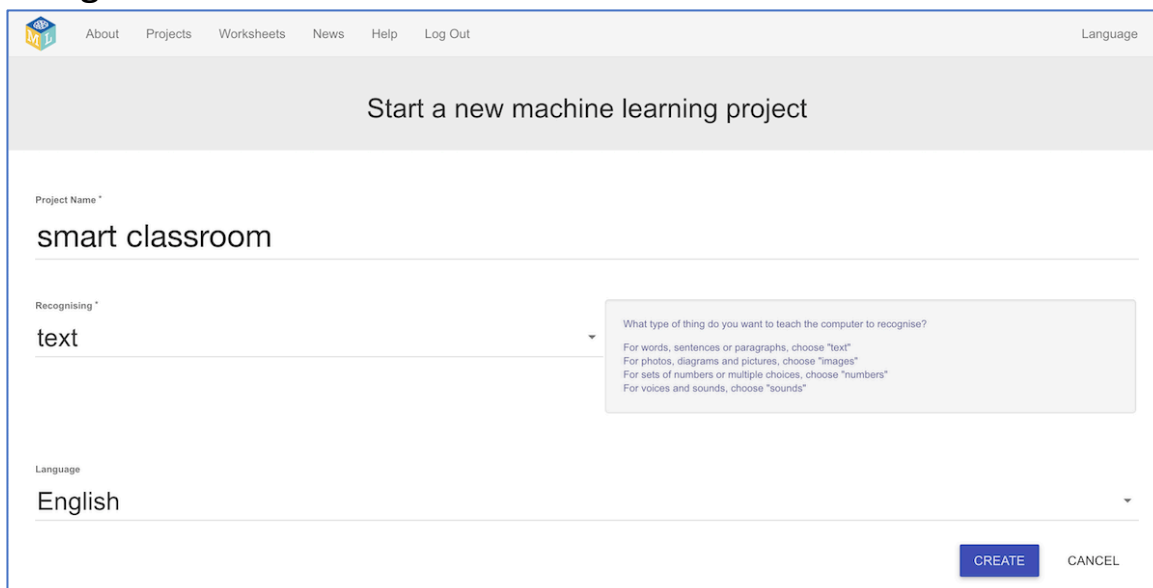
Next, you will teach the computer to recognise commands for different devices by giving it examples of each.

The screenshot displays the Scratch IDE interface. The code editor shows a script for a 'Smart Classroom' project. The script begins with a 'when clicked' event, followed by a 'forever' loop. Inside the loop, the program asks the user for a command and waits. It then uses a 'recognise text' block to check if the confidence of the answer is greater than 70. If not, it says 'Sorry, I'm not sure what you mean' for 2 seconds. If the confidence is high, it checks for specific commands: 'fan_on', 'fan_off', 'lamp_on', and 'lamp_off'. Each command triggers a 'broadcast' message to turn the corresponding device on or off. The stage area shows a virtual classroom with a fan and a lamp, and a text input field for the user's command.



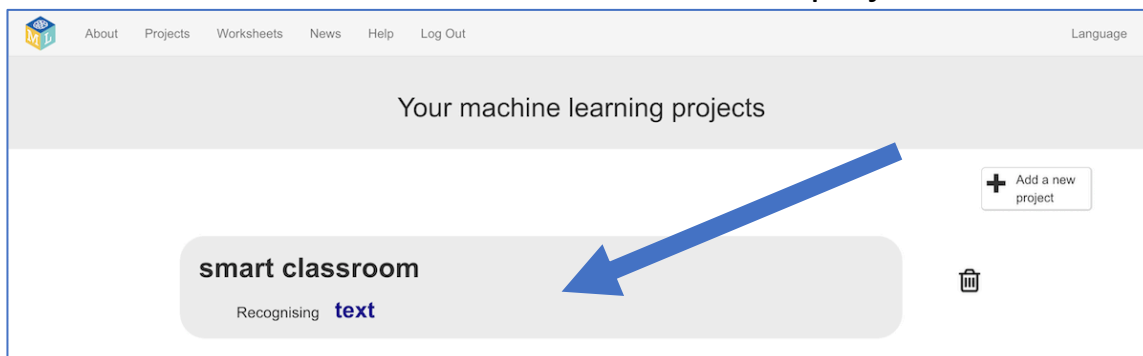
This project worksheet is licensed under a Creative Commons Attribution Non-Commercial Share-Alike License <http://creativecommons.org/licenses/by-nc-sa/4.0/>

1. Go to <https://machinelearningforkids.co.uk/> in a web browser
2. Click on “**Get started**”
3. Click on “**Log In**” and type in your username and password
If you can't remember your username or password, ask your teacher or group leader to reset it for you.
4. Click on “**Projects**” on the top menu bar
5. Click the “**+ Add a new project**” button.
6. Name your project “**smart classroom**” and set it to learn how to recognise “**text**”. Click **Create**

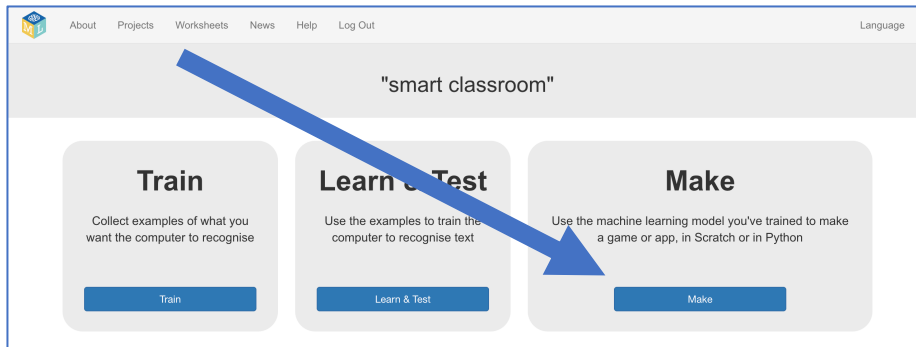


The screenshot shows a web form titled "Start a new machine learning project". At the top, there is a navigation bar with links for "About", "Projects", "Worksheets", "News", "Help", and "Log Out", and a "Language" dropdown. The form fields are: "Project Name" with the text "smart classroom"; "Recognising" with a dropdown menu set to "text", which has a tooltip that reads: "What type of thing do you want to teach the computer to recognise? For words, sentences or paragraphs, choose 'text'. For photos, diagrams and pictures, choose 'images'. For sets of numbers or multiple choices, choose 'numbers'. For voices and sounds, choose 'sounds'"; and "Language" with a dropdown menu set to "English". At the bottom right, there are two buttons: "CREATE" and "CANCEL".

7. You should see “**smart classroom**” in the projects list. Click on it.



8. We'll start by getting a project ready in Scratch. Click **"Make"**

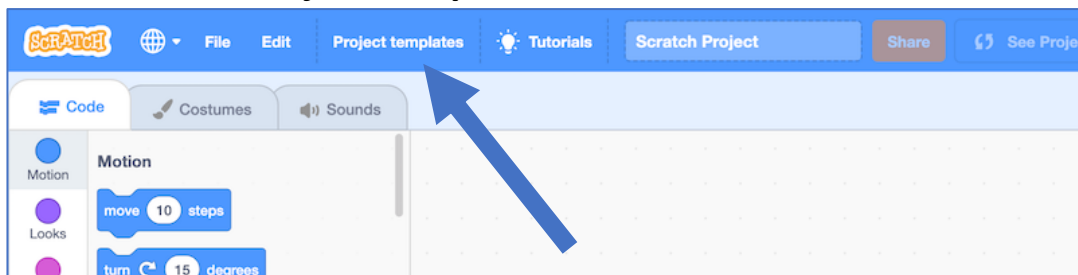


9. Click **"Scratch 3"**

10. Click **"Scratch by itself"**

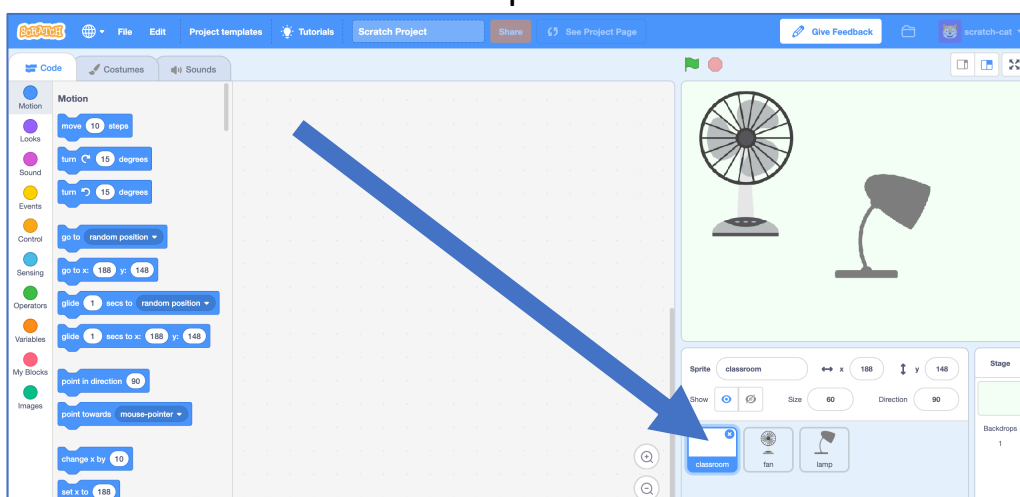
*The page will warn you that you haven't done any machine learning yet, but clicking on **Scratch by itself** will launch Scratch.*

11. Click on **Project templates**



12. Click on the **Smart Classroom** template

13. Click the **"classroom"** sprite so that it is selected as shown below.

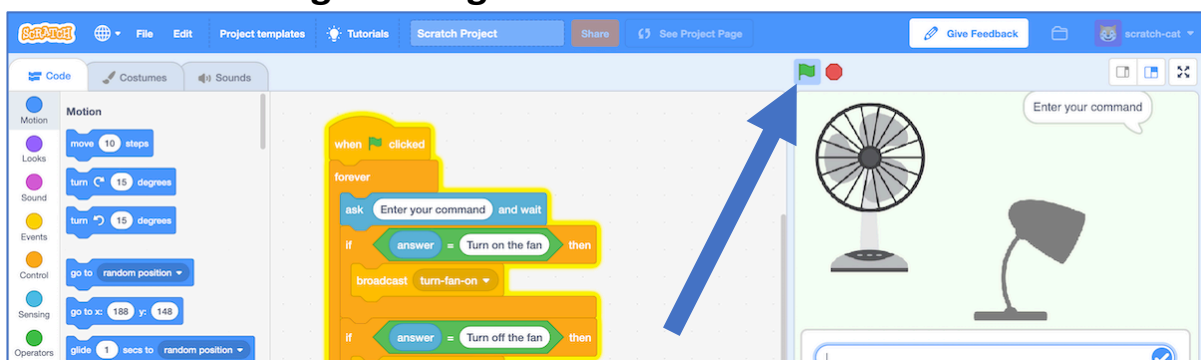


14. Click on the **Scripts** tab and enter the following script.

```
when clicked
  forever
    ask Enter your command and wait
    if answer = Turn on the fan then
      broadcast turn-fan-on
    if answer = Turn off the fan then
      broadcast turn-fan-off
    if answer = Turn on the lamp then
      broadcast turn-lamp-on
    if answer = Turn off the lamp then
      broadcast turn-lamp-off
```

15. Save your project
Click on **File** -> **Save to your computer** to save the project to a file.

16. Click on the **green flag** to test.



17. Type in a message and watch it react!
*Try “Turn on the lamp”, “Turn off the lamp”, “Turn on the fan”, and “Turn off the fan”. They should all work.
Type anything else, and nothing will happen!
Even if you just make a small spelling mistake, it won’t match.*

What have you done so far?

You've programmed the virtual classroom to react to commands using a simple rules-based approach.

If you want it to understand commands phrased differently, you would need to add extra **if** blocks.

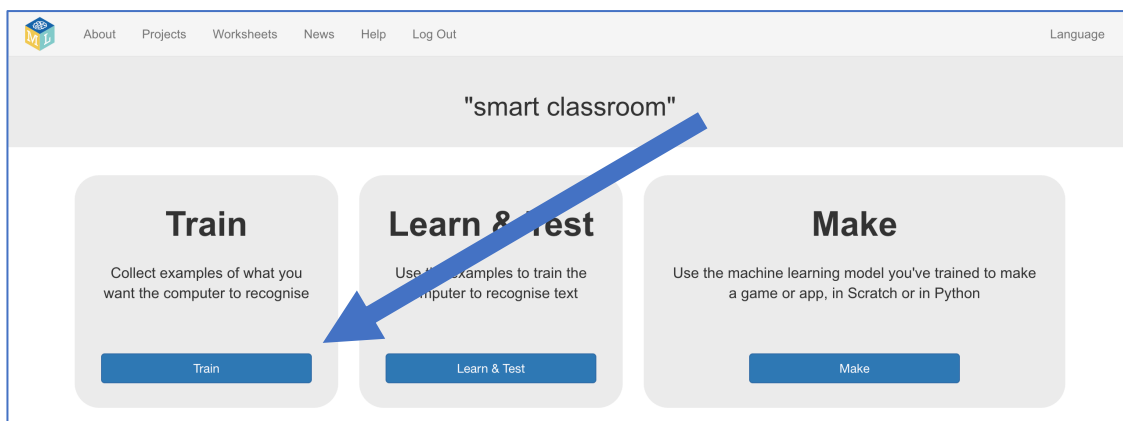
The problem is you need to predict exactly what command the smart assistant will get. Listing every possible message would take forever.

Next, we'll try a better approach: teaching the computer to recognise commands for itself.

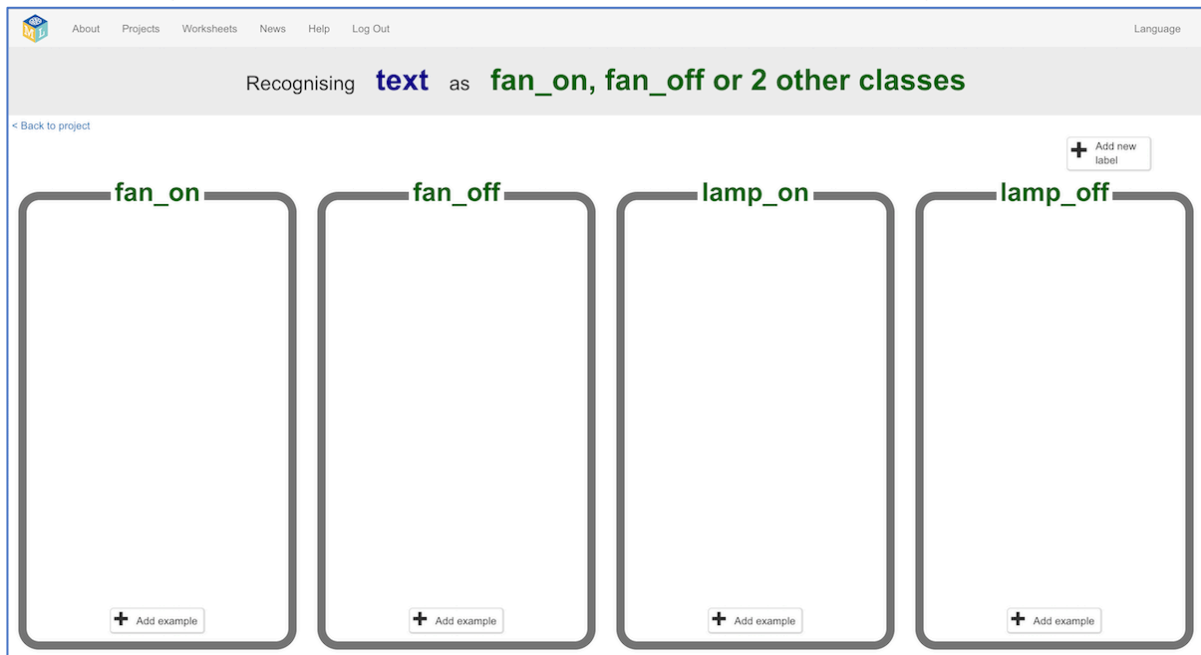
18. Close the Scratch window and go back to the Training tool.

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

20. We need to collect some examples to train the computer.
*Click the **Train** button.*



- 21.** Click on “+ Add new label” and call it “fan on”.
Do that again, and create a second bucket called “fan off”.
Do that again, and create a third bucket called “lamp on”.
Do that again, and create a fourth bucket called “lamp off”.



- 22.** Click on the “Add example” button in the “fan on” bucket, and type in a way to ask for the fan to be turned on.
For example, you could type “Please can you switch on the fan”.

- 23.** Click on the “Add example” button in the “fan off” bucket, and type in a way to ask for the fan to be switched off.
For example, you could type “I want the fan off now”

- 24.** Do the same for the “lamp on” and “lamp off” buckets.

- 25.** Repeat steps 22-24 until you’ve got at least **six** examples of each.
Be imaginative!

Try and think of lots of different ways to ask each command.

For “fan on” you could complain that you’re too hot.

For “fan off” you could complain that it’s too breezy.

For “lamp on” you could complain that it’s too dark or that you can’t see.

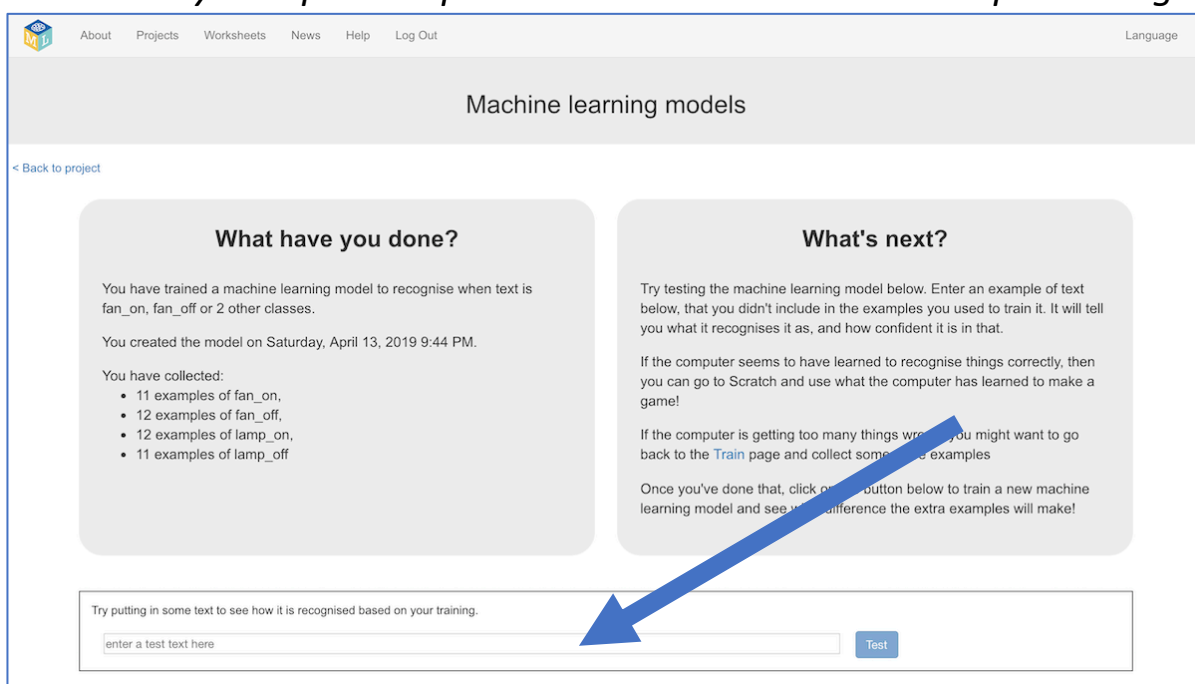
For “lamp off” you could complain that it’s too bright.

26. Click on the “< Back to project” link, then click “Learn & Test”

27. Click on the “Train new machine learning model” button.
As long as you’ve collected enough examples, the computer should start to learn how to recognise commands from the examples you’ve written.

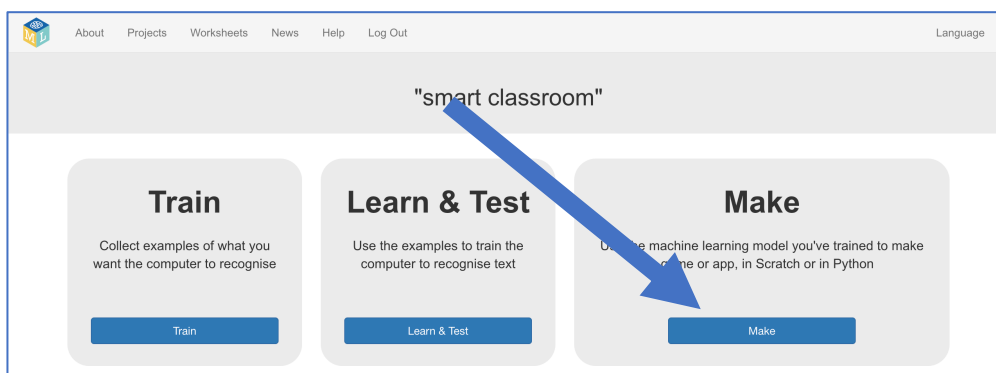
28. Wait for the training to complete. This might take a minute or two. *While waiting, try to complete the machine-learning multi-choice quiz at the bottom of the page.*

29. Once the training has completed, a Test box will be displayed. Try testing your machine learning model to see what it has learned. Type in a command, and press enter. It should be recognised. *Test it with examples that you haven't shown the computer before. If you're not happy with how the computer recognises the messages, go back to step 25, and add some more examples. Make sure you repeat step 27 to train with the new examples though!*



30. Click on the “< Back to project” link

31. Click on **Make**



What have you done so far?

You've started to train a computer to recognise commands.

Instead of writing rules to do this, you are doing it by collecting examples. These examples are being used to train a machine learning “model”.

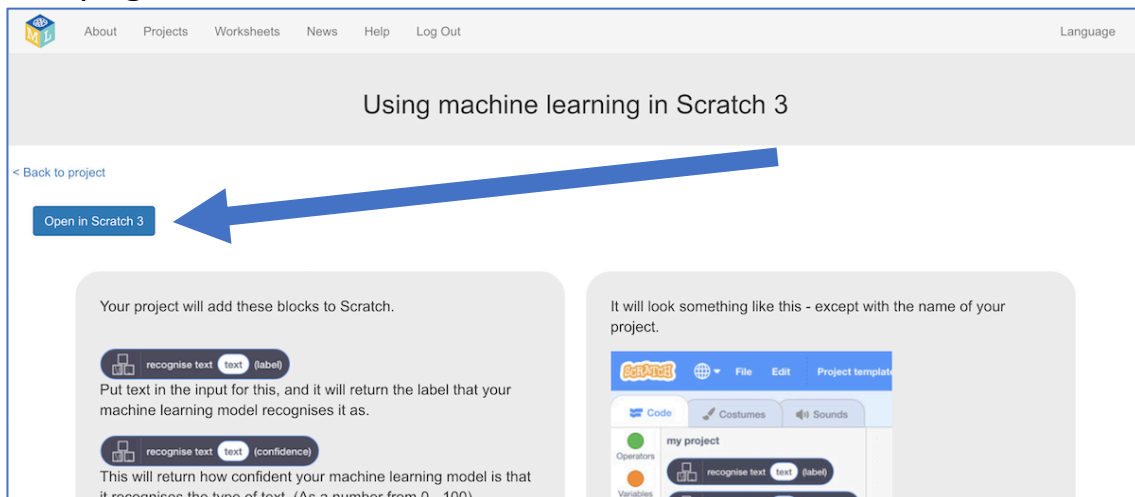
This is called “supervised learning” because of the way you are supervising the computer’s training.

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

32. Click on **Scratch 3**

33. Click on **Open in Scratch 3**

This page has instructions on how to use the new blocks in Scratch



34. Load the Scratch project you saved before.

*Click on **File** -> **Load from your computer***

Click OK when it asks to replace the current project

Tips

More examples!

The more examples you give it, the better the computer should get at recognising your instructions.

Try and be even

Try and come up with roughly the same number of examples for each command.

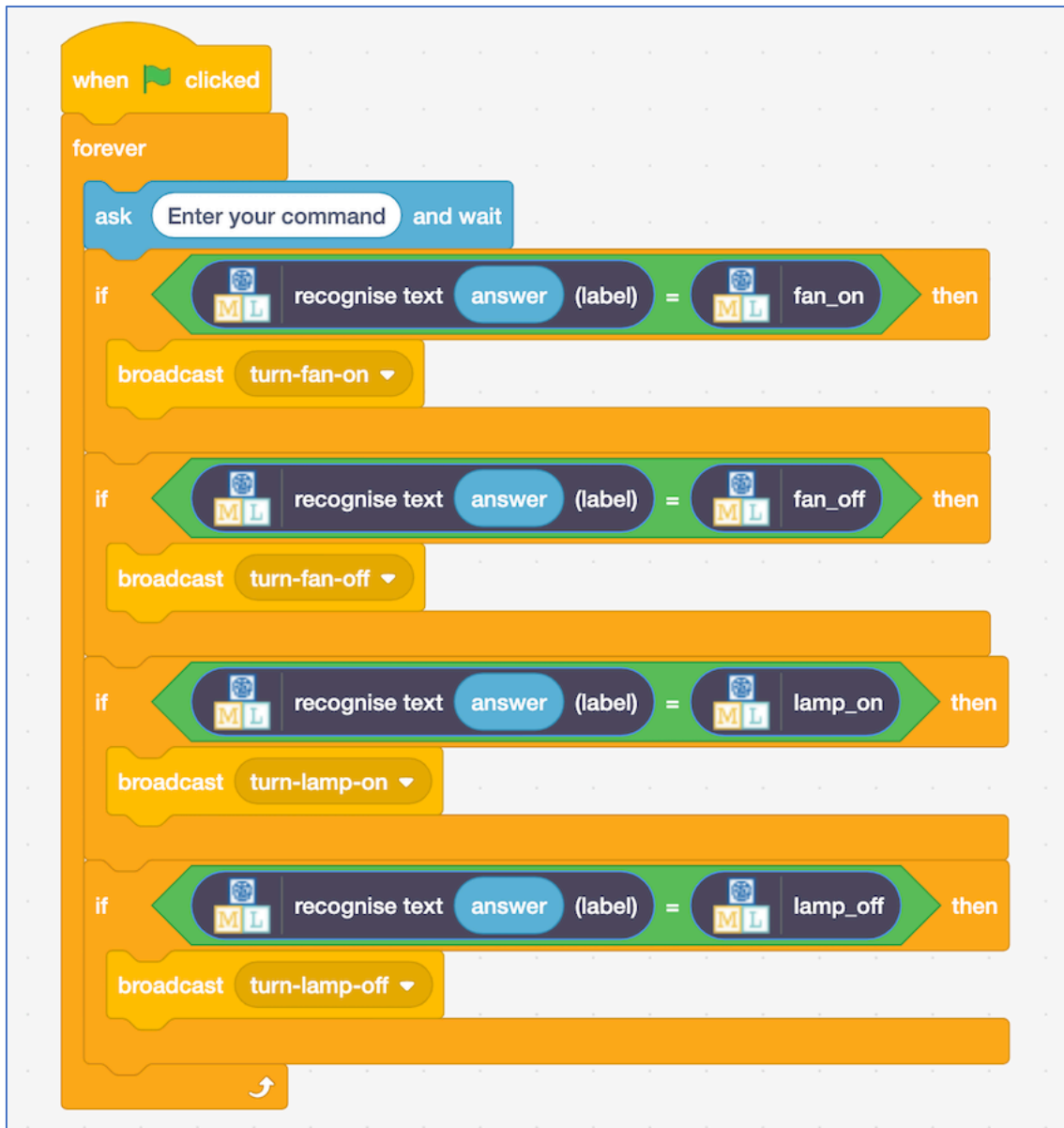
If you have a lot of examples for one command, and not the others, the computer might learn that command is more likely, so you'll affect the way that it learns to recognise messages.

Mix things up with your examples

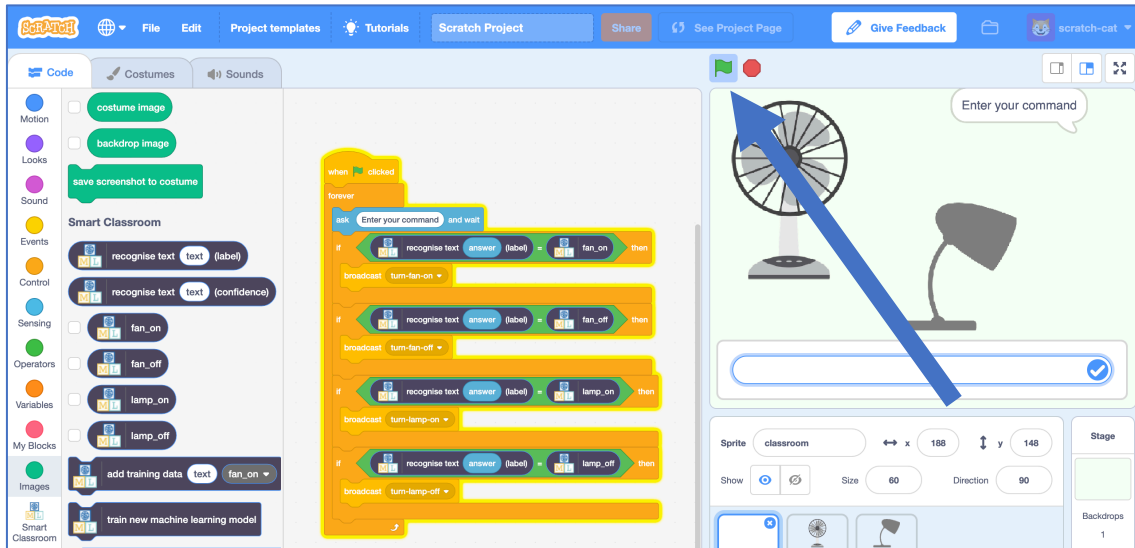
Try to come up with lots of different types of examples.

For example, make sure that you include some long examples and some very short ones.

- 35.** Click on the “**Code**” tab, and update the script to use your machine learning model **instead** of the rules you made before. *The “recognise text ... (label)” block is a new block added by your project. If you give it text, it will return the label for one of the four commands based on the training you’ve given to the computer.*



36. Click the **green flag** to test again.



37. Test your project

Type a command and press enter. The fan or lamp should react to your instructions.

*Make sure you test that this works **even for messages that you didn't include in your training.***

38. Save your project.

*Click **File** -> **Save to your computer***

What have you done so far?

You've modified your Scratch smart classroom assistant to use machine learning instead of your earlier rules-based approach.

Training the computer to be able to recognise instructions for itself should be much quicker than trying to make a list of every possible command.

The more examples you give it, the better it should get at recognising instructions correctly.

- 39.** Leave Scratch open (we'll come back in a moment) but go back to the **Learn & Test** page in the Training tool. Type something in the Test box that has nothing to do with lamps or fans. For example, "make me a cheese sandwich"

< Back to project

What have you done?

You have trained a machine learning model to recognise when text is fan_on, fan_off or 2 other classes.

You created the model on Saturday, April 13, 2019 9:44 PM.

You have collected:

- 11 examples of fan_on,
- 12 examples of fan_off,
- 12 examples of lamp_on,
- 11 examples of lamp_off

What's next?

Try testing the machine learning model below. Enter an example of text below, that you didn't include in the examples you used to train it. It will tell you what it recognises it as, and how confident it is in that.

If the computer seems to have learned to recognise things correctly, then you can go to Scratch and use what the computer has learned to make a game!

If the computer is getting too many things wrong, you might want to go back to the [Train](#) page and collect some more examples

Once you've done that, click on the button below to train a machine learning model and see what difference extra examples will make!

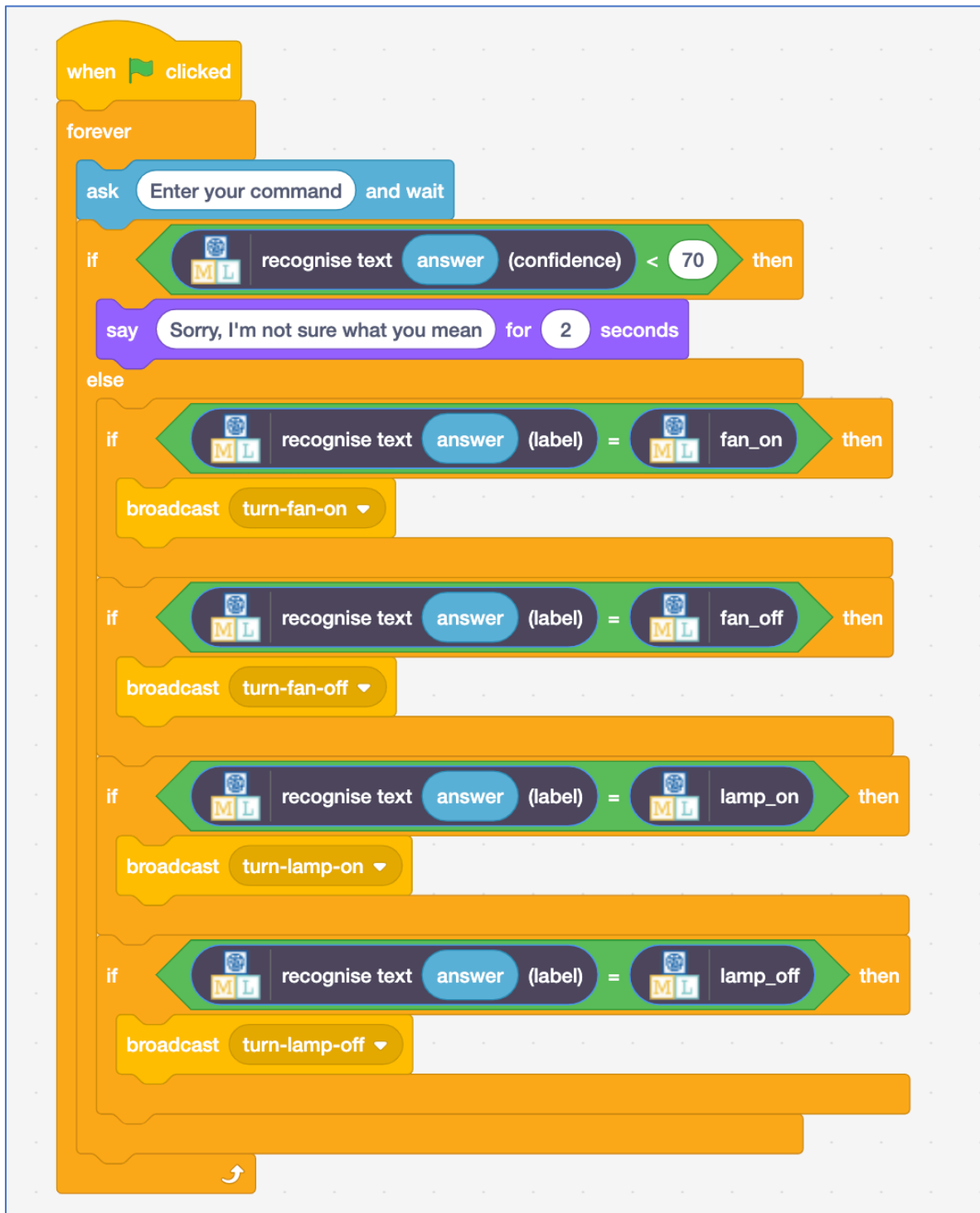
Try putting in some text to see how it is recognised based on your training

Recognised as **lamp_off**
with 21% confidence

- 40.** Look at the confidence score, and check that it's very low. Compare this with the score you get from commands like "turn on the lamp". ***This is the computer's way of telling you that it's not very certain it understands your command, because it doesn't look like what it learned from your examples.***

- 41.** Go back to Scratch.
You can open your saved project from before if you closed the window.

42. Modify the script for the “classroom” sprite so that it uses this confidence score.



43. Click the **green flag** and test again
*Try typing commands that have nothing to do with the fan or lamp.
Try asking for something to be turned on or off.
Check that your classroom reacts in the right way.*

What have you done?

You've trained a smart assistant – like a simple version of the assistants you can get on modern smartphones (like Apple's Siri or Google's Assistant) or virtual assistant devices (like Amazon's Alexa or Google's Home).

You've used it to create a smart classroom assistant in Scratch, using machine learning instead of your earlier rules-based approach.

Training the computer to be able to recognise instructions was hopefully much easier than trying to make a list of every possible command. And the more examples you give it, the better it gets at recognising instructions and the more confident it gets in doing that.

And now, if it's not sure what you mean, it will ask you to try again.

Ideas and Extensions

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

Try another device

Instead of just a fan and a lamp, can you add another device to your smart classroom?

Try different confidence limits

Is 70% the right threshold to use to decide whether the computer has recognised the command?

Experiment with different values until you have a value that works well for your machine learning model.

If you choose a number that is too high, the computer will say "Sorry I'm not sure what you mean" too often.

If you choose a number that is too low, the computer will get too many things wrong.

Do it for real!

Have a look at the smart assistants that developers have made for Amazon's Alexa : <http://amzn.to/2sxy1hw>

Developers made these in the same way that you did this project – creating labels for the commands they wanted it to recognise, and then collecting examples of how those commands might be phrased to train the Alexa to be able to understand them.

Find an Alexa Skill that you think sounds good. Look at the commands it can understand – can you think how you could've trained it?