

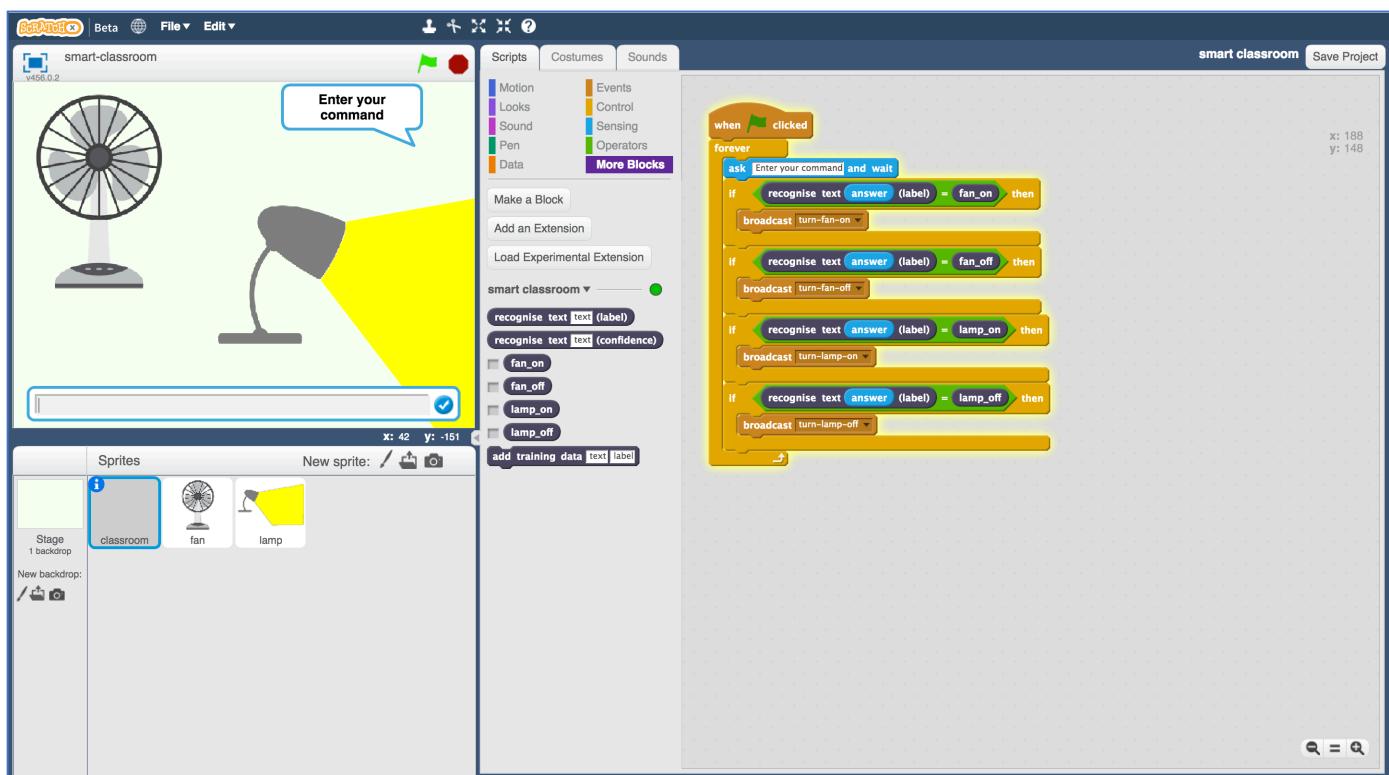
Smart Classroom

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

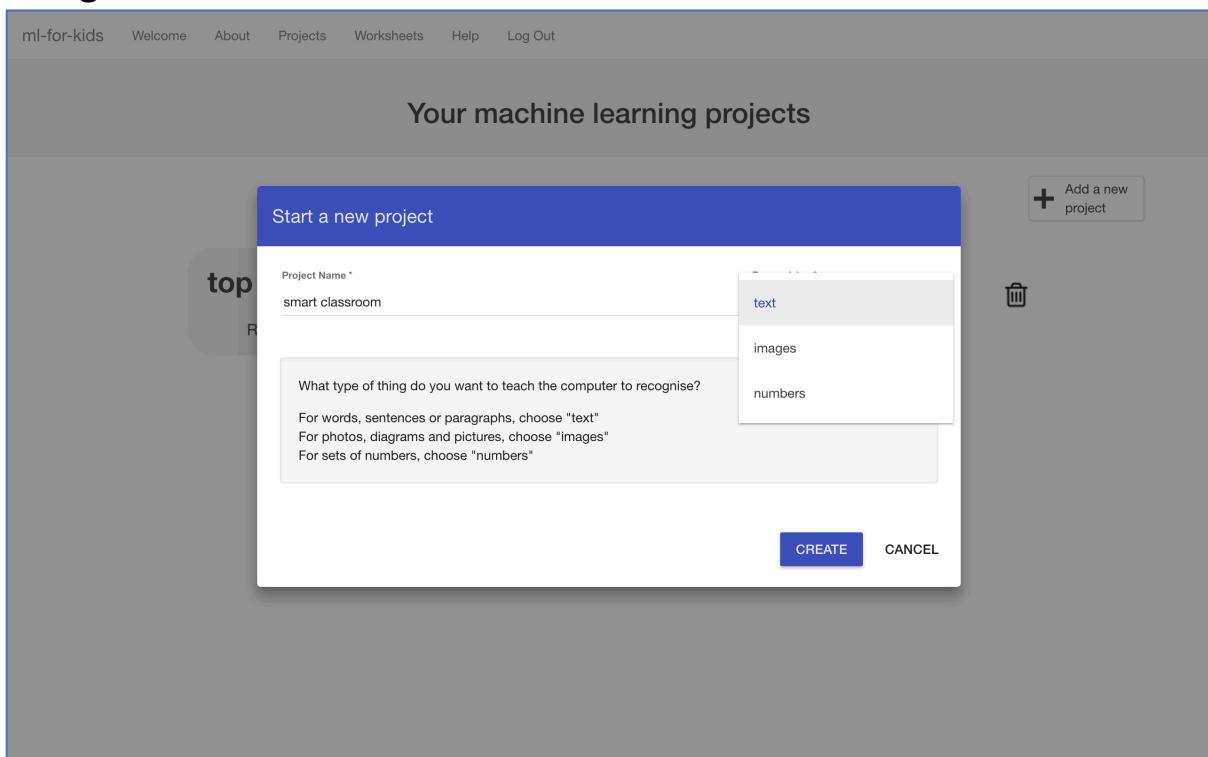
You'll be able to control the devices in the classroom by saying what you want.

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.



1. You'll need the **smart-classroom.sbx** starter file for this project.
If you haven't got this, ask your teacher or group leader.
2. Go to <https://machinelearningforkids.co.uk/> in a web browser
3. Click on “**Get started**”
4. 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.
5. Click on “**Projects**” on the top menu bar
6. Click on the “**+ Add a new project**” button.
7. Name your project “smart classroom” and set it to learn how to recognise “**text**”



- 8.** You should now see “smart classroom” show up in the list of your projects. Click on it.

The screenshot shows a web interface for managing machine learning projects. At the top, there is a navigation bar with links: ml-for-kids, Welcome, About, Projects, Worksheets, Help, and Log Out. Below the navigation bar, the title "Your machine learning projects" is displayed. There are two project cards: "top trumps" and "smart classroom". Each card has a delete icon (trash can) to its right. A button labeled "+ Add a new project" is located in the top right corner of the main content area. The "top trumps" project card contains the text "Recognising numbers as win, draw or lose". The "smart classroom" project card contains the text "Recognising text".

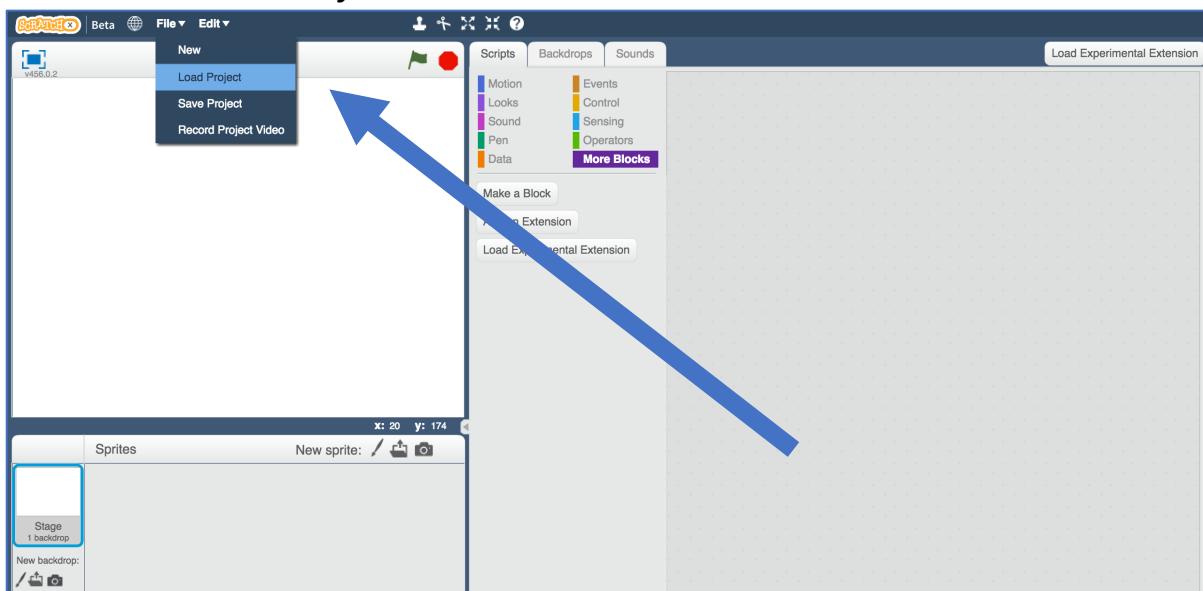
- 9.** We'll start by getting a project ready in Scratch. Click on the **Scratch** button.

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

The screenshot shows a page for the "smart classroom" project. At the top, the project name "smart classroom" is displayed. Below the project name, there are three buttons arranged horizontally: "Train", "Learn & Test", and "Scratch". Each button has a corresponding description below it. The "Train" button says "Collect examples of what you want the computer to recognise." and has a "Train" button. The "Learn & Test" button says "Use the examples to train the computer to recognise text." and has a "Learn & Test" button. The "Scratch" button says "Use the machine learning model you've trained to make a game in Scratch." and has a "Scratch" button.

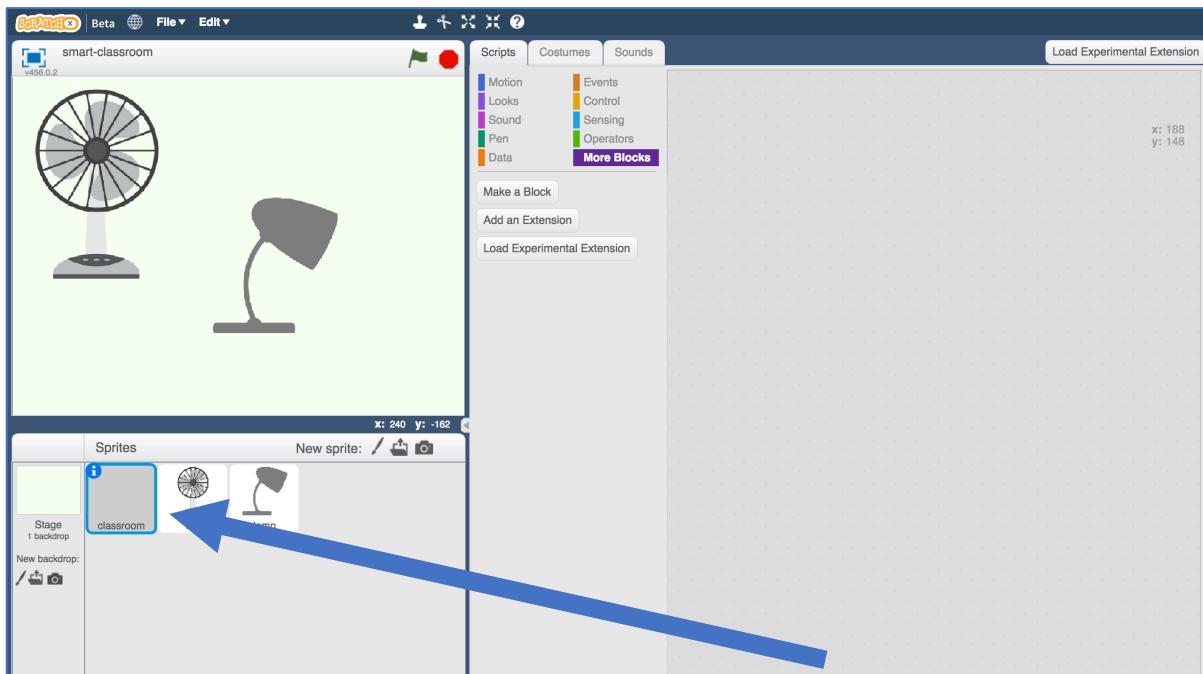
10. Load the smart-classroom.sbx file

Use File -> Load Project as shown below

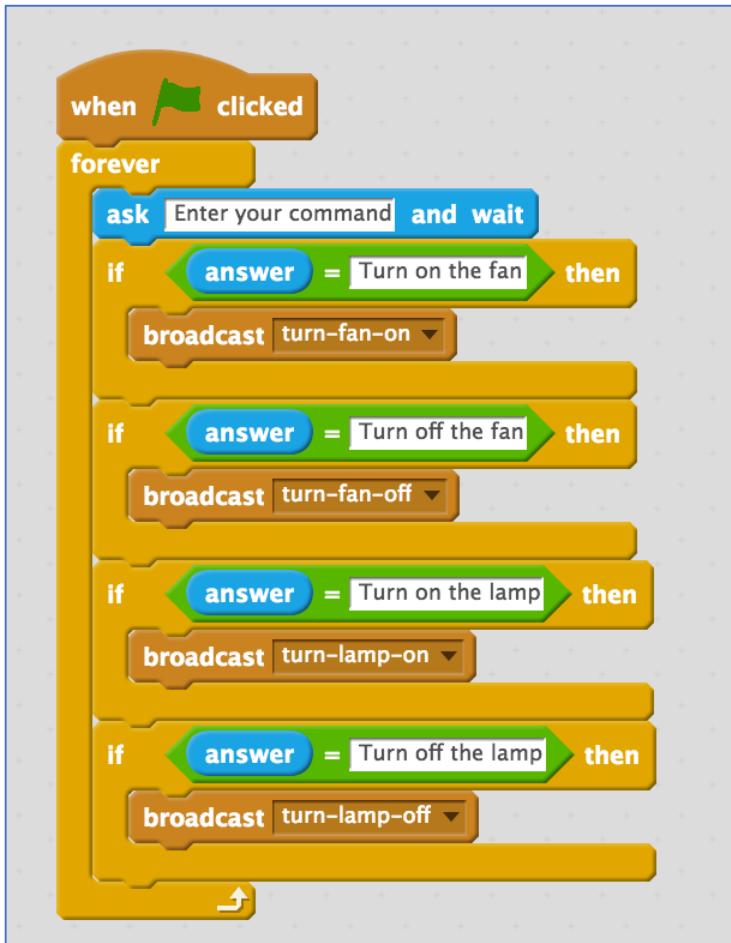


11. You should see two sprites on the stage (fan and lamp) with a third invisible sprite called “classroom”.

Click on the “classroom” sprite so that it is selected as shown below.



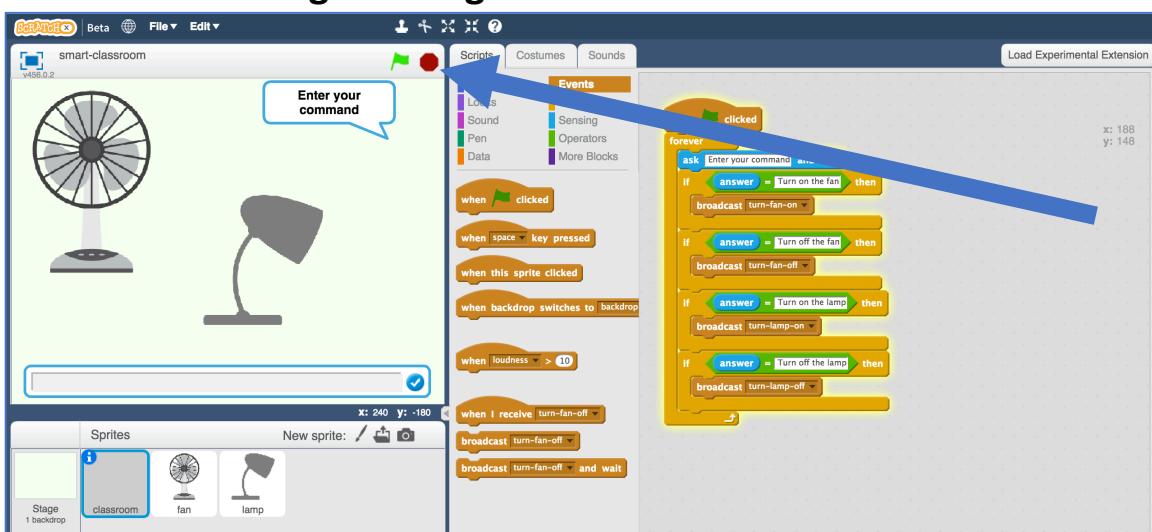
12. Click on the **Scripts** tab and enter the following script.



13. Save your project.

*Click on **File** -> **Save Project** to save the project to a file.*

14. Click on the green flag to test.



15. Type in a message and watch it react!

Type “Turn on the lamp” and press enter. The lamp should turn on.

Try “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 change a capital letter to a lower-case letter, or just make a small spelling mistake. If you don’t get it exactly right, nothing will happen.

What have we done so far?

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

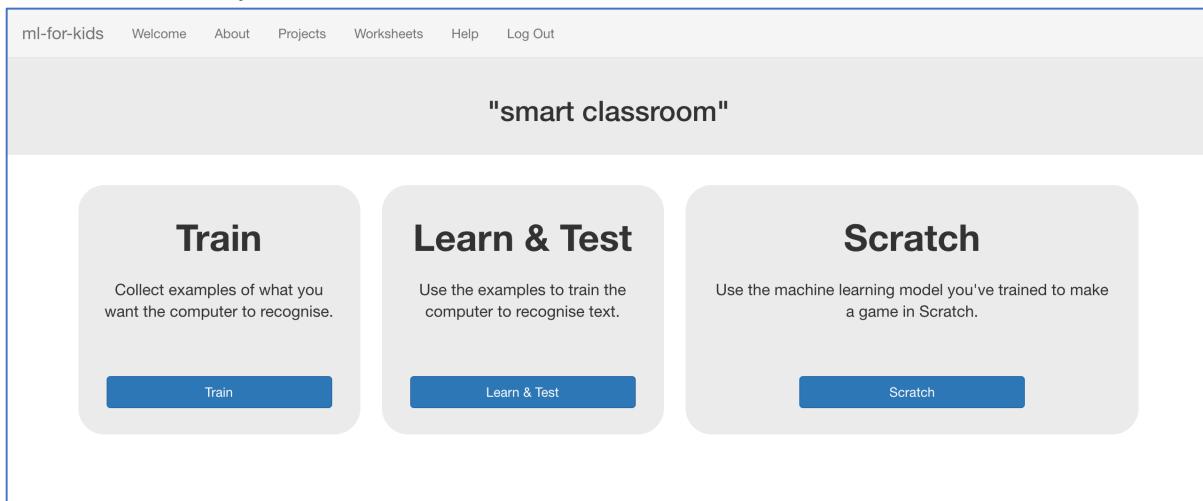
If you want it to react to commands phrased differently as well, you need to add extra **if** blocks.

The problem with this approach is that you need to predict exactly what command the classroom will receive, and making a list of every possible message would take forever.

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

16. Close the Scratch window and go back to the Training tool. Click on the < Back to project link.

17. The first thing we need to do is collect some examples we can use to train the computer. Click on the **Train** button.

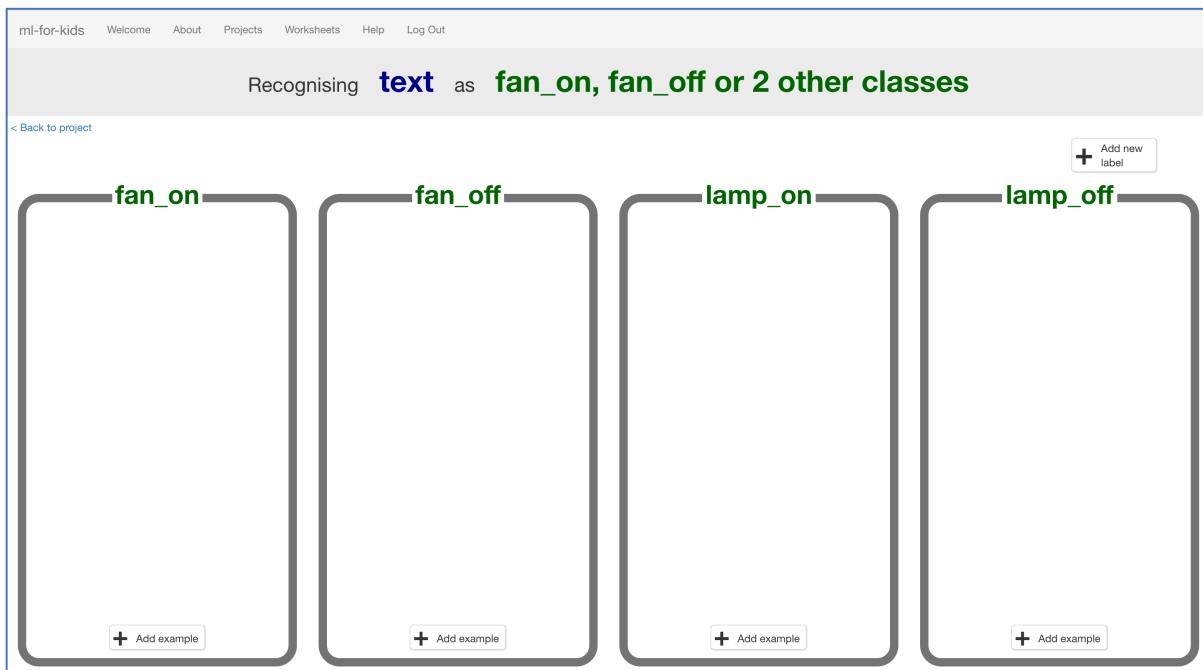


18. Click on “+ Add new label” and call it “fan on”.

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”.



19. 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”.

20. 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”

21. Do the same for the “lamp on” and “lamp off” buckets.

22. Repeat steps 19-21 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.

The screenshot shows a web interface for managing text examples. At the top, a navigation bar includes links for 'ml-for-kids', 'Welcome', 'About', 'Projects', 'Worksheets', 'Help', and 'Log Out'. Below this, a title bar reads 'Recognising **text** as **fan_on, fan_off or 2 other classes**'. Underneath, there are four labeled buckets:

- fan_on**: Examples include "Can we turn the fan on?", "Can you switch on the fan?", "fan on", "I need some air", "I want the fan on.", "I'd like the fan on, please", "I'm too hot", "It's too hot in here", "Please switch the fan on", "Please turn on the fan", and "Turn on the fan".
- fan_off**: Examples include "Can we have the fan off now", "fan off", "I don't want the fan on any more", "I'm cold", "I'm feeling too cold", "It's too breezy", "It's too windy", "It's too windy in here", "Please can you turn off the fan", "Switch off the fan", and "Turn the fan off".
- lamp_on**: Examples include "Can we have some light on?", "Can we have the lamp on?", "I can't see", "I can't see. Let's have some light.", "It's too dark", "It's too dark in here.", "It's too dark. I can't see anything.", "Lamp on", "Light on", "Please turn on the lamp", and "Turn on the lamp".
- lamp_off**: Examples include "Can you turn off the lamp?", "Can you turn the light off", "Could you turn the light off please?", "It's too bright", "Lamp off", "Lamp off please", "Please can you switch the light off", "Please make it darker", "Please turn off the lamp", and "Turn off the lamp".

A 'Add new label' button is located in the top right corner of the interface.

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

24. 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.

The screenshot shows the 'Machine learning models' page. At the top, there is a navigation bar with links: ml-for-kids, Welcome, About, Projects, Worksheets, Help, and Log Out. Below the navigation bar, the title 'Machine learning models' is centered. Underneath the title, there is a link '< Back to project'. The page is divided into two main sections: 'What have you done?' and 'What's next?'. The 'What have you done?' section contains text about collected examples and a bulleted list of 12 examples. The 'What's next?' section contains text about starting training and a link to go back to the Train page. At the bottom of the page, there is a box labeled 'Info from training server:' which contains a 'Train new machine learning model' button.

25. 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.

The screenshot shows the 'Machine learning models' page during the training process. The 'What have you done?' section indicates that training has started since Saturday, July 8, 2017 8:22 PM. It also notes that training may take longer if the server is busy. The 'What's next?' section suggests waiting for the model to finish training or taking a quiz. At the bottom of the page, there is a box labeled 'Info from training server:' which displays the start time (Saturday, July 8, 2017 8:22 PM), current status (Training), and end time (Saturday, July 8, 2017 10:22 PM). There is also a 'Cancel training' button.

26. Once the training has completed, a Test box will be displayed.

Try testing your machine learning model to see what the computer has learned.

Type in a command, and press enter. It should be properly recognised as one of the four commands.

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 19, and add some more examples.

Make sure you repeat step 24 to train with the new examples though!

The screenshot shows a web-based machine learning interface. At the top, there's a navigation bar with links: ml-for-kids, Welcome, About, Projects, Worksheets, Help, and Log Out. Below the navigation is a main title "Machine learning models". A "Back to project" link is visible. The page is divided into two main sections: "What have you done?" and "What's next?".

What have you done?

- You've trained a machine learning model to recognise when text is fan_on, fan_off or 2 other classes.
- You created the model on Saturday, July 8, 2017 8:22 PM.
- You've collected:
 - 12 examples of fan_off,
 - 11 examples of fan_on,
 - 12 examples of lamp_off,
 - 12 examples of lamp_on

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 new machine learning model and see what different the extra examples will make!

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

enter a test text here

Info from training server:

Model started training at:	Saturday, July 8, 2017 8:22 PM
Current model status:	Available
Model will automatically be deleted after:	Saturday, July 8, 2017 10:22 PM

What have we done so far?

You've started to train a computer to recognise commands to control two classroom devices. Instead of trying to write rules to be able 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 new commands.

27. Click on the "[< Back to project](#)" link, then back to the "[Scratch](#)" button.

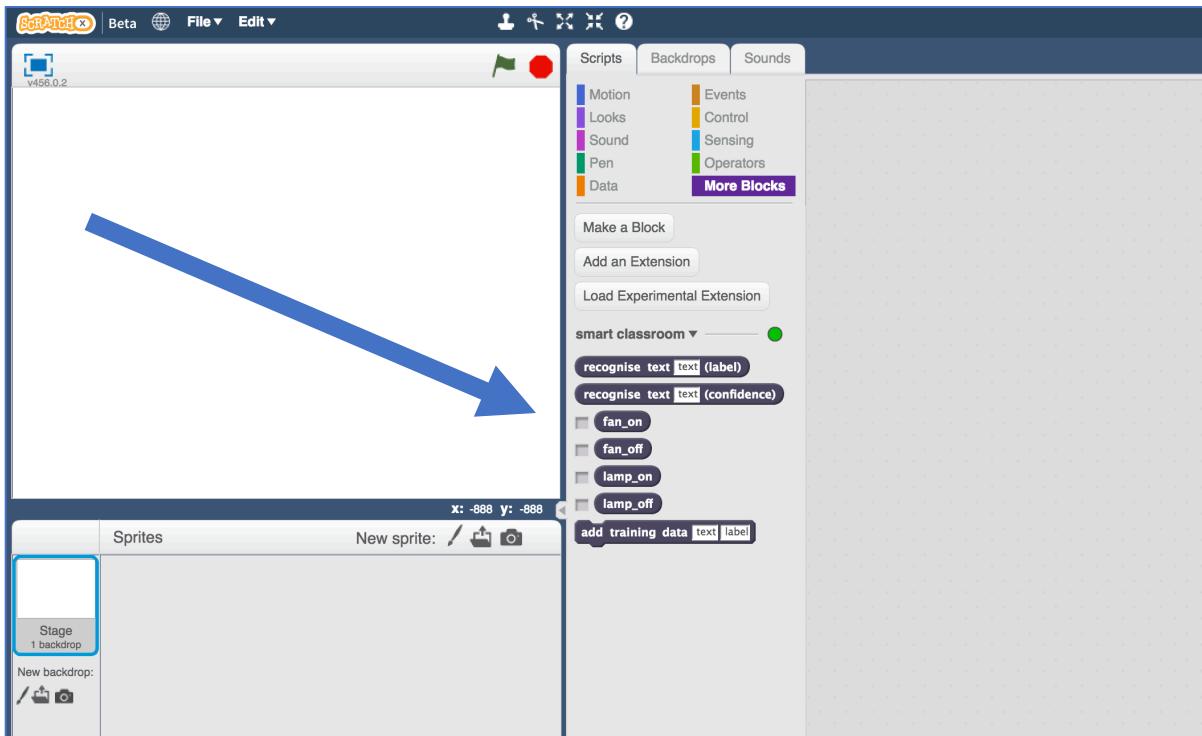
This page will be updated with instructions on how to use the new blocks in Scratch from your project. Keep this page open so can check back on how to use them.

The screenshot shows a web page titled "Using machine learning in Scratch". At the top, there is a navigation bar with links: ml-for-kids, Welcome, About, Projects, Worksheets, Help, and Log Out. Below the title, there is a link "< Back to project". The main content area is divided into two sections:

- Left Section:** This section explains that the project will add blocks to the "More Blocks" tab in Scratch scripts. It includes descriptions and screenshots of the "recognise text [text] (label)" and "recognise text [text] (confidence)" blocks. It also describes the "fan_on", "fan_off", "lamp_on", and "lamp_off" blocks as labels created in the project.
- Right Section:** This section shows a screenshot of the Scratch script editor. It displays a script with a green flag start, followed by an "if [recognise text [answer] (label) = fan_on] then" condition, and a "say [I think that was fan_on]" message block. To the right of the editor, there is a note about colored circles next to the project name indicating training status: a green circle means trained, a yellow circle means not finished, and a red circle means an error.

28. Click on the “Open in Scratch” button at the bottom of that page to launch the Scratch editor.

You should see new blocks in the “More blocks” section from your “smart classroom” project.



29. Load the Scratch project you saved before.

Click on File -> Load 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.

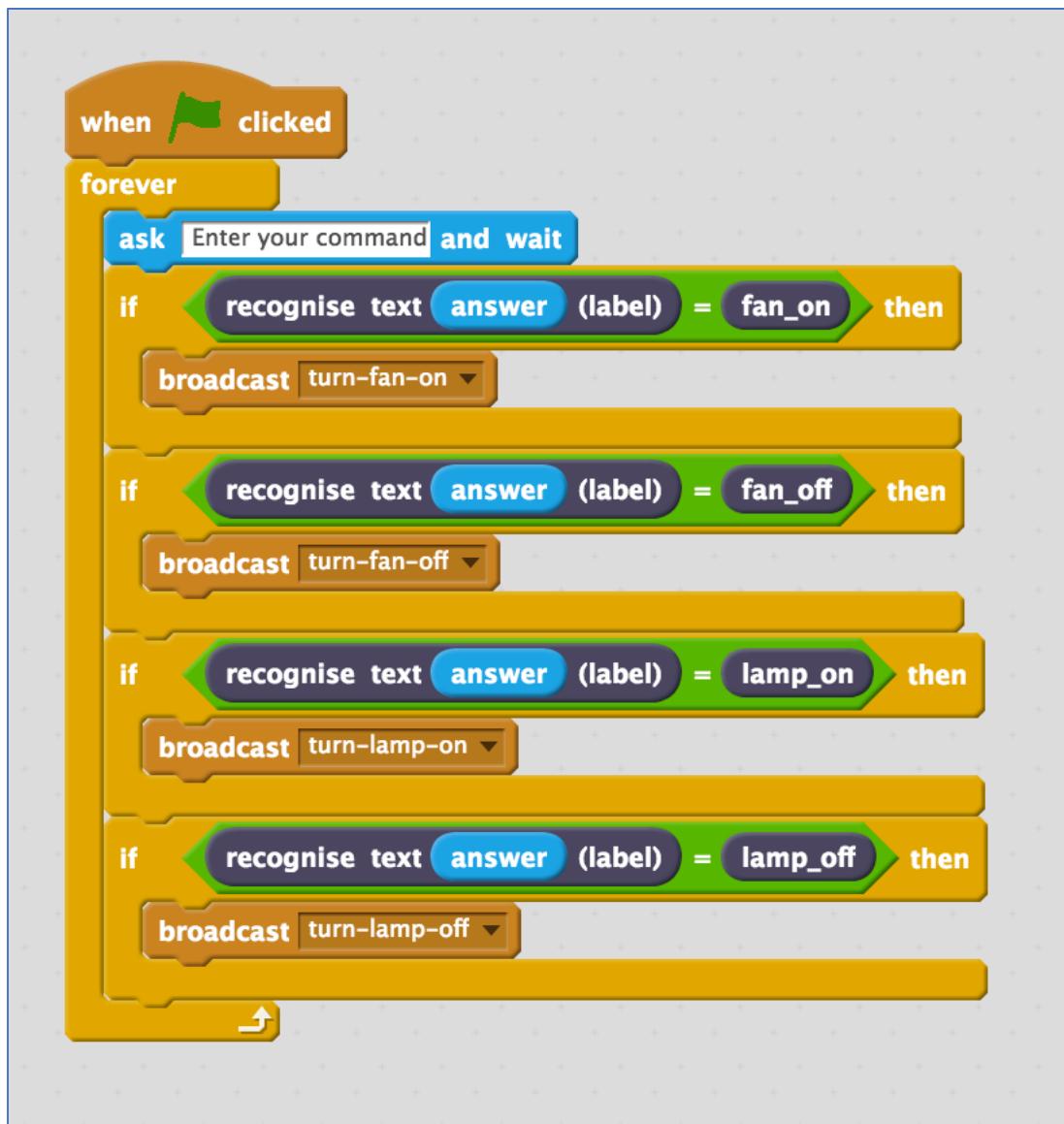
If you have a lot of examples for one command, and not the others, the computer might learn that command is more likely, so you'll affect the way that it learns to recognise messages.

Mix things up with your examples

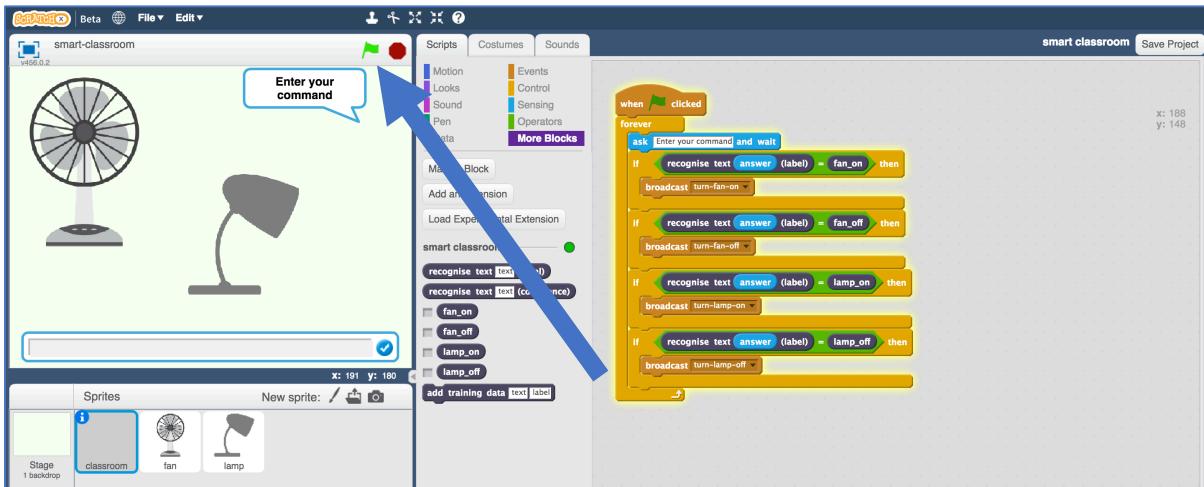
Try to come up with lots of different types of examples. For example, make sure that you include some long examples and some very short ones.

30. Click on the “**Scripts**” 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 a text message, it will return the label for one of the four commands based on the training you’ve given to the computer.



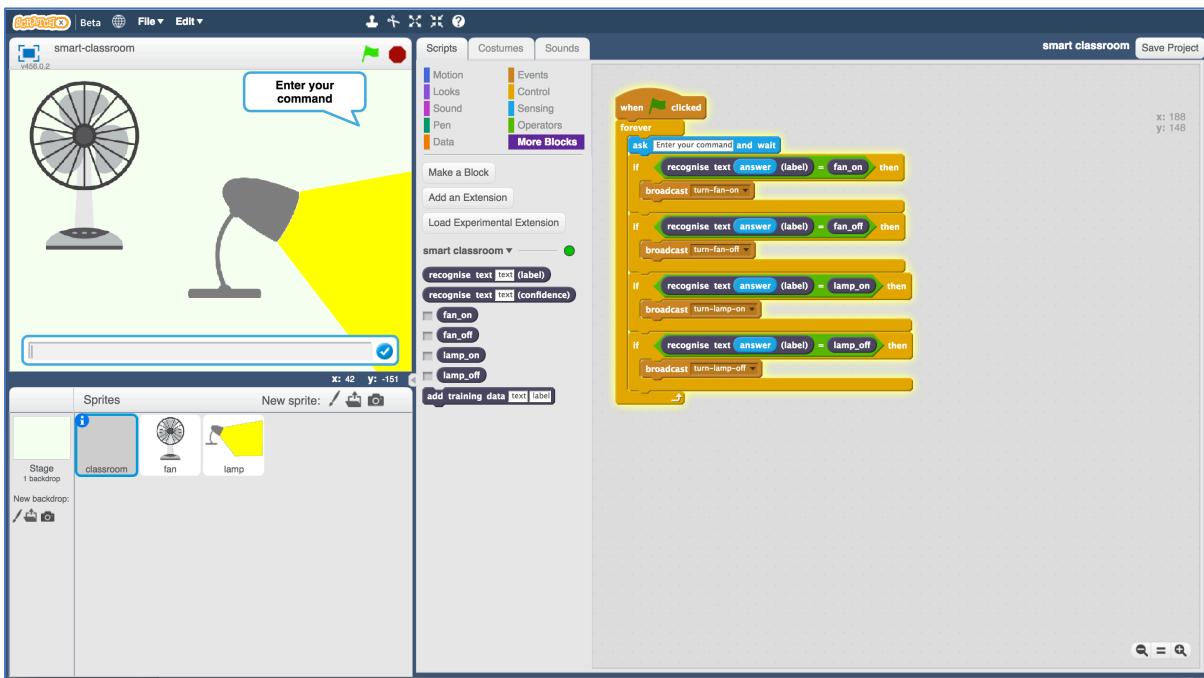
31. Click on the green flag to test again.



32. 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**.



33. Save your project.

File -> Save Project

What have we 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.

34. Leave Scratch open (we'll come back in a moment) but go back to the **Learn & Test** page.

Type a message into 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've trained a machine learning model to recognise when text is fan_on, fan_off or 2 other classes.

You created the model on Saturday, July 8, 2017 8:22 PM.

You've collected:

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

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!

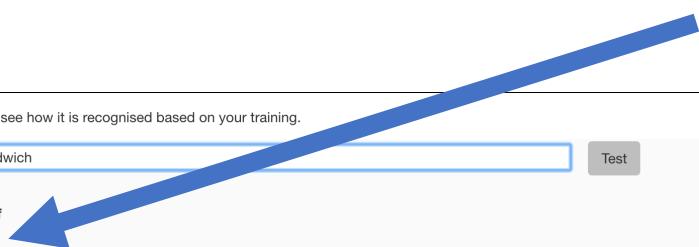
If the computer is getting too many things wrong, you might want to go back to the [Train](#) page and collect some more examples. Once you've done that, click on the button below to train a new machine learning model and see what different the extra examples will make!

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

make me a cheese sandwich

Test

Recognised as **lamp_off** with 21% confidence



35. Look at the confidence score, and check 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.

36. Go back to Scratch.

Open your saved project from before if you closed the window.

37. Modify the script for the "classroom" sprite so that it uses this confidence score.



38. Save your project *You've finished!*

What have we done?

You've trained a smart assistant – like a simple version of the assistants you can get on modern smartphones.

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 in the meantime, while your system is still learning, 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?

Or come up with one of your own?

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.