



Make me happy

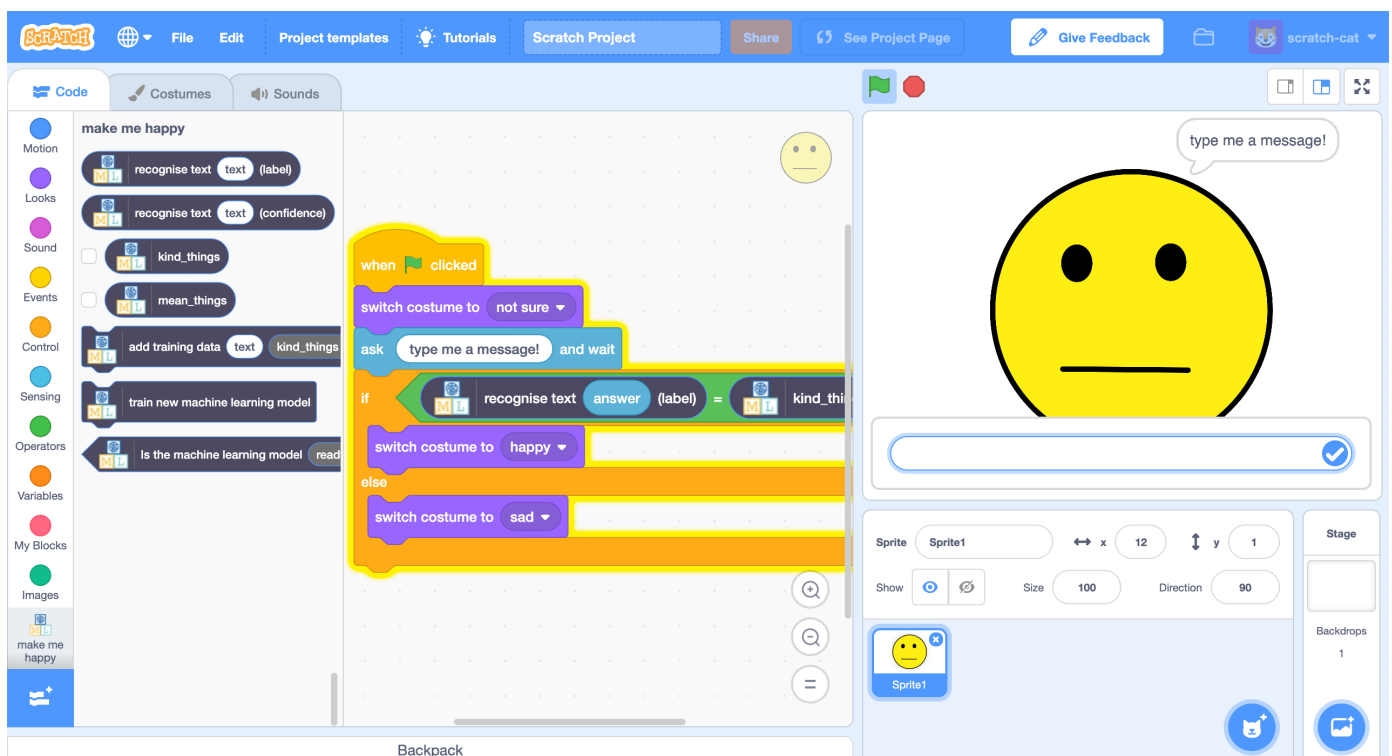
In this project you will make a character that reacts to what you say.

If you compliment it, it will look happy.

If you insult it, it will look sad.

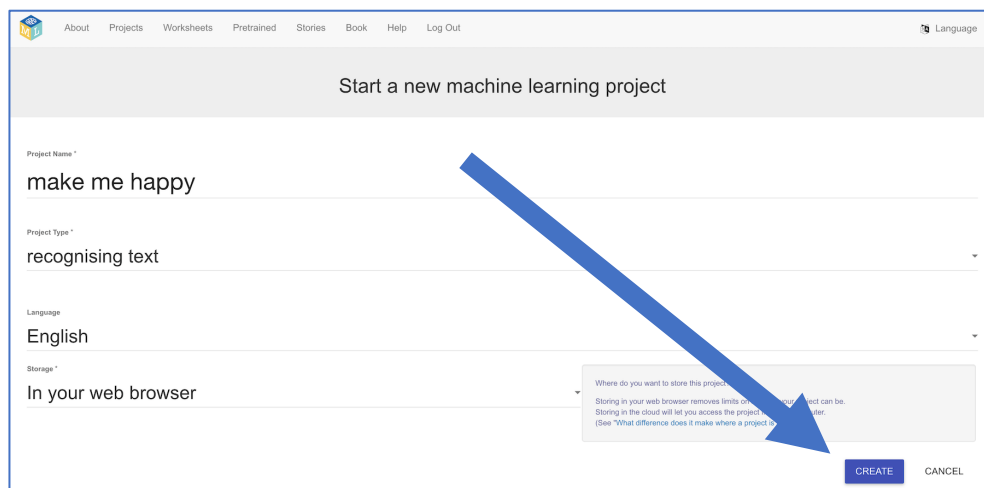
At first, you'll program a list of rules for what is kind and what is mean, and learn why that approach isn't very good.

Next, you will teach the computer to recognise kind messages and mean messages by giving it examples of each.



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 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.
4. Click on “**Projects**” on the top menu bar
5. Click the “**+ Add a new project**” button.
6. Name your project “make me happy” and set it to learn how to **recognise text**.
7. Choose to store your project in the cloud
8. Click the “**Create**” button



The screenshot shows a web form titled "Start a new machine learning project". The form has the following fields:

- Project Name ***: make me happy
- Project Type ***: recognising text
- Language**: English
- Storage ***: In your web browser

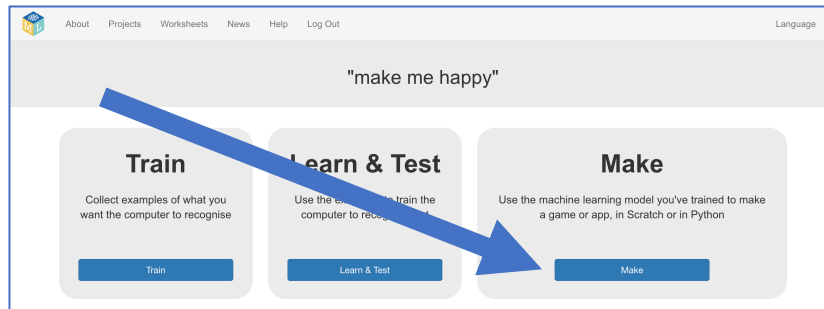
A blue arrow points from the "Project Name" field down to the "CREATE" button. A tooltip is visible over the "Storage" dropdown, explaining the difference between storing in the web browser versus the cloud.

Where do you want to store this project?
Storing in your web browser removes limits on how large your project can be.
Storing in the cloud will let you access the project from anywhere.
(See "What difference does it make where a project is stored" for more details.)

CREATE CANCEL

9. You should now see “**make me happy**” in the list of your projects. Click on it.

10. Start by getting a project ready in Scratch. Click “Make”

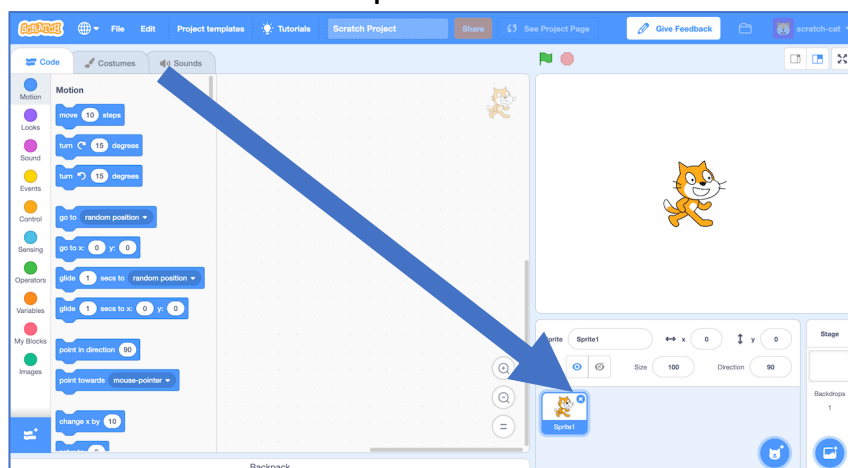


11. Click the **Scratch 3** button

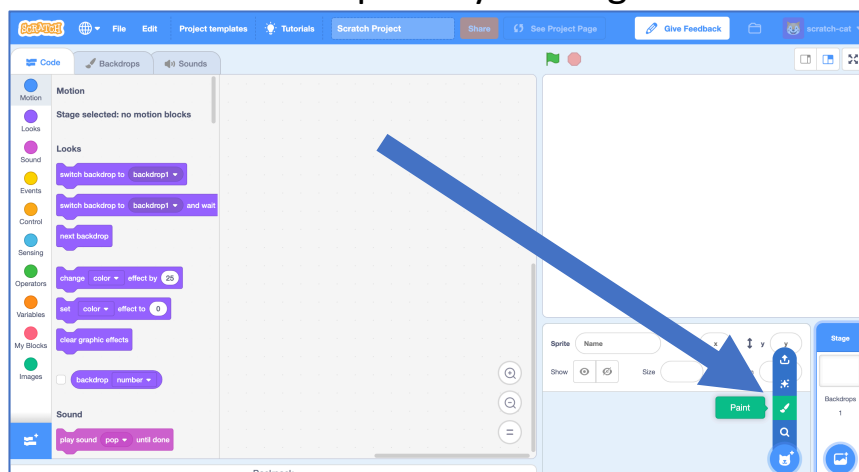
12. Click “Scratch by itself”

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

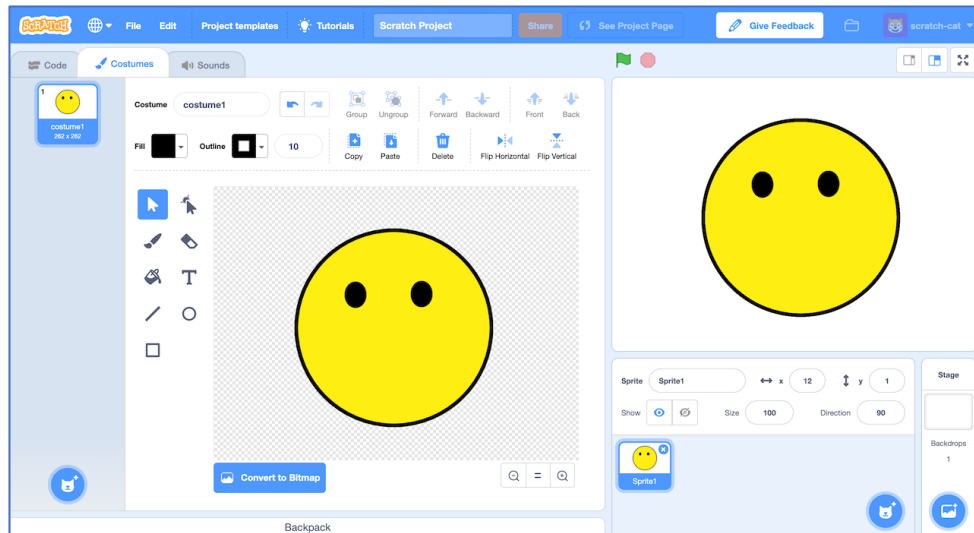
13. Delete the cat sprite



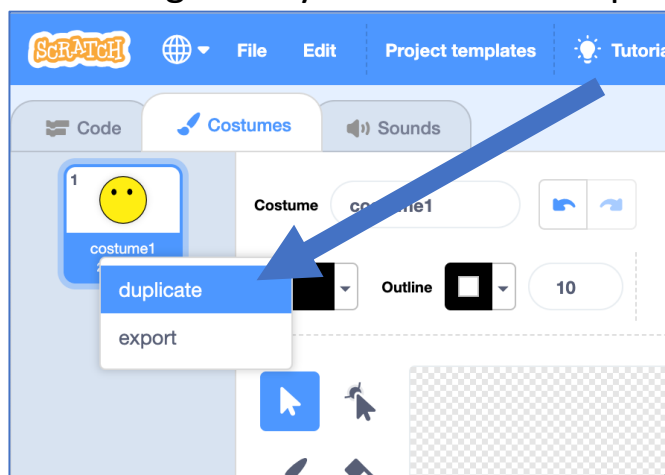
14. Create a new sprite by clicking on the **Paint** icon



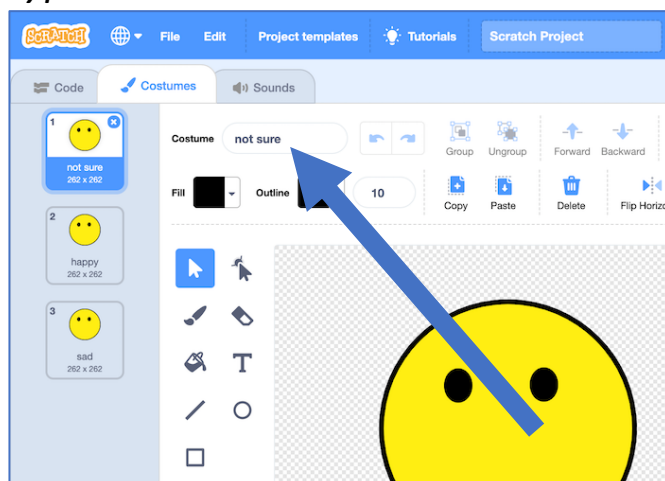
15. Draw a face without a mouth in the **Costumes** tab



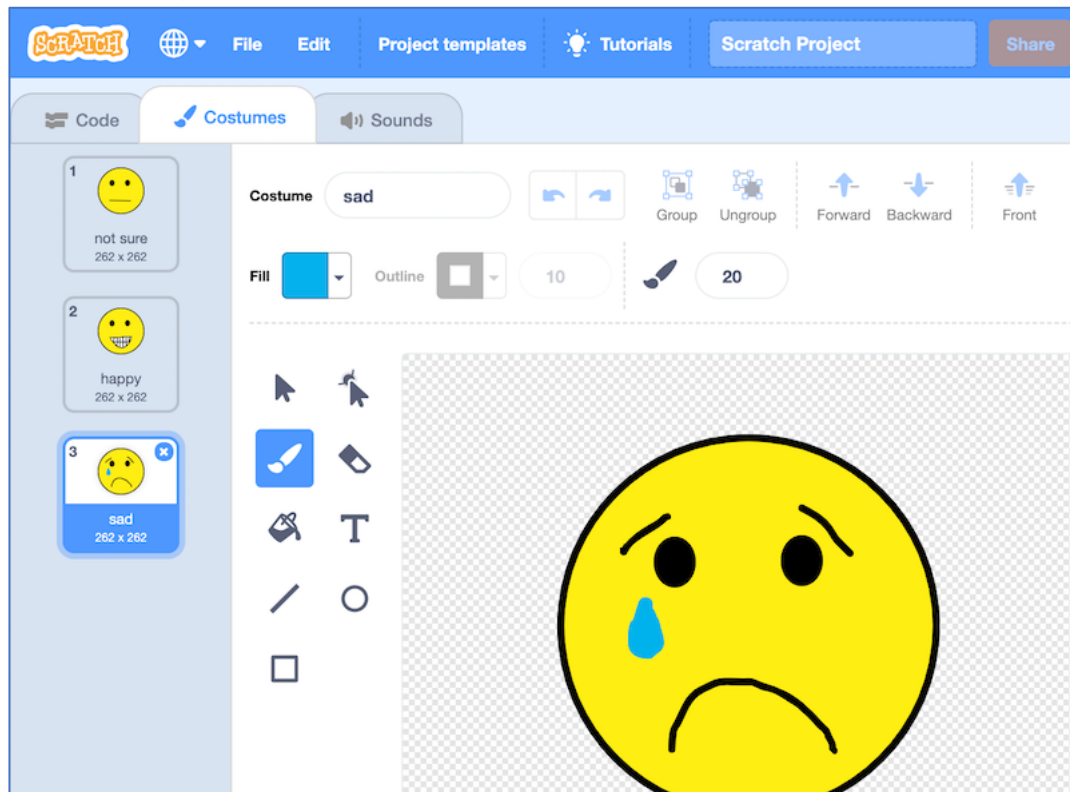
16. Right-click on the costume and click **“duplicate”**. Do that again so you have **three** copies of the costume.



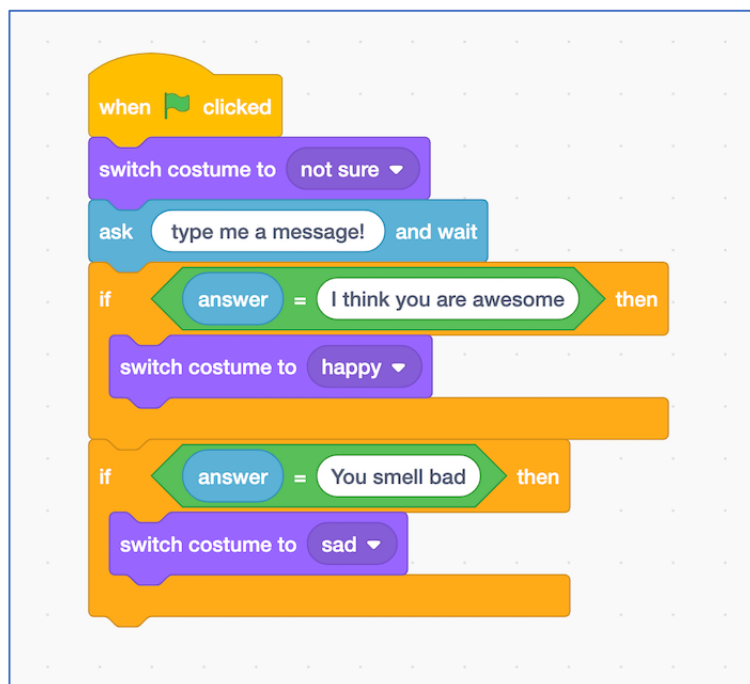
17. Name the three costumes “not sure”, “happy” and “sad” *Type the names into the white box shown by the arrow below.*



- 18.** Draw a mouth on each of the costumes.
The “not sure” face should be a straight line.
The “happy” face should have a smile.
The “sad” face should look sad.



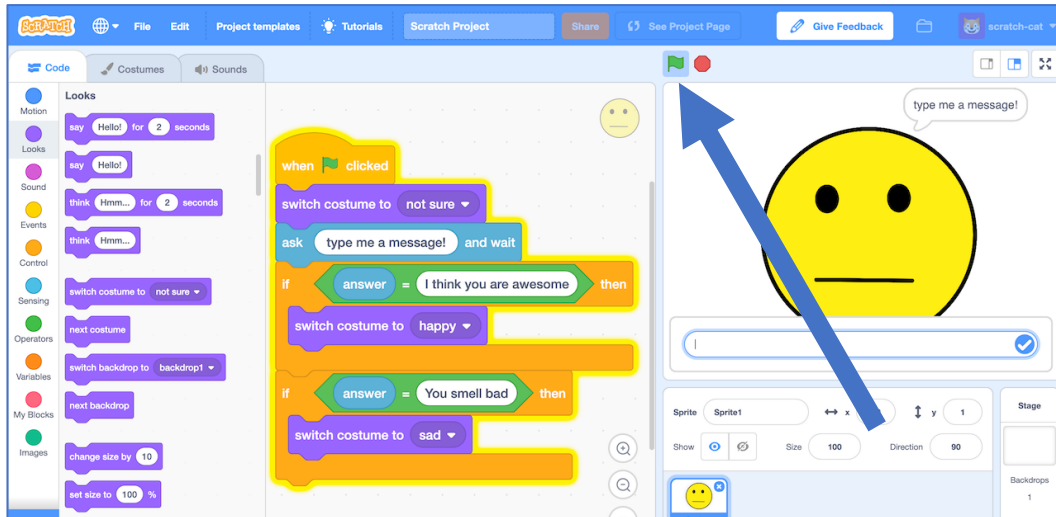
- 19.** Click the “**Code**” tab and enter the following script.



20. Save your project.

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

21. Click the **green flag** to test.



22. Type in a message and watch it react!

Type "I think you are awesome" and press enter. The character smiles. Click the green flag again and type "You smell bad". The character cries. Type anything else, and the character's face won't change.

What have you done so far?

You've created a character that should react to what people type, and programmed it using a simple rules-based approach.

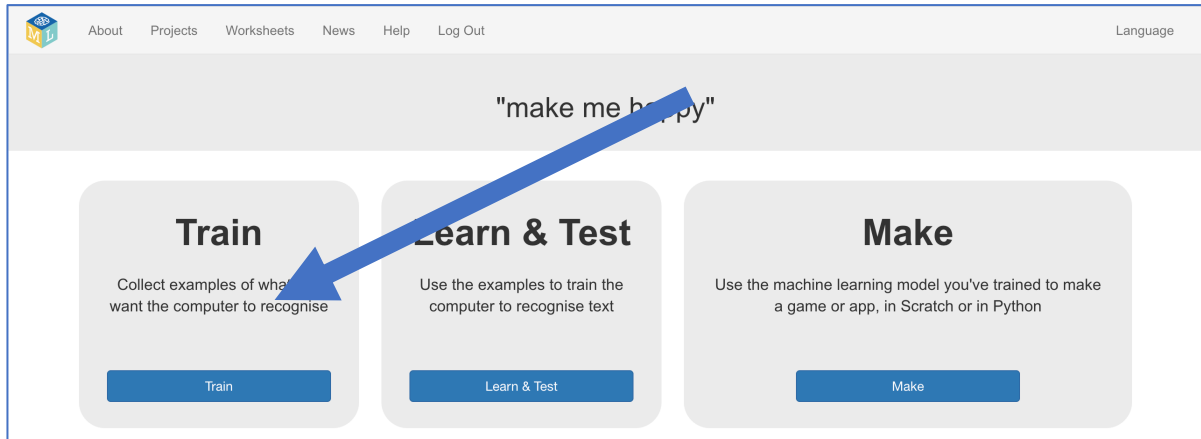
If you want it to react to other messages, you need to add more **if** blocks.

The problem with this is that you need to predict exactly what messages the character will receive. Making a list of every possible message would take forever!

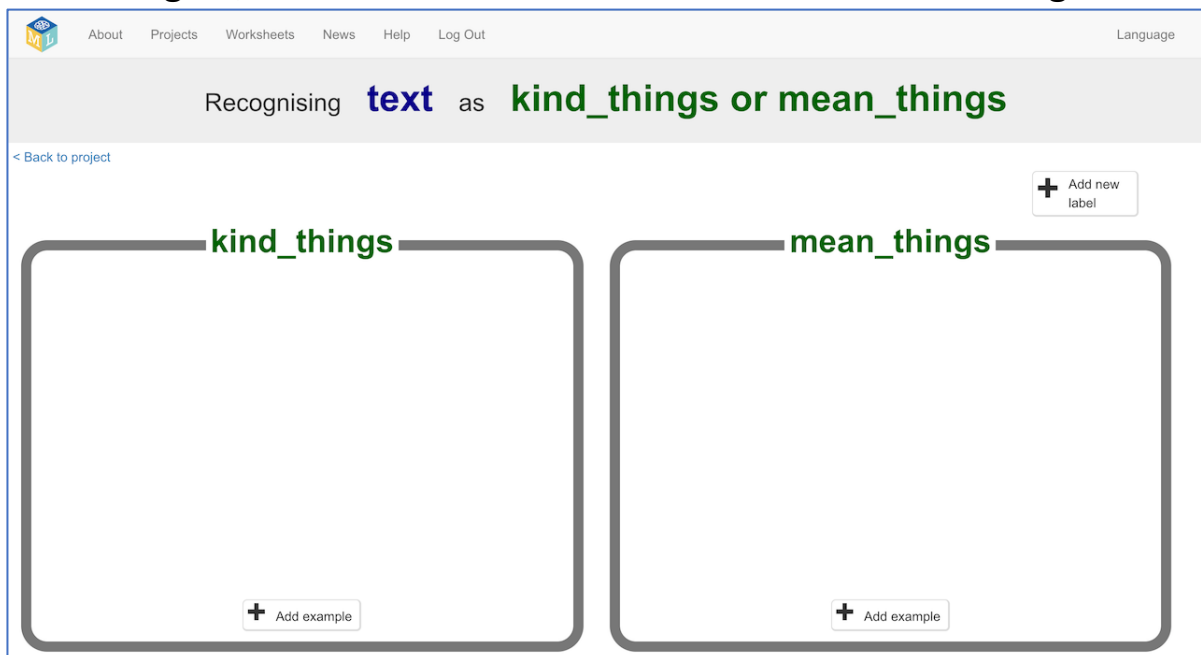
Next, we'll try a better approach – teaching the computer to recognise messages for itself.

23. Close the Scratch window.

24. You need examples to train the computer. Click the “< Back to project” link. Then click the **Train** button.



25. Click on “+ Add new label” and call it “kind things”. Do that again, and create a second bucket called “mean things”.



26. Click the “Add example” button in the “kind things” bucket, and type in the nicest, kindest compliment you can think of.

27. Click on the “Add example” button in the “mean things” bucket, and type in a meanest, cruellest insult you can think of.

28. Repeat steps 26 and 27.

Write at least **six** compliments and at least **six** insults.

The screenshot shows a web interface titled "Recognising text as kind_things or mean_things". It has a navigation bar with links: About, Projects, Worksheets, News, Help, Log Out, and a Language dropdown. Below the title bar, there is a "< Back to project" link and an "Add new label" button. The main area is divided into two columns. The left column is titled "kind_things" and contains six text boxes with the following text: "You're a lovely person", "I appreciate all of the things you do", "Your hair looks great today", "You're my best friend", "I think you're amazing", and "That jacket looks great on you". Below these boxes is an "Add example" button. The right column is titled "mean_things" and contains six text boxes with the following text: "You smell", "I don't like you", "You're as dumb as a bag of rocks", "You're an idiot", "You smell bad", and "I'm fed up with how useless you are". Below these boxes is an "Add example" button. Both columns have a small circle with the number "6" at the bottom right corner.

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

Then click on the “Learn & Test” button.

30. 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 messages from the examples you’ve given to it.

The screenshot shows a web interface titled "Machine learning models". It has a navigation bar with links: About, Projects, Worksheets, News, Help, Log Out, and a Language dropdown. Below the title bar, there is a "< Back to project" link. The main area is divided into two columns. The left column is titled "What have you done?" and contains the text: "You have collected examples of text for a computer to use to recognise when text is kind_things or mean_things." and "You've collected:" followed by a list: "• 6 examples of kind_things," and "• 6 examples of mean_things". The right column is titled "What's next?" and contains the text: "Ready to start the computer's training?" and "Click the button below to start training a machine learning model using the examples you have collected so far" and "(Or go back to the train page if you want to collect some more examples first.)". Below these columns is a section titled "Info from training computer:" which contains a "Train new machine learning model!" button. A blue arrow points from the "Train new machine learning model!" button in the right column to the "Train new machine learning model!" button in the "Info from training computer:" section.

31. Wait for the training to complete. This might take a minute.

- 32.** Once the training has completed, a Test box will be displayed. Try testing your model to see what the computer has learned. Type something kind, and press enter. It should be recognised as kind. Type something mean, and press enter. It should be recognised as mean. *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 28, and add some more examples. Make sure you repeat step 30 to train with the new examples!*

The screenshot shows a web interface for training a machine learning model. It is divided into two main sections: 'What have you done?' and 'What's next?'. A large blue arrow points from the 'What's next?' section down to a 'Test' button in a text input area.

What have you done?

You have trained a machine learning model to recognise when text is kind_things or mean_things.

You created the model on Friday, April 12, 2019 1:05 PM.

You have collected:

- 6 examples of kind_things.
- 6 examples of mean_things

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 difference the extra examples will make!

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

Recognised as **mean_things**
with 87% confidence

What have you done so far?

You've started to train a computer to recognise text as being kind or mean. 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".

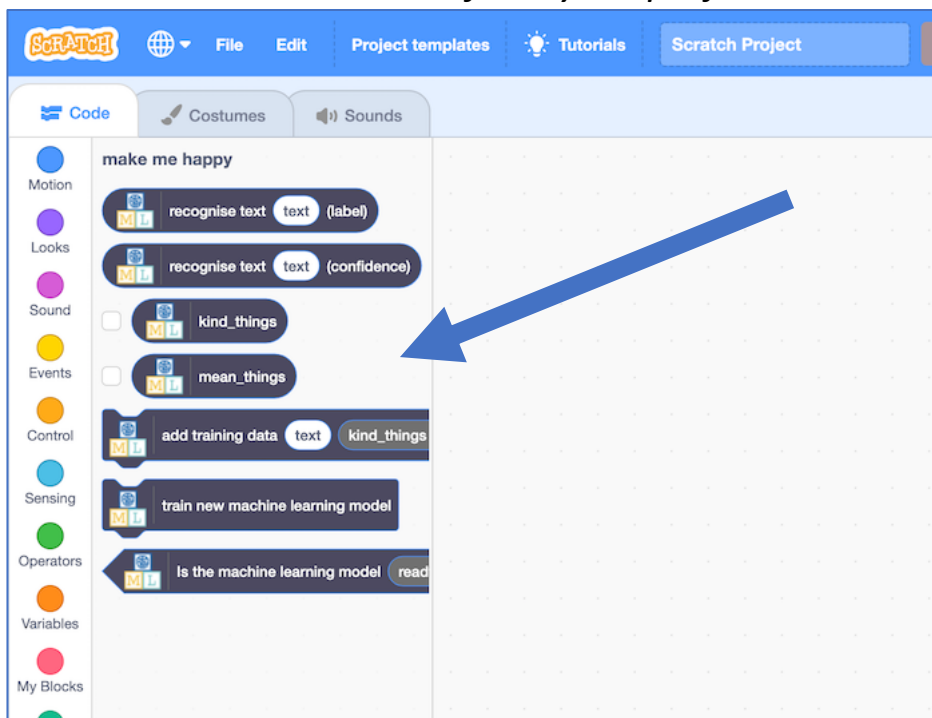
This is called "supervised learning" because of the way you are supervising the computer's training.

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 messages.

33. Click the “< Back to project” link

34. Click the “**Make**” button, then the “**Scratch 3**” button.
*This page has instructions on how to use the new blocks in Scratch.
Keep the page open if you need to check back on how to use them.*

35. Click the “**Open in Scratch**” button to launch the Scratch editor.
You should see new blocks from your project at the bottom of the list.



36. Load the Scratch project that you saved earlier
*Click on **File** -> **Load from your computer***

Tips

More examples!

The more examples you give it, the better the computer should get at recognising whether a message is kind or mean.

Try and be even

Try and come up with roughly the same number of examples for kind and mean.

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

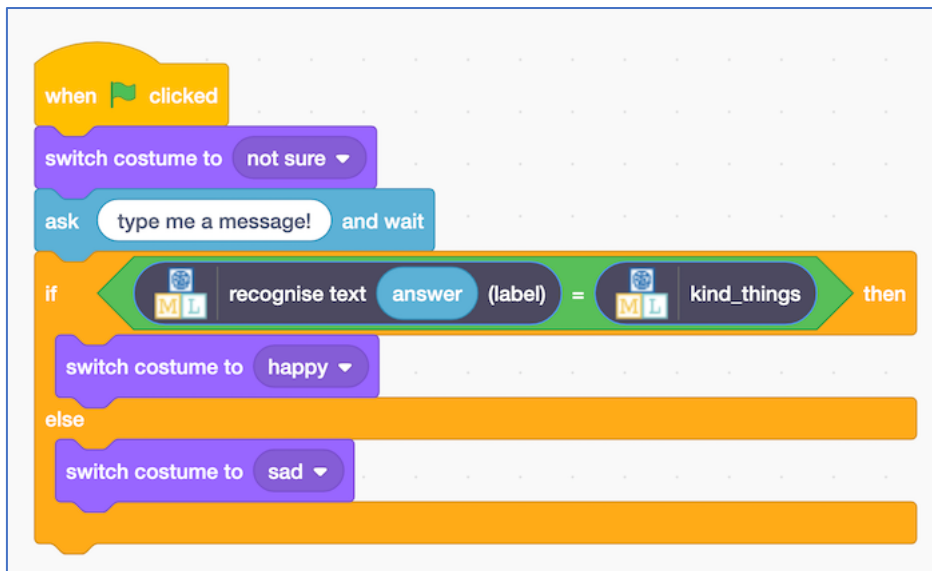
Mix things up with your examples

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

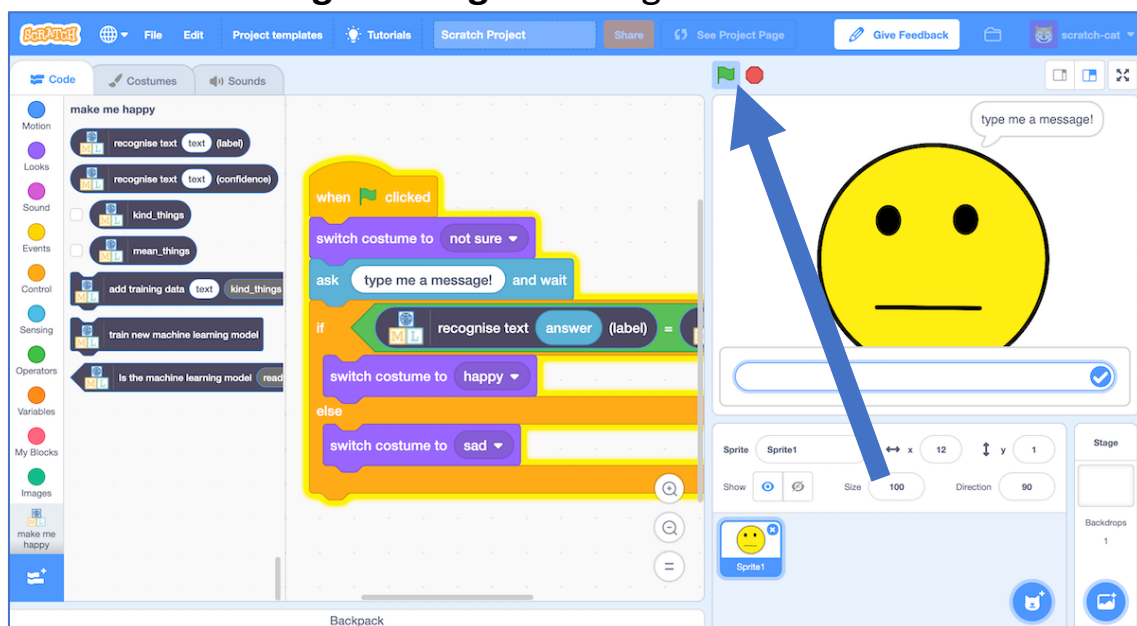
For example, make sure that you include some long examples and some very short ones.

- 37.** 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 some text, it will return either “kind things” or “mean things” based on the training you’ve given to the computer. You can use this to choose the costume to switch to.



- 38.** Click on the **green flag** to test again.

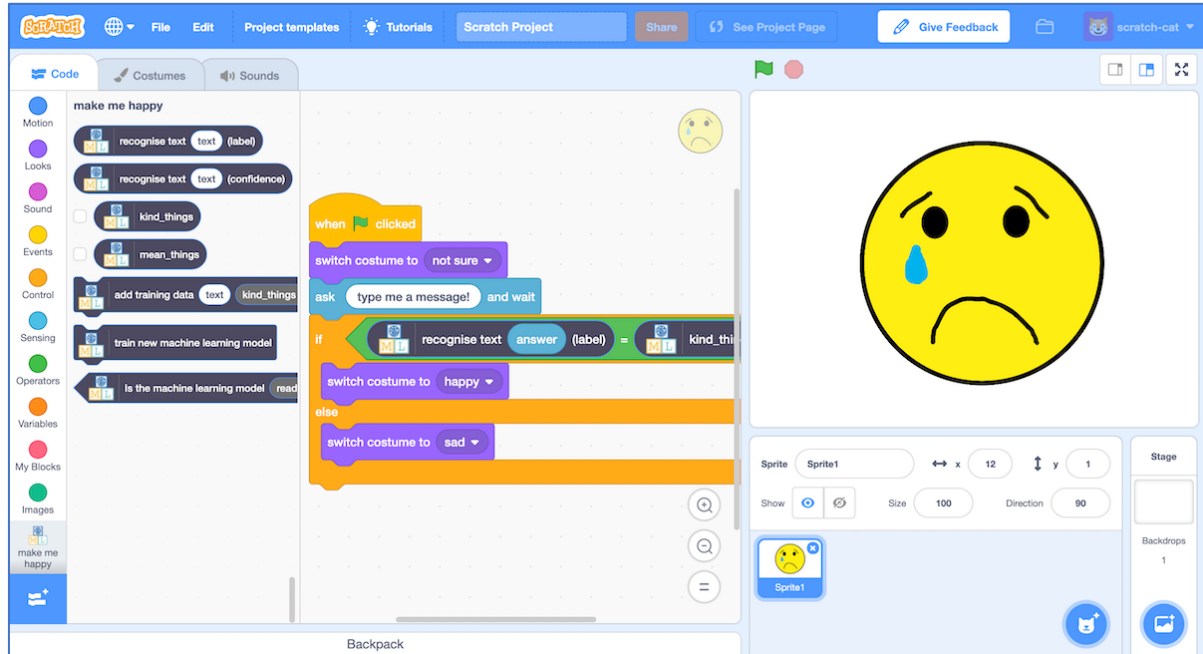


39. Test your project

Type a kind message and press enter. The character should smile.

Click the green flag again. Type a mean and unkind message and press enter. The character should look sad.

This should work for messages that you didn't include in your training.



What have you done?

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

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

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

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?

Write a reply

Instead of just changing the way they look, make your character reply, based on what it recognises in the message!

Try a different character

Instead of a person's face, why not try something different, like an animal?

It could react in different ways, instead of smiling.

For example, you could make a dog that wags their tail if you say something kind to it!

Different emotions

Instead of kind and mean, could you train the character to recognise other types of message?

Real world sentiment analysis

Can you think of examples where it's useful to be able to train a computer to recognise the emotion in writing?