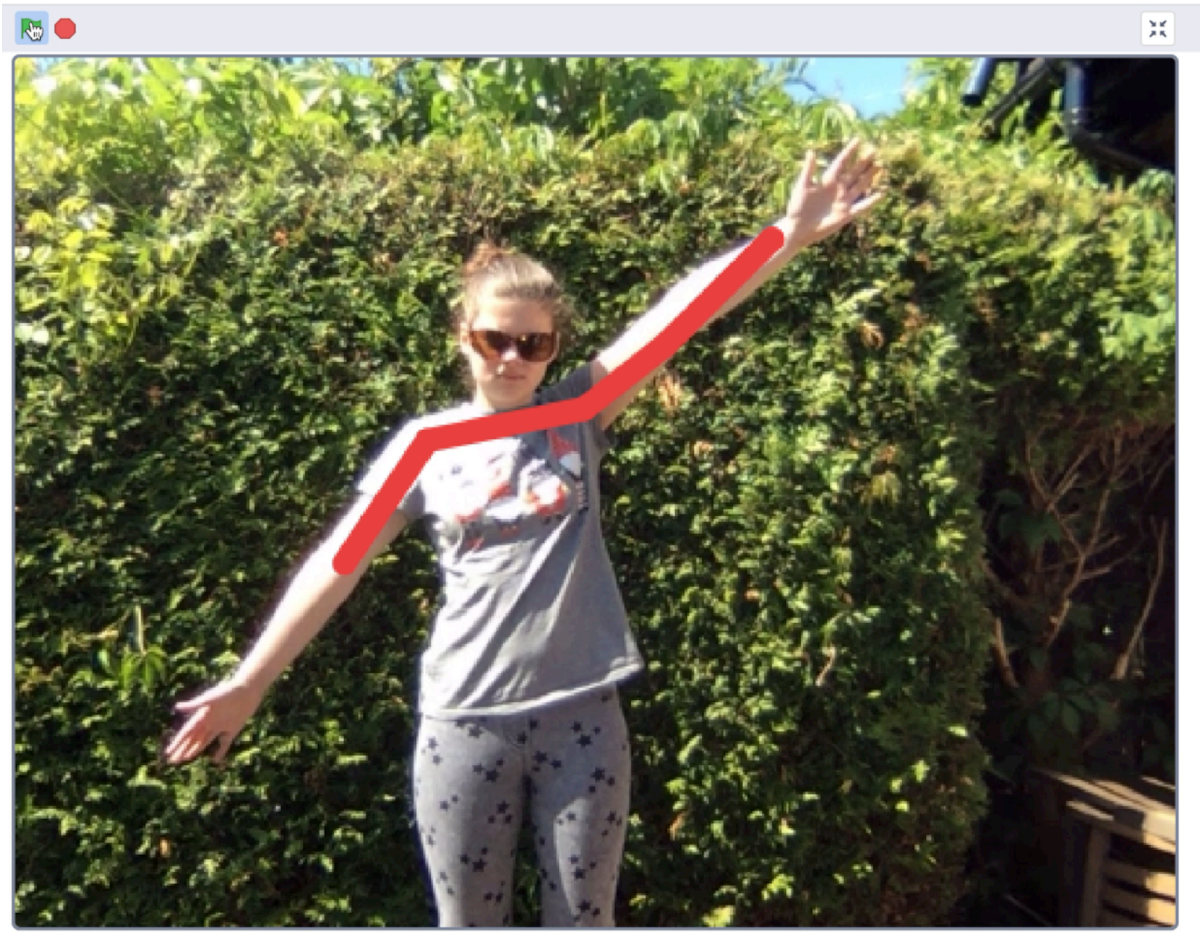
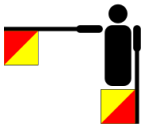
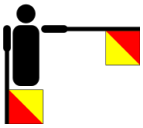
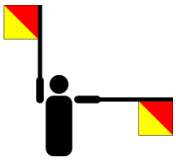
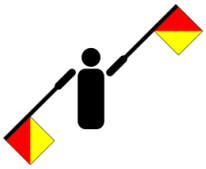
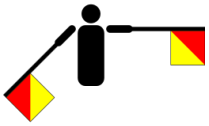
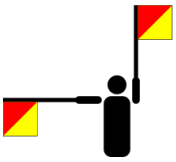
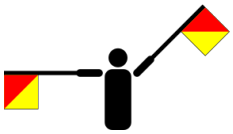
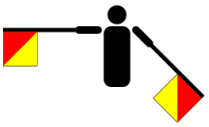
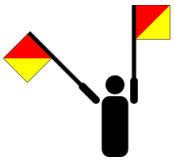
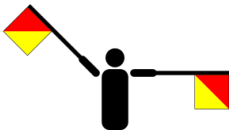
Semaphore

In this project you will make a game to help you learn semaphores. Semaphores is a way of communicating by holding your arms in certain positions (see the next page).

You will train a machine learning model so your game can recognize commands you speak (to let you play without touching the computer).

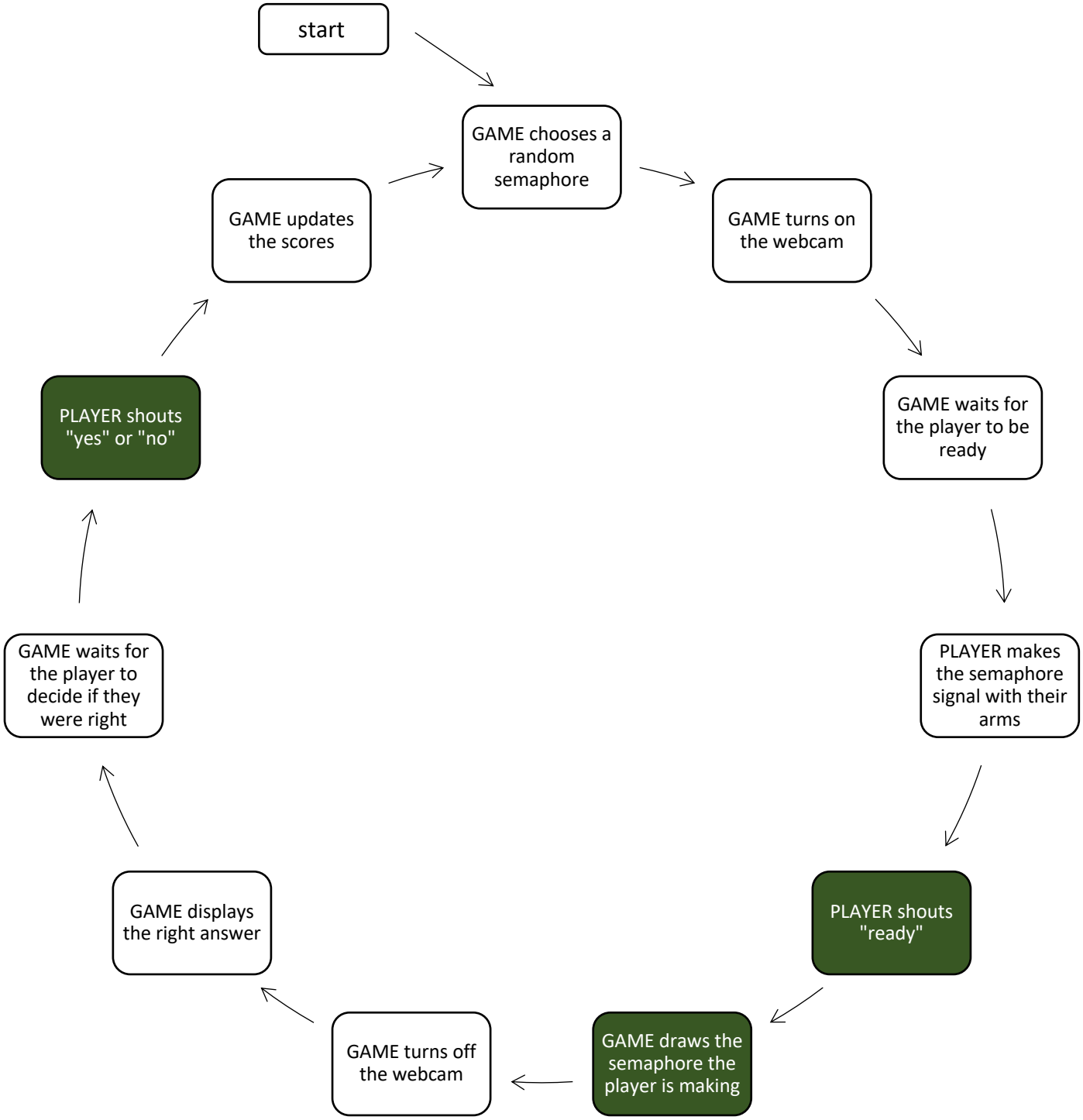
You will use a second machine learning model to find the shape your arms make in the webcam view.



Semaphore			
 A (alpha)	 B (bravo)	 C (charlie)	 D (delta)
 E (echo)	 F (foxtrot)	 G (golf)	 H (hotel)
 I (india)	 J (juliet)	 K (kilo)	 L (lima)
 M (mike)	 N (november)	 O (oscar)	 P (papa)
 Q (quebec)	 R (romeo)	 S (sierra)	 T (tango)
 U (uniform)	 V (victor)	 W (whiskey)	 X (x-ray)
 Y (yankee)	 Z (zulu)		

Design for this project

In this project, you'll make a game to test if you know your semaphore signals!
The dark blocks show where you will be using machine learning, so the computer can:
- recognize the shape you'll be making with your arms
- understand the commands you'll be shouting out (so you don't have to press anything during the game)



1. Go to https://machinelearningforkids.co.uk/ in a web browser

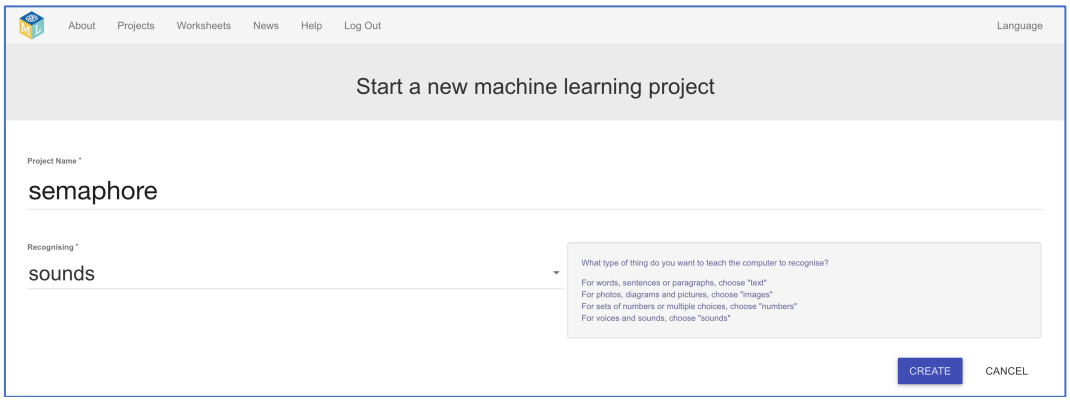
2. Click on "**Get started**"

3. Click on "**Log In**" and type in your username and password
If you don't have a username, ask your teacher to create one for you.

4. Click on "**Projects**" on the top menu bar

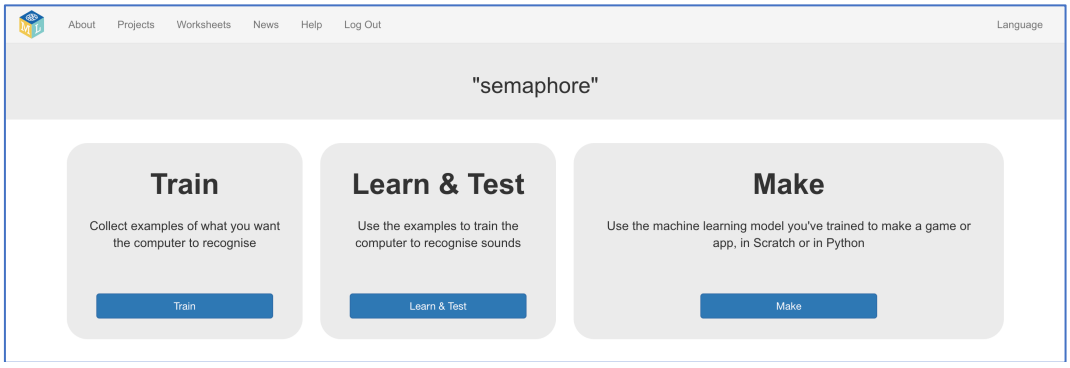
5. Click the "**+ Add a new project**" button.

6. Name your project "semaphore" and set it to learn how to recognise "**sounds**".
Click the "**Create**" button

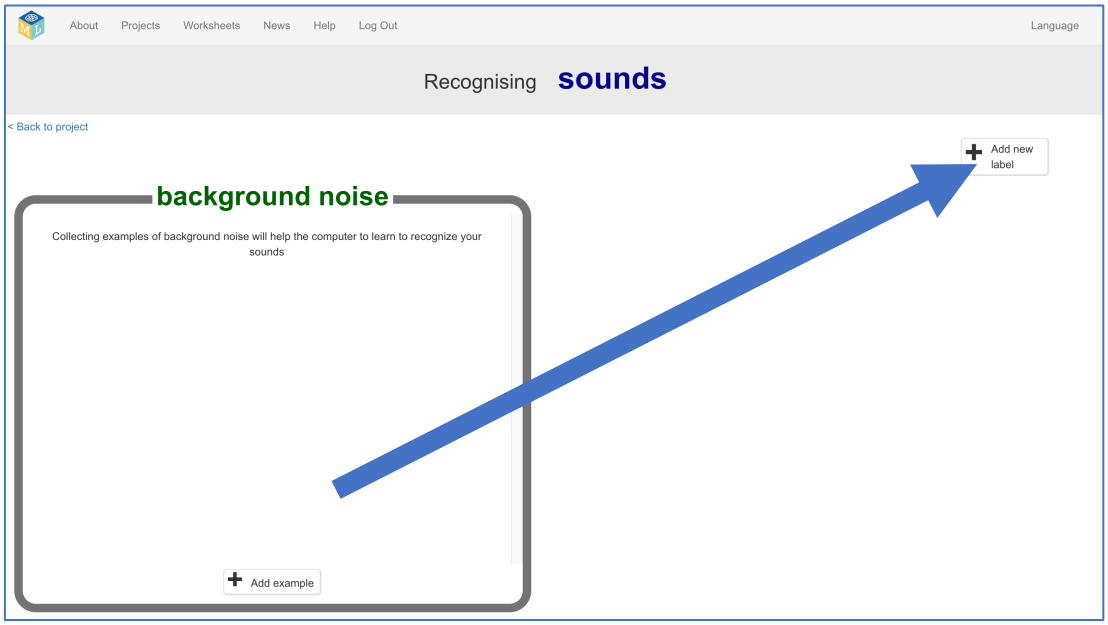


7. You should now see "**semaphore**" in your list of projects. Click on it.

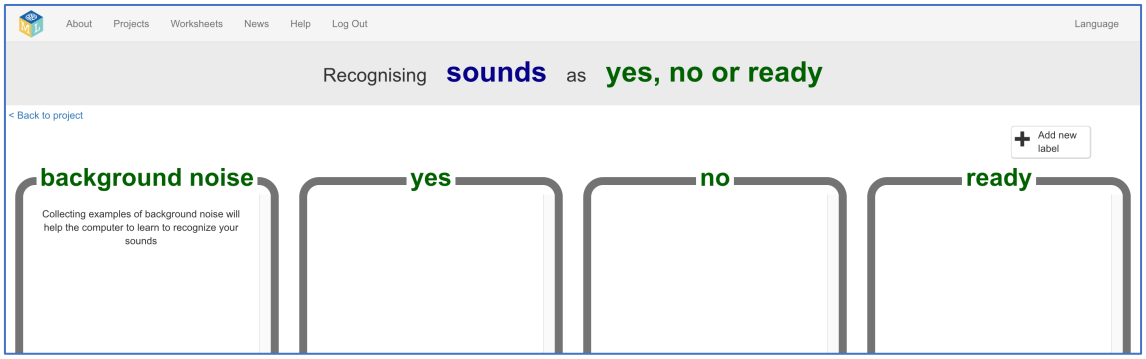
8. Click on "**Train**" to start training the computer to recognize your commands.



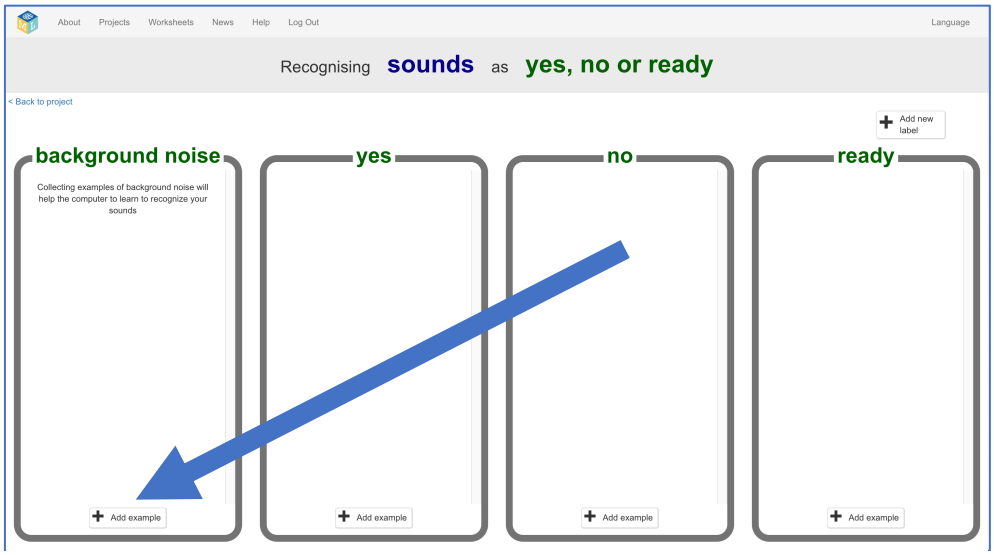
9. Click on the "**+ Add new label**" button to create a bucket for training examples for each of your three commands



10. Create buckets for "yes", "no", and "ready"

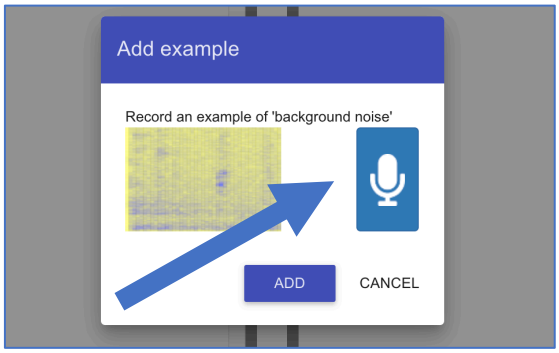


11. Click the "**+ Add example**" button in the background noise bucket.



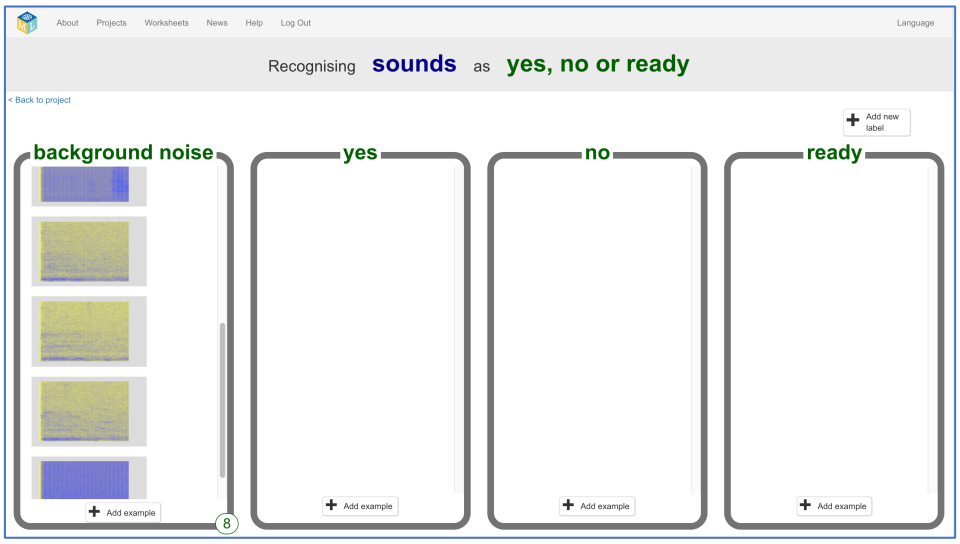
12. Use the "**microphone**" button to record background noise

This helps the computer learn the difference between your commands, and the other things that it can hear.



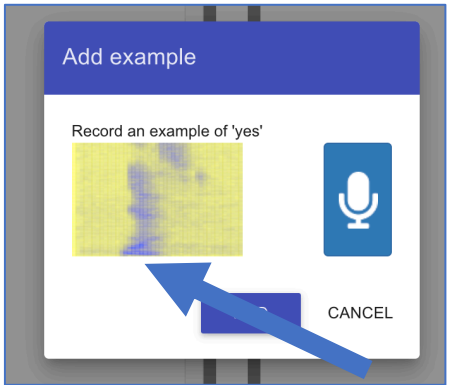
13. Click "**Add**"

14. Repeat until you have at least **eight** examples of background noise



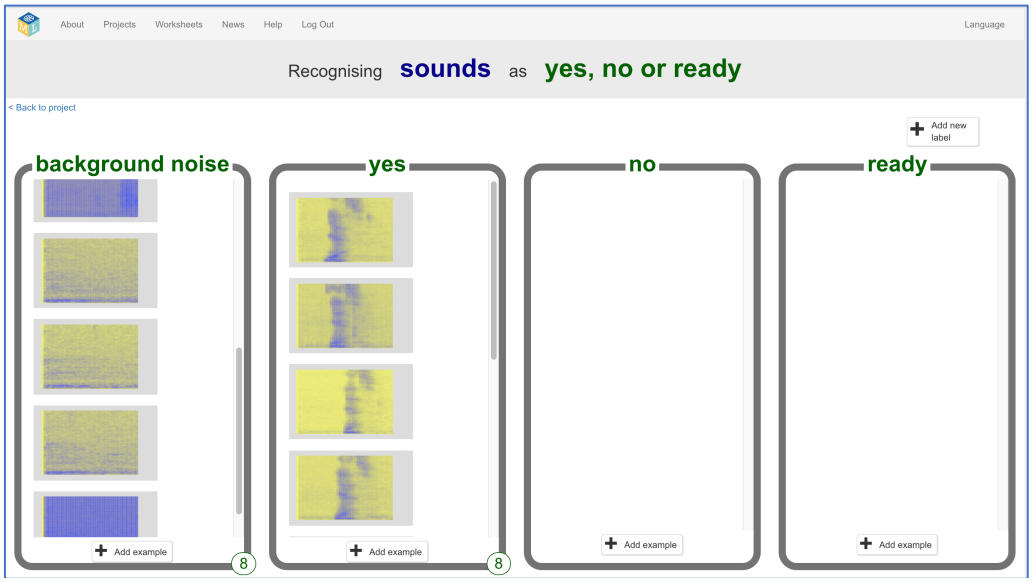
15. Click the "**+ Add example**" button in the "yes" bucket and record yourself saying the word "yes"

The graphic gives you an idea of when you said "yes" in the recording.



16. Repeat until you have at least **eight** examples of you saying “yes”

The more examples you have, the better it should work, but eight is a minimum.



17. Repeat this for the remaining two buckets.

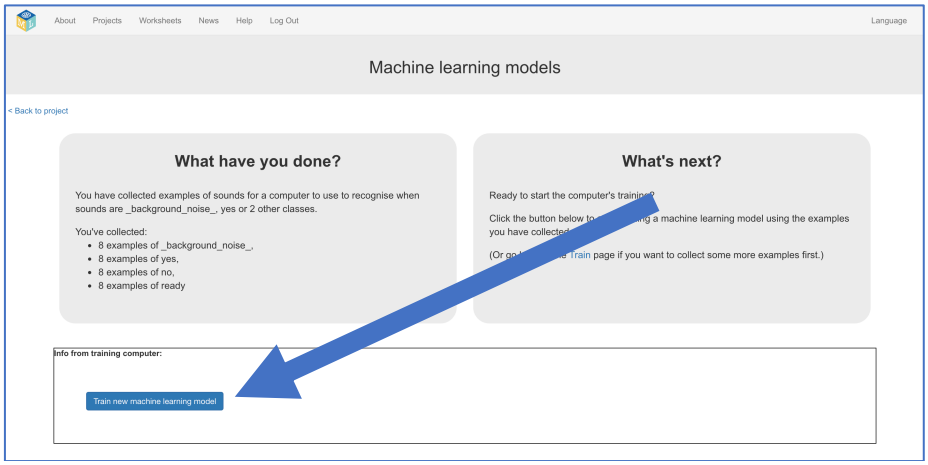
*Record at least **eight** examples of you saying “no” in the “**no**” bucket, and at least **eight** examples of you saying “ready” in the “**ready**” bucket.*

18. Click on the “**< Back to project**” link in the top-left

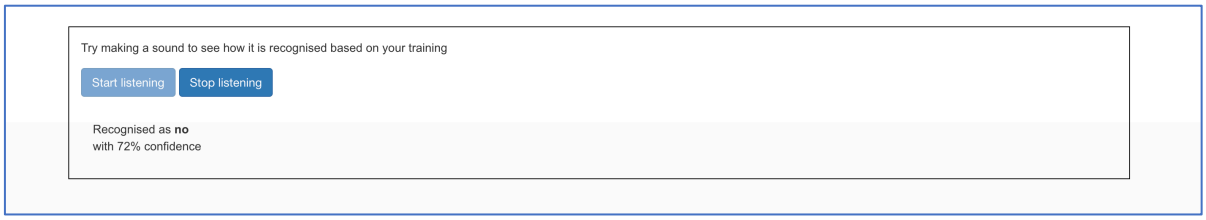
19. Click on “**Learn & Test**”

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

The computer will use the examples you have created to learn how to recognize the sound of you giving each of your three commands.



21. Try out your model to check how well your training has gone
Click on "**Start listening**" and try saying one of your commands.
Does the computer recognize what you say? If it is making too many mistakes, go back to the Train page and add more examples.



22. Click on the "**< Back to project**" link in the top-left

23. Click on "**Make**"

24. Click on "**Scratch 3**"

25. Click the "**Open in Scratch 3**"

What have you done so far?

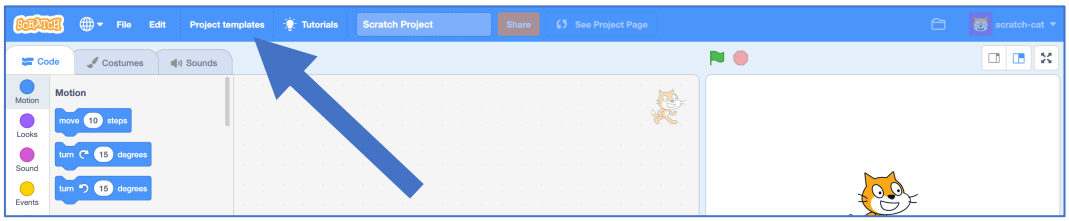
You've trained a machine learning model to recognize your voice saying different commands. This will help you to make a hands-free Scratch project, that you can control with your voice.

The next step is to start making your Scratch project.

Your Scratch project will also need another machine learning model, that you can use with your webcam to recognize the shape that your arms are making to make a semaphore signal.

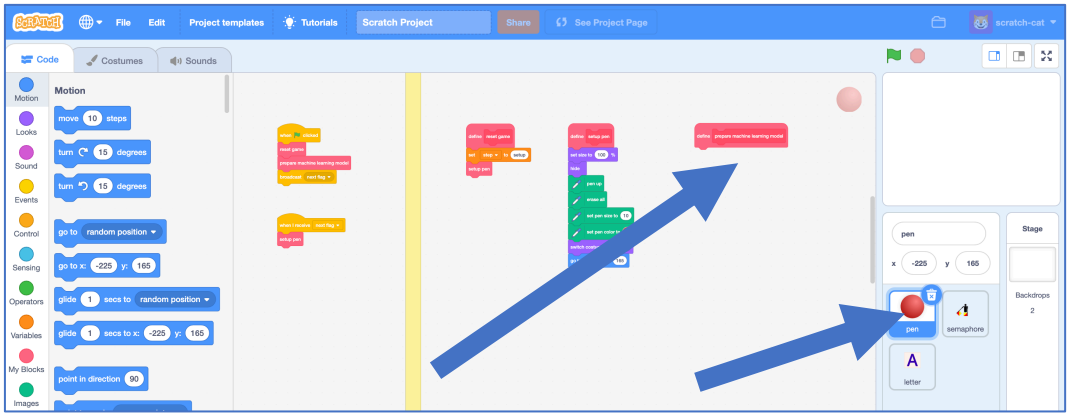
To save time, you won't be training that model yourself as well. You'll use a pre-trained model provided by the Machine Learning for Kids website.

26. Click on “**Project templates**”

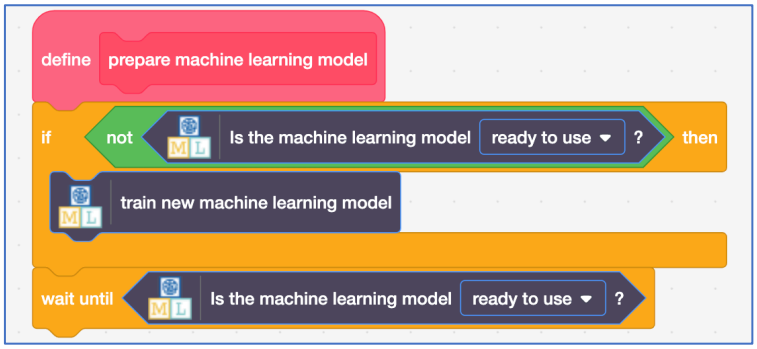


27. Click on the “**Semaphores**” template

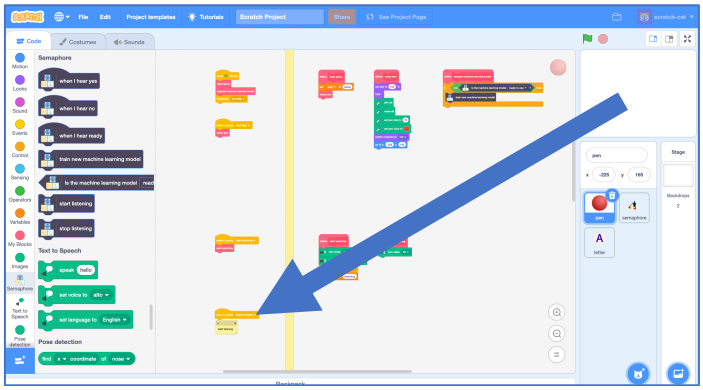
28. Find the “**prepare machine learning model**” script in the “**pen**” sprite



29. Add blocks to the empty script to train your ML model

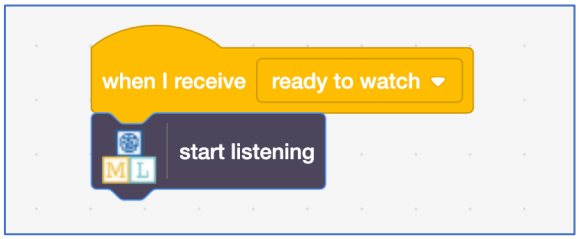


30. Find the “**when I receive ‘ready to watch’**” block

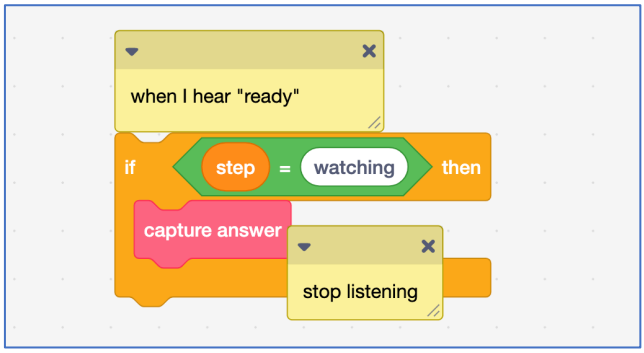


31. Add a "**start listening**" block to the script

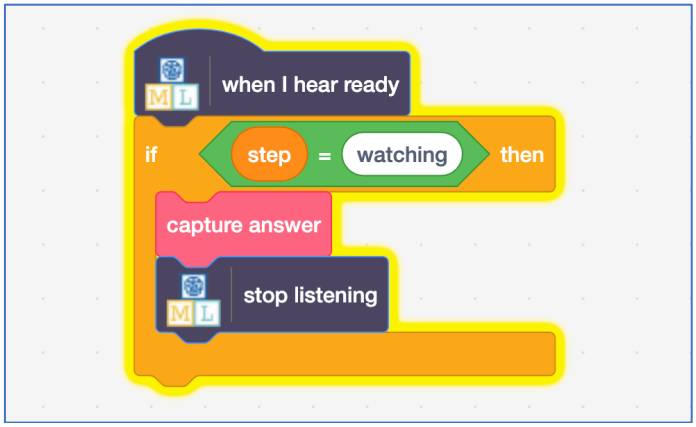
The comment there is to help you, but it isn't needed – you can delete it.



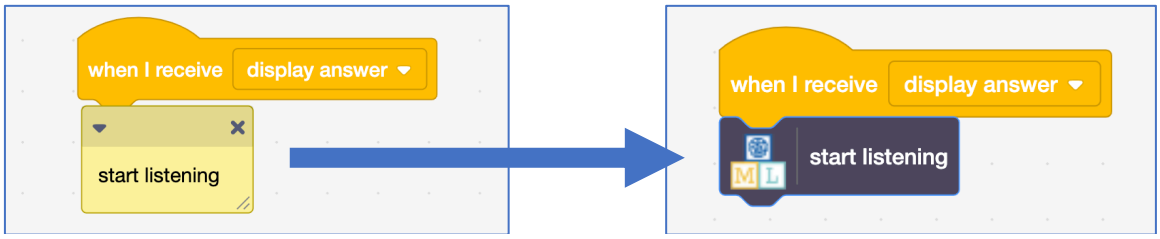
32. Find this script – you'll find it underneath the "ready to watch" block from the previous step



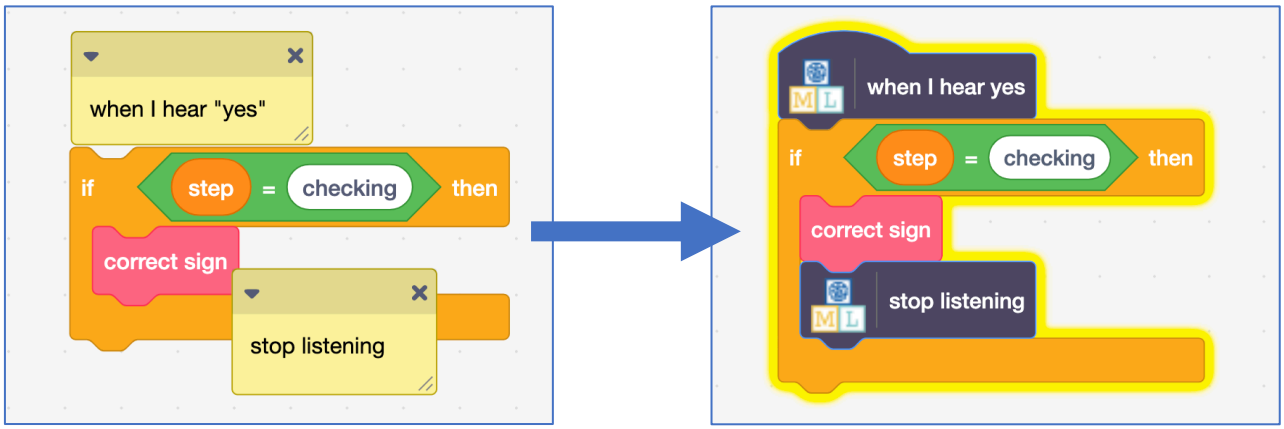
33. Replace the placeholder comments with code blocks from your machine learning project, so that it looks like this.



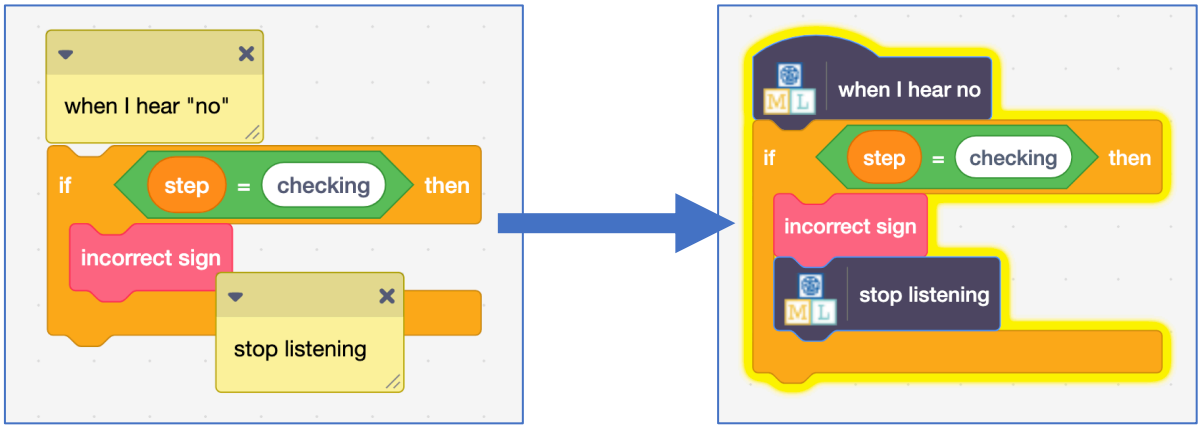
34. Find the "**when I receive 'display answer'**" block underneath, and add a "**start listening**" block to it



35. Update the code as shown underneath using your ML project blocks



36. Update the code as shown underneath using your ML project blocks



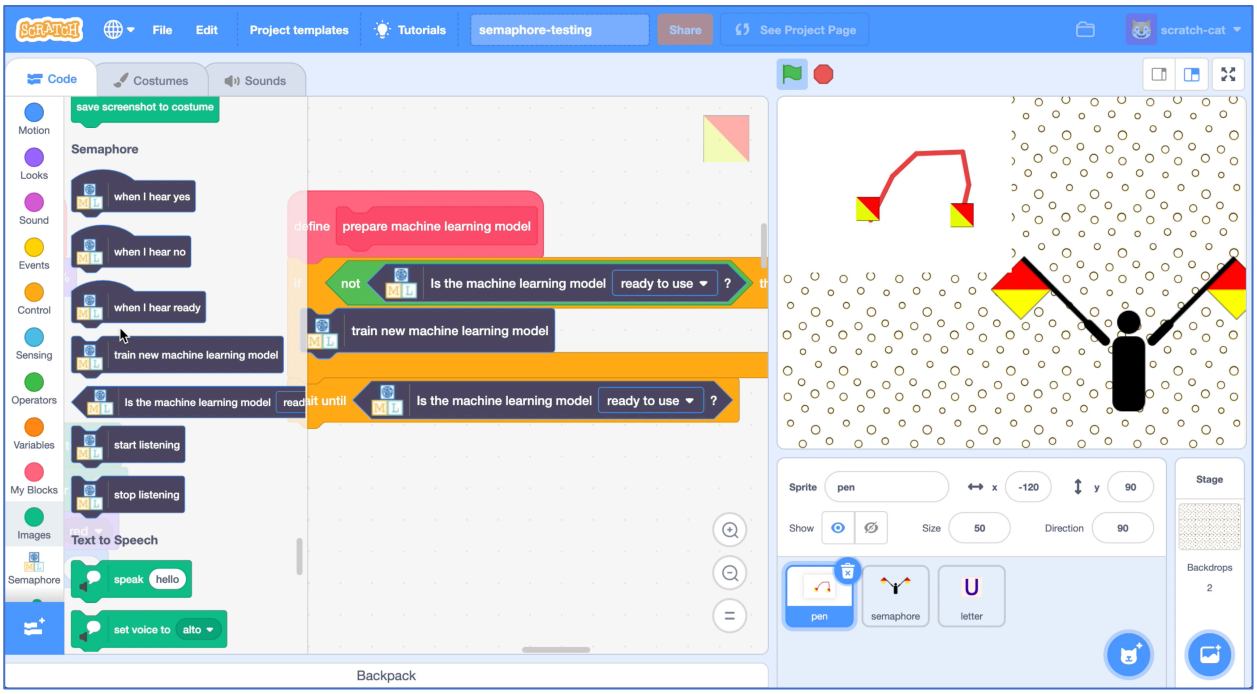
37. Time to test!

Click the Green Flag to start the game. A description of the game is on page 3 if you need a reminder.

The game needs a little space so you can hold out your arms. If you can take your computer outside, even better!

The game will ask you to make a semaphore signal, and start the webcam. Try to step back so you can see your whole body in the webcam view, and when you're ready, shout "ready" so the computer can start to trace out the way your arms are pointing.

It will show you what the right answer is alongside your attempt. Shout "yes" if you think you got it right, or "no" otherwise.



What have you done?

You've created a game using two different machine learning models.

The first machine learning model is a sound recognition model that you've trained to recognize your commands. This is similar to technology used in devices like Amazon's Alexa or Apple's Siri – trained to recognize certain sounds and programmed to take actions in response when heard.

The second machine learning model is a pose detection model. This means it is an image recognition model, that can be used to estimate the pose of a person in the image, by estimating where key body joints are. It was trained using a very large set of training images called Common Objects in Context (often described as "COCO") which were labelled with the locations of wrists, elbows, shoulders, knees, etc.

Using a pre-trained model is common thing in real-world machine learning projects. It is much quicker to use a large set of training examples that someone else has collected, than it is to create your own training examples.

Tips

Play by yourself
The pose detection model you're using has been trained to recognize a single person in the webcam. It will not work as well if more than one person is visible.

Add new training examples if you need to
We recorded training examples indoors, then took the laptop outside to test. The background noise was so different outside, and our voices sounded so different when shouted from a distance, that our model wasn't good at recognizing commands.

We added more training examples, recorded outside where we were testing. This made it work much better.

If you need to do this, remove the "if" block you added in step 29. That "if" block stops you training a new machine learning model.

Make the game easier
Learning every semaphore symbol is difficult! To make the game easier, delete some costumes in the semaphore sprite. The game only asks questions based on the costumes left in there.

Try training your own image model
You had to tell the computer if your signals were correct or not. This would be a good job for a machine learning model!

Try training another image machine learning model, using the tracing line drawn over your arms as training examples. You could use this to let your game recognize if you got the semaphore signal right or not.