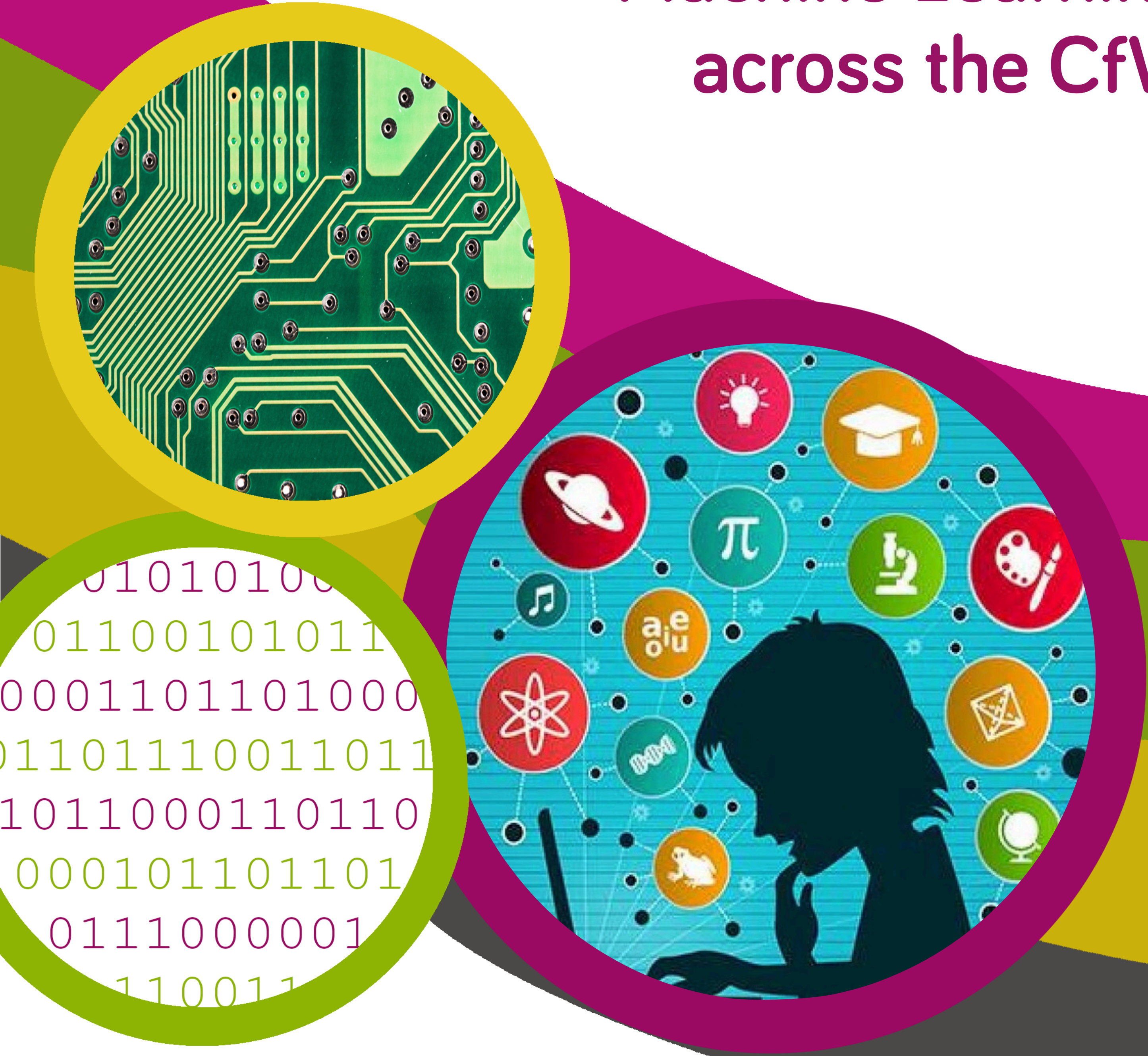
technocamps

Machine Learning across the CfW



UNDEB EWROPEAIDD
EUROPEAN UNION

Llywodraeth Cymru
Welsh Government

Cronfa Gymdeithasol Ewrop
European Social Fund

Prifysgol Abertawe
Swansea University

Cardiff University
Prifysgol Caerdydd

Prifysgol Bangor University

Cardiff Metropolitan University
Prifysgol Metropolitan Caerdydd

i.t.wales

Prifysgol Aberystwyth University

Prifysgol Glyndŵr Wrecsam
Wrexham Glyndŵr University

University of South Wales
Prifysgol De Cymru

Introduction

Overview

Machine Learning is a tool that is becoming more prevalent in society as software and technology continues to develop. It can be implemented across all the Areas of Learning and Experience, reinforcing learning in the classroom and improving digital literacy in the process.

In today's world digital literacy is an essential skill for learners to develop. The technological requirements for jobs are ever increasing, and a strong start in digital skills will prepare learners and give them an advantage.

Online Resources

Digital Resources:
https://tc1.me/educonf22resources

Youtube Tutorials:
https://tc1.me/progacrosscurriculum

More ideas for MLfK

Health and Wellbeing

- Predict Healthy Foods from Contents

Mathematics and Numeracy

- Make Predictions from Statistics

Science and Technology

- Predicting Classes of Animals

Languages, Literacy and Communication

- Recognising Authors
- Decoding Secret Codes

Expressive Arts

- Recognising Artists
- Recognising Musicians

Humanities

- Predicting Location of Landscapes

About MLfK

Machine Learning for Kids is a powerful tool that allows learners to train their own A.I. projects.

Educators can make an account which allows projects to be saved and class accounts to be created for group projects.

Learners are unable to make their own accounts, so the trained A.I. project itself cannot be saved (unless it is a group project), and will only last 4 hours.

Note: any code can be saved for future use, but will no longer have an A.I. to communicate with. The project can be recreated and the saved file loaded in.

Using MLfK

Go to: machinelearningforkids.co.uk

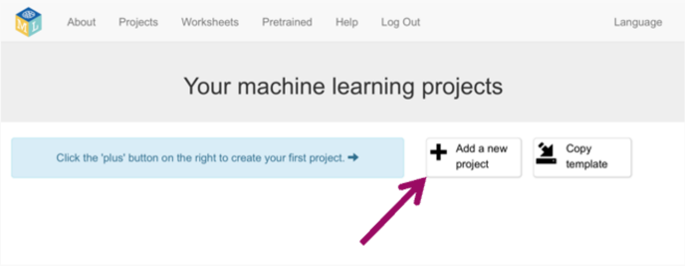
Click on Get Started

Click on Try It Now / Log in

Note: On an educators account, the Teacher menu has a Student Management page, where group accounts can be created or altered. This includes the login details for the class.

On sign-up as an educator, these accounts can be provided for you.

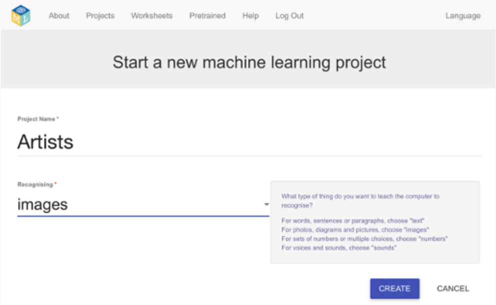
Making a Project



Your machine learning projects

Click the 'plus' button on the right to create your first project. ➔

On your projects page click Add a New Project to begin.



Start a new machine learning project

Project Name * Artists

Recognising * images

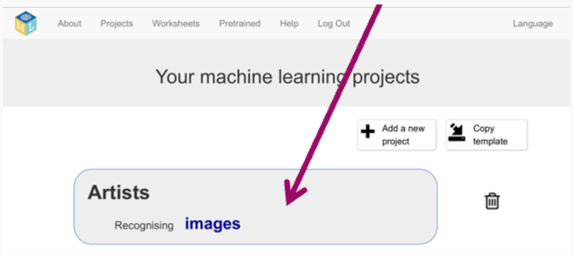
What type of thing do you want to teach the computer to recognise?

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"

Name your project and decide what you'll be training the A.I. to recognise.

Note: Under the worksheets tab on MLfK there are countless project examples and templates. All have thorough worksheets free for download and some even include Teacher Guidance sheets.

Using Your Project



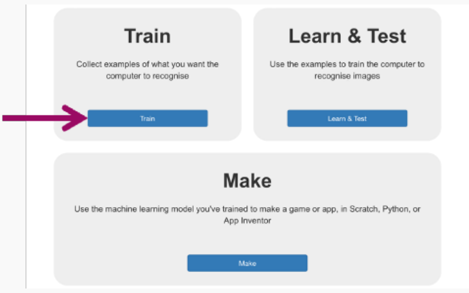
Your machine learning projects

Artists

Recognising images

Back on your projects page click on the project to open it.

These are the three steps to any project:
Training the A.I. with some data, having the A.I. learn from the data, and making something using the A.I.



Train

Collect examples of what you want the computer to recognise

Learn & Test

Use the examples to train the computer to recognise images

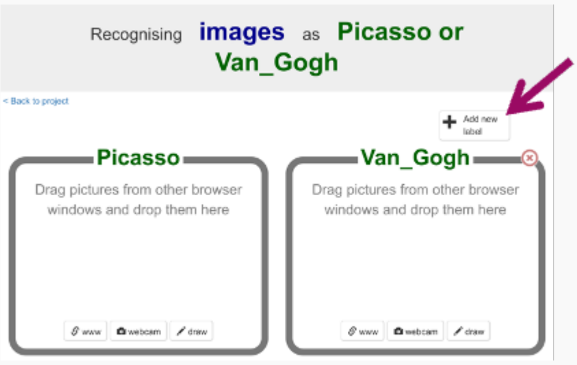
Make

Use the machine learning model you've trained to make a game or app, in Scratch, Python, or App Inventor

Training Your Project

To train your project you will have to add labels, these are the buckets in which you will store your data.

You should use at least two labels so that the A.I. has something to contrast.



Recognising images as Picasso or Van_Gogh

< Back to project
Add new label
Picasso
Drag pictures from other browser windows and drop them here
www
webcam
draw
Van_Gogh
Drag pictures from other browser windows and drop them here
www
webcam
draw

Many labels may be used, but the more options you have the more examples of each you'll need for the A.I. to differentiate.

Add Examples

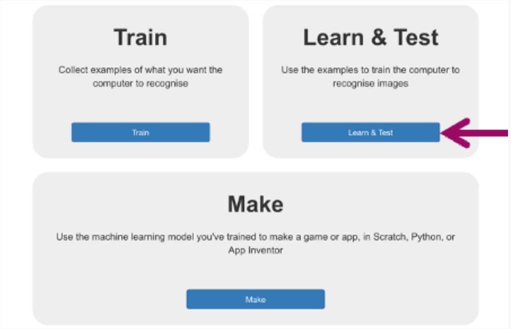
Now examples must be added to your labels!

- For text/number projects exaples can be typed or copy/pasted from elsewhere.
- For image projects images can be copied from the web (not uploaded), taken on webcam or drawn.
- For sound projects samples can be recorded with the microphone only.

Note: This only applies to training within MLfK, training from within Scratch or Python will allow uploads to take place.

At least 5 examples are required for each label (8 for sound). As a general rule the more data the better the model will perform. However, try to keep the examples approximately even across labels.

Train the A.I.



Train

Collect examples of what you want the computer to recognise

Learn & Test

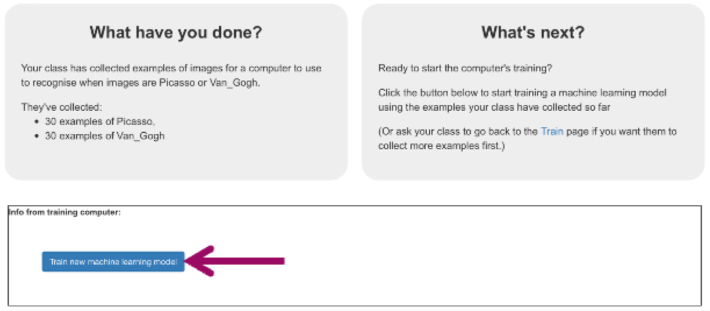
Use the examples to train the computer to recognise images

Make

Use the machine learning model you've trained to make a game or app, in Scratch, Python, or App Inventor

The A.I can be trained within MLfK by using the Learn & Test option.

This will inform you how many examples you've collected and whether you need more.



What have you done?

Your class has collected examples of images for a computer to use to recognise when images are Picasso or Van_Gogh.

They've collected:

- 30 examples of Picasso,
- 30 examples of Van_Gogh

What's next?

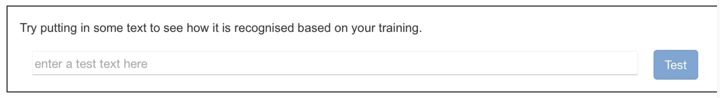
Ready to start the computer's training?

Click the button below to start training a machine learning model using the examples your class have collected so far

(Or ask your class to go back to the Train page if you want them to collect more examples first.)

Info from training computer:

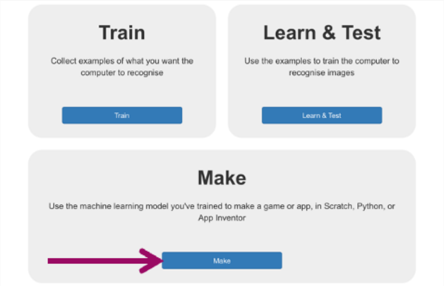
Click Train New Machine Learning Model to begin training.
This will take approximately a minute.



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

Testing can then be done from within MLfK.

Making a Program



Train

Collect examples of what you want the computer to recognise

Learn & Test

Use the examples to train the computer to recognise images

Make

Use the machine learning model you've trained to make a game or app, in Scratch, Python, or App Inventor

Once the A.I. has been trained we can choose to make something with it.

Click on Make and choose one of the options.

- Scratch 3 will open a version of Scratch in a seperate window (this cannot be done through the Scratch website as MLfK blocks are provided).
- Python will provide you with the relevant Python files and the API key for that particular project.

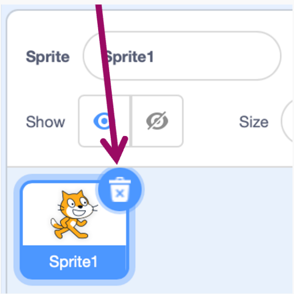
Recognising Artists with Scratch

This example will walk through how to make a program in Scratch that uses A.I. to recognise the paintings of different artists.

Many of the elements shown here are also applicable to creating machine learning programs that recognise sounds, text or numbers.

Note: For this program only the labels need to be created in MLfK. No data needs to be added to the project and no training needs to be done.

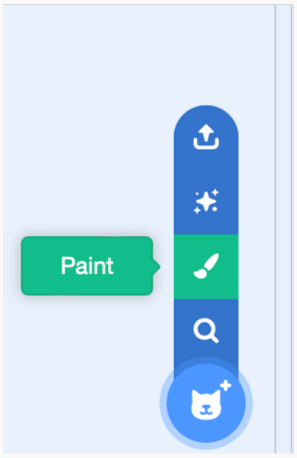
Creating Sprites


Sprite Sprite1

We will begin by deleting the default Scratch sprite.

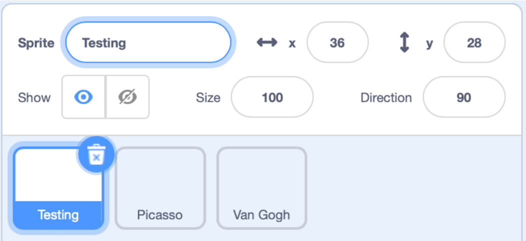
This is usually helpful as it allows learners to create a sprite that is relevent to the program they're creating.

When making new sprites we can choose to upload, paint or choose from premade sprites.

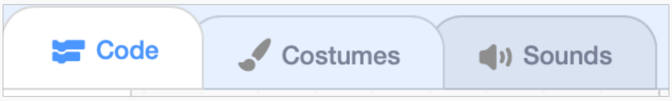


For this program we will paint three new sprites called **Testing**, **Picasso** and **Van Gogh**.

Note: we will not actually paint anything.


Sprite Testing x 36 y 28
Show Size 100 Direction 90

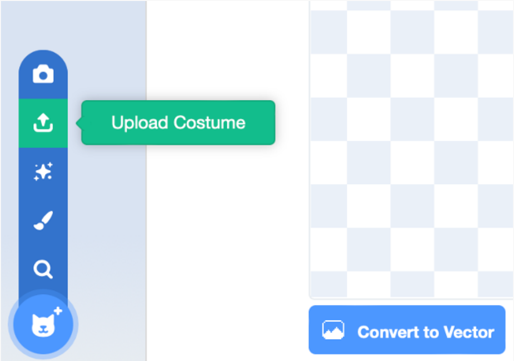
Uploading Costumes



In the Costumes tab we can upload costumes for our sprite.

For this program we will be uploading saved pictures of our artists.

We'll need training images for each artist in their own sprites, and then testing images from both artists in the testing sprite.

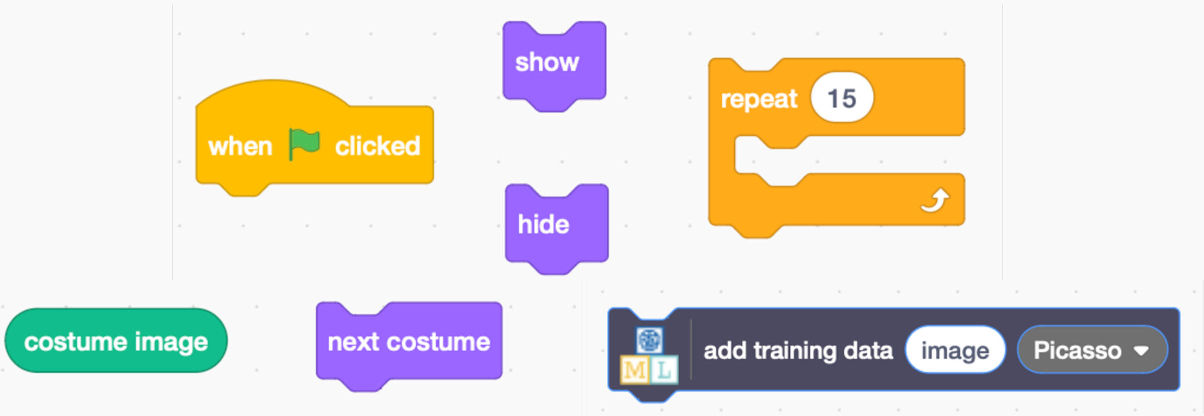


Picasso Sprite

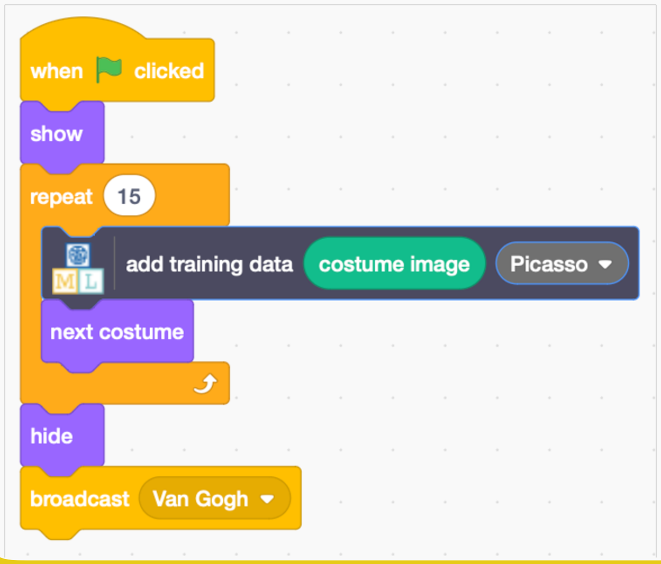
The code for this sprite is only required to add training data to MLfK.

In a project where data is added directly into the MLfK labels, this code is not required.

Here are the blocks:



Picasso Sprite Complete



```
when ⚑ clicked
show
repeat 15
    add training data  costume image  Picasso ▾
    next costume
hide
broadcast  Van Gogh ▾
```

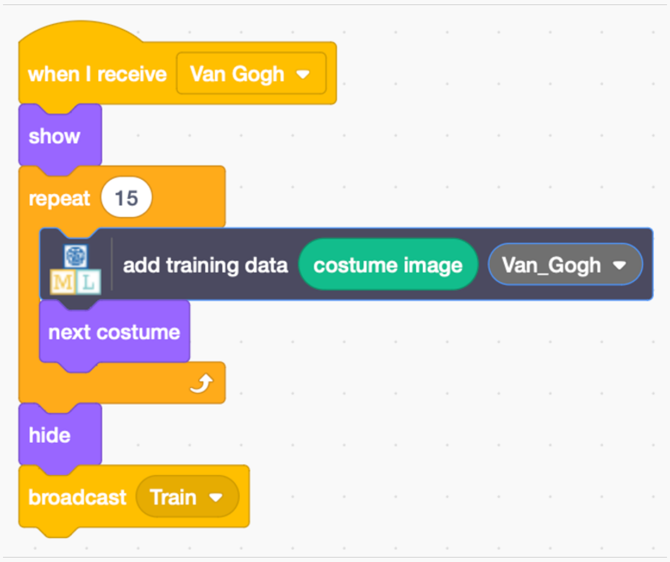
Here is the completed code for the Picasso sprite.

This will cycle through all 15 costumes and add them to the data set.

For a different number of costumes change the repeat value.

Van Gogh Sprite

The code for the Van Gogh sprite can be dragged and dropped in from Picasso for simplicity.

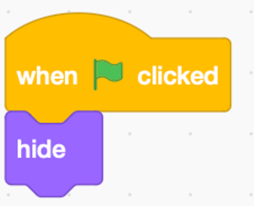


```
when I receive  Van Gogh ▾
show
repeat 15
    add training data  costume image  Van_Gogh ▾
    next costume
hide
broadcast  Train ▾
```

However there are a couple of changes required:

- The first block needs to receive the message from Picasso instead of starting on the Green Flag click.
- The add training data needs to be changed to Van Gogh.
- A new message needs to be broadcast to begining training the MLfK A.I.

Testing Sprite (Green Flag)



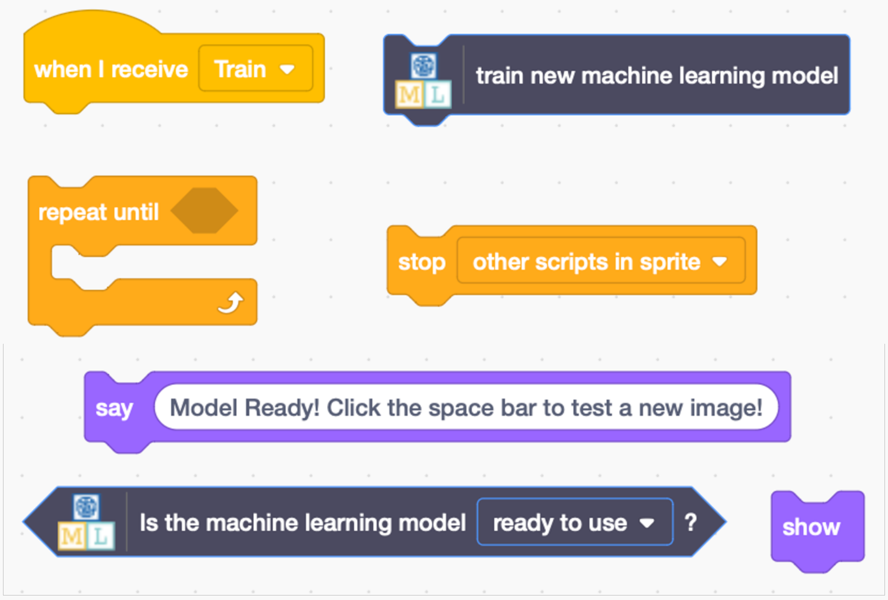
```
when 🏳 clicked
hide
```

This piece of code just ensures the training images can be seen while they are cycled through.

It is in no way necessary to the program.

Testing Sprite (Training)

These are the blocks that will train the MLfK model from within Scratch.

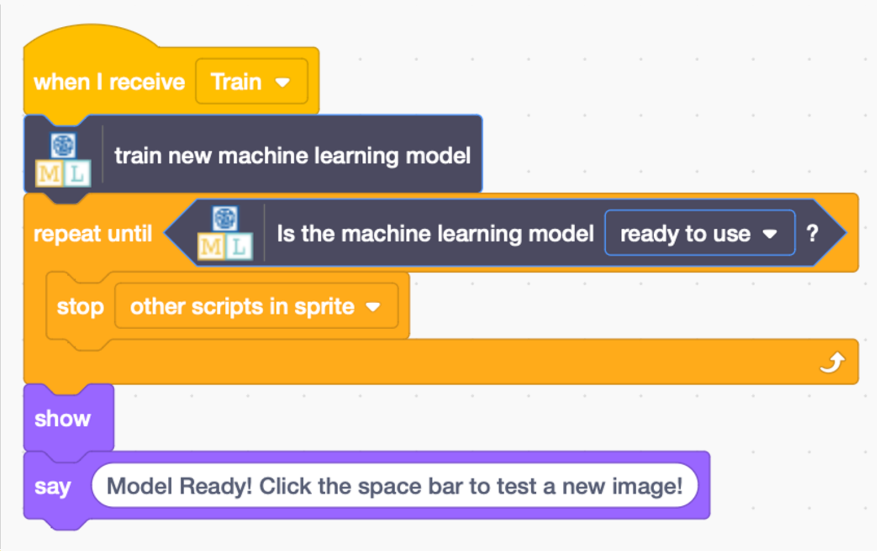


```
when I receive [Train ▾]
train new machine learning model
repeat until < >
stop [other scripts in sprite ▾]
say (Model Ready! Click the space bar to test a new image!)
<Is the machine learning model [ready to use ▾] ?>
show
```

Like the previous sprites these are not necessary as this can be done within the MLfK website. They have been used for this program as images cannot be uploaded to MLfK.

This could be used as an extension task.

Testing Sprite (Training) Complete



```
when I receive [Train ▾]
train new machine learning model
repeat until <Is the machine learning model [ready to use ▾] ?>
    stop [other scripts in sprite ▾]
end
show
say [Model Ready! Click the space bar to test a new image!]
```

This is the complete code required to train the MLfK model.

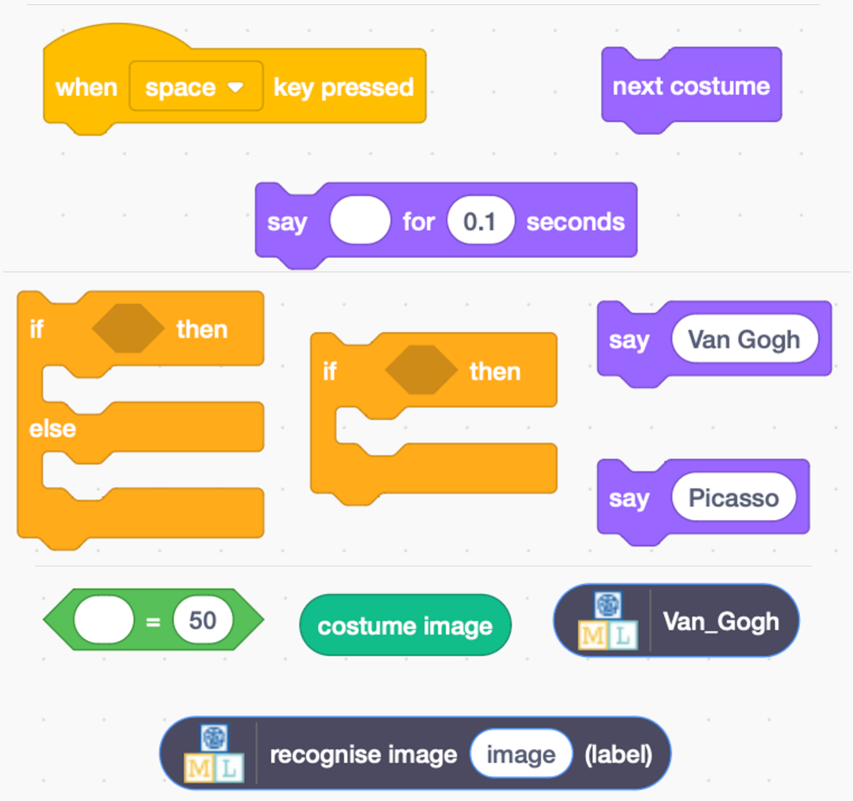
All other code is stopped until the model is trained.

Testing Sprite (Testing)

These are the blocks required to test your machine learning program within Scratch.

This is the only part of the program that is necessary to use the A.I.

Note: This can still be done within MLfK, however making use of the A.I. adds to the learning experience.

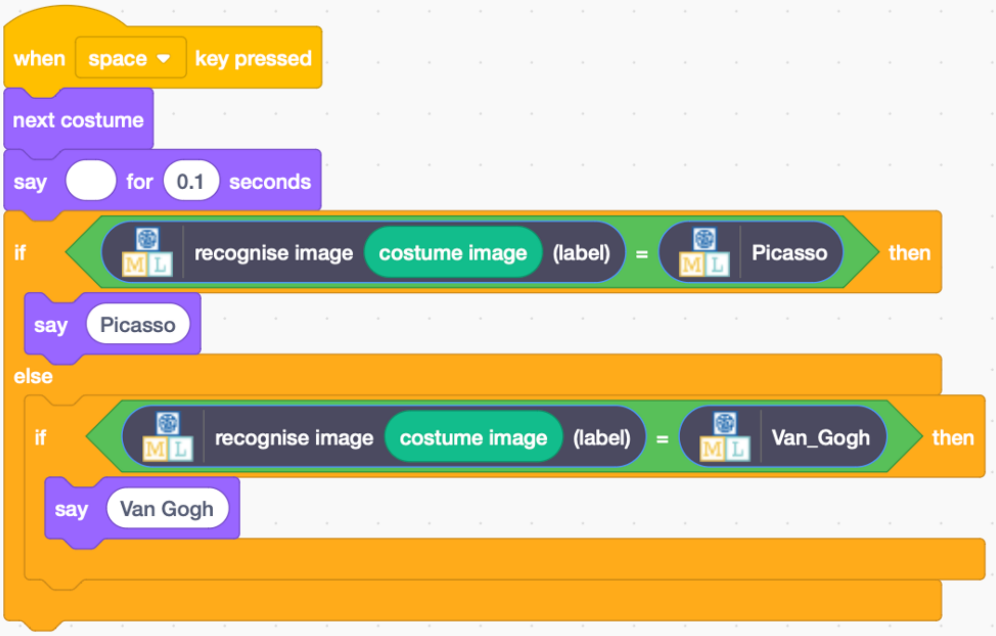


```
when [space ▾] key pressed
next costume
say [ ] for (0.1) seconds
if < > then
else
if < > then
say [Van Gogh]
say [Picasso]
< ( ) = (50) >
(costume image)
(Van_Gogh)
(recognise image (image) (label))
```

Testing Sprite (Testing) Complete

This is the completed code for the MLfK program to be tested within Scratch.

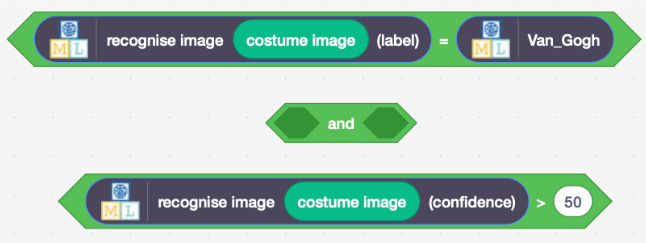
This code can be applied to many different machine learning models and expanded upon or altered for different scenarios.



```
when space ▾ key pressed
next costume
say ( ) for 0.1 seconds
if < recognise image (costume image) (label) = Picasso > then
    say Picasso
else
    if < recognise image (costume image) (label) = Van_Gogh > then
        say Van Gogh
```

Other Ideas

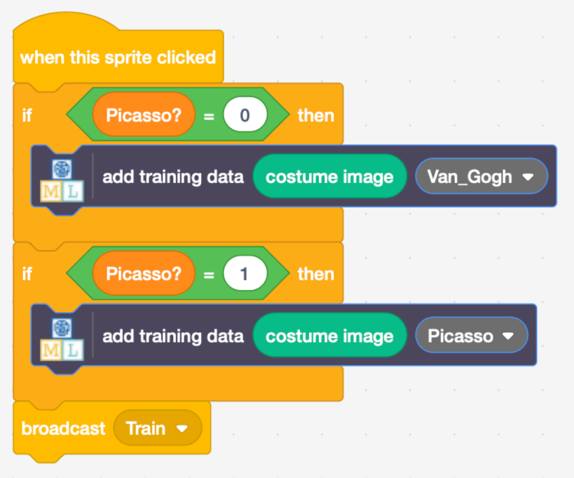
Check Confidence Levels



```
<(recognise image (costume image) (label)) = (Van_Gogh)>
and
<(recognise image (costume image) (confidence)) > (50)>
```

Using an **and** block we can check for whether an image matches one of the labels but also check that the A.I. is confident of the result to some extent.

Train Further With Test Images



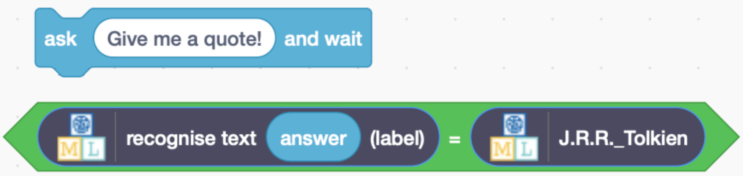
```
when this sprite clicked
if <(Picasso?) = (0)> then
    add training data (costume image) (Van_Gogh ▾)
if <(Picasso?) = (1)> then
    add training data (costume image) (Picasso ▾)
broadcast (Train ▾)
```

We can send our testing data back into the MLfK to labels to be trained on the A.I.

In this code, for each image we test, the A.I. adds that image to it's training data.

The **Picasso?** variable is set to 0 or 1 when the A.I. recognises the image as either a Van Gogh or a Picasso

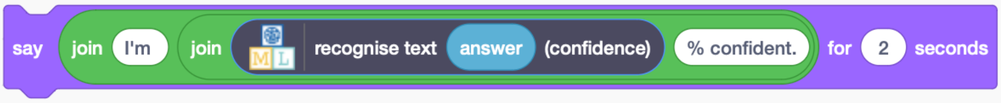
Using Text Inputs



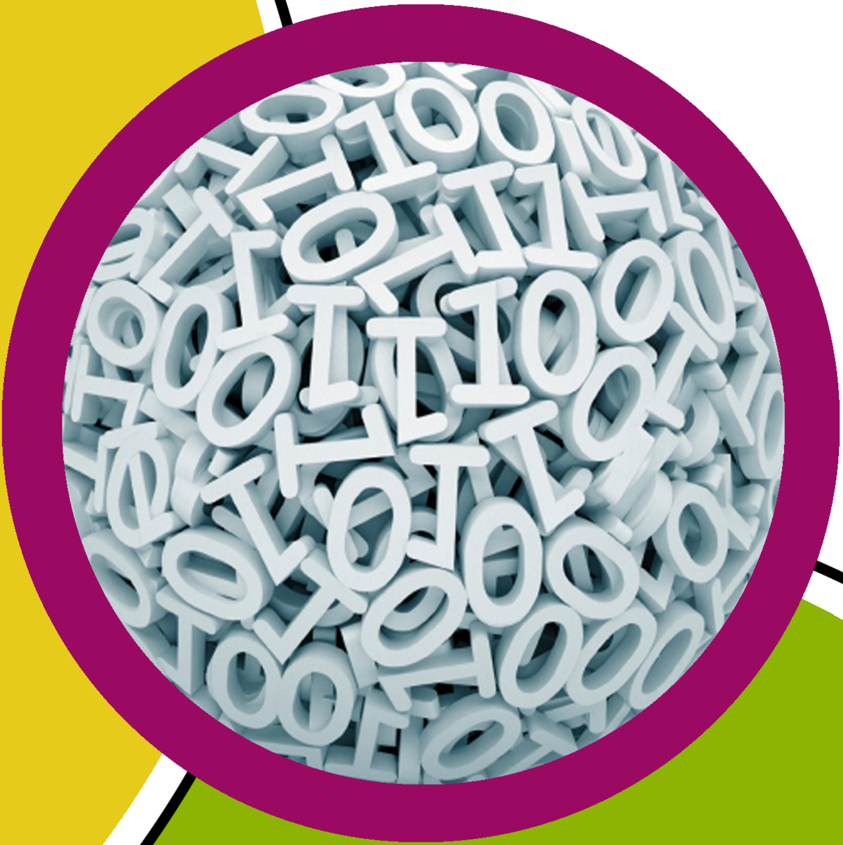
```
ask (Give me a quote!) and wait
<(recognise text (answer) (label)) = (J.R.R._Tolkien)>
```

To recognise text use the **ask** block to input text into Scratch, and the **answer** block to use it.

Or even have the A.I. say how confident it is at a result.



```
say (join (I'm) (join (recognise text (answer) (confidence)) (% confident.))) for (2) seconds
```



technocamps

@Technocamps

Find us on Facebook