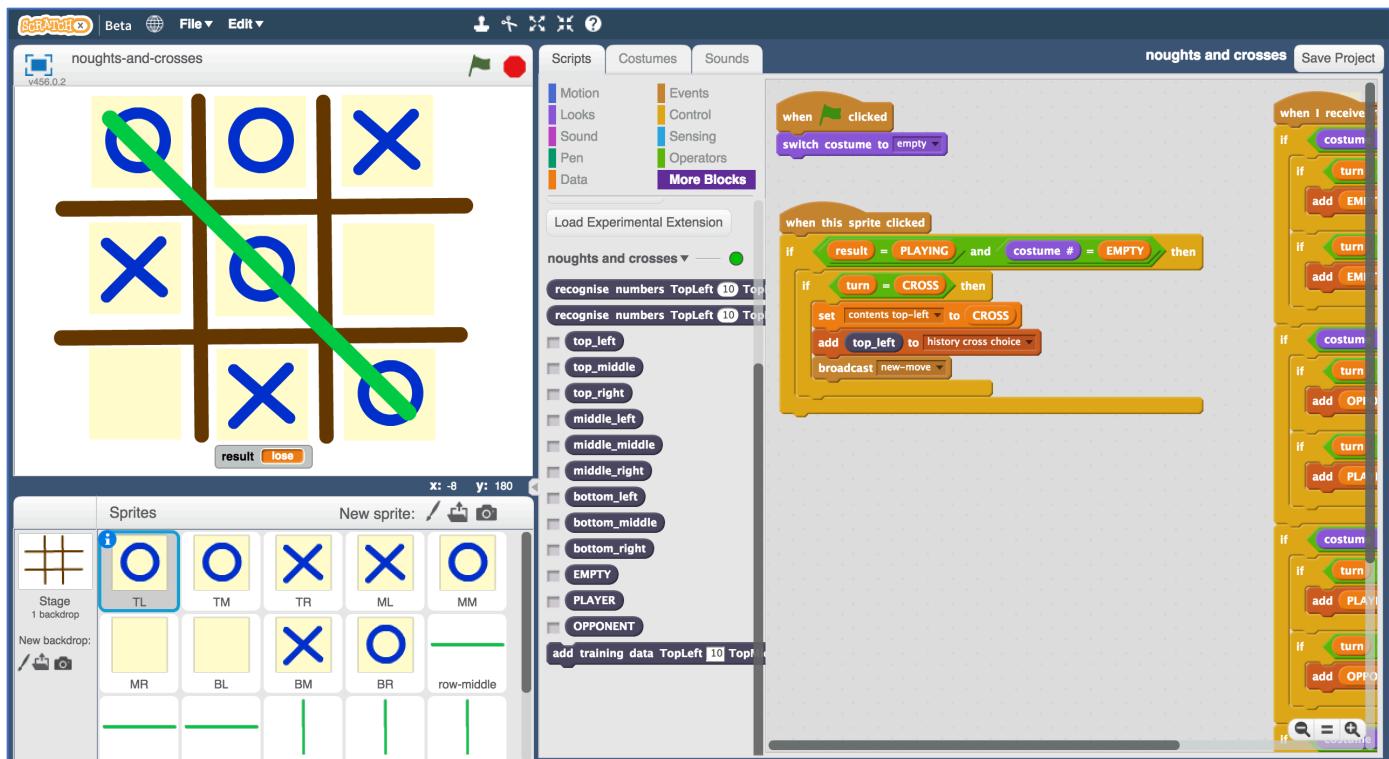


Noughts & Crosses

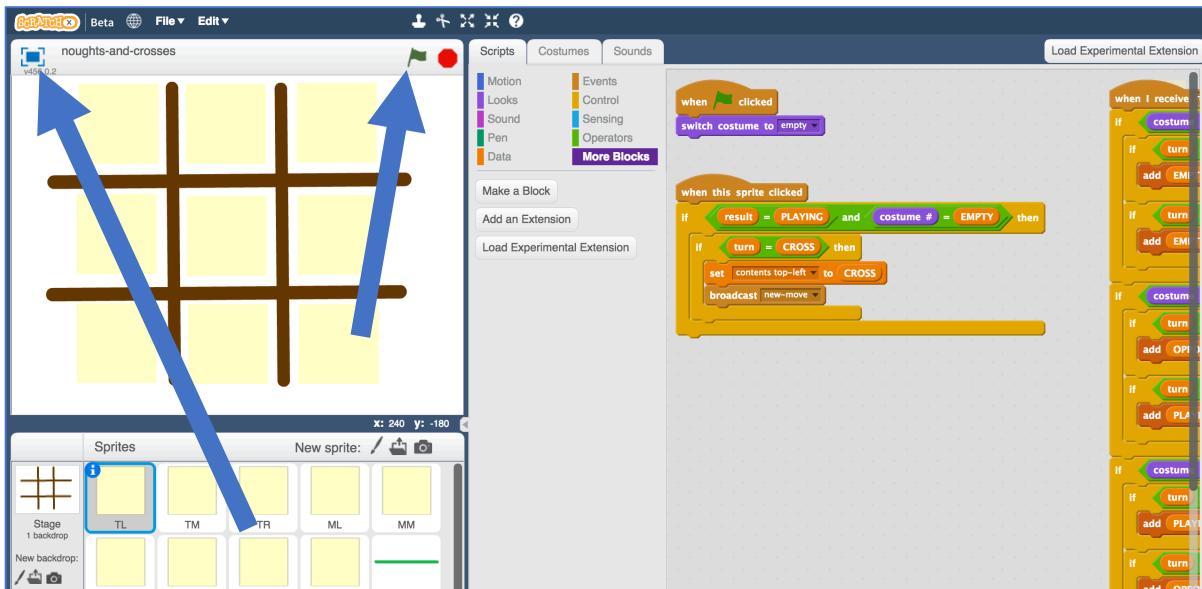
In this project you will create a noughts and crosses game in Scratch that is able to learn from how you play.

You won't give it instructions for how to play, or tell it what the objective or rules of the game are.

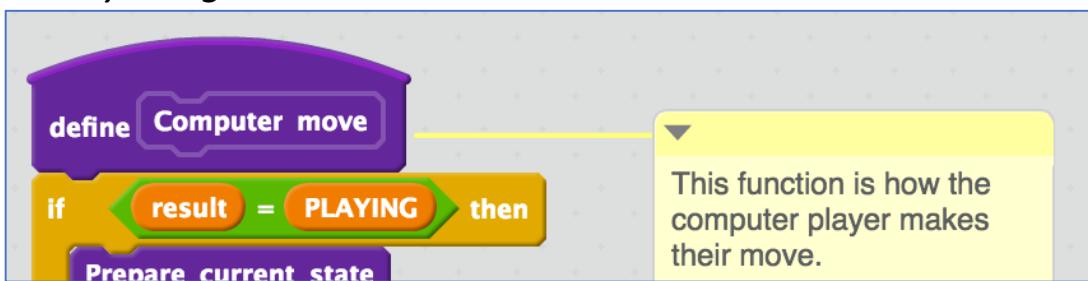
Instead, you'll show it examples of you playing the game. When it's seen enough examples to start trying to play for itself, you'll tell it when it beats you.



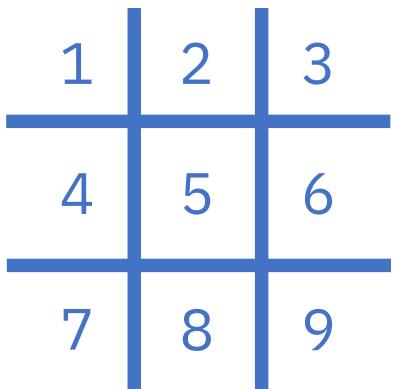
1. Go to <https://machinelearningforkids.co.uk/scratchx> in a browser.
2. Open the **noughts-and-crosses.sbx** starter file for this project.
Click File -> Load Project
If you haven't got this file, ask your teacher or group leader.
3. Click the **full-screen** button, and then click the Green Flag



4. Play a few games of noughts and crosses
You are CROSS (X), the computer is playing as NOUGHTS (O).
Click the green flag to start a new game, then click on the game board.
5. Can you see how the computer is choosing where to put its moves?
When you think you've worked out the computer's strategy, look at the Computer move block in the Stage
Were you right?

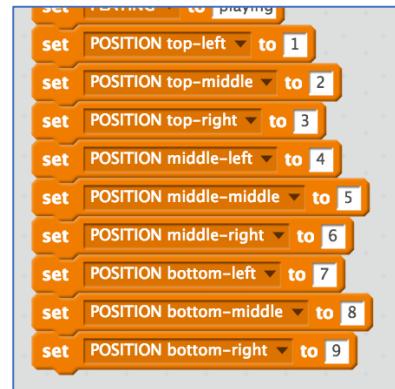


Representing noughts and crosses in Scratch



The positions of spaces on the noughts and crosses board are numbered from 1 to 9.

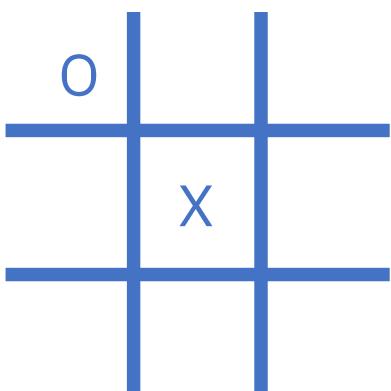
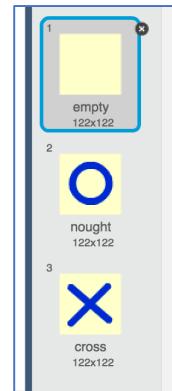
Data constants are used to make it easier to refer to them in scripts.



Empty = 1
O = 2
X = 3

An empty space is shown in costume 1.
A nought is shown in costume 2.
A cross is shown in costume 3.

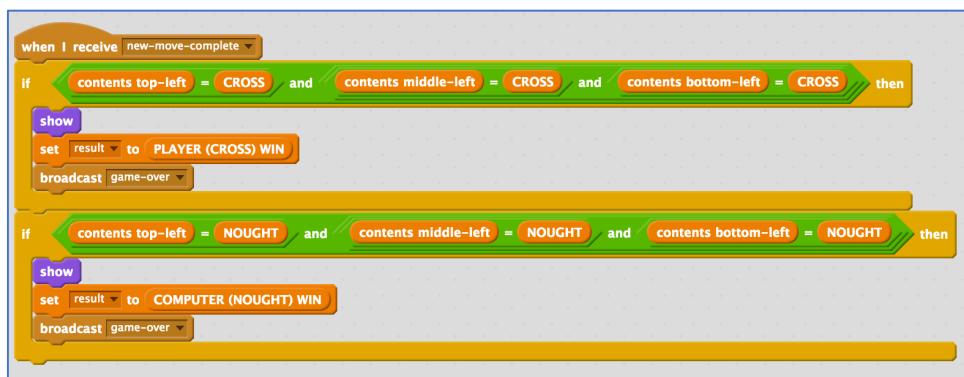
Data constants are used to make it easier to refer to these in scripts.



Variables are used to store the current state of the game.

For example, at this point:

contents top-left = 2
contents middle-middle = 3
contents bottom-right = 1



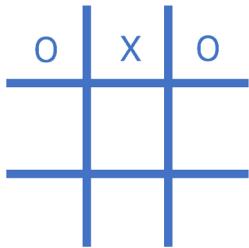
Each of the green row and column sprites check to see if someone has won.

This happens after every move.

What are you going to do?

You're going to train a computer to play noughts and crosses. You'll do this by showing it examples of how you play the game.

Imagine the board looks like this and it's X's turn.

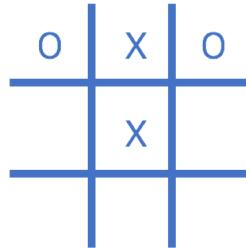


Imagine you decide to put your X in the centre space.

top-left	opponent
top-middle	player
top-right	opponent
middle-left	empty
middle-middle	empty
middle-right	empty
bottom-left	empty
bottom-middle	empty
bottom-right	empty

choice : middle-middle

Imagine the board looks like this and it's O's turn.



Imagine you decide to put your O in the bottom middle space.

top-left	player
top-middle	opponent
top-right	player
middle-left	empty
middle-middle	opponent
middle-right	empty
bottom-left	empty
bottom-middle	empty
bottom-right	empty

choice : bottom-middle

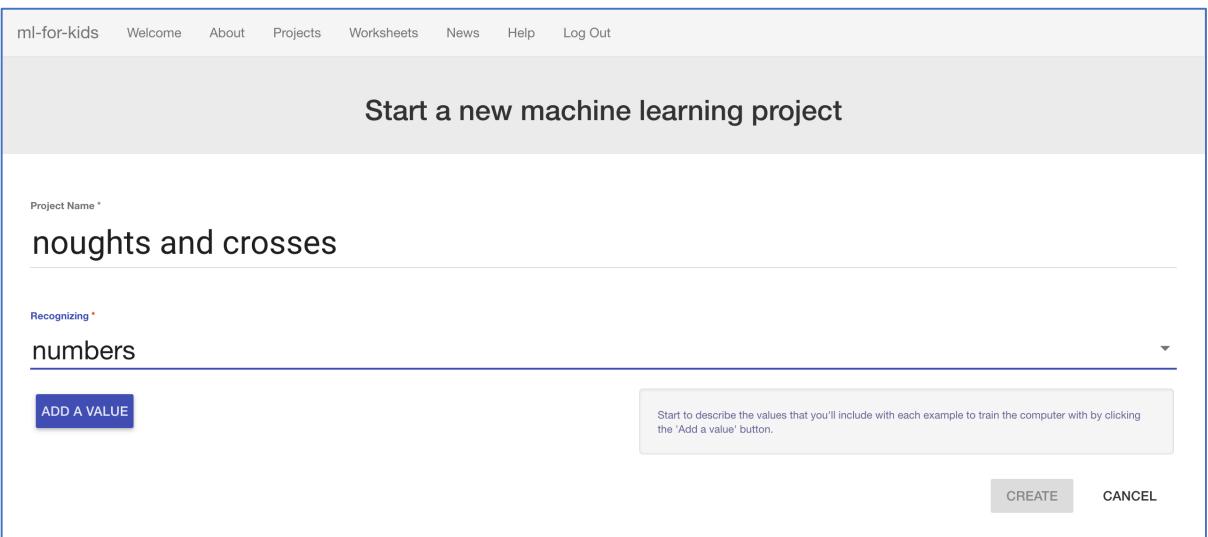
Using “opponent” and “player” instead of “nought” and “cross” means the computer can learn from both nought and cross moves.

You'll only use examples of moves from the player that wins the game.

If you (X) win, use your moves as examples to train the computer.
If the computer (O) wins, use the computer's moves to train with.

These **examples of moves that lead to winning** will teach the computer how to play to win!

- 6.** Close the Scratch window.
- 7.** Go to <https://machinelearningforkids.co.uk/> in a web browser
- 8.** Click on “**Get started**”
- 9.** 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.
- 10.** Click on “**Projects**” on the top menu bar
- 11.** Click on the “**+ Add a new project**” button.
- 12.** Name your project “noughts and crosses” and set it to learn how to recognise “**numbers**”



The screenshot shows a web-based form for creating a new machine learning project. At the top, there is a navigation bar with links: ml-for-kids, Welcome, About, Projects, Worksheets, News, Help, and Log Out. Below the navigation bar, the main title is "Start a new machine learning project". The first input field is labeled "Project Name *" and contains the text "noughts and crosses". The second input field is labeled "Recognizing *" and contains the text "numbers". Below these fields is a button labeled "ADD A VALUE". To the right of the input fields is a tooltip box containing the text: "Start to describe the values that you'll include with each example to train the computer with by clicking the 'Add a value' button." At the bottom right of the form are two buttons: "CREATE" and "CANCEL".

- 13.** Click “Add a value” and name a value “TopLeft” and make it “multiple choice”.

The screenshot shows a user interface for defining a value. On the left, there is a text input field labeled "Value 1 *" containing the text "TopLeft". To its right is a dropdown menu labeled "Type of value *" with the option "multiple-choice" selected. Below this, a sub-section titled "Choices:" contains a button labeled "add a choice". To the right of the main input area is a tooltip-like box with the following text:
If TopLeft can be described as numbers, choose "number".
If it can be described as choosing from a few options, choose "multiple-choice".

ADD ANOTHER VALUE

- 14.** Type “EMPTY” into the “add a choice” box and press Enter
Type “PLAYER” into the “add a choice” box and press Enter
Type “OPPONENT” into the “add a choice” box and press Enter
These are the possible contents for the top-left box in the noughts and crosses board. It can be empty, or it can have the player’s own shape (e.g. cross) in it, or it can have the opponent’s shape in it (e.g. cross).

The screenshot shows the same form interface as before, but now with three choices listed under "Choices": "EMPTY", "PLAYER", and "OPPONENT". Each choice has a small red "X" icon to its right. Below the choices is a button labeled "add a choice". To the right of the main input area is a tooltip-like box with the following text:
Type in another choice to use in your multiple-choice list, then press Enter.

ADD ANOTHER VALUE

- 15.** Click “Add another value” again and repeat to add values for the other eight positions on the board
Each example is the state of the board before a move that led to a win. TopMiddle, TopRight, MiddleLeft, MiddleMiddle, MiddleRight, BottomLeft, BottomMiddle, BottomRight

It's very important that you spell "EMPTY", "PLAYER" and "OPPONENT" in the same way for all nine positions.

Project Name: noughts and crosses

Recognising * numbers

Value 1 * TopLeft	Type of value * multiple-choice	X	Choices: EMPTY ⚡ PLAYER ⚡ OPPONENT ⚡
Value 2 * TopMiddle	Type of value * multiple-choice	X	Choices: EMPTY ⚡ PLAYER ⚡ OPPONENT ⚡
Value 3 * TopRight	Type of value * multiple-choice	X	Choices: EMPTY ⚡ PLAYER ⚡ OPPONENT ⚡
Value 4 * MiddleLeft	Type of value * multiple-choice	X	Choices: EMPTY ⚡ PLAYER ⚡ OPPONENT ⚡
Value 5 * MiddleMiddle	Type of value * multiple-choice	X	Choices: EMPTY ⚡ PLAYER ⚡ OPPONENT ⚡
Value 6 * MiddleRight	Type of value * multiple-choice	X	Choices: EMPTY ⚡ PLAYER ⚡ OPPONENT ⚡
Value 7 * BottomLeft	Type of value * multiple-choice	X	Choices: EMPTY ⚡ PLAYER ⚡ OPPONENT ⚡
Value 8 * BottomMiddle	Type of value * multiple-choice	X	Choices: EMPTY ⚡ PLAYER ⚡ OPPONENT ⚡
Value 9 * BottomRight	Type of value * multiple-choice	X	Choices: EMPTY ⚡ PLAYER ⚡ OPPONENT ⚡

ADD ANOTHER VALUE **CREATE** **CANCEL**

16. Click **Create**. You should see “noughts and crosses” show up in the list of your projects. Click on it.

Your machine learning projects

+ Add a new project

newspapers
Recognising **text** as **Daily_Express, Daily_Mirror or 2 other classes**

noughts and crosses
Recognising **numbers**

17. Click the “Train” button

ml-for-kids Welcome About Projects Worksheets News Help Log Out

"noughts and crosses"

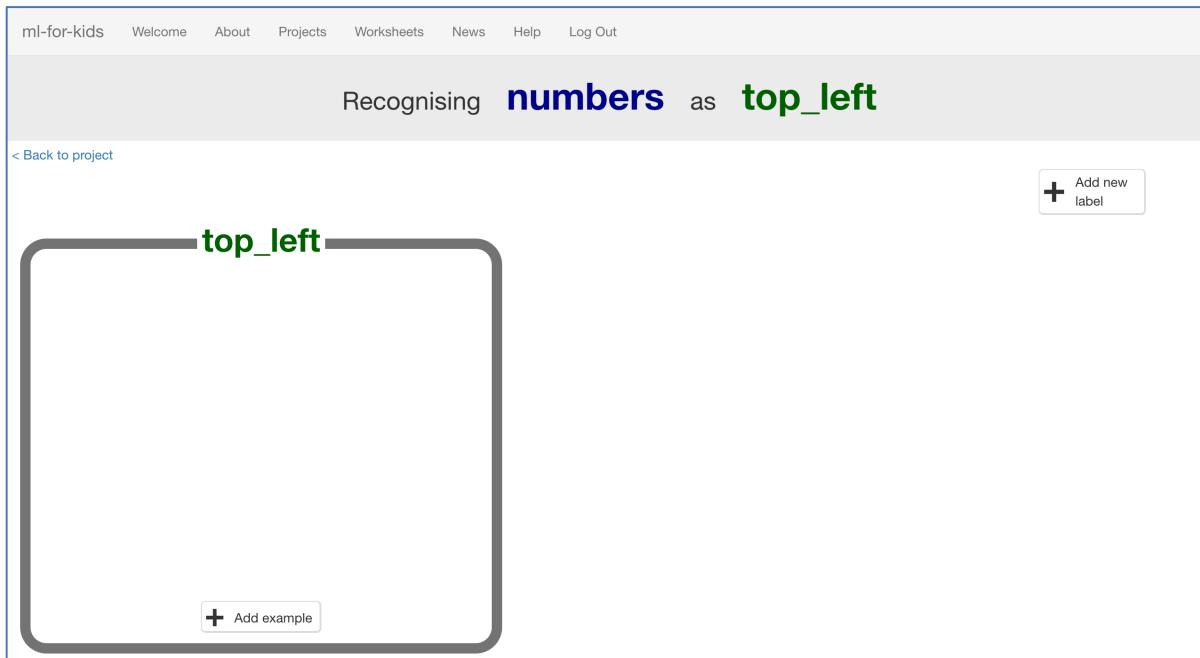
Train
Collect examples of what you want the computer to recognise.
Train

Learn & Test
Use the examples to train the computer to recognise numbers.
Learn & Test

Scratch
Use the machine learning model you've trained to make a game in Scratch.
Scratch

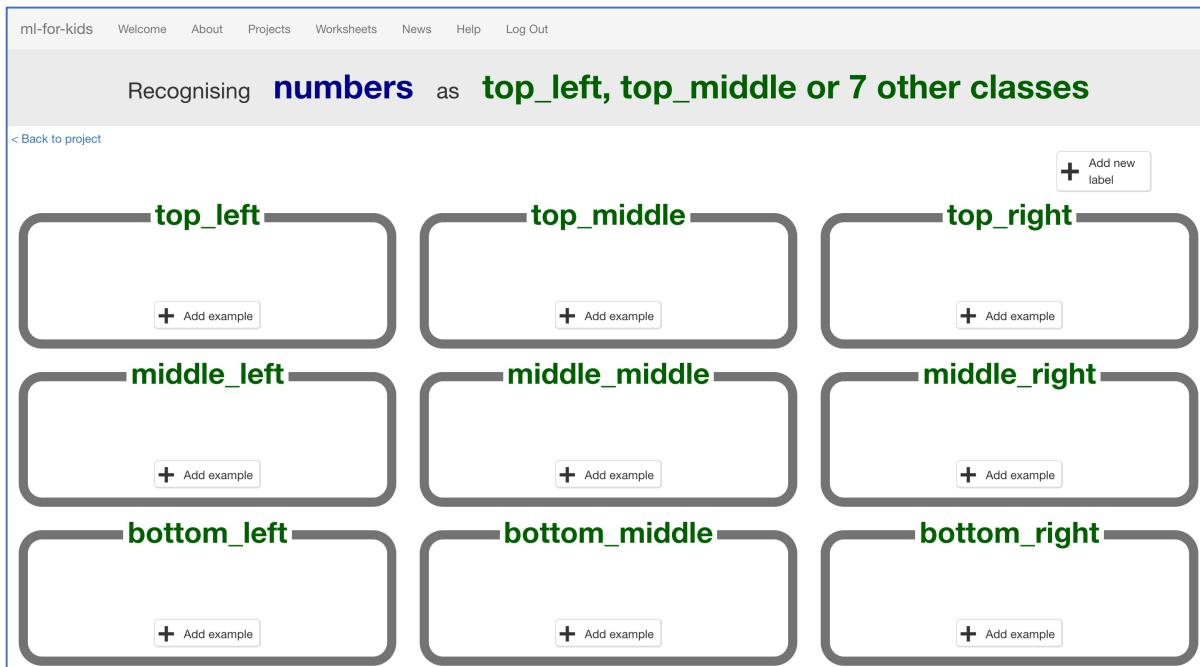
18. Click “+ Add new label” and create a label called “top left”

Examples of making a move in the top-left box (in games that eventually ends in a win) will go in this bucket.



19. Click “+ Add new label” again and create labels for the other eight spaces on the board.

“top middle”, “top right”, “middle left”, “middle middle”, “middle right”, “bottom left”, “bottom middle”, “bottom right”



20. Click the “< Back to project” link then click **Scratch**

21. Click the **Open in Scratch** button

It will warn you that you haven't trained the computer yet – but that's okay, as you'll use Scratch to collect the training examples.

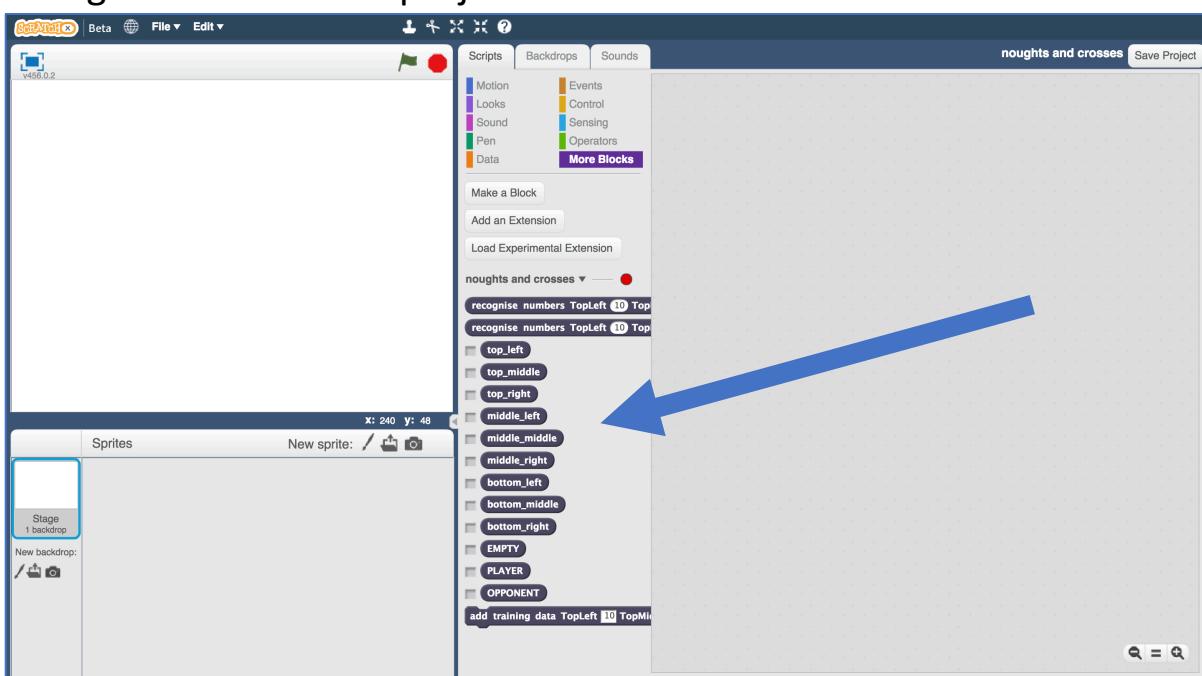
Click the “go straight into Scratch now” link.

The screenshot shows a web-based interface for a machine learning project. On the left, a text area explains that the project will add specific blocks to the 'More Blocks' tab in Scratch. It lists several 'recognise numbers' blocks for positions like TopLeft, TopMiddle, etc., and notes that these represent labels created in the project. Below this, a script example is shown:

```
if [recognise numbers TopLeft v] then
  say [I think that was top_left!]
```

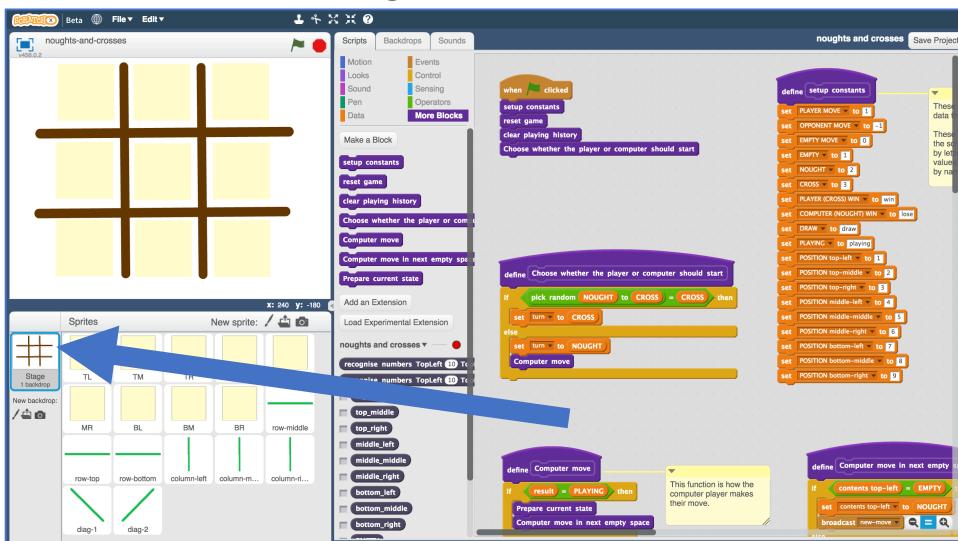
On the right, a preview window shows the Scratch interface with the 'More Blocks' tab selected. A green circle next to the project name indicates the model is trained. A legend explains the colored circles: green means trained, yellow means not finished, and red means an error. The Scratch stage shows a blue arrow pointing towards the bottom right.

22. You should see new blocks in the “More blocks” section from your “noughts and crosses” project.



23. Open the “noughts-and-crosses.sbx” starter project file again.
Click **File -> Load Project**

24. Click on the “Stage”



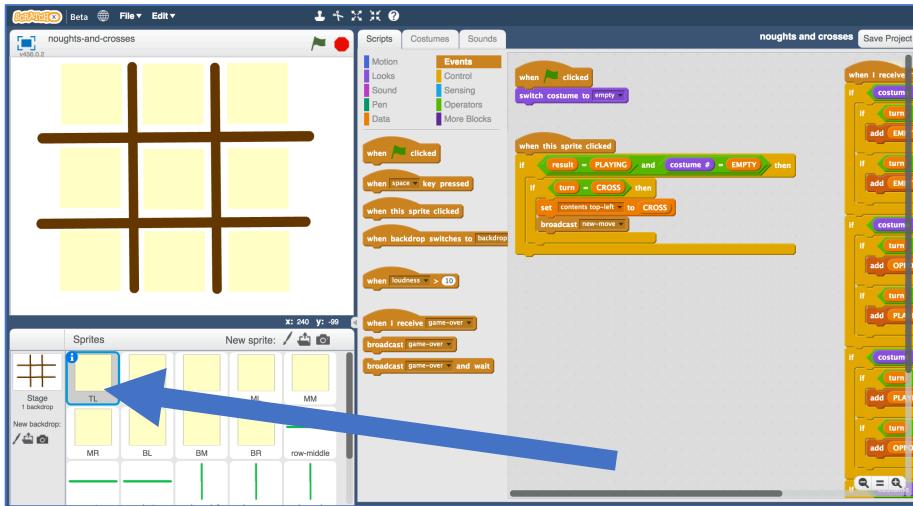
25. Modify the custom block “**setup constants**” to use the multiple choice values you created

Drag the “*EMPTY*”, “*PLAYER*” and “*OPPONENT*” blocks from your “noughts and crosses” project into the top three spaces in the script like in the picture below.



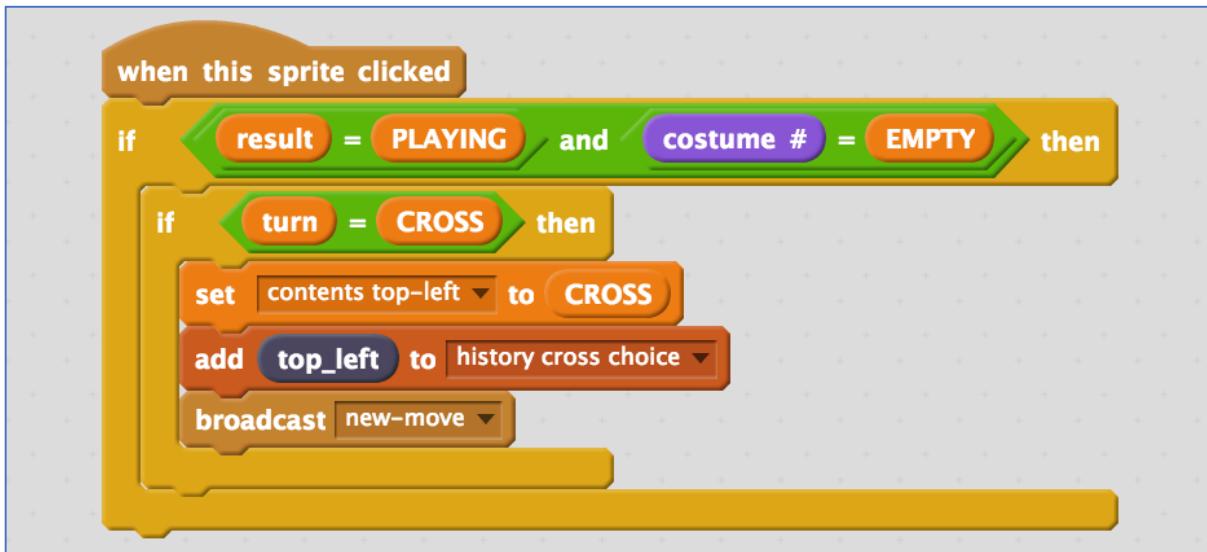
26. Click the TL (top-left space) sprite

You'll update the script to store when you click on this space for your move.



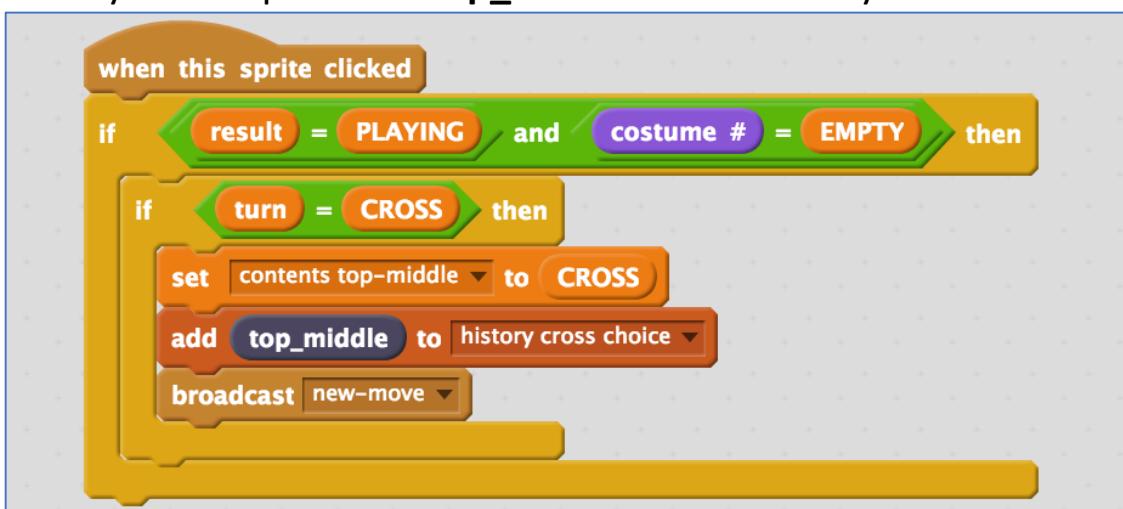
27. Modify the “When this sprite clicked” script to add `top_left` to the history of cross choices.

You just need to add a single block to get the script to look like this:



28. Click the TM (top-middle) sprite.

Modify the script to add **top_middle** to the history of cross choices.



29. Repeat for the other seven board sprites

Add *top_right* to the *TR* sprite. Add *middle_left* to the *ML* sprite.

Add *middle_middle* to the *MM* sprite. Add *middle_right* to the *MR* sprite.

Add *bottom_left* to the *BL* sprite. Add *bottom_middle* to the *BM* sprite.

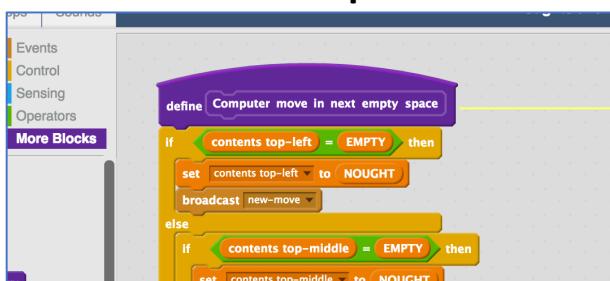
Add *bottom_right* to the *BR* sprite.



30. Click on the “Stage” again

You'll update the script to store when the computer makes its move

31. Find the “Computer move in next empty space” script



32. Modify the script to add each of the computer's moves to the history of nought choices.

You only need to add the add CHOICE to 'history nought choice' blocks



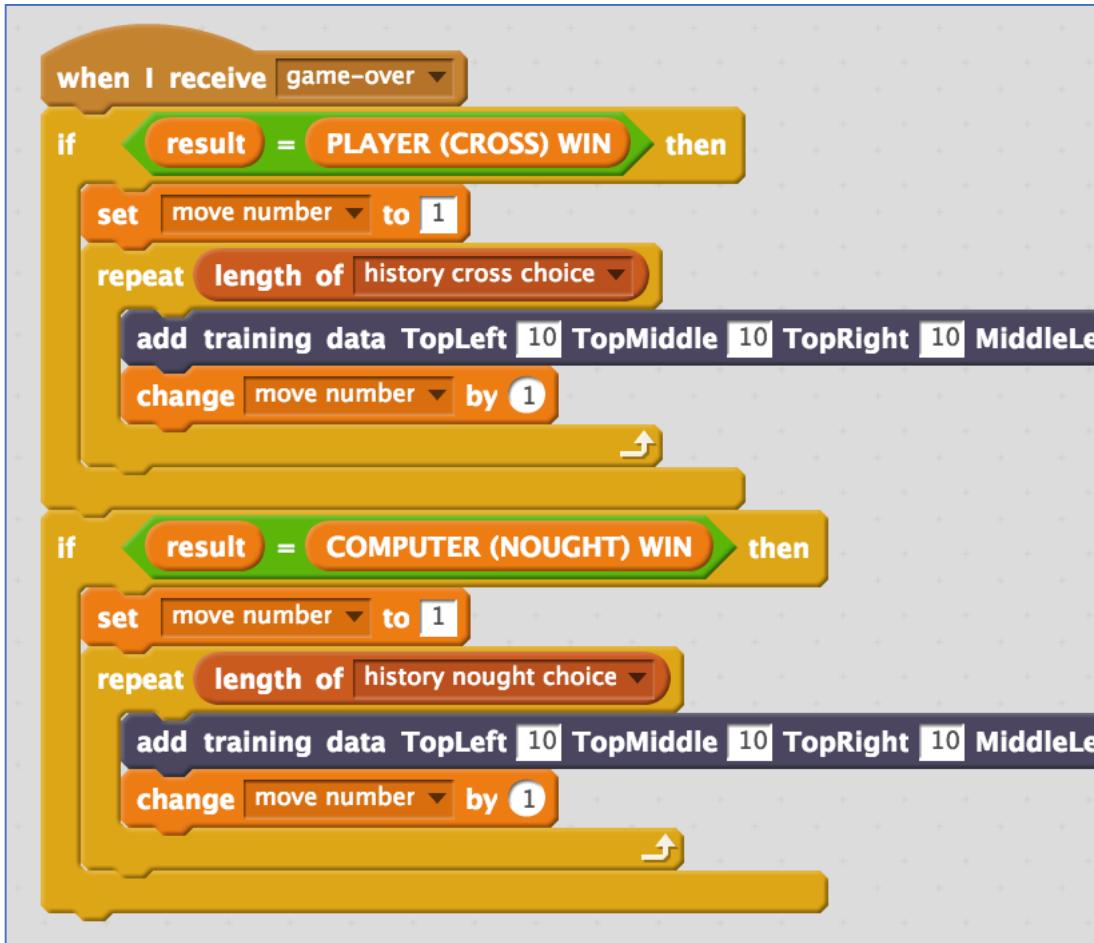
```

broadcast new-move ▾
else
  if contents middle-left = EMPTY then
    set contents middle-left ▾ to NOUGHT
    add middle_left to history nought choice ▾
    broadcast new-move ▾
  else
    if contents middle-middle = EMPTY then
      set contents middle-middle ▾ to NOUGHT
      add middle_middle to history nought choice ▾
      broadcast new-move ▾
    else
      if contents middle-right = EMPTY then
        set contents middle-right ▾ to NOUGHT
        add middle_right to history nought choice ▾
        broadcast new-move ▾
      else
        if contents bottom-left = EMPTY then
          set contents bottom-left ▾ to NOUGHT
          add bottom_left to history nought choice ▾
          broadcast new-move ▾
        else
          if contents bottom-middle = EMPTY then
            set contents bottom-middle ▾ to NOUGHT
            add bottom_middle to history nought choice ▾
            broadcast new-move ▾
          else
            if contents bottom-right = EMPTY then
              set contents bottom-right ▾ to NOUGHT
              add bottom_right to history nought choice ▾
              broadcast new-move ▾
            else
              set result ▾ to DRAW
              broadcast game-over ▾
The space De dra

```

33. Create the following script (still in the Stage)

This will add all of the history of moves made by the winning player to the training data that you will use to train the computer.



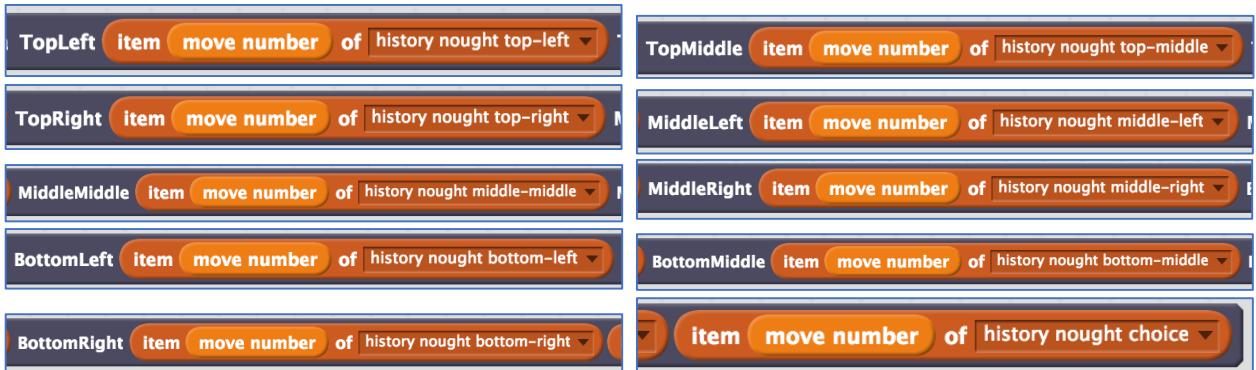
34. Add the following items to the first “add training data” block

If the game ends with the player (CROSS) winning, all the moves you want to add to the training data should be from the cross histories.

And you end with the choice of move that was made.

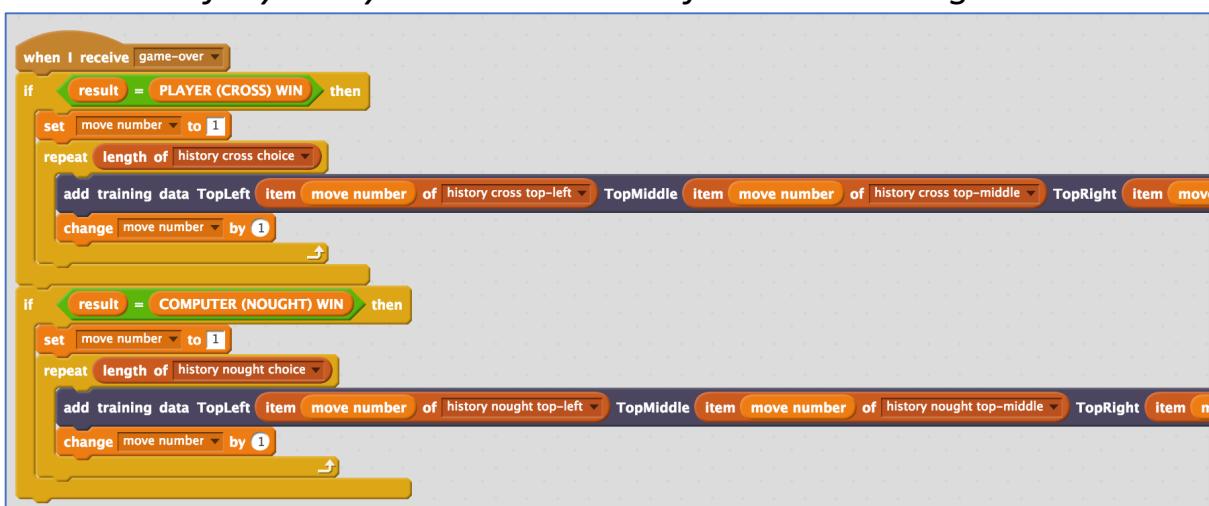


35. Add the following items to the **second** “add training data” block
*If the game ends with the computer (NOUGHT) winning, all the moves you want to add to the training data should be from the **nought** histories. And you end with the choice of move that was made.*



36. The final script should look like this

*Double-check that the top “add training data” row is all from “history nought” lists, and matches the board space. (e.g. TopLeft with top-left)
 Double-check that the bottom “add training data” row is all from “history cross” lists, and matches the board space. (e.g. TopLeft with top-left)
 Do this carefully – any mistakes will confuse the training.*



37. Save your project
*Click **File -> Save Project***

38. Play a few games

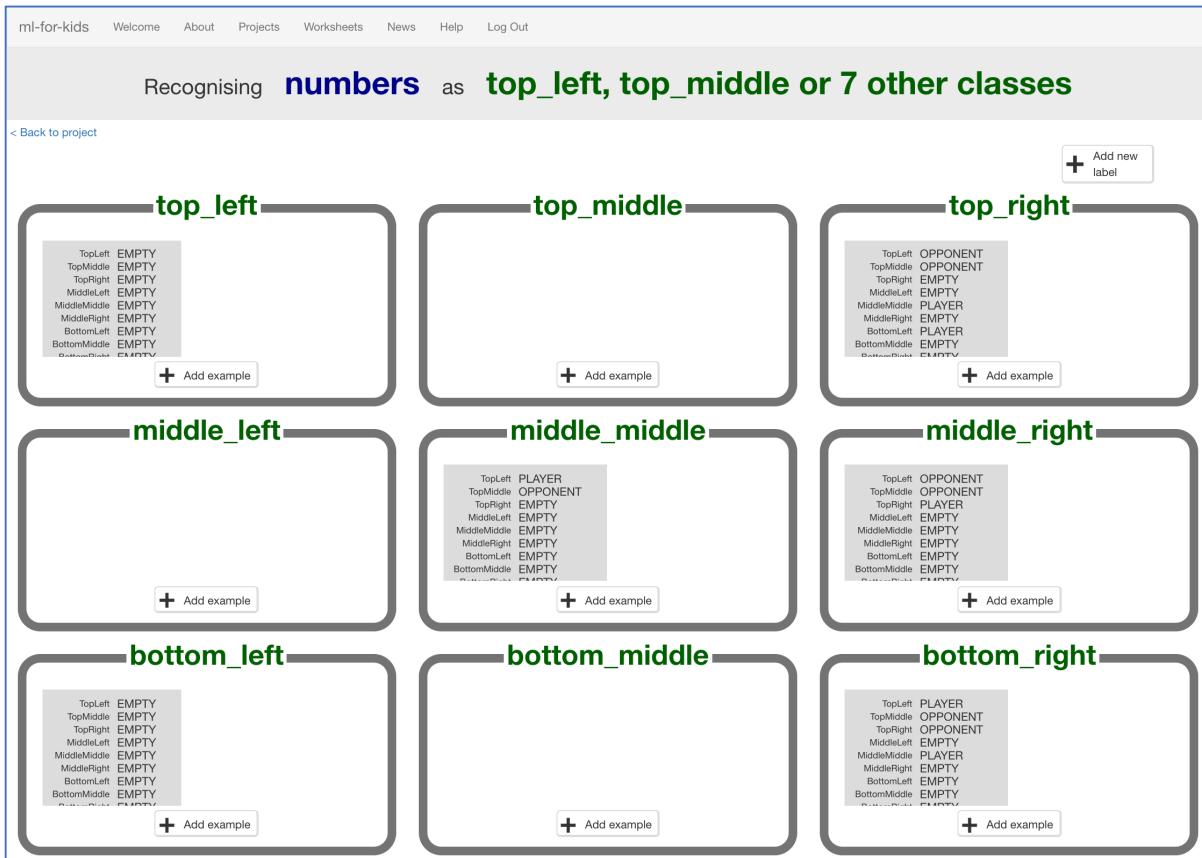
*Click on the **Green Flag** as you did before.*

It is better to play in full-screen mode to avoid accidentally moving sprites.

39. Go back to the training page

Leaving the Scratch window open, go back to the training tool window.

*Click the “**< Back to project**” link and then click “**Train**”*



40. Look at your training so far

Each item is the state of the board at the time you or the computer made a move (in a game that you or the computer won).

The bucket that the item is in is the move that you made.

If there are any empty buckets – like top_middle, middle_left and bottom_middle in the screenshot before – that means you haven’t made a move in that space in a game that you won yet.

- 41.** Click the “< Back to project” link. Click the “Learn & Test” button.
- 42.** If there is a “Train new machine learning model” button you can go to step **47**. Otherwise, carry on to step **43**.
- 43.** You don't have enough examples to train the computer yet
For the computer to know when it is a good idea to choose any space on the board, you need at least 5 examples of where you chose that space and ended up winning.
This page shows you how many examples you have so far. Look to see which one(s) you need more examples for.

ml-for-kids Welcome About Projects Worksheets News Help Log Out

Machine learning models

< Back to project

What have you done?

You've collected examples of numbers for a computer to use to recognise when numbers are top_left, top_middle or 7 other classes.

You've collected:

- 1 example of bottom_left,
- 2 examples of bottom_right,
- 2 examples of middle_middle,
- 1 example of middle_right,
- 1 example of top_left,
- 2 examples of top_right

What's next?

Keep going!

Go back to the [Train](#) page and collect more examples for each of the labels.

The more you can get, the better it should learn, but you need at least five examples of each as an absolute minimum.

- 44.** Leave the “Learn & Test” window open.
Go back to the **Scratch** window.

- 45.** Play more games.
Try starting from a different position each time to get a variety of examples.
Try starting from positions that you know you need more examples of.

46. When you think you've got at least 5 examples of each space, go back to the “Learn & Test” window and **refresh** the page.
If there is still no “Train new machine learning model” button, you need to go back to step 44 and try again.

The screenshot shows a web application interface for machine learning. At the top, there is a navigation bar with links: ml-for-kids, Welcome, About, Projects, Worksheets, News, Help, and Log Out. Below the navigation bar, the title "Machine learning models" is displayed. Underneath the title, there is a link "< Back to project". The main content area is divided into two sections: "What have you done?" on the left and "What's next?" on the right. The "What have you done?" section contains text about collecting examples of numbers for a computer to recognise when numbers are top_left, top_middle or 7 other classes. It lists the collected examples: 6 examples of bottom_left, 5 examples of bottom_middle, 8 examples of bottom_right, 5 examples of middle_left, 12 examples of middle_middle, 8 examples of middle_right, 5 examples of top_left, 5 examples of top_middle, and 9 examples of top_right. The "What's next?" section asks if the user is ready to start the computer's training. It provides instructions to click the "Train new machine learning model" button to start training a machine learning model using the examples collected so far. It also offers an alternative option to go back to the Train page if the user wants to collect more examples first. At the bottom of the page, there is a box labeled "Info from training computer:" which contains the text "Train new machine learning model".

47. Click the “Train new machine learning model” button at the bottom of the page.

What have you done so far?

You’re teaching a computer to play noughts and crosses.

So far, you’ve updated a Scratch noughts and crosses game so that it can collect examples of how you play and add them to a set of examples. And you’ve used those examples to train a machine learning “model”.

The next step is to use that model to let the computer decide what move to make – instead of just going for the next empty space every time.

48. Close the Scratch window.

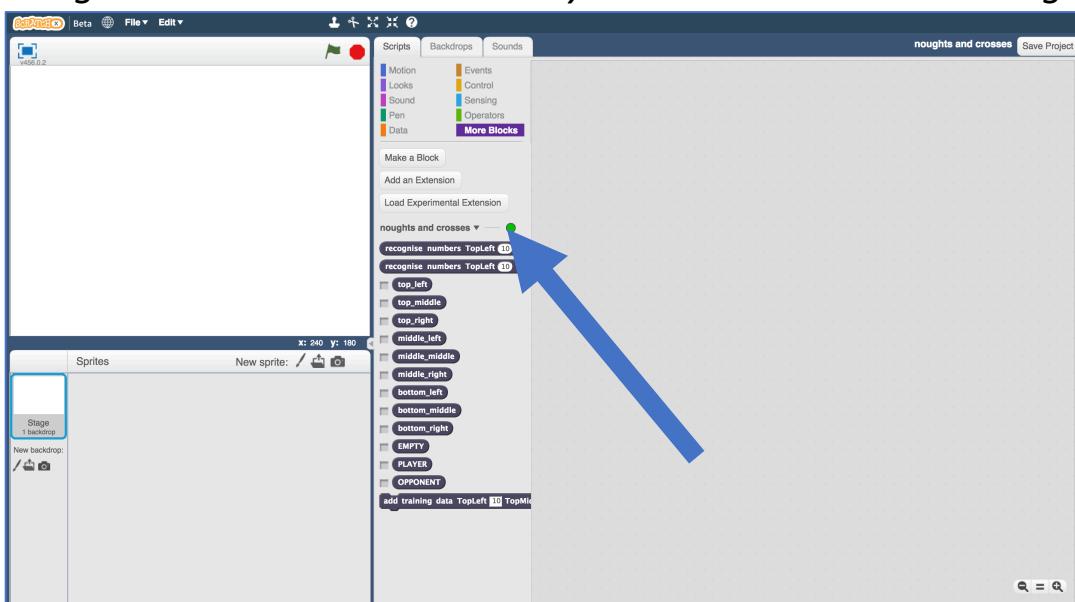
Make sure you save your project first!

49. In the training page, click the “< Back to project” link.

Then click the **Scratch** button.

50. Click the **Open in Scratch** button.

The green circle means this time you have a machine learning model.



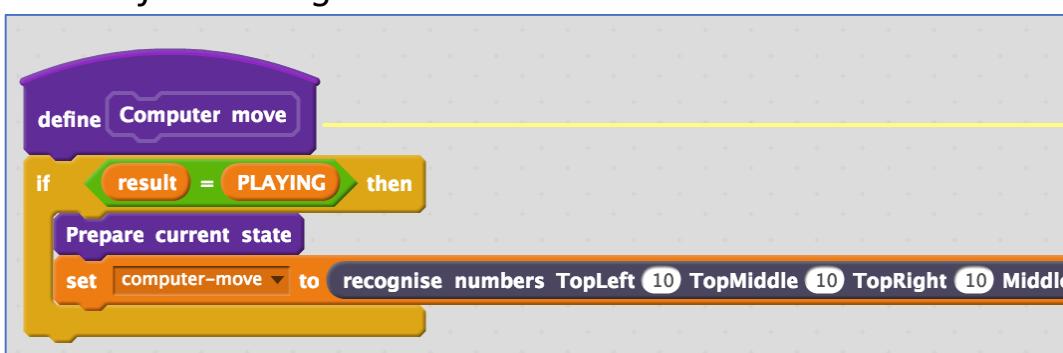
51. Open your saved project

*Click **File** -> **Load Project***

52. Click on the **Stage** and find the **Computer move** script

53. Modify the **Computer move** script so that it looks like this

Use the first “recognise numbers” block that ends with “label”



54. Add the current state for each space on the board to the “recognise numbers” block

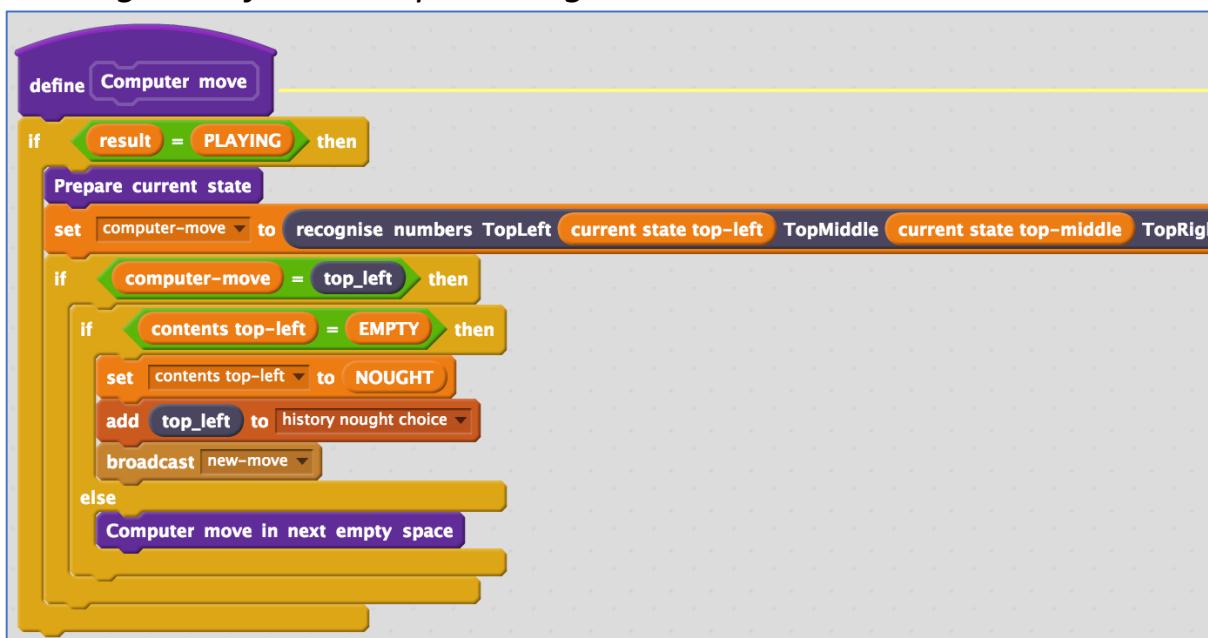


55. Modify the script to look like this

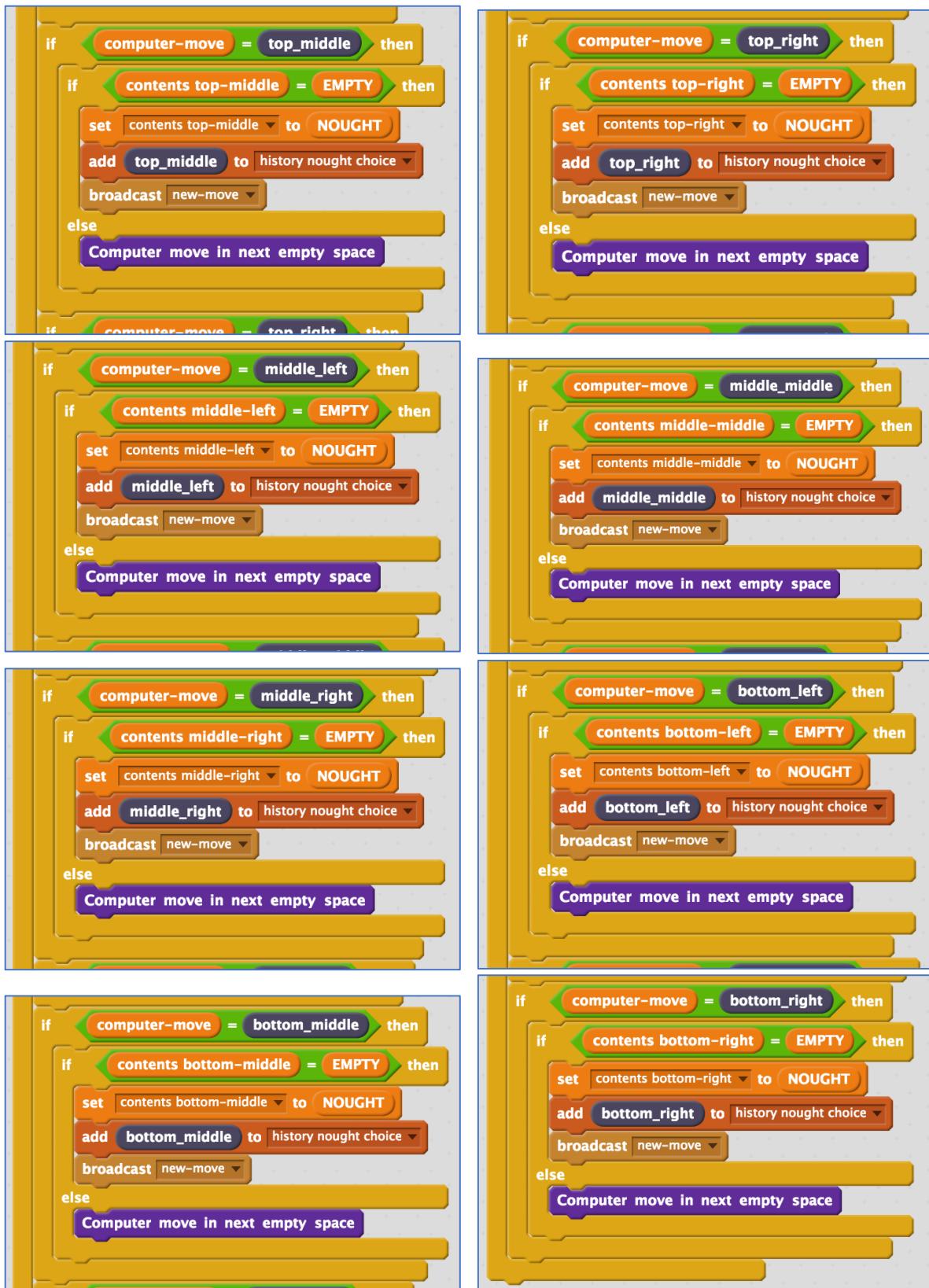
This script will use the machine learning model to let the computer choose where to have a move... (as long as it chooses the top-left space).

While the computer is still learning, it may get things wrong. So there is also a check in the script to make sure that the computer has chosen an EMPTY space! If it's chosen a space that already has a nought or cross in, it'll just pick the next empty space.

It will also add the choice it's made to the history, so it can be used in training data if it ends up winning.



56. Update the **Computer move** script to handle the other choices the machine learning model could make. This is all one long script.
Duplicate the "if computer-move" block by right-clicking on it to save time.



57. The whole script should end up looking like this



58. Save your project

*Click **File** -> **Save Project***

59. Play against the computer by clicking on the **Green Flag**

Use full-screen to avoid moving sprites accidentally.

Avoid playing the same game over and over again. Choose different spaces to give the computer a variety of examples of how to play.

60. When it starts to feel like you're playing the same games over and over again, go back to the Learn & Test screen, and use the new examples you've collected

*Click the "**Train a new machine learning model**" button again*

The screenshot shows the 'Machine learning models' screen. At the top, there's a header bar with the title 'Machine learning models'. Below it, a link 'Back to project' is visible. The main area is divided into two sections: 'What have you done?' on the left and 'What's next?' on the right.

What have you done?

You've trained a machine learning model to recognise when numbers are top_left, top_middle or 7 other classes.

You created the model on Sunday, October 8, 2017 12:19 AM.

You've collected:

- 16 examples of bottom_left,
- 14 examples of bottom_middle,
- 19 examples of bottom_right,
- 11 examples of middle_left,
- 23 examples of middle_middle,
- 15 examples of middle_right,
- 11 examples of top_left,
- 11 examples of top_middle,
- 14 examples of top_right

What's next?

Try testing the machine learning model below. Enter an example of numbers 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!

61. Every time you train a new machine learning model, you will need to re-open the Scratch project so that it starts using the new model.

*Click **File** -> **Load Project***

Open your saved project, even though you already have it open!

*Click **OK** when it asks if you want to replace the current project.*

That will make sure you're using the latest model.

62. Go back to step 59 – and repeat

Play against the computer, and try to vary your playing as much as possible.

Once you're struggling to keep coming up with new games, click the Train new machine learning model button again, and re-load your saved project in Scratch.

Try to repeat this process a few times until your machine learning model starts getting good!

Tips

Don't be kind!

You might be tempted to go easy on the computer when you're playing against it, particularly when it's just starting to learn and is playing very badly.

For example, you might have two crosses-in-a-row next to a blank space and could win. But instead, you might feel sorry for it doing badly and put a cross somewhere else instead to give it a chance.

Don't.

It is learning from the way that you play. If you don't complete a three-in-a-row when you can, you will be teaching it that it should do that.

If you want it to get better quickly, **play as well as you can.**

Mix things up with your examples

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

For example, start from a different position on the board on every turn.

What have you done?

You've trained a computer to play noughts and crosses.

You didn't have to describe the rules to the computer.

You didn't tell it that it should try to get three noughts in a row.

You didn't describe the difference between rows, columns or diagonals.

(The rules are in the Scratch game, but that doesn't count – that wasn't used in the machine learning model).

Instead, you showed it how you play, by collecting examples of decisions that you made when you win.

When it makes decisions that leads to it winning, this is added to its training data, so it can be even more confident in that approach in future.

This is called “reinforcement learning” because when it does something good you are “reinforcing” this by rewarding it.

Did you know?

People have been learning about machine learning by training a computer to play noughts and crosses for decades!

One famous example was **Donald Michie** – a British artificial intelligence researcher. During World War II, Michie worked at Bletchley Park as a code breaker.

In 1960, he developed “**MENACE**” – the Machine Educable Noughts And Crosses Engine. This was one of the first programs able to learn how to play noughts and crosses perfectly.

As he didn’t have a computer he could use, Michie built MENACE using 304 matchboxes and coloured glass beads.

Each matchbox represented a possible state of the board – like the examples that you’ve been collecting in your training data.

He put beads in the matchboxes to show how often a choice led to a win – the number of beads in the matchbox was like the number of times an example shows up in one of the buckets you created for your training data.

