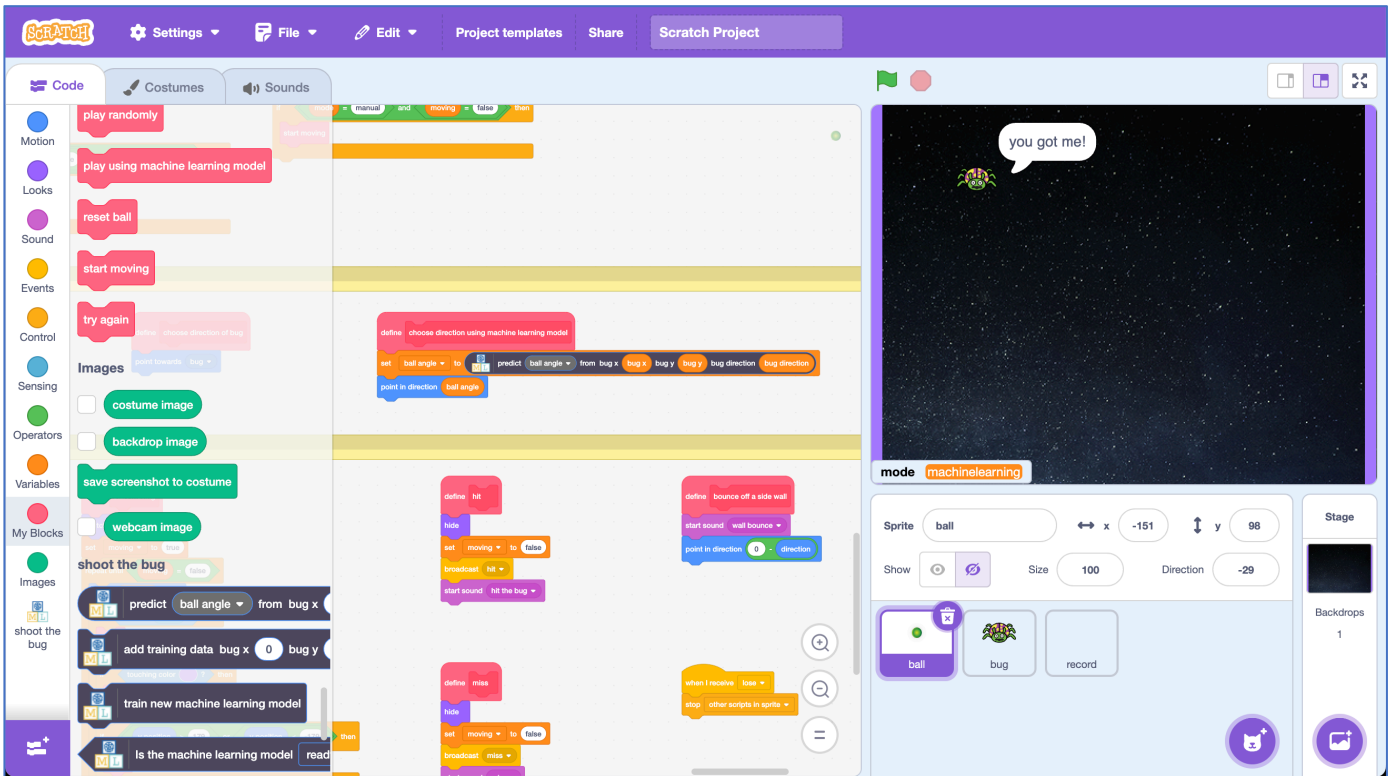
Shoot the bug

In this project you will train a computer to play a simple arcade game.

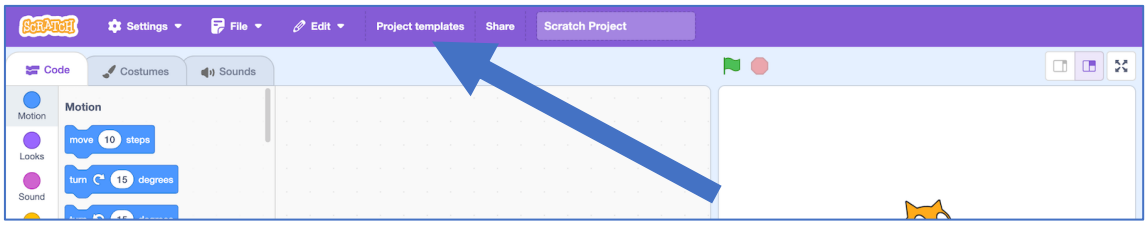
The game is based on shooting balls at a target.

You will teach the computer to be able to play this game by collecting examples of shots that hit, so that it can learn to make predictions about the shot it should take.



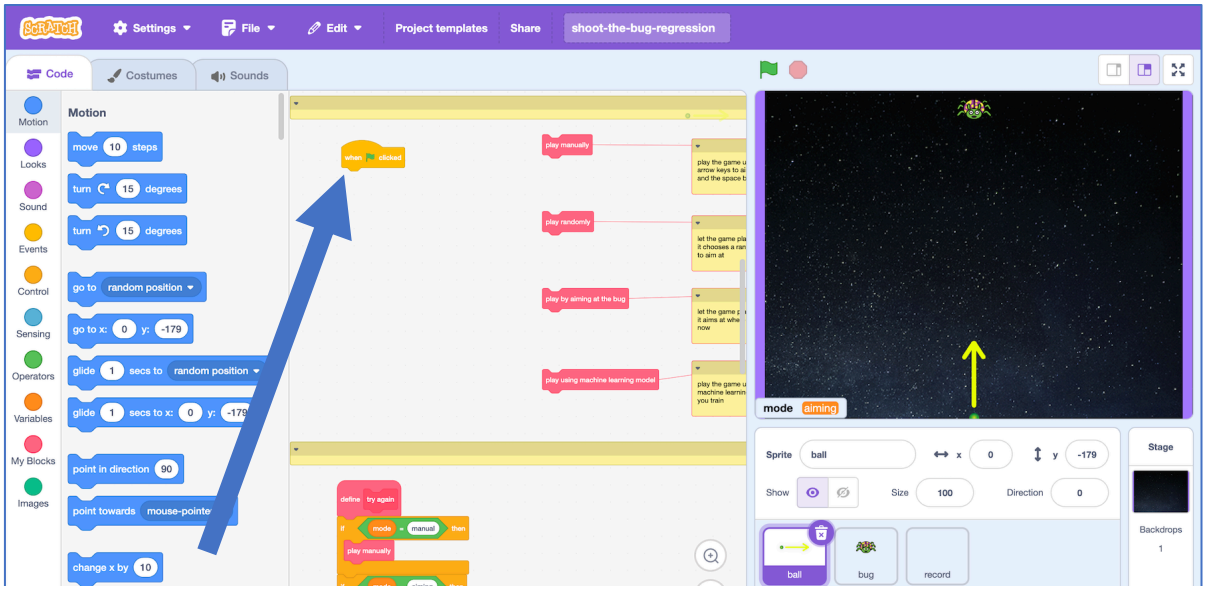
1. Go to https://machinelearningforkids.co.uk/scratch

2. Click on "**Project templates**"



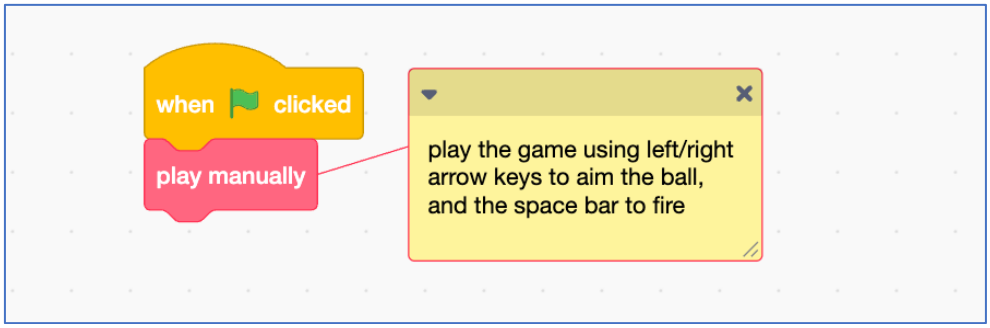
3. Click on the "**Shoot the bug**" template

4. Find the "**when Green Flag clicked**" block in the "**ball**" sprite



5. Attach the "**play manually**" block to the green flag block

(You can find it just to the right of the Green Flag block)



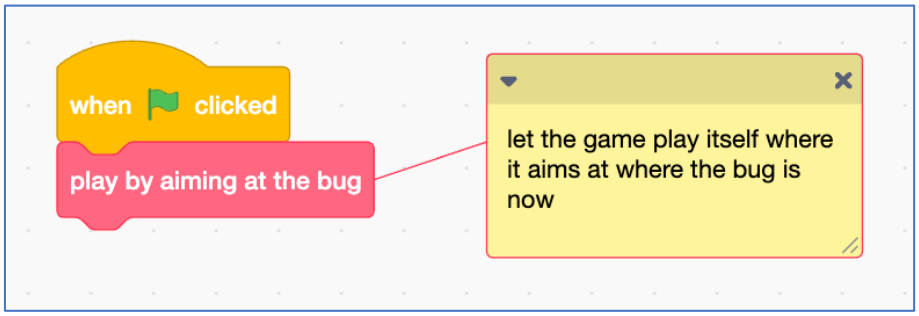
6. Click the **Green Flag** and try to shoot the bug!

Use the arrow keys to aim, then press the space bar when you're ready.

Try playing a few times to get used to how the game works.

7. Try using the "**play by aiming the bug**"

This will fire the ball for you, aiming at the current location of the bug when it fires.



What have you done so far?

You played a game in Scratch. Each time you play, the bug appears at a random location, and starts moving towards the bottom of the screen. The aim of the game is to shoot a ball at the bug before it reaches the bottom.

In this project, you are going to get the computer to decide what angle it should shoot at, based on the location of the bug.

You have seen that if it just aims at where the bug is now, it will often miss, because the bug is always moving. The computer needs to learn to aim in front of where the bug is, based on the speed the bug is moving.

You could do this by writing code to calculate the correct angle to launch at, based on the location. (If you have time, give that a try to compare!)

But, for this project, you're going to train the computer so that it learns for itself how to shoot at the bug.

You'll collect examples of the game being played and use that to train a machine learning "model" that can predict the correct angle to fire the ball at.

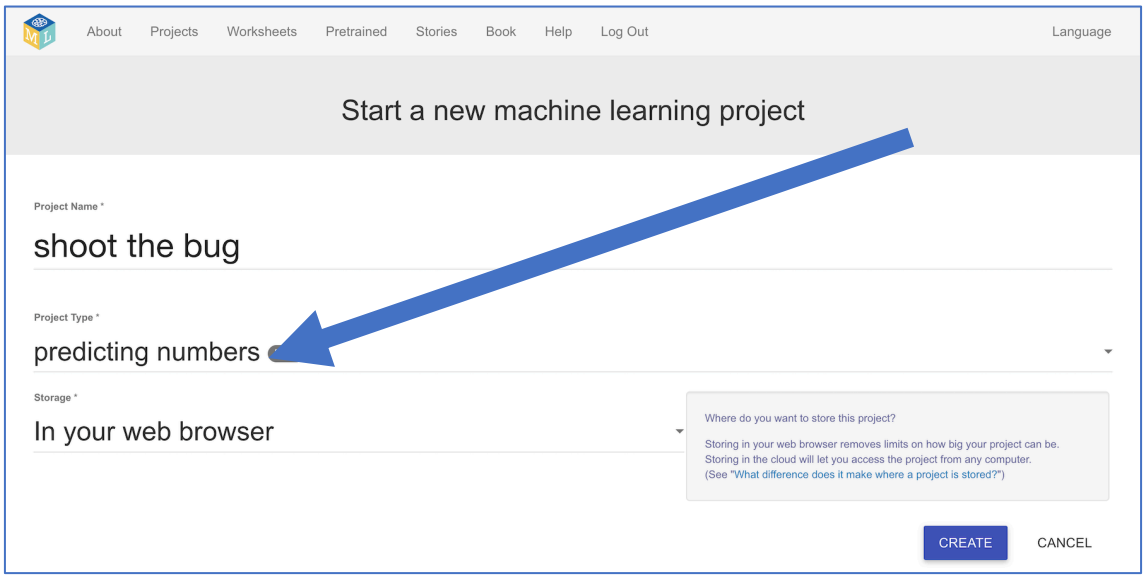
8. Go to https://machinelearningforkids.co.uk in a web browser

9. Click on "**Get started**"

10. Click on "**Try it now**"

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

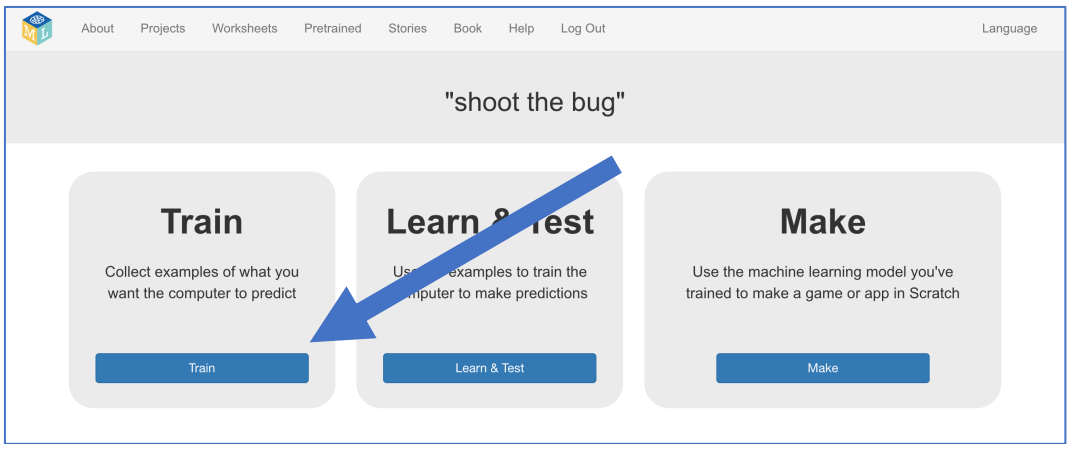
12. Name your project "shoot the bug" and set it to learn how to "**predict numbers**".

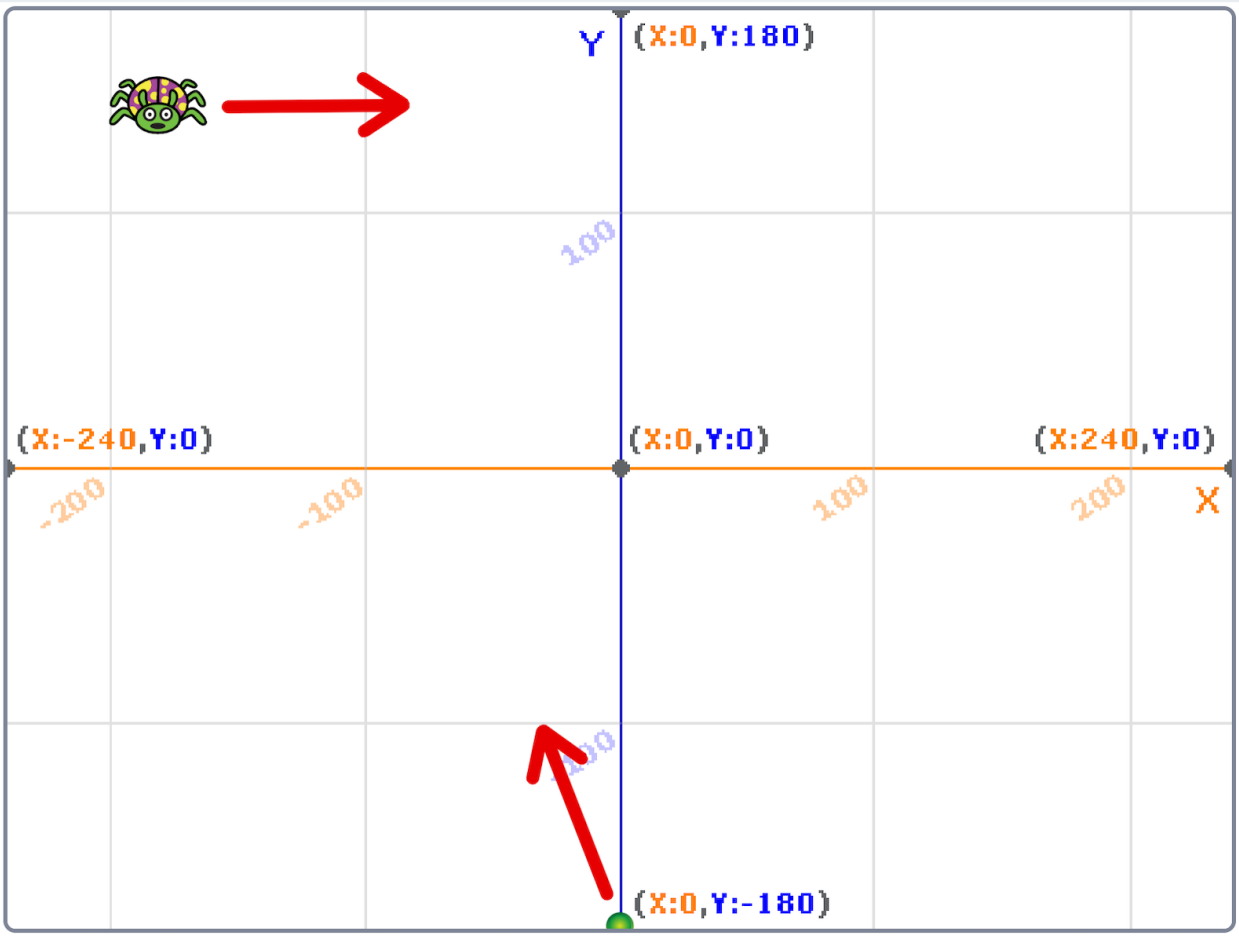


13. Click on "**Create**"

14. "**shoot the bug**" will be added to your list of projects. Click on it.

15. Click the "**Train**" button





Values the computer will use to make a prediction:

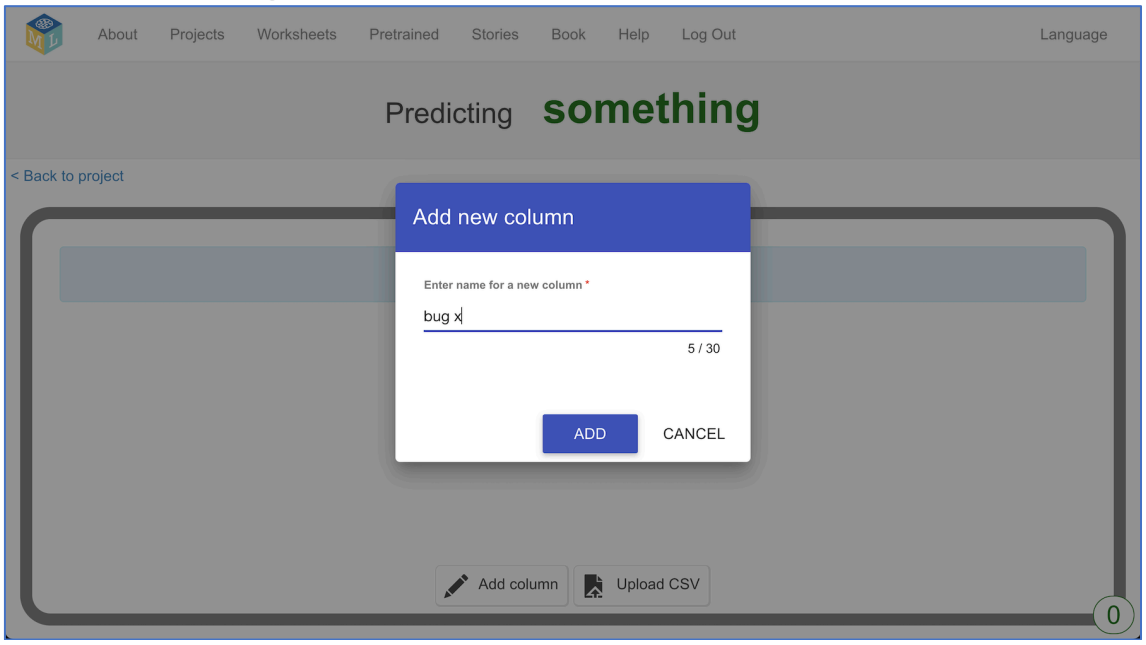
Name	What is it?	A positive number means…	A negative number means…	Example in screenshot
`bug x`	x-coordinate of the current location of the bug	The bug is on the **right side** of the stage	The bug is on the **left side** of the stage	`-180`
`bug y`	y-coordinate of the current location of the bug	The bug is at the **top** of the stage	The bug is at the **bottom** of the stage	`140`
`bug direction`	How far the bug will move (horizontally) in its next move	The bug is moving left-to-right →	The bug is moving right-to-left ←	`30`

Value the computer will learn to predict:

Name	What is it?	A positive number means…	A negative number means…	Example in screenshot
`ball angle`	direction to fire the ball at	The ball will fire towards the **right**	The ball will fire towards the **left**	`-20`

16. Click on **“Add column”**

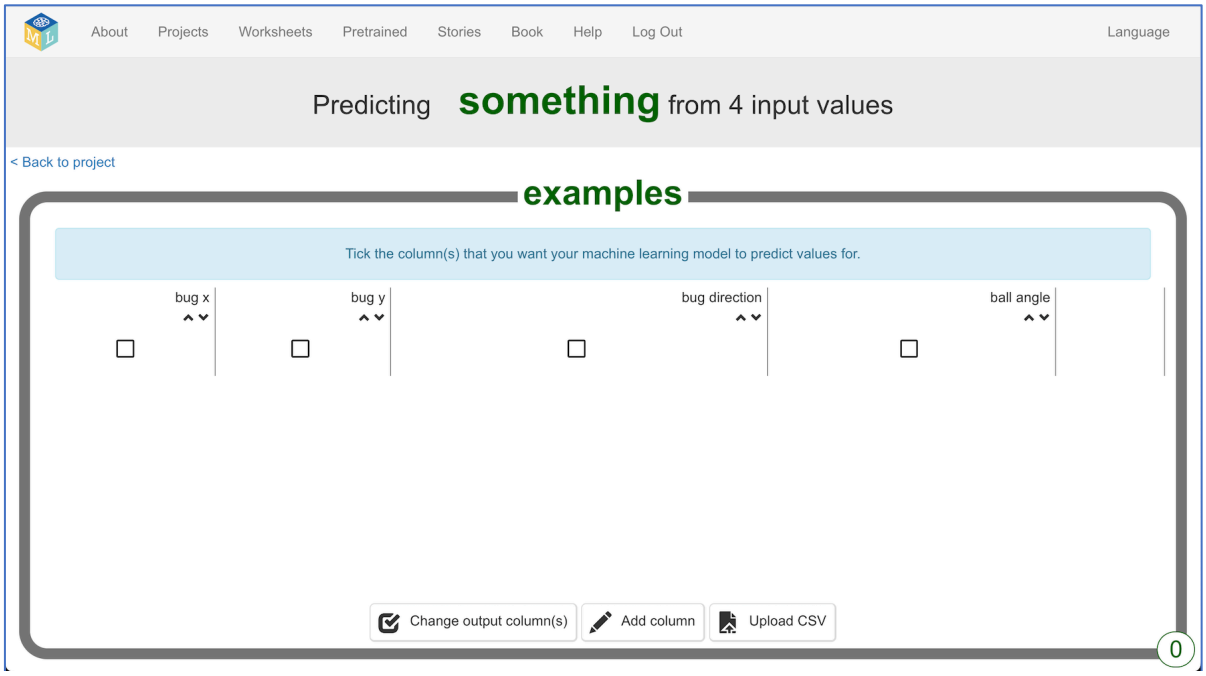
17. Type “`bug  x`” and click “**Add**”



18. Click on “**Add column**”, type “`bug  y`”, and click “**Add**”

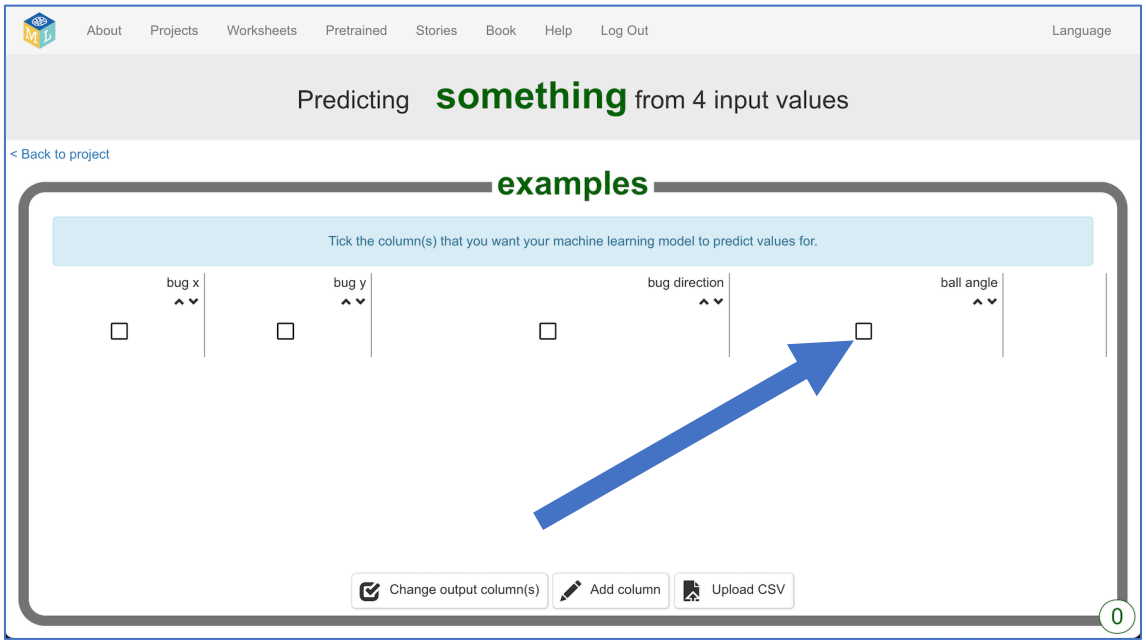
19. Click on “**Add column**”, type “`bug  direction`”, and click “**Add**”

20. Click on “**Add column**”, type “`ball  angle`”, and click “**Add**”

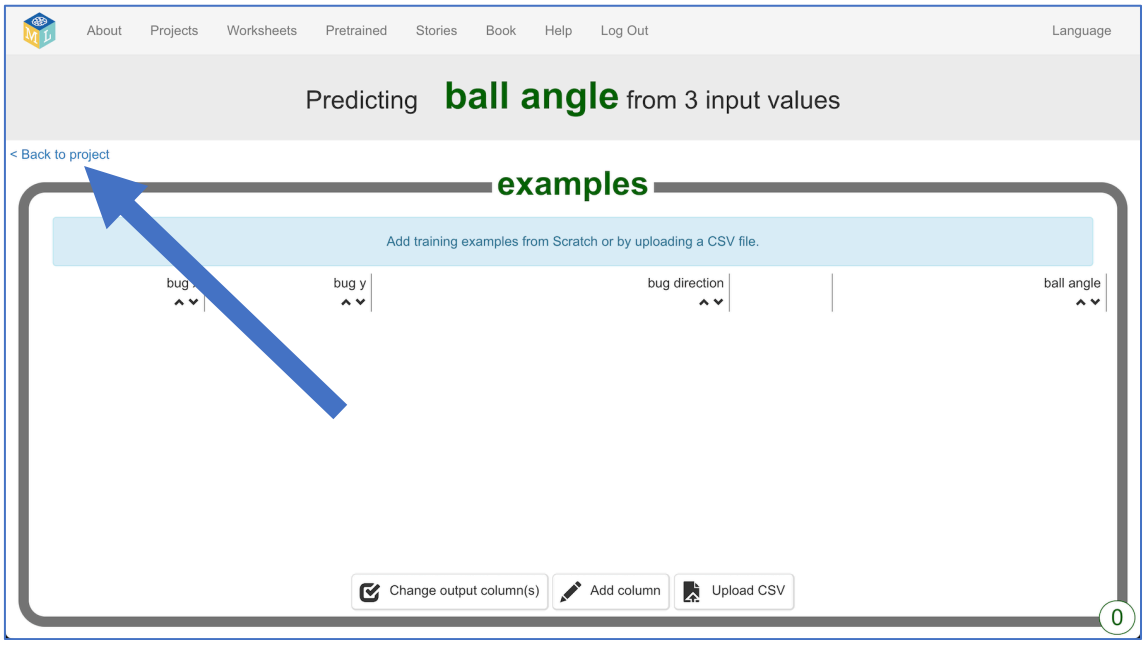


21. Click the tick box under “ball angle”

This is to choose that the “ball angle” value is the one that you want the computer to learn to predict



22. Click “**Back to project**”



23. Click the “**Make**” button

24. Click on “**Scratch 3**”

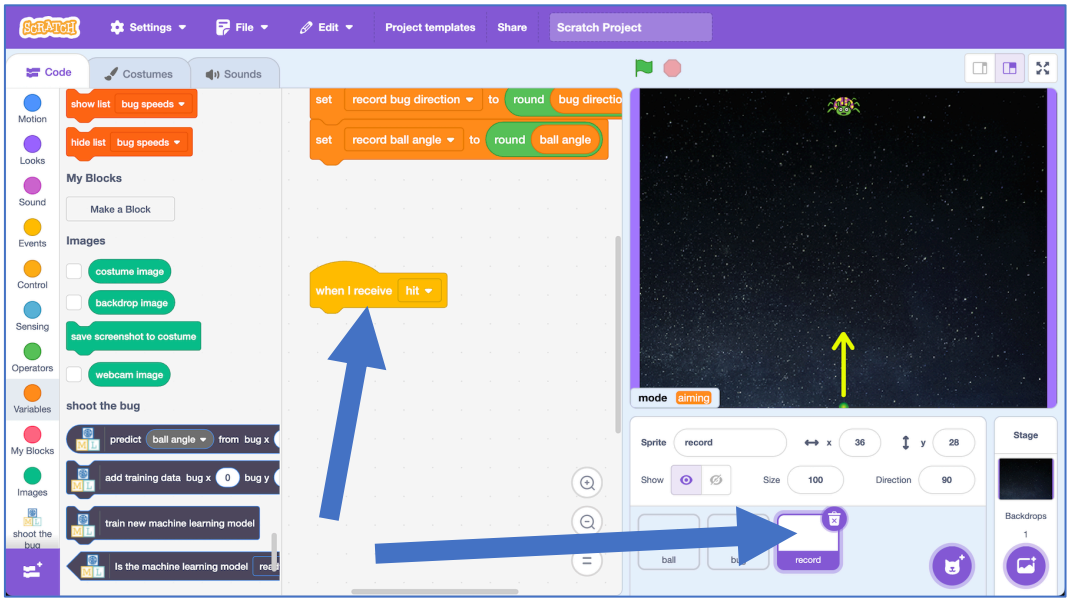
25. Click the “**straight into Scratch**” button

The page will warn you that you don’t have a machine learning model yet. That is okay. You are going to use Scratch to collect the training examples you will use to train a model.

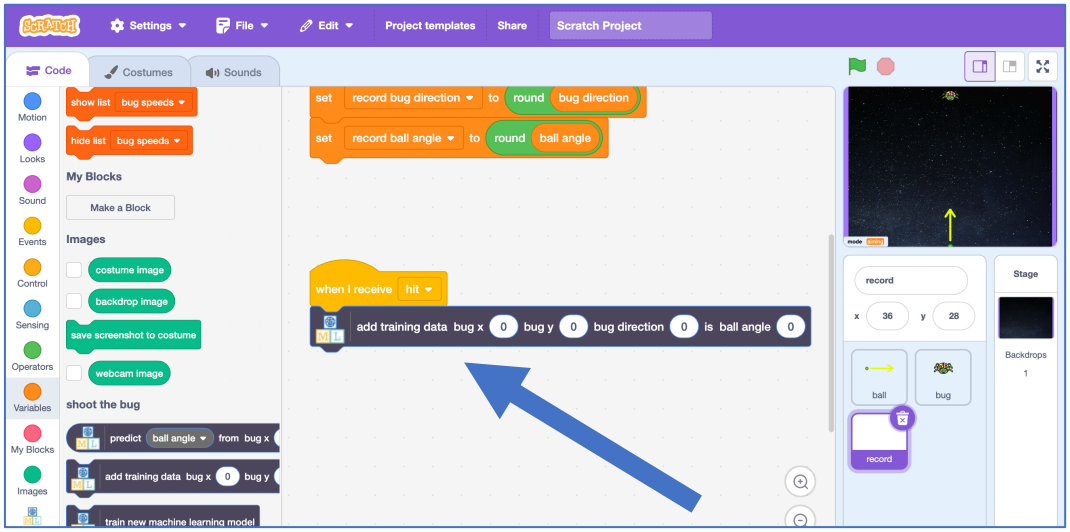
26. In the Scratch window that opens, click on “**Project templates**”

27. Open the “**Shoot the bug**” template again

28. Click on the “**record**” sprite and find the “**when I receive hit**” block

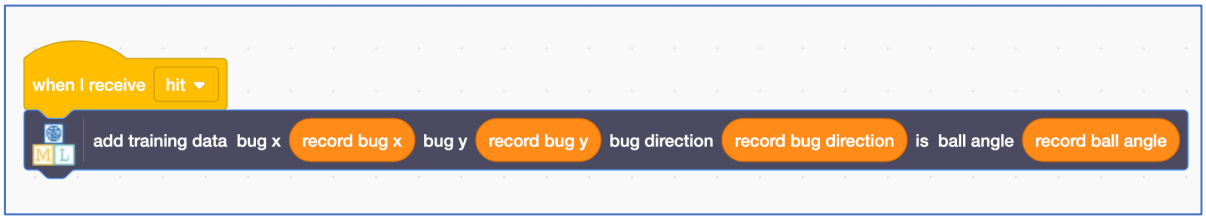


29. Add an “**add training data**” block to it

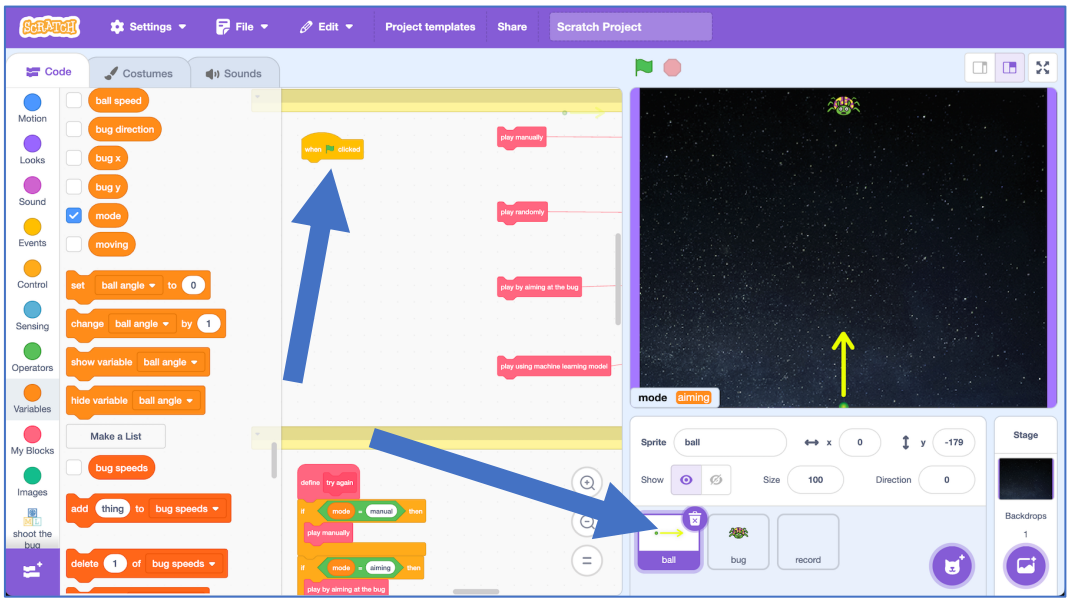


30. Add variables to the "**add training data**" block to match this screenshot.

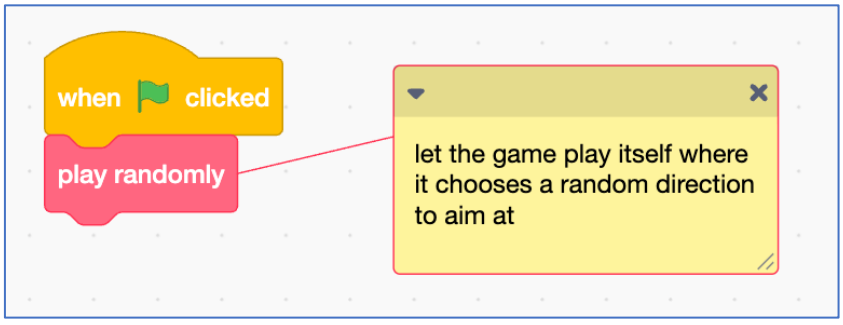
*Make sure you use the variables that start with "**record**"*



31. Click the "**ball**" sprite and find the "**when Green Flag clicked**" block



32. Connect the "**play randomly**" block to it



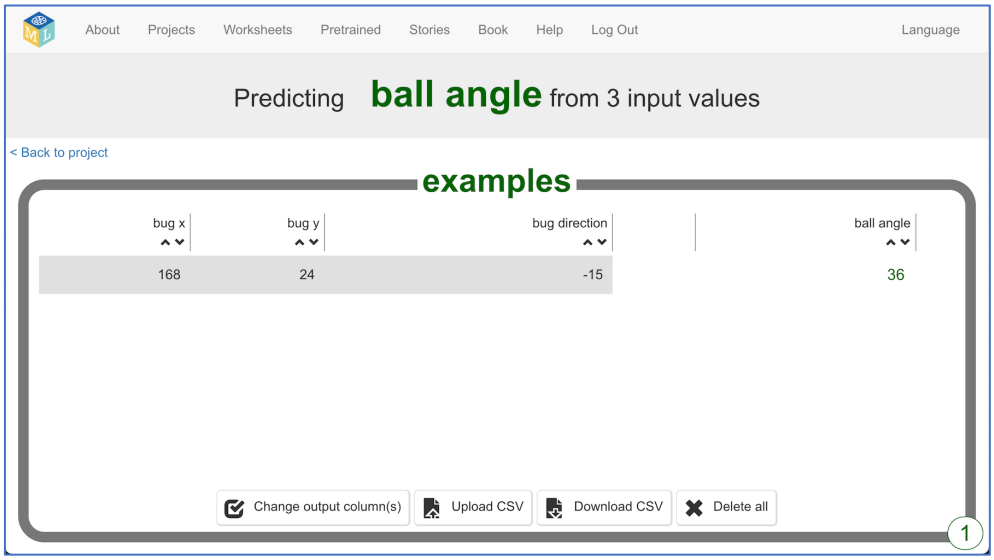
33. Click the **Green Flag**

34. Wait until the ball hits the bug

The game will keep firing in random directions until a ball hits the bug.

35. In the training window, go back to the "**Train**" page

You should see the first example has been added to your training data



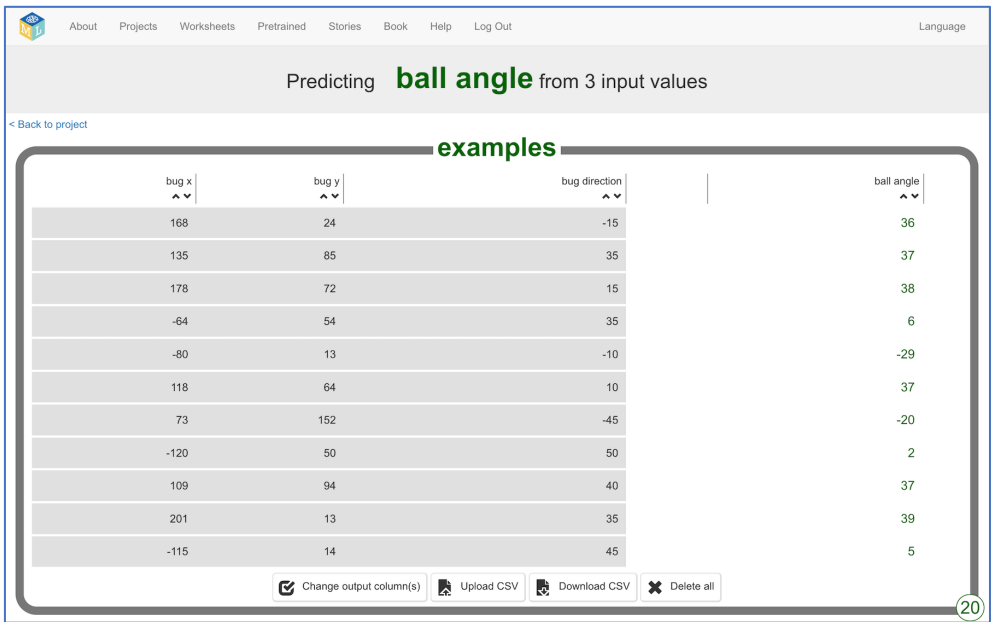
36. Go back to the Scratch window, and click on the Green Flag again

37. Repeat this to collect more training examples

When the ball hits the bug (or the bug catches you!) click the Green Flag again to start again.

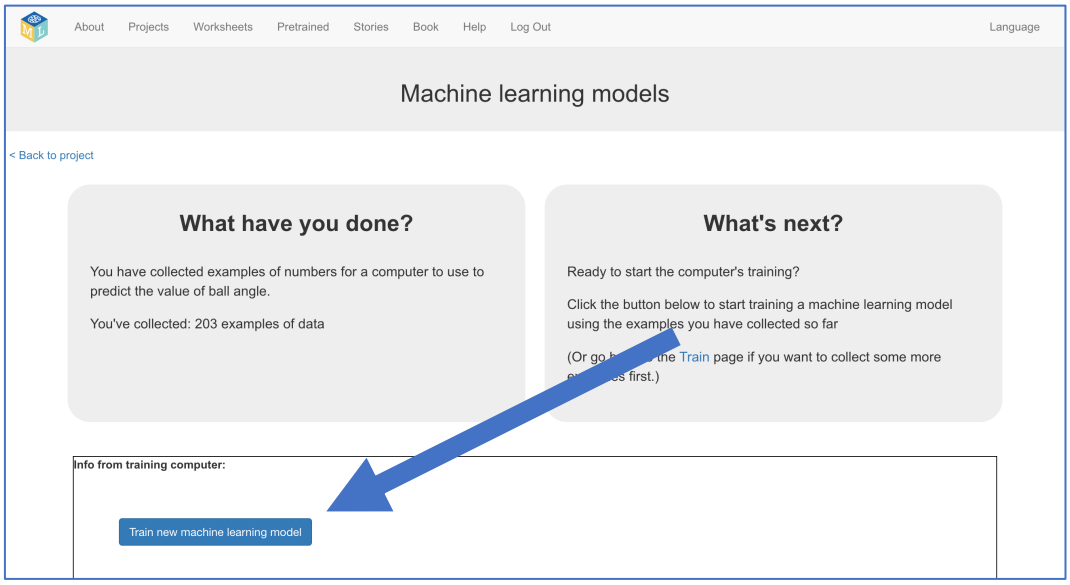
Each time, the bug starts in a different location and moves at a different speed. Your Scratch project will collect all of these to train the computer.

38. Go back to the training window and refresh the Train page to review the examples you have collected



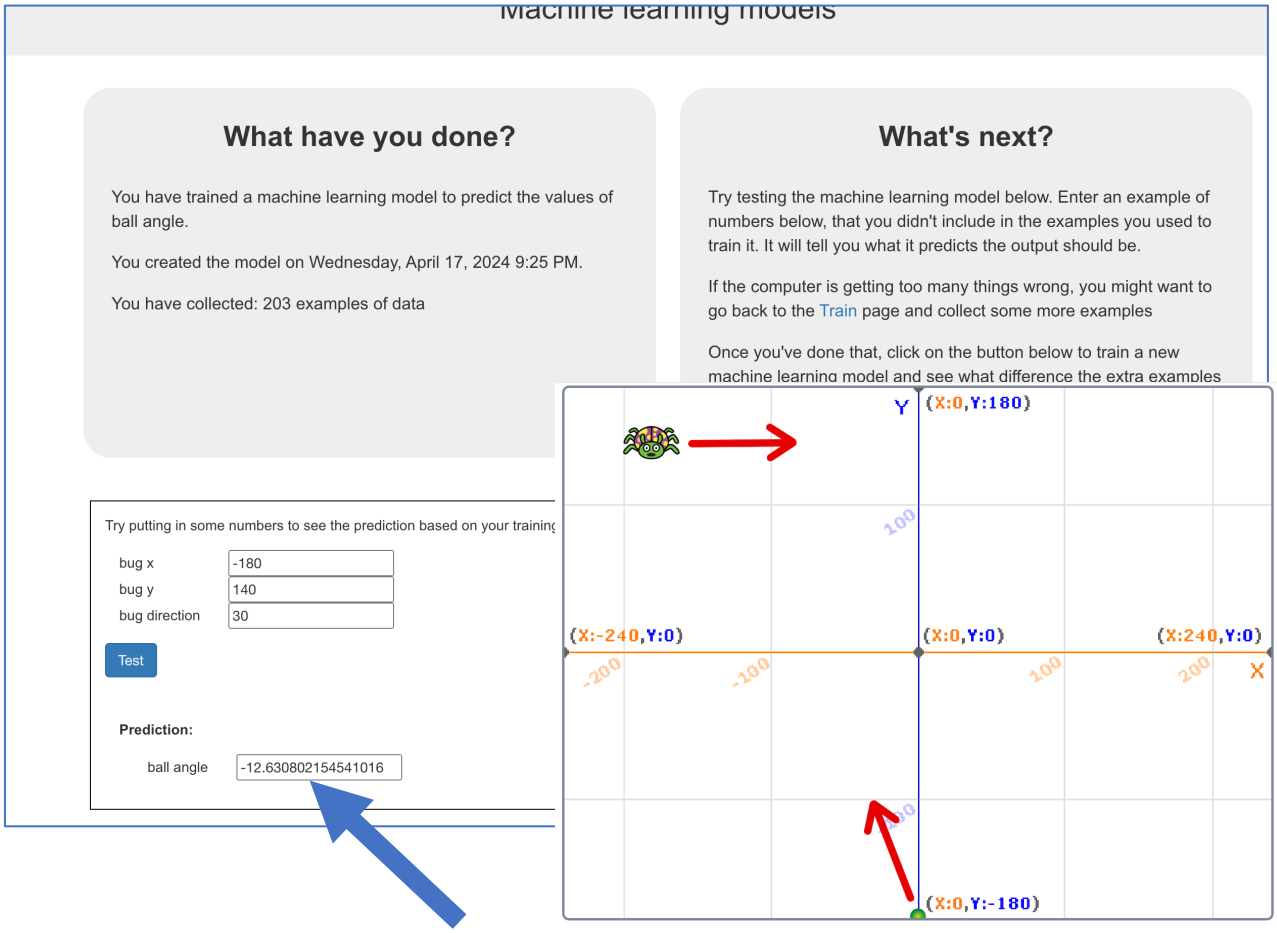
39. Train a machine learning model using your examples

*Click on "**Back to project**", then click the "**Learn & Test**" button.*
*Then click on the "**Train new machine learning model**" button*

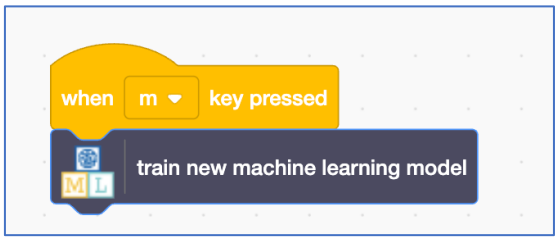


40. Try testing your model to see what angles it predicts

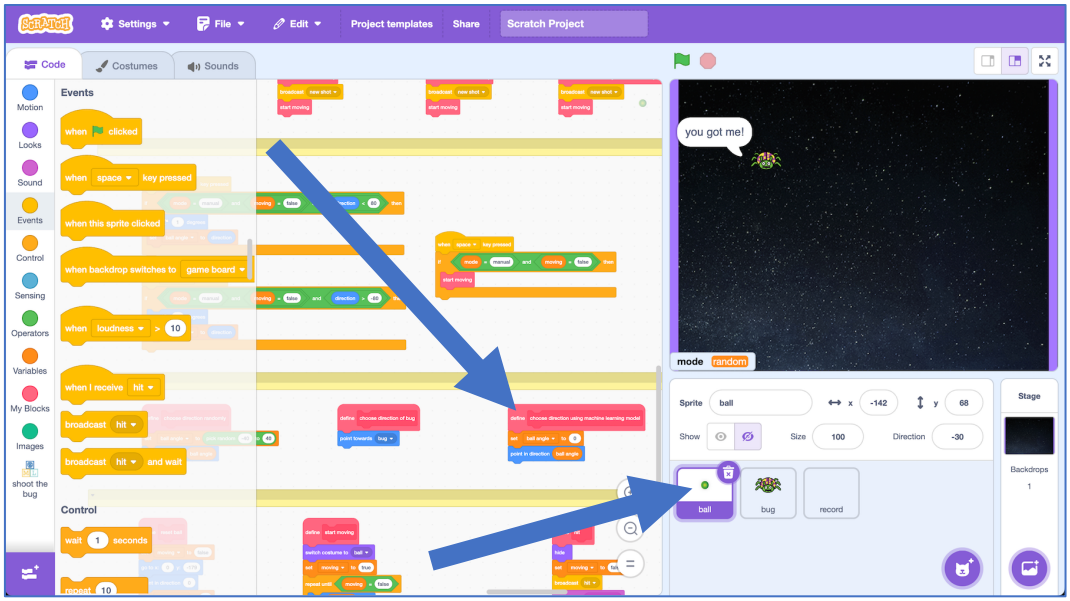
This is the prediction my model made for the values in my earlier diagram
Do you think your model is making good predictions?



41. In the Scratch window, add this code to the “ball” sprite

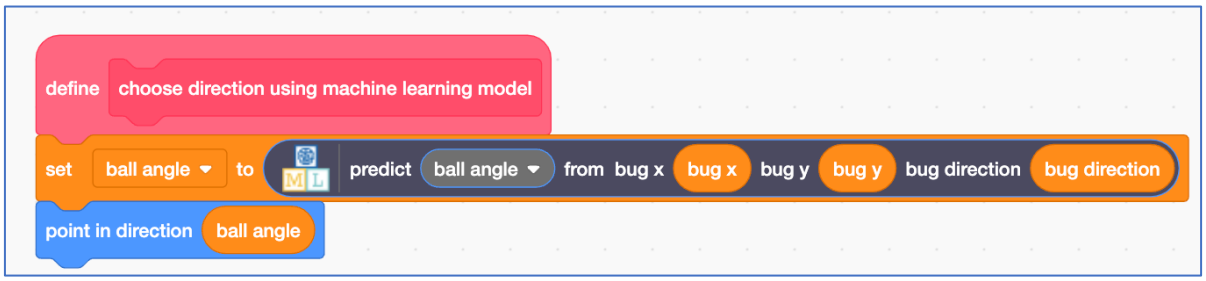


42. Find the “**choose direction using machine learning model**” block



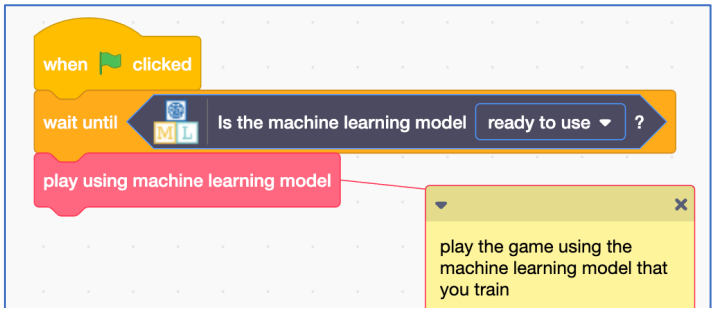
43. Update the code for this block to use your machine learning model

*Make sure you use the right variables: **bug x**, **bug y**, and **bug direction***



44. Find the “**when Green Flag clicked**” block again.

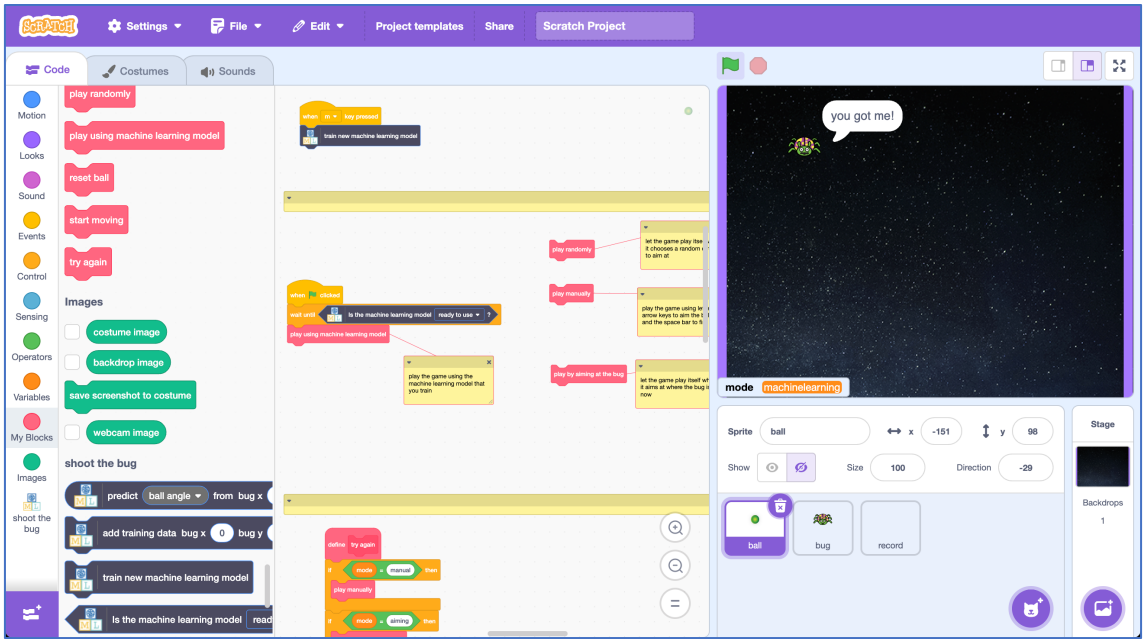
Remove the “play randomly” block, and change it to use your model



45. Press the "**m**" button on your keyboard to train a new model

46. Click on the **Green Flag** again

How good is your machine learning model at choosing angles?



What have you done so far?

You've trained a computer to play a game. Instead of working out the equation to calculate the angle to fire the ball, you did it by collecting examples. These examples were used to train a machine learning model.

The computer learned from patterns in the examples. It used these to make predictions about the angle to fire the ball at. The more examples it had to learn from, the better it will probably be.

Because you still have the "add training data" blocks in your script, you are still collecting more training examples every time you play. This means the more time you let your machine learning model play the game, the better it should get at playing.

Press the m button again to train a new model using the extra examples.

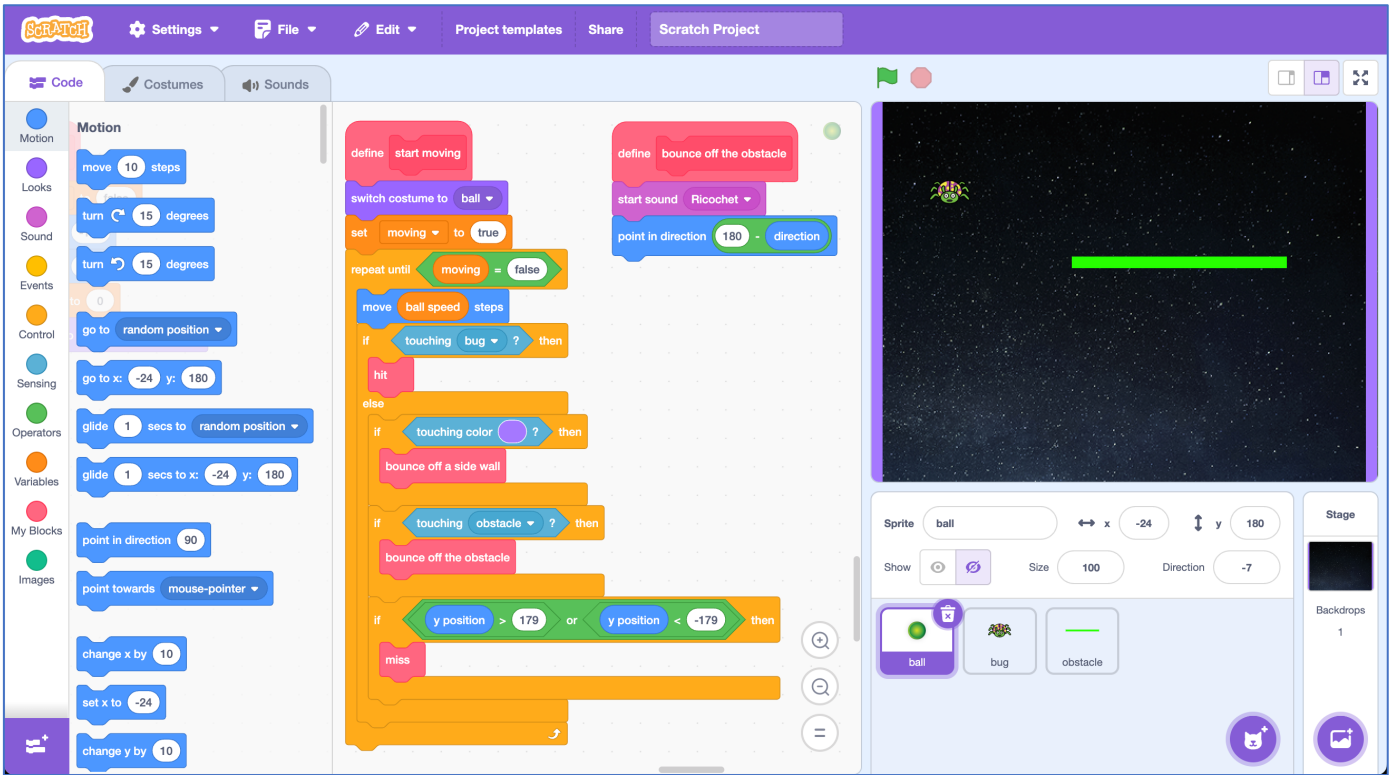
Is this a good use of machine learning?

We use machine learning when we want computers to do things that would be complicated for us to write instructions for it to follow.

We avoid machine learning if the time it takes to collect training examples would be longer than just writing the instructions for how to do the task.

Compare the effort to collect the training examples to train the computer to play this game, with the effort it would've taken you to calculate the angle to fire at. Do you think this game is a good use of machine learning?

What if the game was made harder? What if there was an obstacle to get around?



Try adding a new sprite called "obstacle" and modify the code for the "ball" sprite so that it bounces off of the obstacle.

The equations to calculate the correct angle to fire at would be even more complicated.

This makes it an even better use of machine learning. (But it would need more training examples for the computer to learn how to play, because it's a more complex task.)