

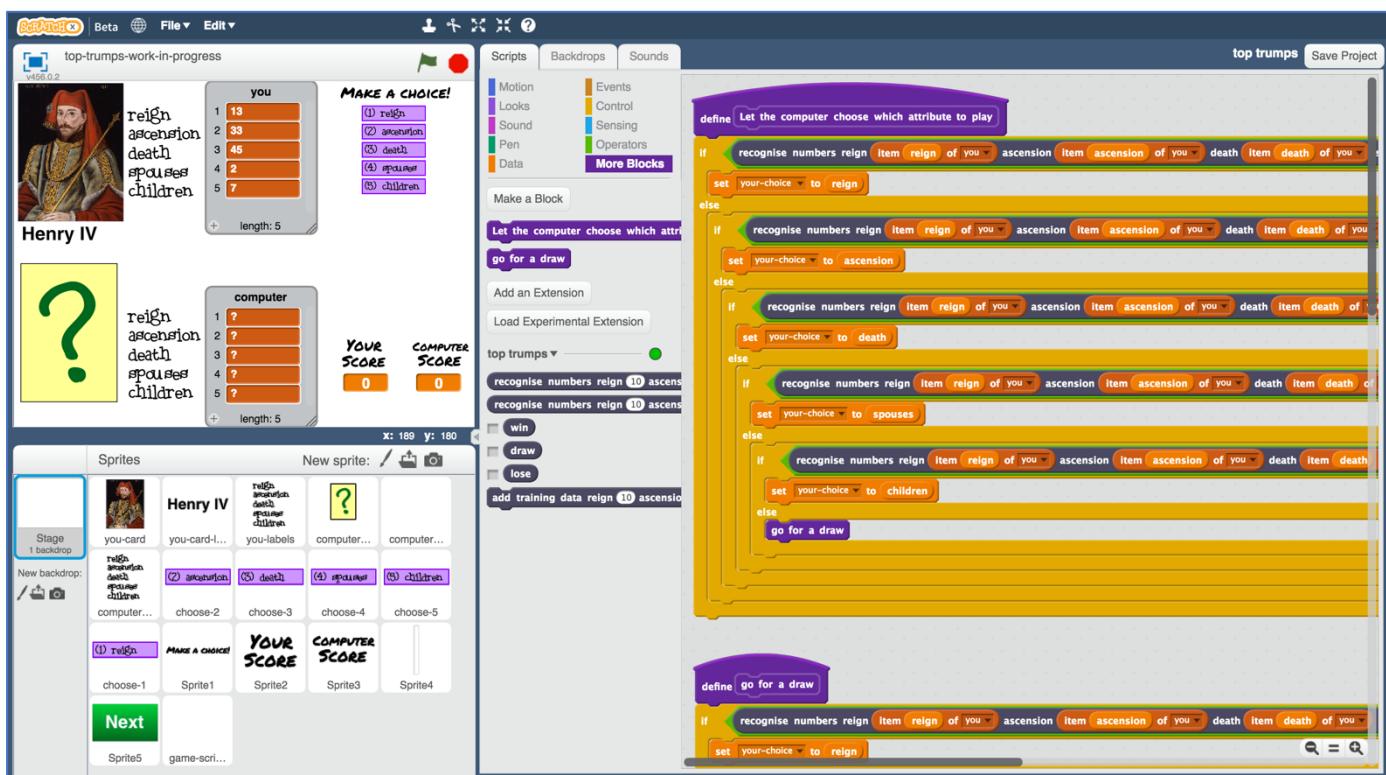
Top Trumps

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

For some values on the cards, you win by having the highest number. For others, you win by having the lowest. The range of numbers for different values will vary.

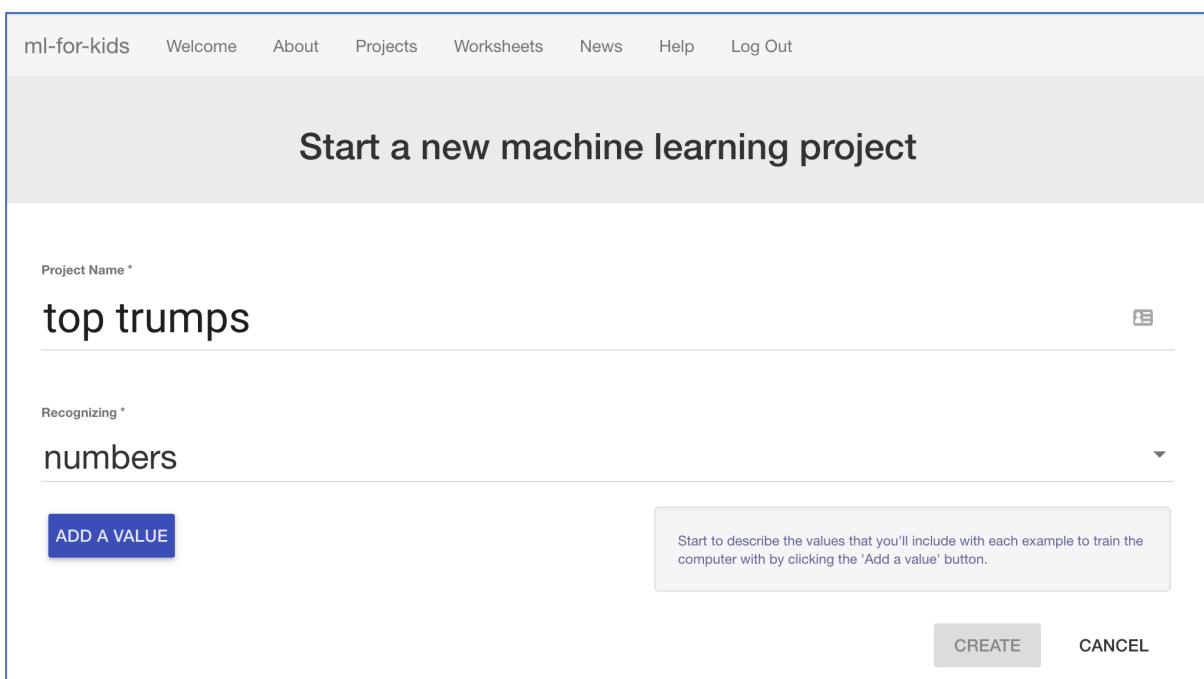
The aim will be for the computer to learn how to play the game well without you having to give it a list of all the cards or tell it the rules.

Instead, you'll try two different ways of training the computer to play the game by giving it examples of the game being played.



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. You'll need the **top-trumps.sbx** starter file for this project.
If you haven't got this, ask your teacher or group leader.
2. Go to <https://machinelearningforkids.co.uk/> in a web browser
3. Click on “**Get started**”
4. 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.
5. Click on “**Projects**” on the top menu bar
6. Click the “**+ Add a new project**” button.
7. Name your project “**top trumps**” 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 value "top trumps". To the right of this field is a small "X" icon for deletion. The second input field is labeled "Recognizing *" and contains the value "numbers". Below these fields is a blue "ADD A VALUE" button. To the right of the "Recognizing" field 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".

8. Click the “Add a value” button five times.

That will give you five text boxes to type names into.

Enter the following names into these boxes, in this order:

- * reign
- * ascension
- * death
- * spouses
- * children

Set the type for all of these to “number”

These will be where you put the numbers that are on a Top Trumps card.

Project Name *

top trumps

Recognizing *

numbers

Value 1 *	Type of value *	<input type="text" value="reign"/>	<input type="button" value="X"/>
		<input type="text" value="number"/>	<input type="button" value="▼"/>
Value 2 *	Type of value *	<input type="text" value="ascension"/>	<input type="button" value="X"/>
		<input type="text" value="number"/>	<input type="button" value="▼"/>
Value 3 *	Type of value *	<input type="text" value="death"/>	<input type="button" value="X"/>
		<input type="text" value="number"/>	<input type="button" value="▼"/>
Value 4 *	Type of value *	<input type="text" value="spouses"/>	<input type="button" value="X"/>
		<input type="text" value="number"/>	<input type="button" value="▼"/>
Value 5 *	Type of value *	<input type="text" value="children"/>	<input type="button" value="X"/>
		<input type="text" value="number"/>	<input type="button" value="▼"/>

ADD ANOTHER VALUE

CREATE CANCEL

9. Click “Add another value” one more time

Call this new value “choice”

Set its type to “multiple-choice”

Add “reign”, “ascension”, “death”, “spouses”, “children” as the choices for this value.

This is where you will put which value you choose from the Top Trumps card.

The screenshot shows a project configuration interface with the following details:

- Project Name:** top trumps
- Recognizing:** numbers
- Value 1:** reign (Type of value: number)
- Value 2:** ascension (Type of value: number)
- Value 3:** death (Type of value: number)
- Value 4:** spouses (Type of value: number)
- Value 5:** children (Type of value: number)
- Value 6:** choice (Type of value: multiple-choice)
 - Choices:
 - reign
 - ascension
 - death
 - spouses
 - children

Buttons at the bottom: ADD ANOTHER VALUE, CREATE, CANCEL.

10. Click the “Create” button

- 11.** You should now see “**top trumps**” in the list of your projects.
Click on it.

Your machine learning projects

top trumps
Recognising **numbers**

smart classroom
Recognising **text** as **fan_on, fan_off or 2 other classes**

journey to school

+ Add a new project

- 12.** Start by getting a project ready in Scratch. Click the **Scratch** button.
*The next page will warn you that you haven't done any machine learning yet, but clicking on the **Scratch by itself** link will launch Scratch.*

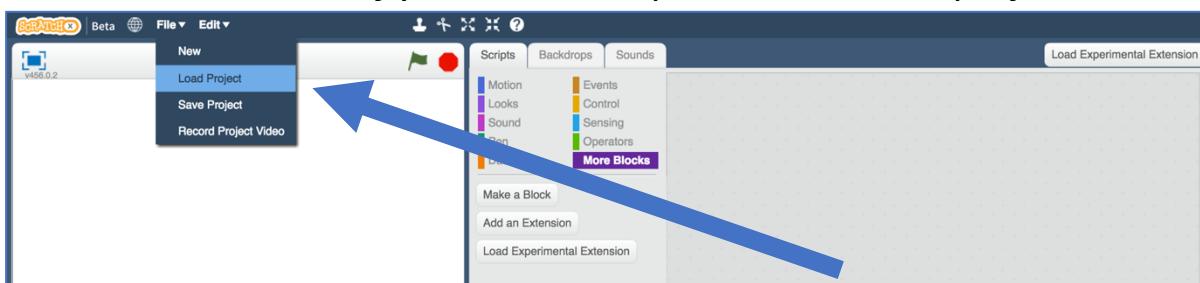
"top trumps"

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

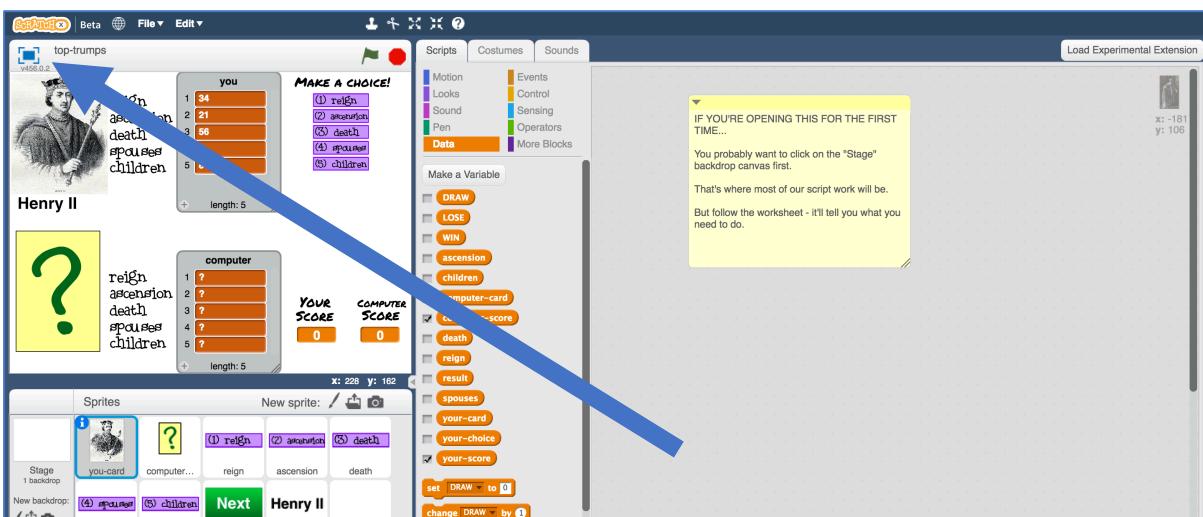
Learn & Test
Use the examples to train the computer to recognise numbers.
Learn & Test

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

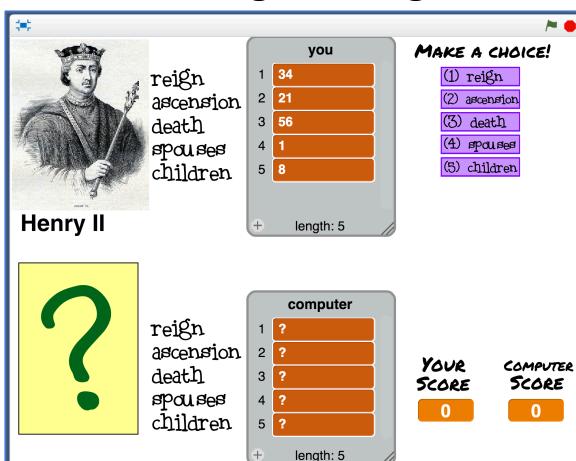
- 13.** Load the top-trumps.sbx file
Use **File -> Load Project** as shown below
Click **OK** when it asks if you want to replace the current project.



- 14.** This is Top Trumps based on the Kings and Queens of England.
Click the **full-screen button**.



- 15.** Click the green flag to start



The top half of the screen is **you**.

The bottom half is the **computer**.

When you click the Green Flag to start the game, you can't see the computer's card yet.

It's all just question marks.

Choose a value from your king or queen by **clicking the purple button next to it**

For example, in the screenshot above, my card is Henry II.

1) He reigned for **34** years.

(If I choose this and he was King longer than the computer's card, I'll win)

2) He ascended to the throne when he was **21**.

(If I choose this and he became King quicker than the computer's card, I'll win)

3) He died when he was **56**.

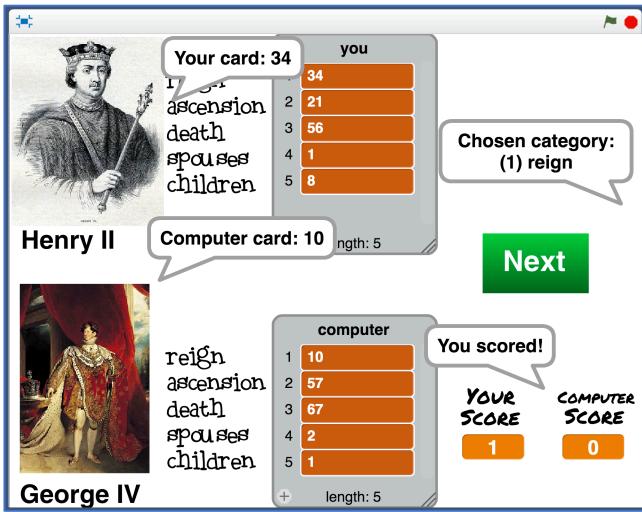
(If I choose this and he lived longer than the computer's card, I'll win)

4) He had **1** spouse.

(If I choose this and he had more spouses than the computer's card, I'll win)

5) He had **8** children.

(If I choose this and he had more children than the computer's card, I'll win)



When you choose a value, the computer card is revealed, and you see if you win or lost.

The score in the bottom right corner is updated.

Click on the green Next button to move onto the next card and play again.

If you win or tie, it's your turn again.

If you lose, the computer will get to choose the next value instead.

16. Play a few rounds of the game against the computer.

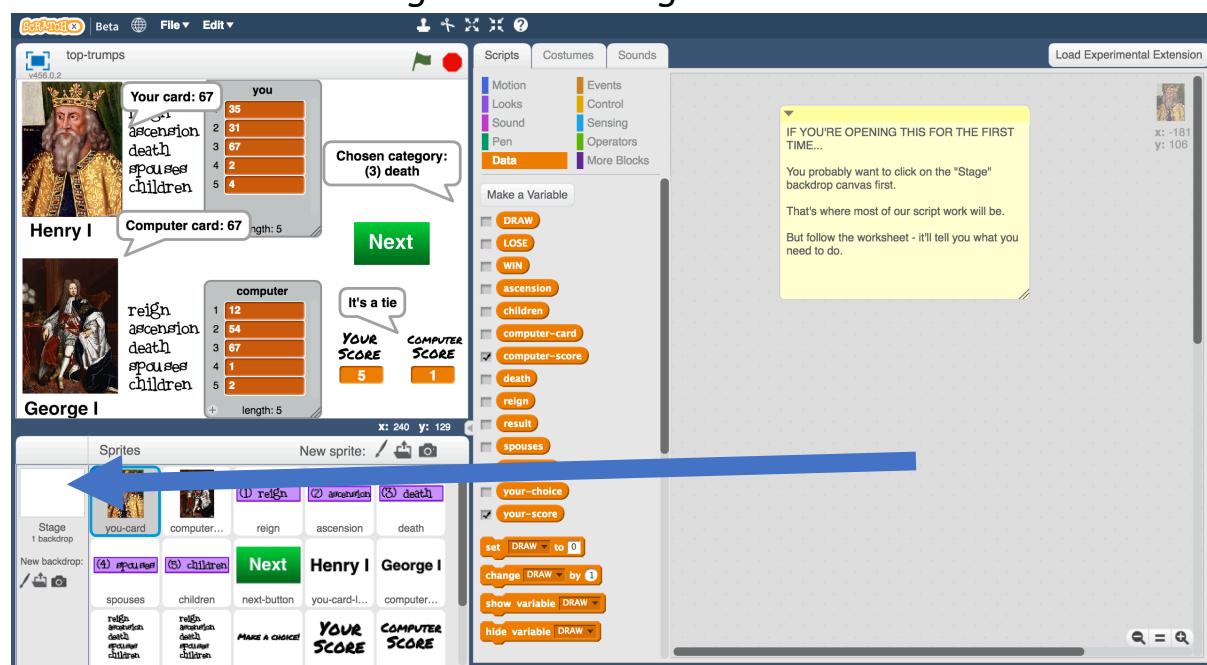
Try to work out how the computer is choosing values to play.

When you think you've worked out how the computer is playing, move onto the next step.

17. Click on the **full-screen** button again to go back to normal view.

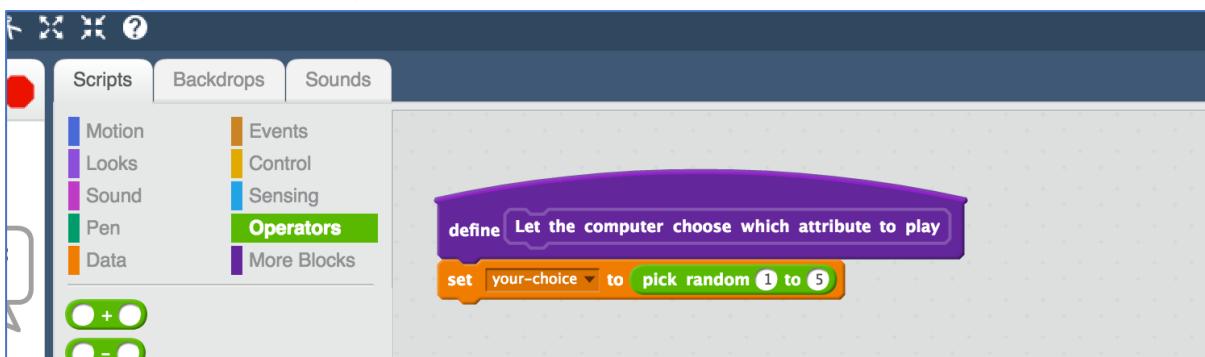
Then click on the **Stage**

This is the white rectangle above "Stage" shown with the arrow below.



18. The script on the **Stage** shows how the computer has been coded.
The computer always picks “reign”.
Did you get it right?

19. Change the script so that the computer chooses a value at random when it’s the computer’s turn.
Choosing from 1 (reign) to 5 (children) at random.



20. Click the **green flag** to reset the scores to 0. Go back to full-screen and play the game again.
Stop when either you or the computer reaches 10 points. Who won?

What have you done so far?

You’ve set up a bot to play Top Trumps and given it a simple strategy: choose values at random.

But people don’t play like that. We learn how to choose which value would give us the best chance of winning. We do this based on the cards we’ve seen before, and on our understanding of the rules.

Next, you’ll try a simple way to train the computer using a few experiences of seeing how the game is played.

21. Go back to Scratch.

Play the game, and write down what happens. You can do this by filling in the tables on the next page.

Only write down the numbers from your cards.

You need:

5 examples of a round where you won.

5 examples of a round where you drew.

5 examples of a round where you lost.

An example row is included in each table to show you what I mean.

Examples where you **won**, and the computer lost

The values from your card					which value did you pick?
reign	ascension	death	spouses	children	
23	24	48	1	9	ascension

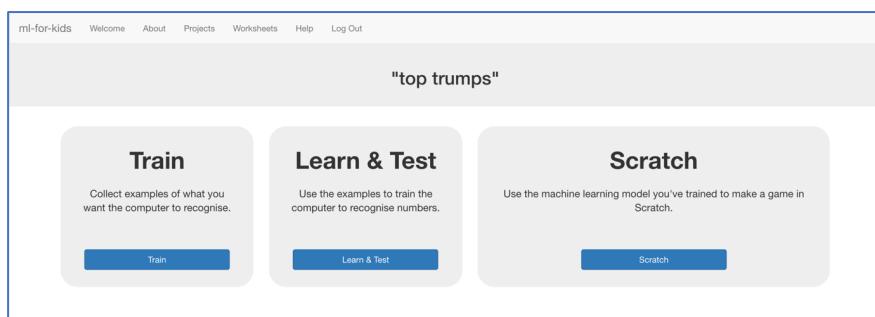
Examples where it was a **draw**

The values from your card					which value did you pick?
reign	ascension	death	spouses	children	
21	36	58	1	9	spouses

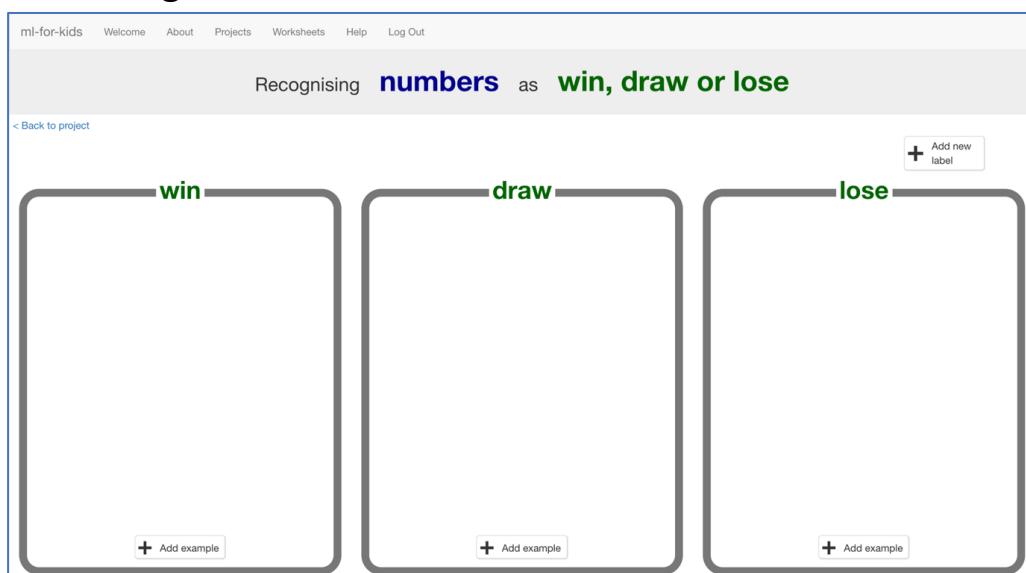
Examples where you **lost**, and the computer won

The values from your card					which value did you pick?
reign	ascension	death	spouses	children	
13	38	51	1	3	death

- 22.** Close Scratch. Click the “< Back to project” link.
Click on the **Train** button.



- 23.** Click on “+ Add new label” and call it “win”.
Do that again, and create a second bucket called “draw”.
Do that again, and create a third bucket called “lose”.



24. Click the “Add example” button in the “win” bucket, and type in the values from the first row in your “win” table.

The screenshot shows a web-based application interface for managing data buckets. At the top, there's a navigation bar with links like 'ml-for-kids', 'Welcome', 'About', 'Projects', 'Worksheets', 'Help', and 'Log Out'. Below the navigation, the title 'Recognising numbers as win, draw or lose' is displayed. On the left, there's a 'win' bucket containing four cards with data: 'reign 33 ascension 43 death 76 spouses 1 children 9 choice death', 'reign 9 ascension 59 death 68 spouses 1 children 6 choice death', 'reign 21 ascension 18 death 40 spouses 1 children 10 choice death', and 'reign 9 ascension 25 death 34 spouses 1 children 1 choice ascension'. Each card has a '+ Add example' button at the bottom. In the center, a modal window titled 'Add new example' is open, asking 'Enter an example of 'win''. It contains four input fields with the values 'reign', 'ascension', 'death', and 'spouses'. At the bottom of the modal are 'ADD' and 'CANCEL' buttons. To the right of the modal, there's a 'lose' bucket with a '+ Add new label' button.

25. Do this for all the examples in the “win” table.

Now click on the “Add example” button in the “draw” bucket, and do the same for all the rows in the “draw” table.

Then type in all the examples from the “lose” table into the “lose” bucket.

This screenshot shows the 'ml-for-kids' application interface with three main buckets: 'win', 'draw', and 'lose'. The 'win' bucket on the left contains four cards with data. The 'draw' bucket in the center contains two cards with data. The 'lose' bucket on the right contains five cards with data. Each card displays a list of numerical values for various categories. At the bottom of each bucket, there is a '+ Add example' button.

26. Click the “< Back to project” link. Click the “Learn & Test” button.

27. 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 commands from the examples you’ve written.

The screenshot shows a web application interface for training a machine learning model. At the top, there is a navigation bar with links: ml-for-kids, Welcome, About, Projects, Worksheets, 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?**

You've collected examples of numbers for a computer to use to recognise when numbers is win, draw or lose.

You've collected:

 - 5 examples of draw,
 - 5 examples of lose,
 - 5 examples of win
- What's next?**

Ready to start the computer's training?

Click the button below to start training a machine learning model using the examples you've collected so far.

(Or go back to the Train page if you want to collect some more examples first.)

At the bottom left of the main content area, there is a small box labeled "Info from training server:" containing the text "Train new machine learning model".

28. Once the training has completed, a Test box will be displayed.

Test your machine learning model to see what the computer has learned.

Type in all the values from a card into the ‘reign’, ‘ascension’, ‘death’, ‘spouses’, and ‘children’ boxes.

Then pick the value you would choose from the ‘choice’ drop-down list.

The computer will tell you whether it thinks you will win, lose or draw based on that choice.

Machine learning models

[< Back to project](#)

What have you done?

You've trained a machine learning model to recognise when numbers are win, draw or lose.

You created the model on Tuesday, September 26, 2017 11:20 PM.

You've collected:

- 5 examples of draw,
- 5 examples of lose,
- 5 examples of win

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!

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

reign	12
ascension	54
death	67
spouses	1
children	2
choice	death

[Test](#)

Recognised as **win**
with 100% confidence

What have you done so far?

You've started to train a computer to learn about Top Trumps.

The examples help the computer learn what values to expect in cards: the range of numbers for each value, how often it should expect to see high values, how often it should expect to see low values.

The examples also help the computer to learn what numbers will cause it to win or lose, without you needing to tell it what the rules are.

29. Click the “< Back to project” link, then back to the “Scratch” button. This page will be updated with instructions on how to use the new blocks in Scratch from your project. Keep this page open so you can check back on how to use them.

The screenshot shows a web page titled "Using machine learning in Scratch". It includes a "Back to project" link, a sidebar with ML block descriptions, a Scratch script editor, and a status bar indicating training progress.

ML Block Descriptions:

- recognise numbers reign 1 ascension 2 death 3 spouses 4 children 5 choice 6 label**: Put numbers in the input for this, and it will return the label that your machine learning model recognises it as.
- recognise numbers reign 1 ascension 2 death 3 spouses 4 children 5 choice 6 (confidence)**: This will return how confident your machine learning model is that it recognises the type of numbers. (As a number from 0 - 100).
- win draw lose**: These blocks represent the labels you've created in your project, so you can use their names in your scripts.

This means you can do something like this:

```

if [recognise numbers reign 1 ascension 2 death 3 spouses 4 children 5 choice 6 (label)] then
  say [I think that was win]

```

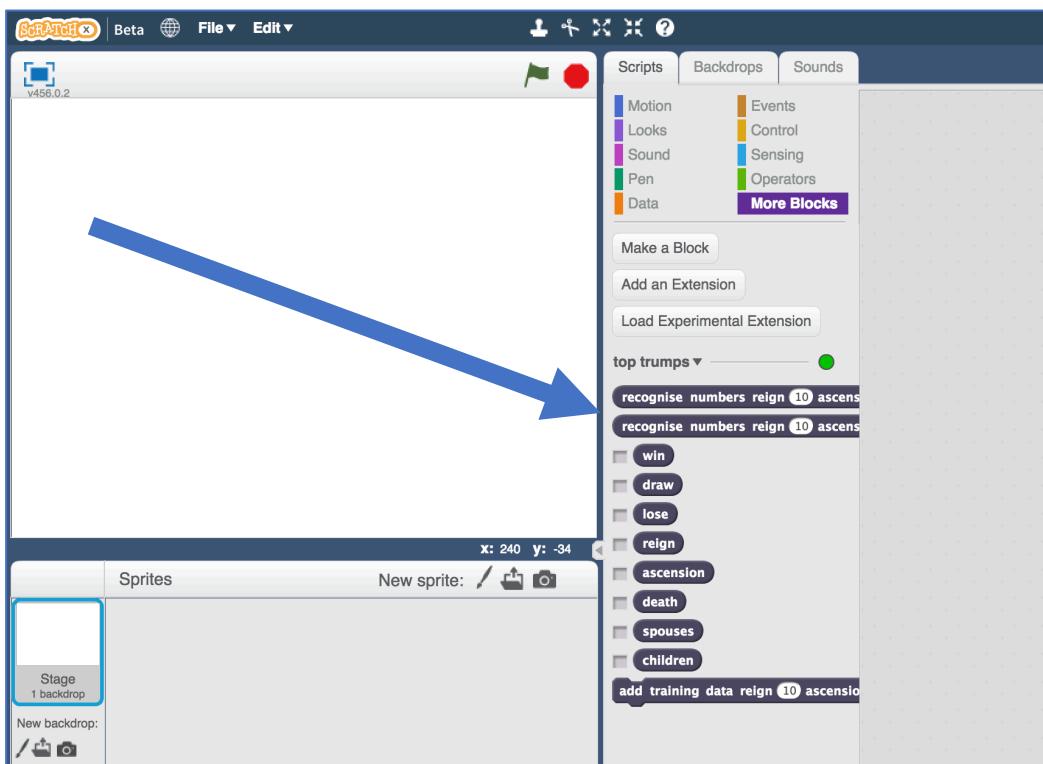
Status Bar:

It will look something like this - except with the name of your project.

The coloured circle next to your project name tells you if your machine learning model is trained and ready to go.

- Green circle: means your model is trained and ready to go
- Yellow circle: means your model hasn't finished training yet
- Red circle: means something went wrong. Go back to the Learn & Test page to see what went wrong with training.

30. Click the “Open in Scratch” button at the bottom of that page to launch the Scratch editor.
You should see new blocks in the “More blocks” section from your “top trumps” project.



31. Load the “top-trumps.sbx” Scratch project you opened before.

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

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

32. Click on the **Stage** (as you did before) to go back to the Script for how the computer chooses its moves.

33. You need to change the script so that the computer uses the machine learning model that you’ve started to train.

Your script needs to:

1) Make a prediction for all the possible choices on its card

(reign / ascension / death / spouses / children)

2) Choose the one that the computer predicts will give it a **win**

3) If the computer doesn’t predict any of the choices will give it a win, then choose the one that the computer predicts will give it a **draw.**

4) If the computer doesn’t predict any choice will give it a win or draw, then choose one at **random and hope for the best!**

This is a long and complicated script.

Take it slowly - we’ll build it up in stages.

Step 1 – Create this – to check for a win for a single value (reign)

(Use the label “lose” because the computer is going to choose which value to go for based on your card. For it to win, it wants to choose a value that it has learned will make you lose).

“reign” in the choice space is the dark blue “reign” block from the “More Blocks” tab because this is one of the multiple-choice options

```
if [recognise numbers reign item reign of you v ascension item ascension of you v death item death of you v spouses item spouses of you v children item children of you v] choice reign (label) = lose then
  set [your-choice v] to [reign]
else
  
```

“reign” in the orange data spaces is the orange “reign” block from the “Data” tab

Step 2 – Duplicate that. Put the copy into the else block.

Change the choice and the set-your-choice value from “reign” to “ascension”.

You should end up with this:

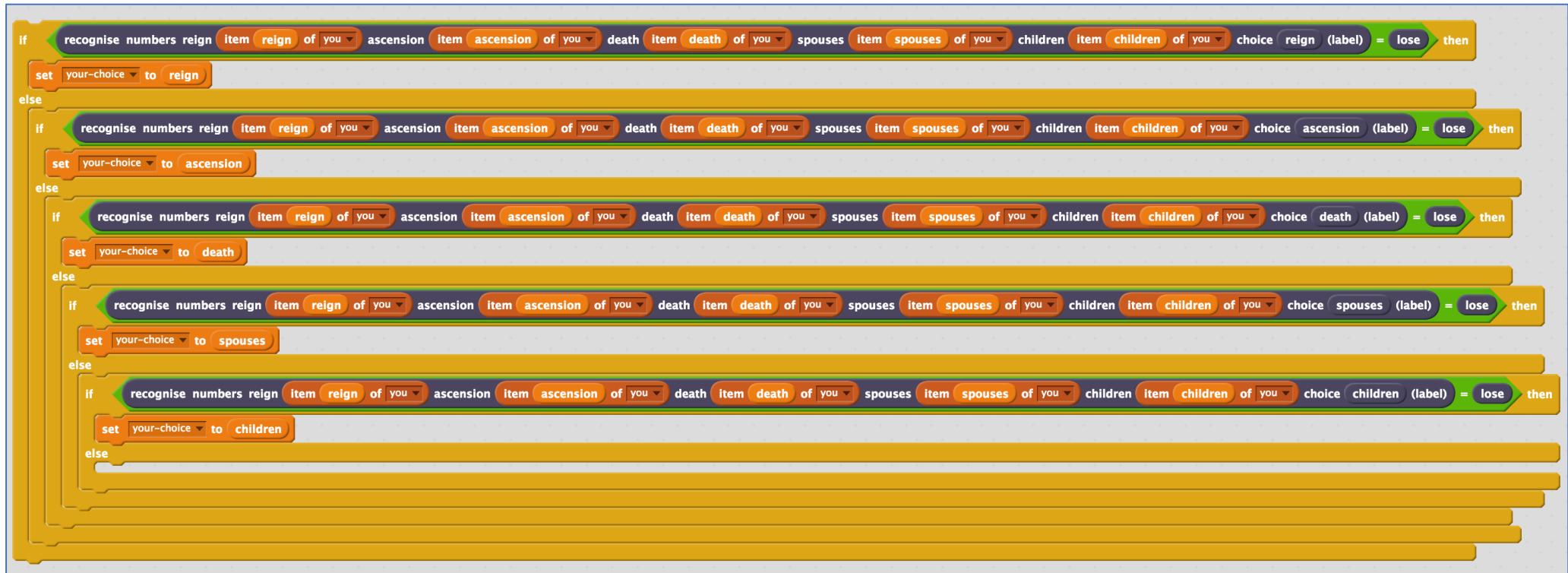
```
if [recognise numbers reign item reign of you v ascension item ascension of you v death item death of you v spouses item spouses of you v children item children of you v] choice reign (label) = lose then
  set [your-choice v] to [reign]
else
  if [recognise numbers reign item reign of you v ascension item ascension of you v death item death of you v spouses item spouses of you v children item children of you v] choice ascension (label) = lose then
    set [your-choice v] to [ascension]
  else
    
```

Step 3 – Duplicate again and change the choice and set-your-choice to “death”. Put that copy into the else block.

Repeat for “reign”

Repeat again for “children”.

You should end up with this:

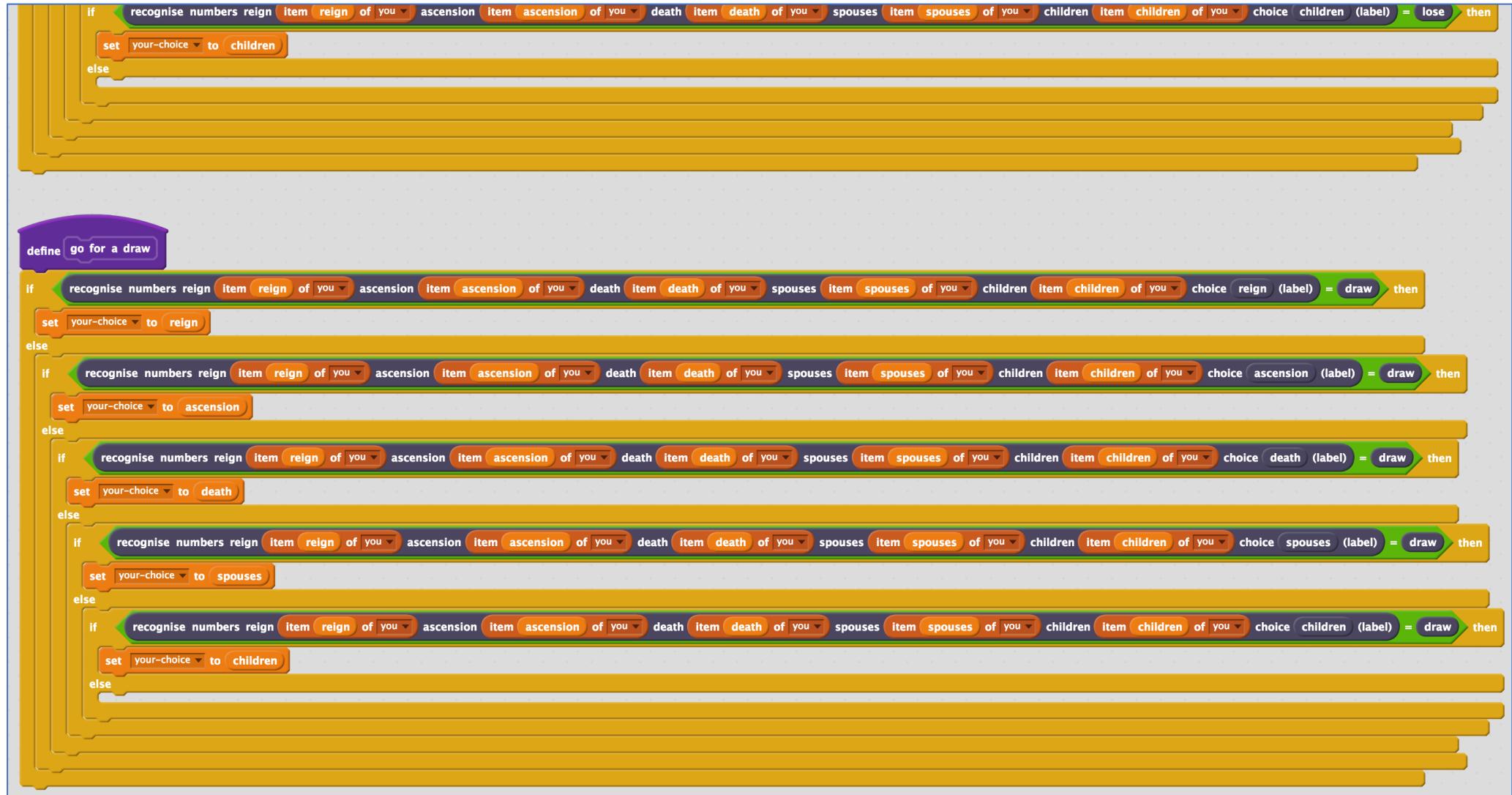


Step 4 – Click “Make a Block” and create a block called “go for a draw”.

Duplicate everything we just did into this new “go for a draw” block.

Change all the “lose” values to “draw”.

You should end up with this new block:



Step 5 – Add a random choice block in the last else of “go for a draw”

Step 6 – Join up the first script you made to “Let the computer choose which attribute to play”

Step 7 – Add “go for a draw” to the last else in “Let the computer choose which attribute to play”

```
define go for a draw
  if [recognise numbers reign item reign of you v ascension item ascension of you v death]
    set [your-choice v] to [reign]
  else
    if [recognise numbers reign item reign of you v ascension item ascension of you v death]
      set [your-choice v] to [ascension]
    else
      if [recognise numbers reign item reign of you v ascension item ascension of you v death]
        set [your-choice v] to [death]
      else
        if [recognise numbers reign item reign of you v ascension item ascension of you v spouses]
          set [your-choice v] to [spouses]
        else
          if [recognise numbers reign item reign of you v ascension item ascension of you v children]
            set [your-choice v] to [children]
          else
            set [your-choice v] to [pick random 1 to 5]
```

Step 5

```
define Let the computer choose which attribute to play
  if [recognise numbers reign item reign of you v ascension item ascension]
    set [your-choice v] to [reign]
  else
    if [recognise numbers reign item reign of you v ascension item ascension]
      set [your-choice v] to [ascension]
    else
      if [recognise numbers reign item reign of you v ascension item ascension]
        set [your-choice v] to [death]
      else
        if [recognise numbers reign item reign of you v ascension item ascension]
          set [your-choice v] to [spouses]
        else
          if [recognise numbers reign item reign of you v ascension item ascension]
            set [your-choice v] to [children]
          else
            go for a draw
```

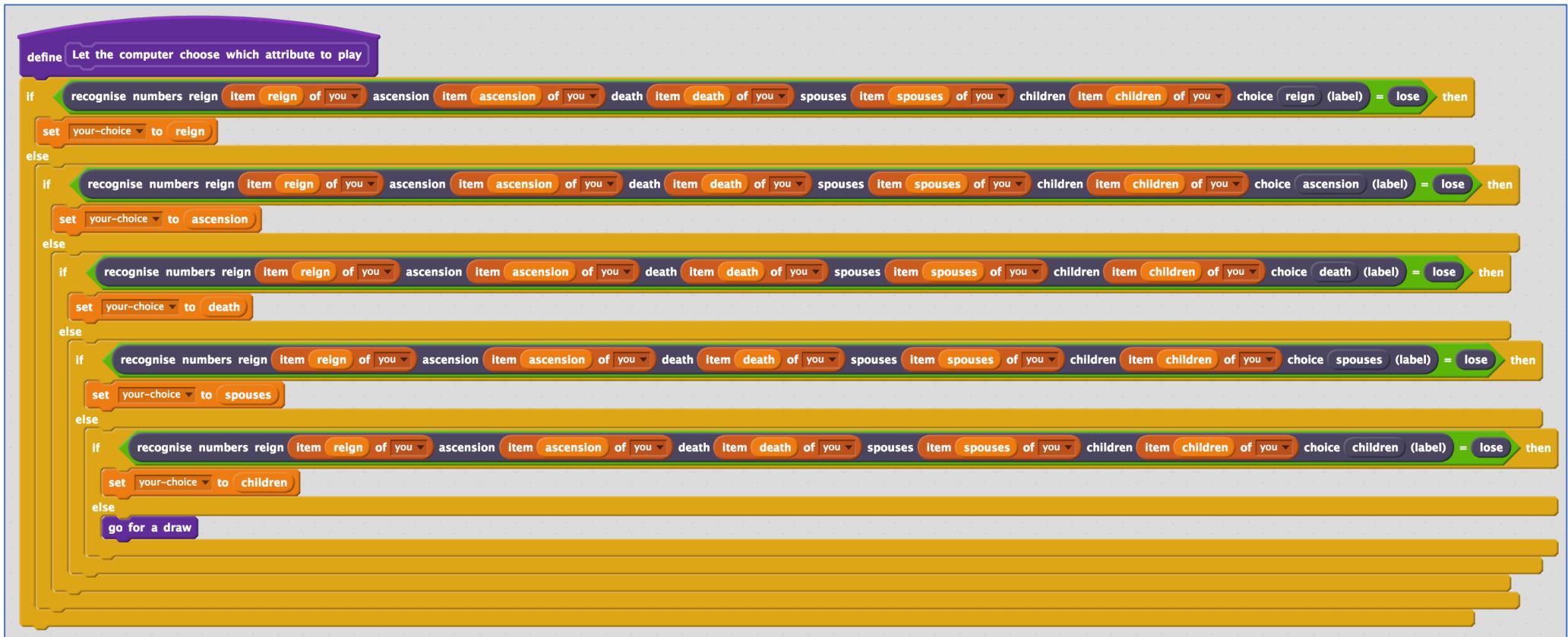
Step 6

Step 7

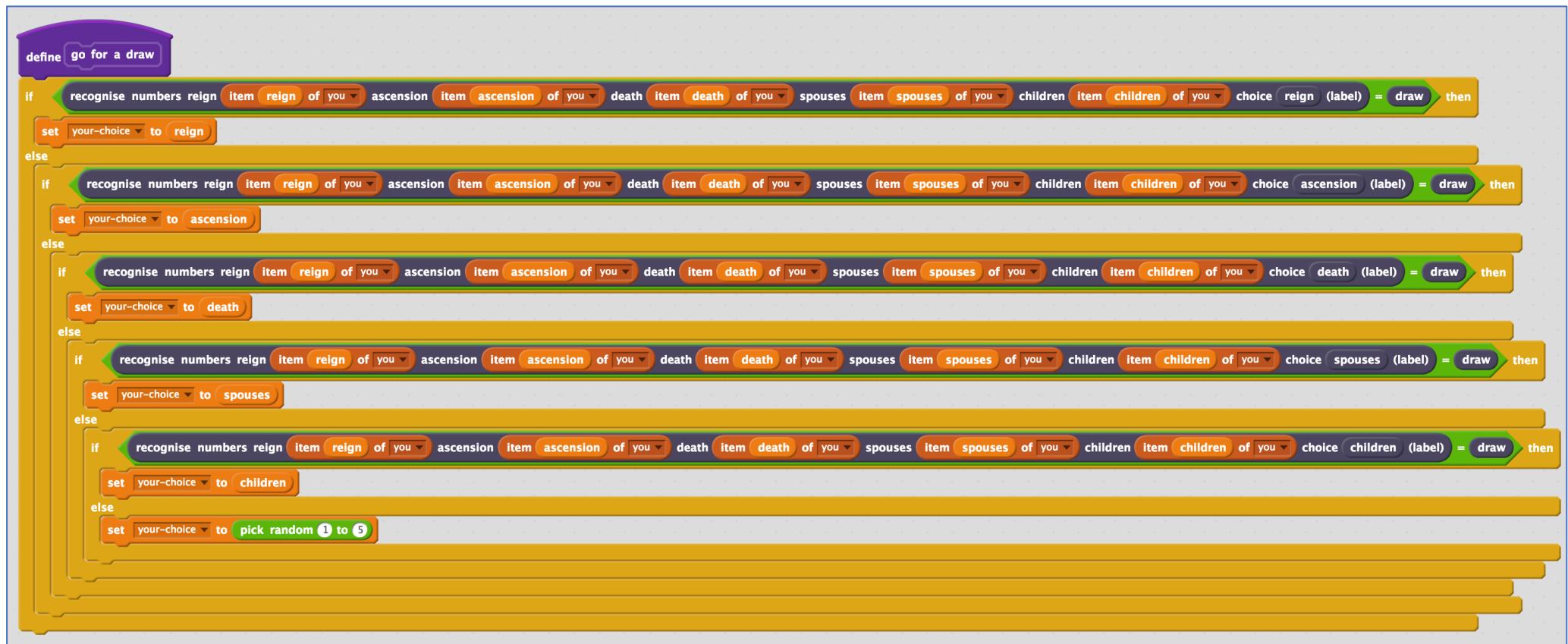
```
define go for a draw
  if [recognise numbers reign item reign of you v ascension item ascension]
    set [your-choice v] to [reign]
  else
    if [recognise numbers reign item reign of you v ascension item ascension]
```

018

The final script looks like this (part 1):



The final script looks like this (part 2):



That's it – well done!

It might be good to get someone to double-check your script at this point. It is easy to make a mistake.

34. Click the **fullscreen** button, and then click on the **green flag** again.

Play the game again.

Make a note of the score.

How good is the computer after learning from just five examples of each possible outcome?

35. Save your project.

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

What have you done so far?

You've modified your Scratch Top Trumps bot to use machine learning instead of your earlier random approach.

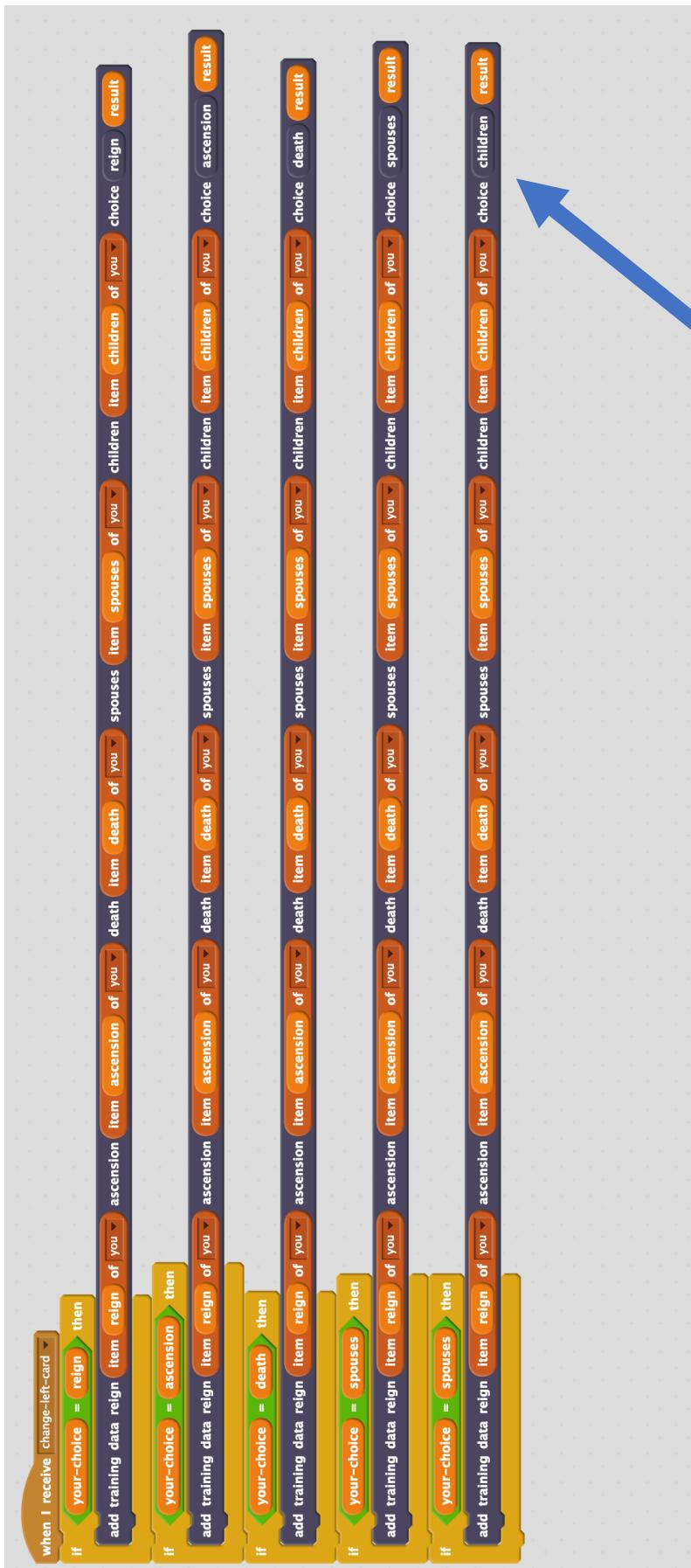
You haven't collected nearly enough examples to train a good model yet. The computer won't have seen enough examples of the game being played to have learned the types of values to expect, or the values that are more likely to win. Its predictions will often be wrong.

It needs more examples. Lots more examples.

Collecting, writing down, and then typing in the training examples was very slow. You probably don't want to have to do that a lot more.

Next, you'll modify the game to collect new training examples as part of playing the game. This will mean the more you play, the more examples the computer will have to learn from, and the better it should get.

36. Still in the “Stage”, in the same place as the scripts we wrote before, add the following script.



After every hand, the outcome of whether you won, lost or it was a draw will be added to the training data.

Choices should use the dark blue blocks from the “More Blocks” tab which represent the multiple-choice options

37. Save your project.

Click on *File* -> *Save Project*

38. Click the **fullscreen** button, and then click the **green flag** again.

Play the game again for a while to collect training examples.

Make a note of the score.

The longer you play the game, the more examples the computer will have to learn from.

39. Close the Scratch window.

40. Click the “< Back to project” link and then click the “Train” button.

41. You should see examples from your game in the training buckets.

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

Recognising numbers as win, draw or lose

[- Back to project](#)

[+ Add new label](#)

win

reign 33	reign 9	reign 9
ascension 43	ascension 25	ascension 59
death 76	death 34	death 88
spouses 1	spouses 1	spouses 1
children 9	children 1	children 6
choice death	choice ascension	choice death

reign 34	reign 21	reign 37
ascension 33	ascension 18	ascension 17
death 68	death 40	death 55
spouses 2	spouses 1	spouses 6
children 19	children 10	children 10
choice ascension	choice death	choice death

reign 9	reign 9	reign 25
ascension 25	ascension 25	ascension 28
death 34	death 34	death 52
spouses 1	spouses 1	spouses 1
children 1	children 1	children 8
choice spouses	choice children	choice spouses

reign 23	reign 12	reign 39
ascension 28	ascension 37	ascension 0
death 52	death 49	death 49
spouses 1	spouses 1	spouses 1
children 8	children 19	children 1
choice children	choice death	choice death

reign 9	reign 34	reign 25
ascension 25	ascension 21	ascension 44
death 34	death 56	death 70
spouses 1	spouses 1	spouses 1
children 1	children 9	children 6
choice death	choice reign	choice spouses

reign 25 reign 37 reign 50

[+ Add example](#)

draw

reign 56	reign 63	reign 3
ascension 9	ascension 18	ascension 51
death 45	death 51	death 57
spouses 1	spouses 1	spouses 2
children 9	children 9	children 20
choice spouses	choice spouses	choice spouses

reign 59	reign 10	reign 10
ascension 22	ascension 57	ascension 57
death 81	death 57	death 57
spouses 1	spouses 2	spouses 2
children 15	children 1	children 1
choice spouses	choice spouses	choice children

reign 20	reign 34	reign 10
ascension 38	ascension 21	ascension 57
death 59	death 56	death 67
spouses 1	spouses 1	spouses 2
children 10	children 8	children 1
choice spouses	choice reign	choice spouses

reign 10	reign 13	reign 13
ascension 57	ascension 30	ascension 30
death 67	death 43	death 43
spouses 2	spouses 0	spouses 0
children 1	children 0	children 0
choice children	choice spouses	choice reign

reign 15	reign 13	reign 15
ascension 43	ascension 28	ascension 24
death 76	death 52	death 48
spouses 1	spouses 1	spouses 1
children 9	children 1	children 3
choice ascension	choice reign	choice ascension

reign 9 reign 15 reign 15

[+ Add example](#)

lose

reign 44	reign 10	reign 10
ascension 25	ascension 57	ascension 57
death 59	death 47	death 47
spouses 1	spouses 2	spouses 2
children 0	children 1	children 1
choice death	choice spouses	choice children

reign 10	reign 15	reign 23
ascension 57	ascension 40	ascension 40
death 67	death 56	death 48
spouses 2	spouses 1	spouses 1
children 1	children 2	children 9
choice ascension	choice death	choice death

reign 33	reign 23	reign 0
ascension 43	ascension 28	ascension 38
death 76	death 52	death 65
spouses 1	spouses 1	spouses 1
children 9	children 1	children 3
choice ascension	choice reign	choice ascension

reign 15	reign 23	reign 9
ascension 24	ascension 24	ascension 34
death 56	death 48	death 34
spouses 1	spouses 1	spouses 1
children 2	children 9	children 1
choice reign	choice children	choice ascension

reign 15	reign 34	reign 22
ascension 40	ascension 10	ascension 10
death 56	death 56	death 33
spouses 1	spouses 2	spouses 2
children 2	children 8	children 0
choice reign	choice death	choice ascension

reign 9 reign 15 reign 22

[+ Add example](#)

42. Click the “< Back to project” link and then click on the “Learn & Test” button.

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

The screenshot shows a web page titled "Machine learning models". 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, there is a section titled "Machine learning models". On the left, there is a "What have you done?" section containing the following text:
You've trained a machine learning model to recognise when numbers are win, draw or lose.
You created the model on Tuesday, September 26, 2017 11:20 PM.
You've collected:

- 12 examples of draw,
- 16 examples of lose,
- 33 examples of win

On the right, there is a "What's next?" section containing the following text:
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!

44. Click on the “< Back to project” link and then click the “Scratch” button. Click the “Open in Scratch” button.

45. Open your saved project again.

*Use **File** -> **Load Project***

46. Play the game again.

Is it getting any better? Does the computer win more often now?

47. Repeat steps 38 – 46 to collect more examples, and then train a new machine learning model with them.

You need to close the Scratch window every time you train a machine learning model for Scratch to start using the new trained model.

What have you done?

You've modified your Scratch Top Trumps bot to use machine learning.

Instead of training the bot being a separate thing, you've made the computer learn by playing the game. This means you don't need to wait for the computer to have learned before it can start playing. It can start playing (even if it loses a lot at first), straight away. And by playing the game, it will learn from those experiences how to get better.

You haven't told the computer what to do, but allowed it to try out different choices and discover what choices are more likely to help it to win.

This is called "reinforcement learning". When it makes a good choice, this is reinforced by the computer being told that it has won.

An example of training a Top Trumps bot

Your results will be different to this.

But these were the results I got from training my bot.

	Score	
	Human	Computer
No training – computer choosing at random	72	28
Trained with 100 examples	47	53
Trained with 200 examples	38	62
Trained with 300 examples	29	71
Trained with 400 examples	25	75
Trained with 500 examples	27	73
Trained with 600 examples	29	71
Trained with 700 examples	27	73

In general, more training is better.

*There were times where the computer did worse after more training.
Why do you think that was?*

*After a certain point, the computer's scores stopped improving, even
after I kept adding more and more training.*

Why do you think that was?

*Compare these results with the results from your bot. How has your
bot learned from the training you've given it?*