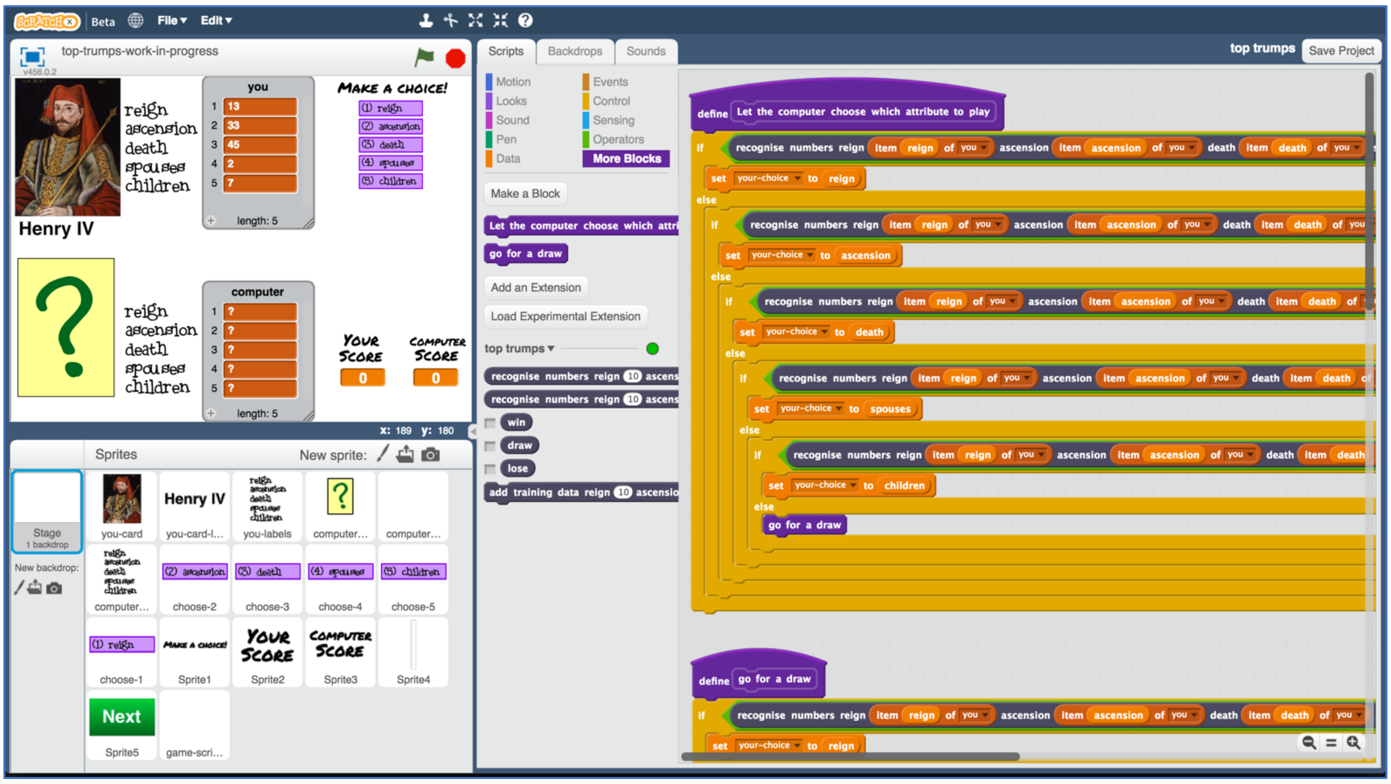
Top Trumps

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.



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

define go for a draw
if recognise numbers reign (item reign of you) ascension (item ascension of you) death (item death of you
    set your-choice to reign
```

What will you be doing?

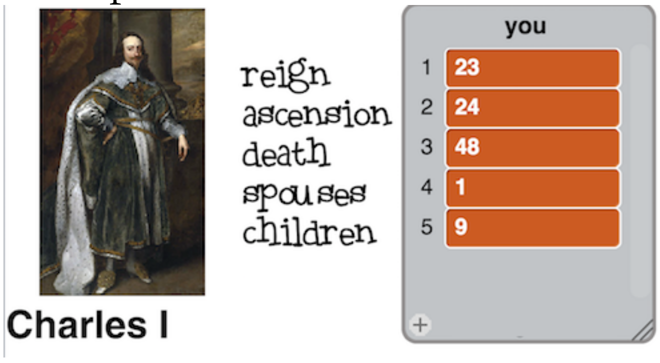
You'll be training a machine learning model to play a Top Trumps game based on Kings and Queens of England.

Each card is based on a different King or Queen.

Each card has five numbers on it:

Reign – how long they were king or queen for
Ascension – how old they were when they became king or queen
Death – how old they were when they died
Spouses – how many husbands or wives they had
Children – how many children they had

For example:



Charles I

1) He reigned for **23** years
If I choose this, and he was King longer than my opponent's card, I'll win
2) He ascended to the throne when he was **24**
If I choose this, and he became King earlier than my opponent's card, I'll win
3) He died when he was **48**
If I choose this, and he lived longer than my opponent's card, I'll win
4) He had **1** spouse
If I choose this, and he had more spouses than my opponent's card, I'll win
5) He had **9** children
If I choose this, and he had more children than my opponent's card, I'll win.

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

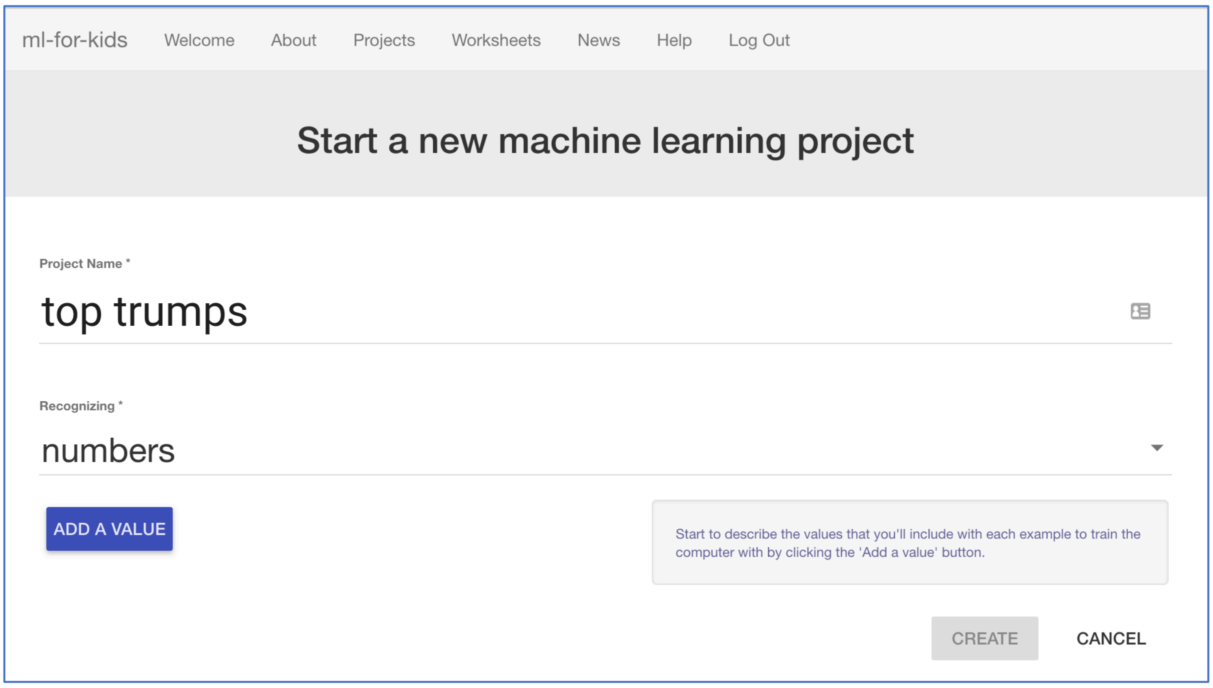
2. Click on "**Get started**"

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

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

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

6. Name your project "top trumps" and set it to learn how to recognise "**numbers**"



Start a new machine learning project

Project Name *
top trumps

Recognizing *
numbers

Start to describe the values that you'll include with each example to train the computer with by clicking the 'Add a value' button.

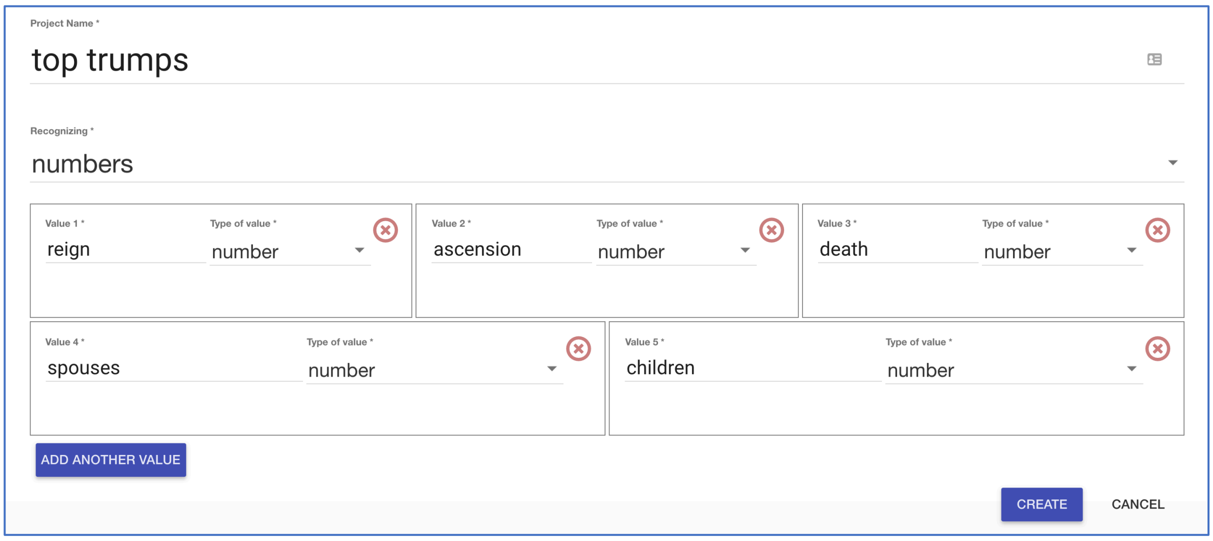
7. Click the "**Add a value**" button five times.
That will give you five text boxes to type names into.

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.

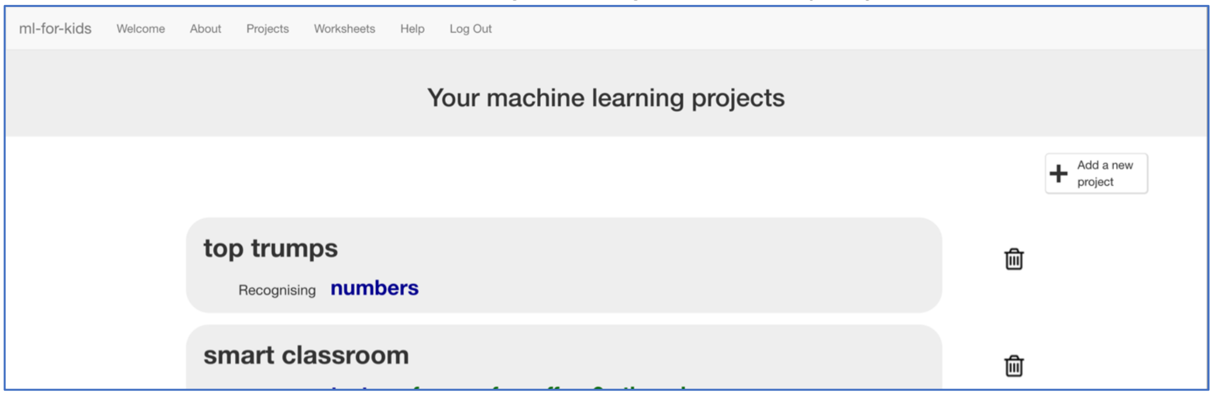

Project Name *
top trumps

Recognizing *
numbers

Value	Type of value
Value 1 *: reign	number
Value 2 *: ascension	number
Value 3 *: death	number
Value 4 *: spouses	number
Value 5 *: children	number

8. Click the "**Create**" button

9. You should now see "**top trumps**" in the projects list. Click on it.

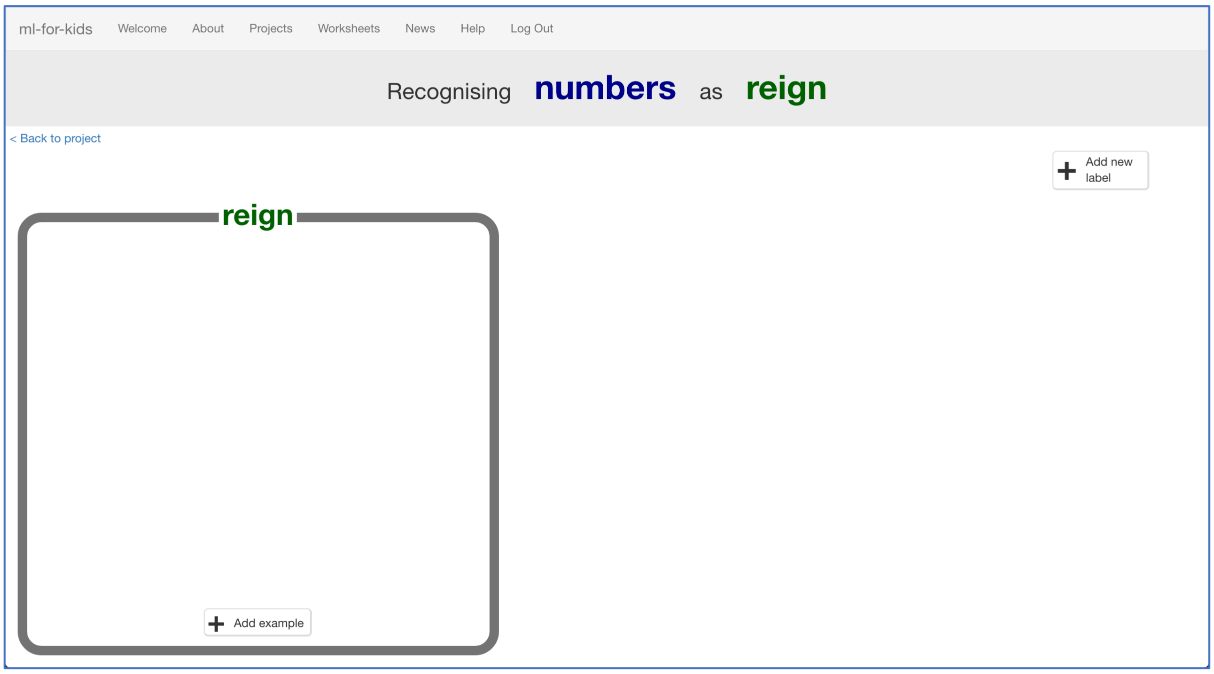


Your machine learning projects

top trumps
Recognising numbers

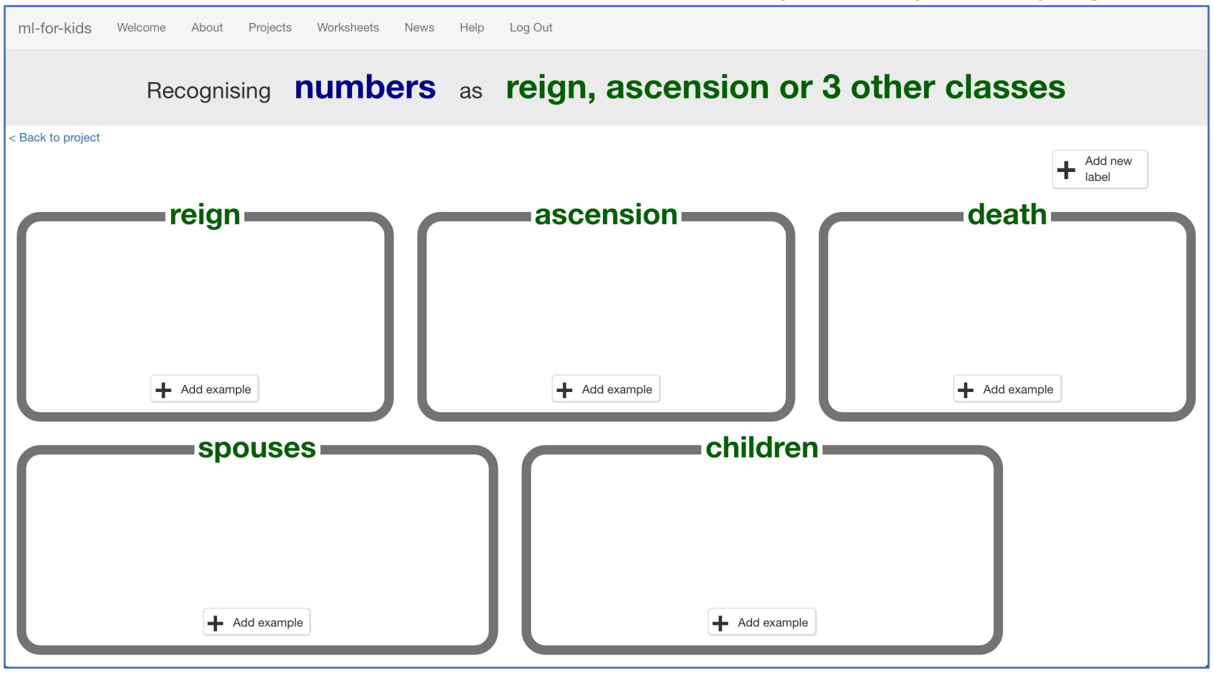
smart classroom

10. Click the "**+ Add new label**" button, and create a label called "reign"



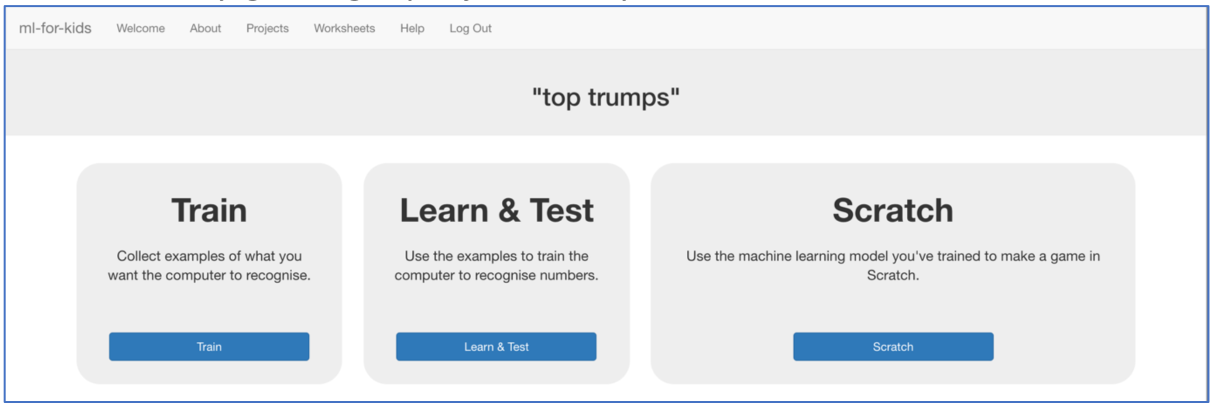
11. Use the "**+ Add new label**" button four more times to create labels for "ascension", "death", "spouses", and "children"

These are the choices available in each turn of the Top Trumps game.



12. Click the **"< Back to project**" link

13. Start by getting a project ready in Scratch. Click the **Scratch** button.



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

"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

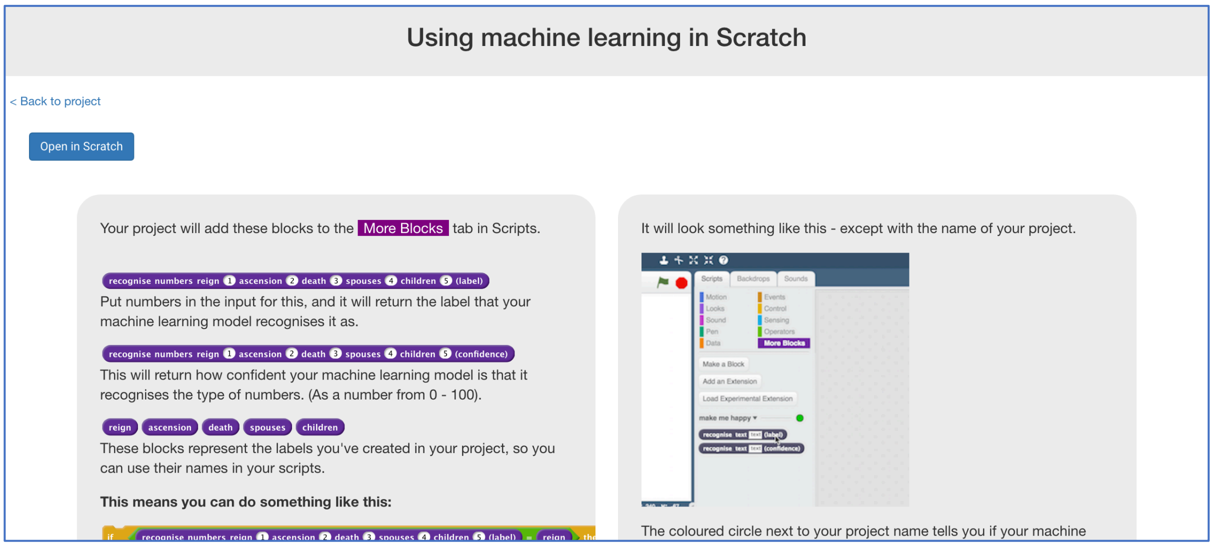
Scratch

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

Scratch

14. Click the "**Open in Scratch**" button

*It will warn you that you haven't trained a machine learning model yet, but that's okay as you'll be using Scratch to collect training data, so click the "**straight into Scratch**" button.*



Using machine learning in Scratch

< Back to project

Open in Scratch

Your project will add these blocks to the More Blocks tab in Scripts.

recognise numbers reign 1 ascension 2 death 3 spouses 4 children 5 (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 (confidence)

This will return how confident your machine learning model is that it recognises the type of numbers. (As a number from 0 - 100).

reign ascension death spouses children

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:

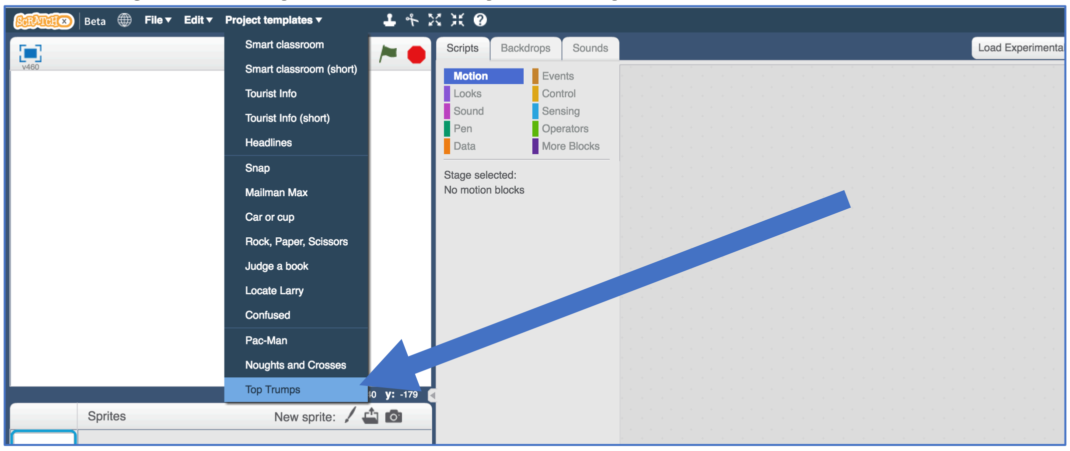
if recognise numbers reign 1 ascension 2 death 3 spouses 4 children 5 (label) = reign the

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

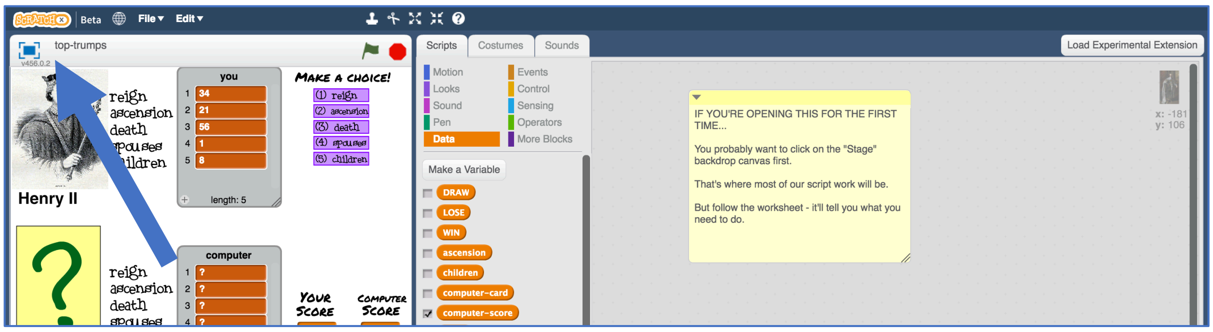
The coloured circle next to your project name tells you if your machine

15. Load the Top Trumps project template

*Use **Project templates -> Top Trumps** as shown below*



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



IF YOU'RE OPENING THIS FOR THE FIRST TIME...

You probably want to click on the "Stage" backdrop canvas first.

That's where most of our script work will be.

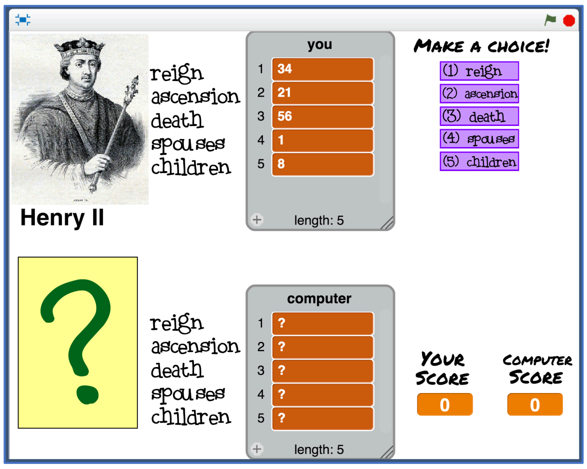
But follow the worksheet - it'll tell you what you need to do.

17. 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, 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**

When you choose, the computer card is revealed, and you see if you won or lost.

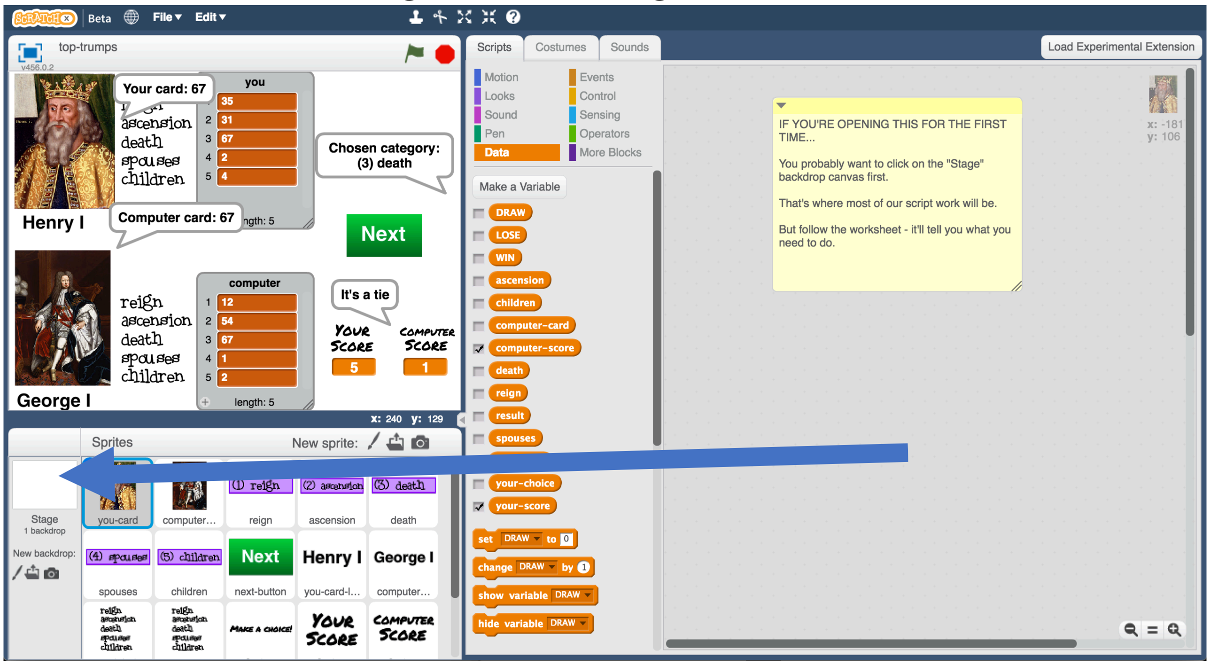
The score in the bottom right corner is updated.

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.

18. 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.

19. 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.



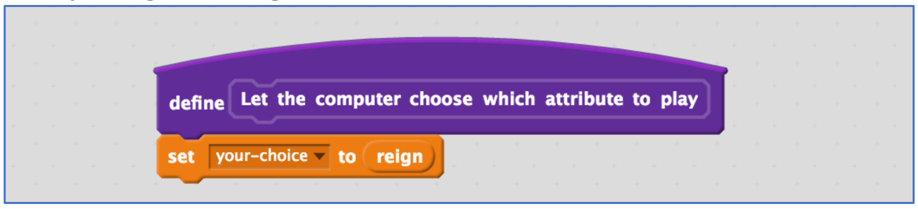
IF YOU'RE OPENING THIS FOR THE FIRST TIME...

You probably want to click on the "Stage" backdrop canvas first.

That's where most of our script work will be.

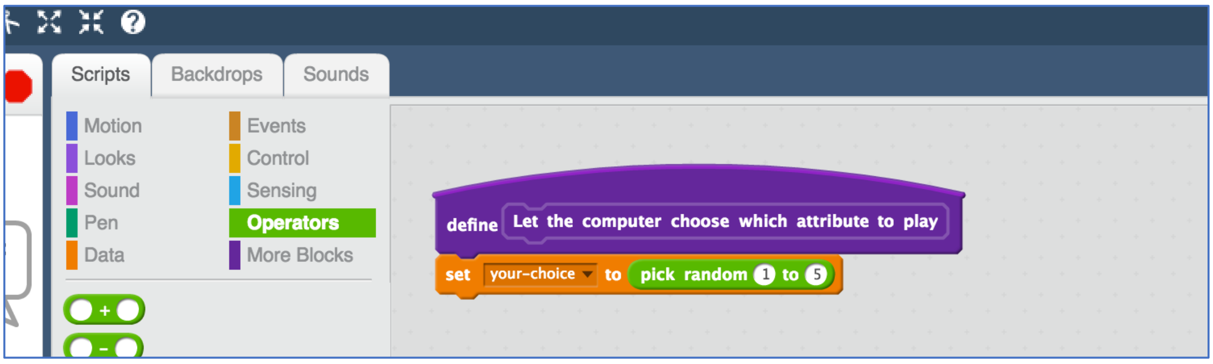
But follow the worksheet - it'll tell you what you need to do.

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



```
define Let the computer choose which attribute to play
set your-choice to reign
```

21. 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.



```
define Let the computer choose which attribute to play
set your-choice to pick random 1 to 5
```

22. 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?

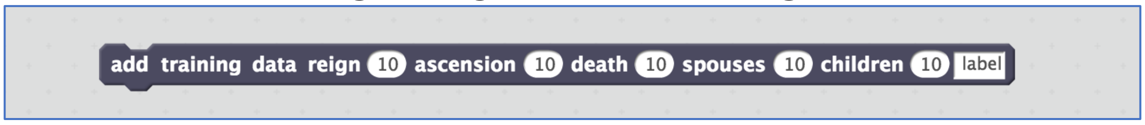
What have you done so far?

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

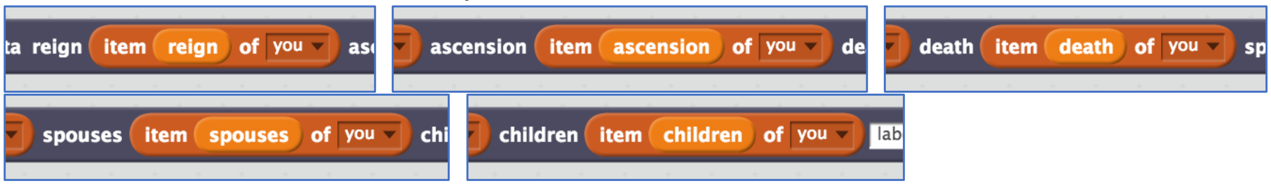
But people don't play like that. We learn how to choose which value would give us the best chance of winning. We do this based on the cards we've seen before, and on our understanding of the rules.

Next, you'll create a Scratch script that collects training examples using the moves that you make.

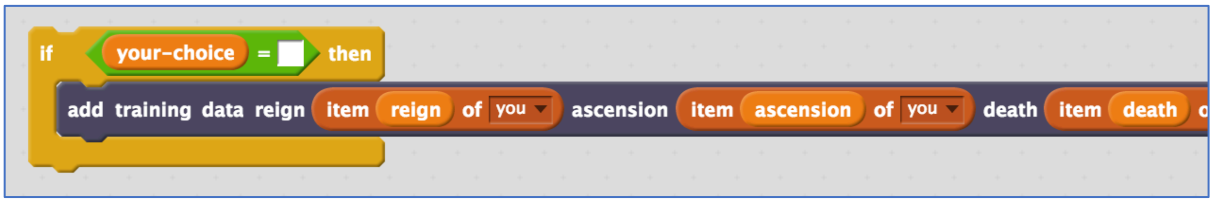
23. Still on the **Stage**, drag the "**add training data**" block to the canvas


add training data reign 10 ascension 10 death 10 spouses 10 children 10 label

24. Add the values from your card to the block


ta reign item reign of you as
ascension item ascension of you de
death item death of you sp
spouses item spouses of you chi
children item children of you lab

25. Surround the "add training data" with an "**if**" block like this:

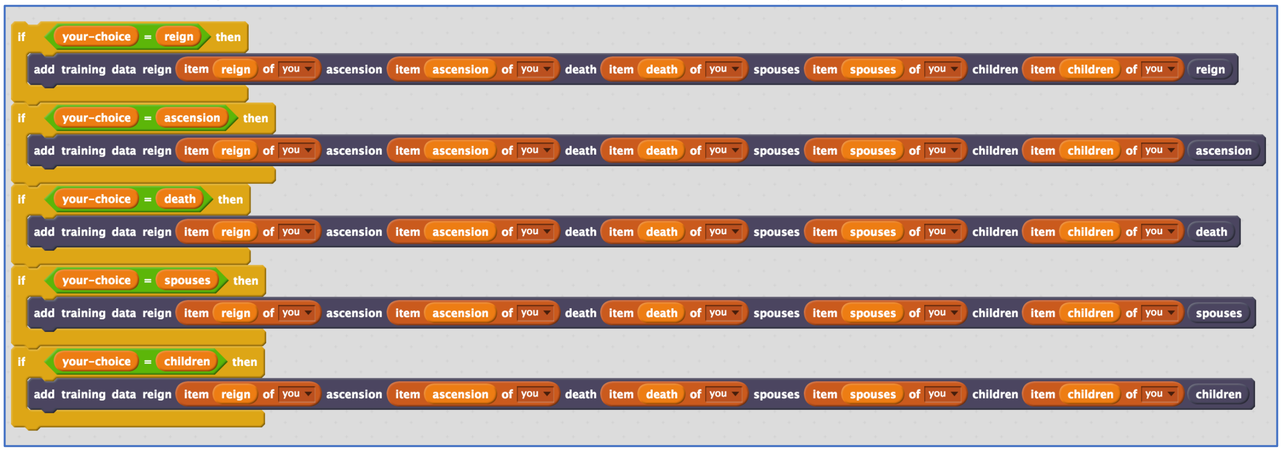

if your-choice = then
add training data reign item reign of you ascension item ascension of you death item death

26. Duplicate it five times – once for each possible choice of value

Right-click on the "if" and then choose "Duplicate"

27. Add the choices to each block

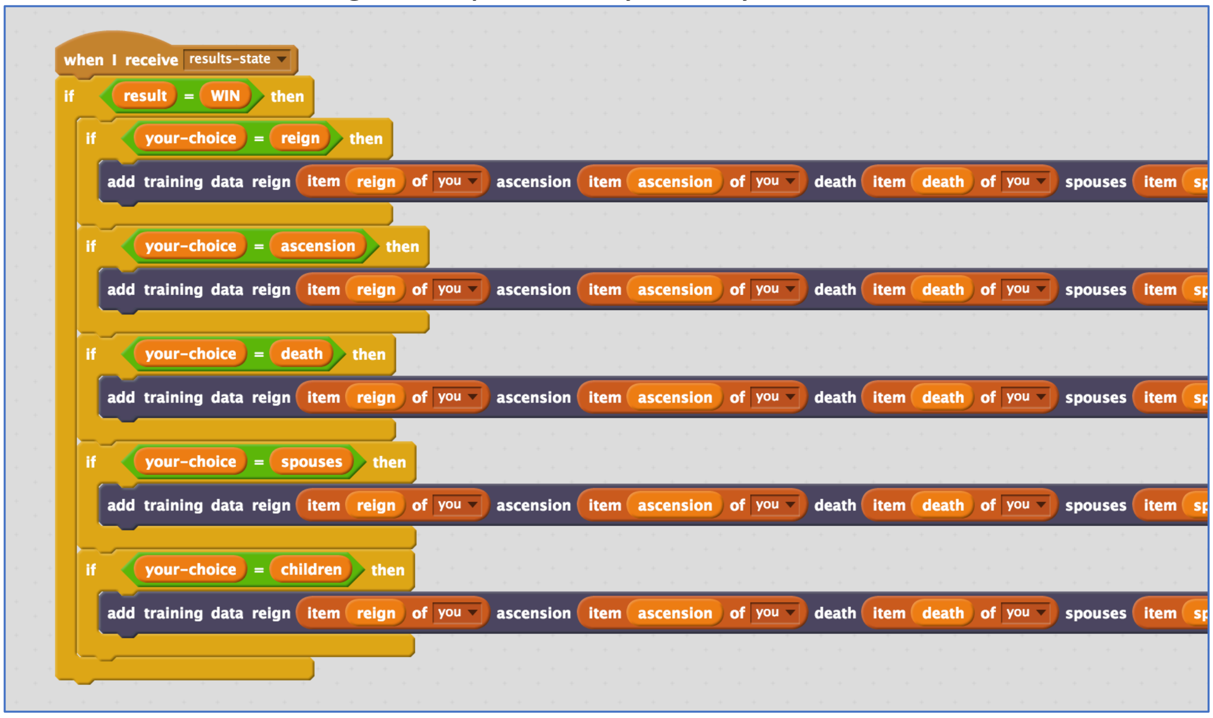
"if" choices should use orange options from "Data"
"add training data" choices should use dark options from "More blocks"



```
if your-choice = reign then
    add training data reign (item reign of you) ascension (item ascension of you) death (item death of you) spouses (item spouses of you) children (item children of you) reign
if your-choice = ascension then
    add training data reign (item reign of you) ascension (item ascension of you) death (item death of you) spouses (item spouses of you) children (item children of you) ascension
if your-choice = death then
    add training data reign (item reign of you) ascension (item ascension of you) death (item death of you) spouses (item spouses of you) children (item children of you) death
if your-choice = spouses then
    add training data reign (item reign of you) ascension (item ascension of you) death (item death of you) spouses (item spouses of you) children (item children of you) spouses
if your-choice = children then
    add training data reign (item reign of you) ascension (item ascension of you) death (item death of you) spouses (item spouses of you) children (item children of you) children
```

28. Finish the script so that this is called every time you win a hand

The values that were on your card, and the choice you made, will be added to the training examples every time you win.



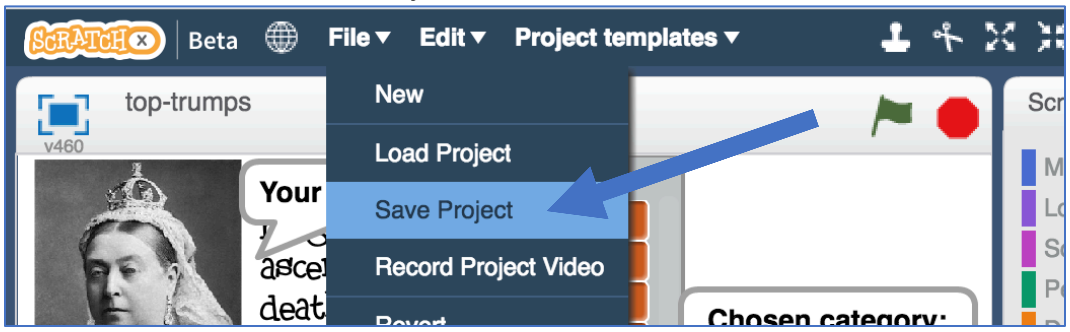
```
when I receive results-state
if result = WIN then
    if your-choice = reign then
        add training data reign (item reign of you) ascension (item ascension of you) death (item death of you) spouses (item sp
    if your-choice = ascension then
        add training data reign (item reign of you) ascension (item ascension of you) death (item death of you) spouses (item sp
    if your-choice = death then
        add training data reign (item reign of you) ascension (item ascension of you) death (item death of you) spouses (item sp
    if your-choice = spouses then
        add training data reign (item reign of you) ascension (item ascension of you) death (item death of you) spouses (item sp
    if your-choice = children then
        add training data reign (item reign of you) ascension (item ascension of you) death (item death of you) spouses (item sp
```

29. Click the full-screen button and green flag again.

Play games until your score reaches **10**.

30. Save your project

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



31. Leave the Scratch window open.

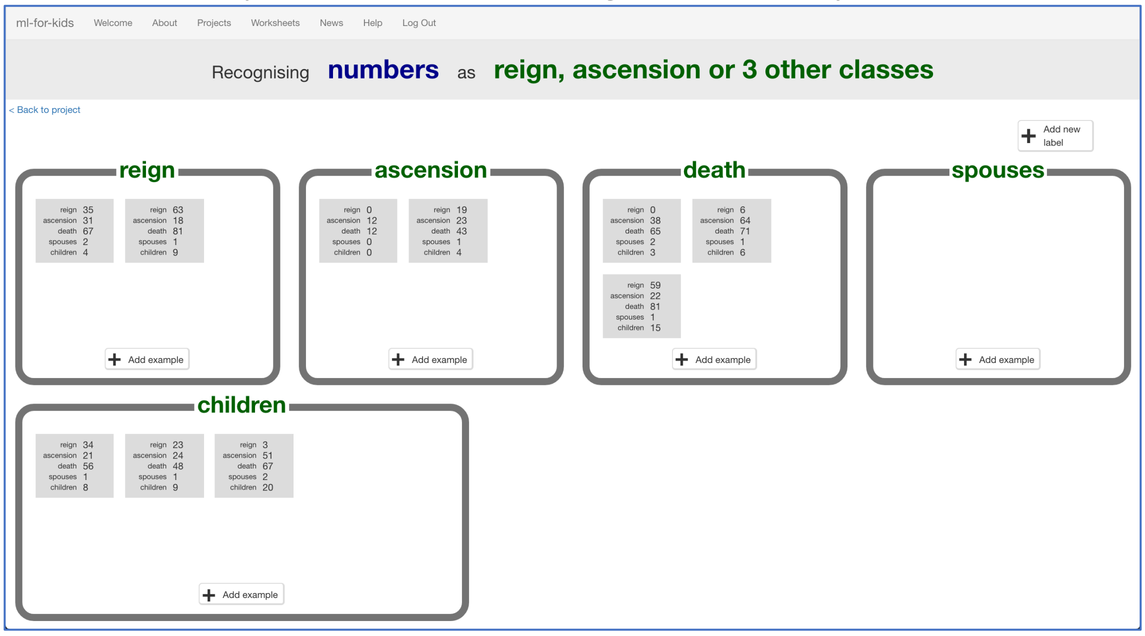
In the training window, click the "**< Back to project**" link.

32. Click the "**Train**" button

33. Check your training data

The ten winning moves that you made should have been added to your training examples.

Each example contains the numbers that were on your card. The bucket that the example is in is the winning choice that you made.



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

Recognising **numbers** as **reign, ascension or 3 other classes**

< Back to project

Add new label

reign

reign	35	63
ascension	31	18
death	67	81
spouses	2	1
children	4	9

Add example

ascension

reign	0	19
ascension	12	23
death	12	43
spouses	0	1
children	0	4

Add example

death

reign	0	6	59
ascension	38	64	22
death	65	71	81
spouses	2	1	1
children	3	6	15

Add example

spouses

Add example

children

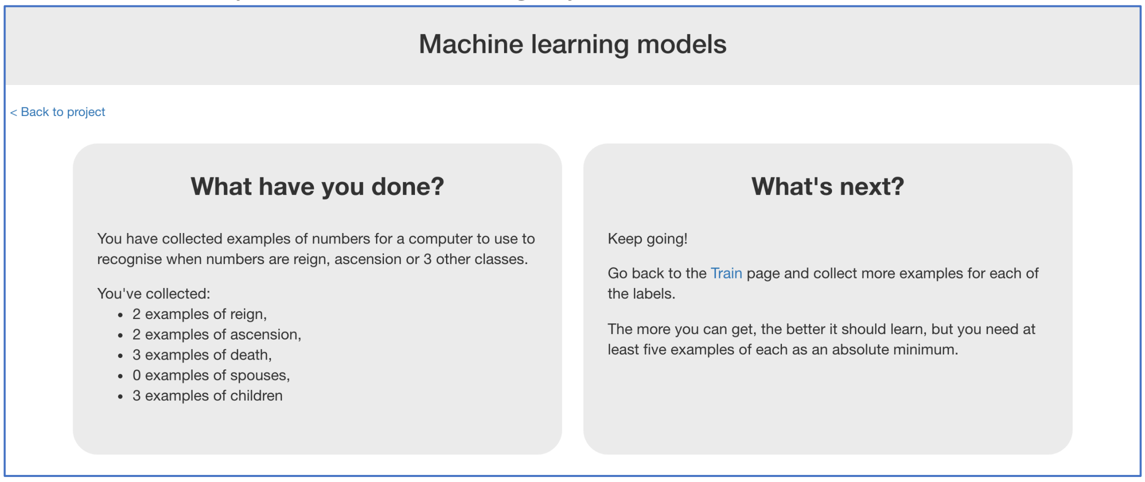
reign	34	23	3
ascension	21	24	51
death	56	48	67
spouses	1	1	2
children	8	9	20

Add example

34. Click the "**< Back to project**" link. Then click "**Learn & Test**"

35. The training page won't let you train a model yet

The ten examples aren't enough yet to train a model.



Machine learning models

< Back to project

What have you done?

You have collected examples of numbers for a computer to use to recognise when numbers are reign, ascension or 3 other classes.

You've collected:

- 2 examples of reign,
- 2 examples of ascension,
- 3 examples of death,
- 0 examples of spouses,
- 3 examples of children

What's next?

Keep going!

Go back to the Train page and collect more examples for each of the labels.

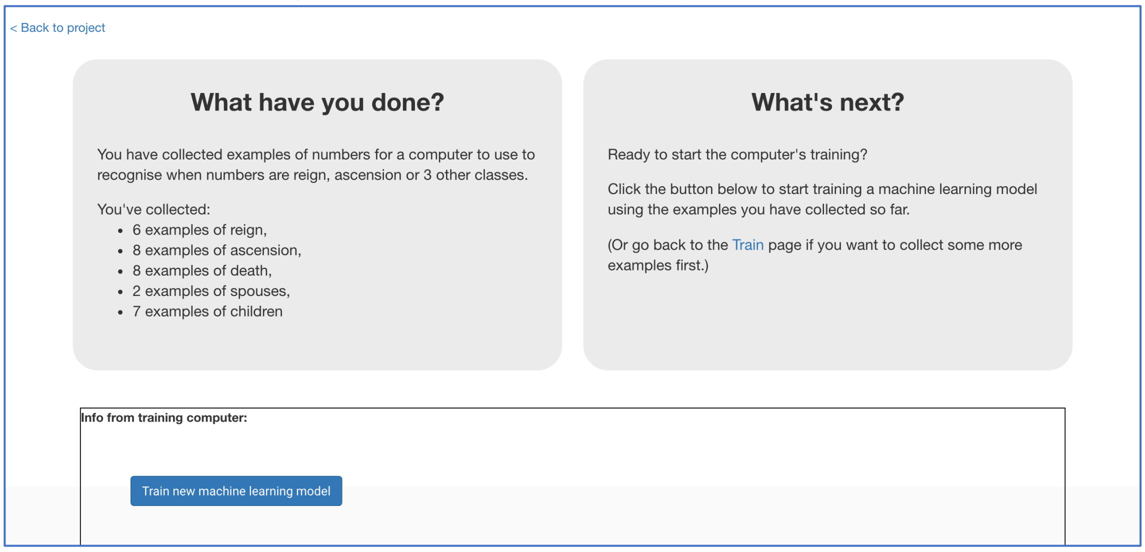
The more you can get, the better it should learn, but you need at least five examples of each as an absolute minimum.

36. Leave the page open.

Go back to your game in Scratch, and play more games.

37. After a while, go back to the "Machine learning models" page, and refresh the page.

Keep doing this until you've got enough examples for the "**Train new machine learning model**" button to appear.



< Back to project

What have you done?

You have collected examples of numbers for a computer to use to recognise when numbers are reign, ascension or 3 other classes.

You've collected:

- 6 examples of reign,
- 8 examples of ascension,
- 8 examples of death,
- 2 examples of spouses,
- 7 examples of children

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 have collected so far.

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

Info from training computer:

38. Close the Scratch window.

What have you done so far?

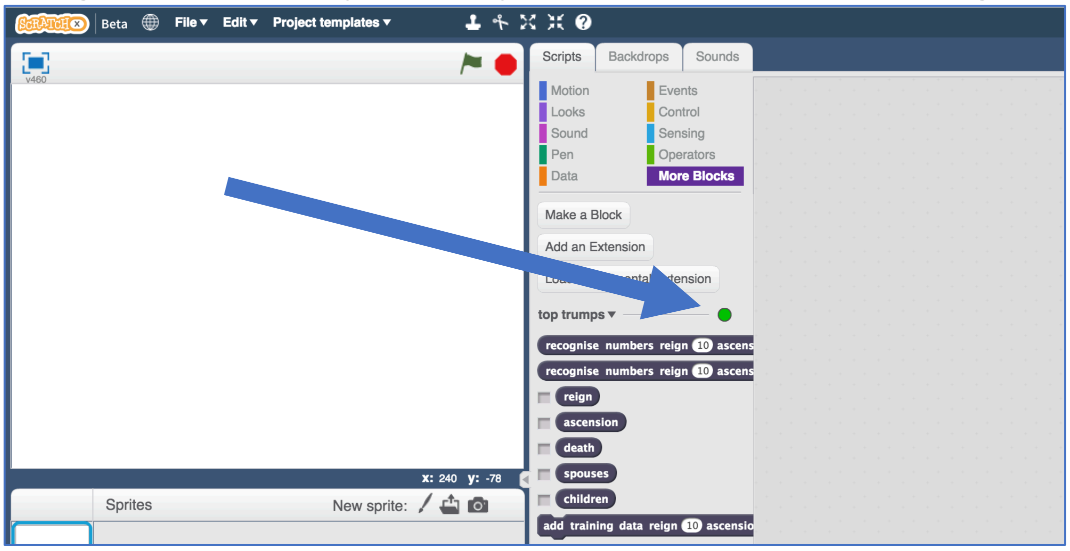
You've started to train a computer to learn about Top Trumps.

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 are likely to help it to win, without you needing to tell it what the rules are.

39. Click the "**< Back to project**" link. Then click the "**Scratch**" button

The green icon tells you that you have a machine learning model this time.

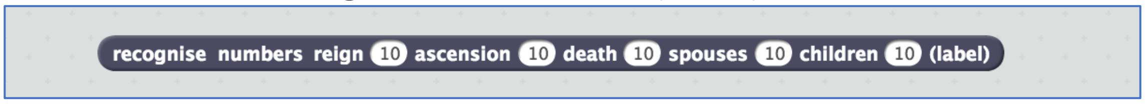


40. Open the Scratch project you saved before

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

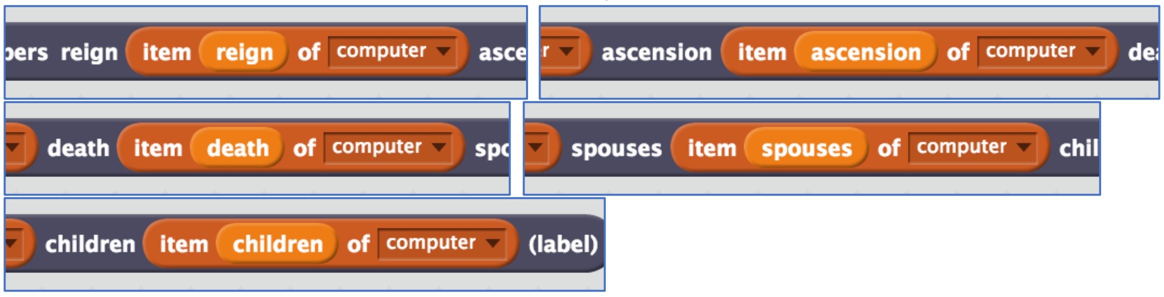
41. Click on the "**Stage**" to get to where you added the script before

42. Add the “recognise numbers … (label)” block



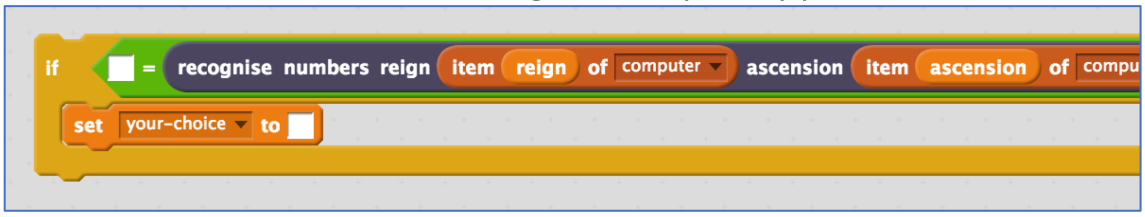
```
recognise numbers reign 10 ascension 10 death 10 spouses 10 children 10 (label)
```

43. Add the values from the computer’s card to the block



```
bers reign item reign of computer asce
r ascension item ascension of computer de
death item death of computer spo
spouses item spouses of computer chil
children item children of computer (label)
```

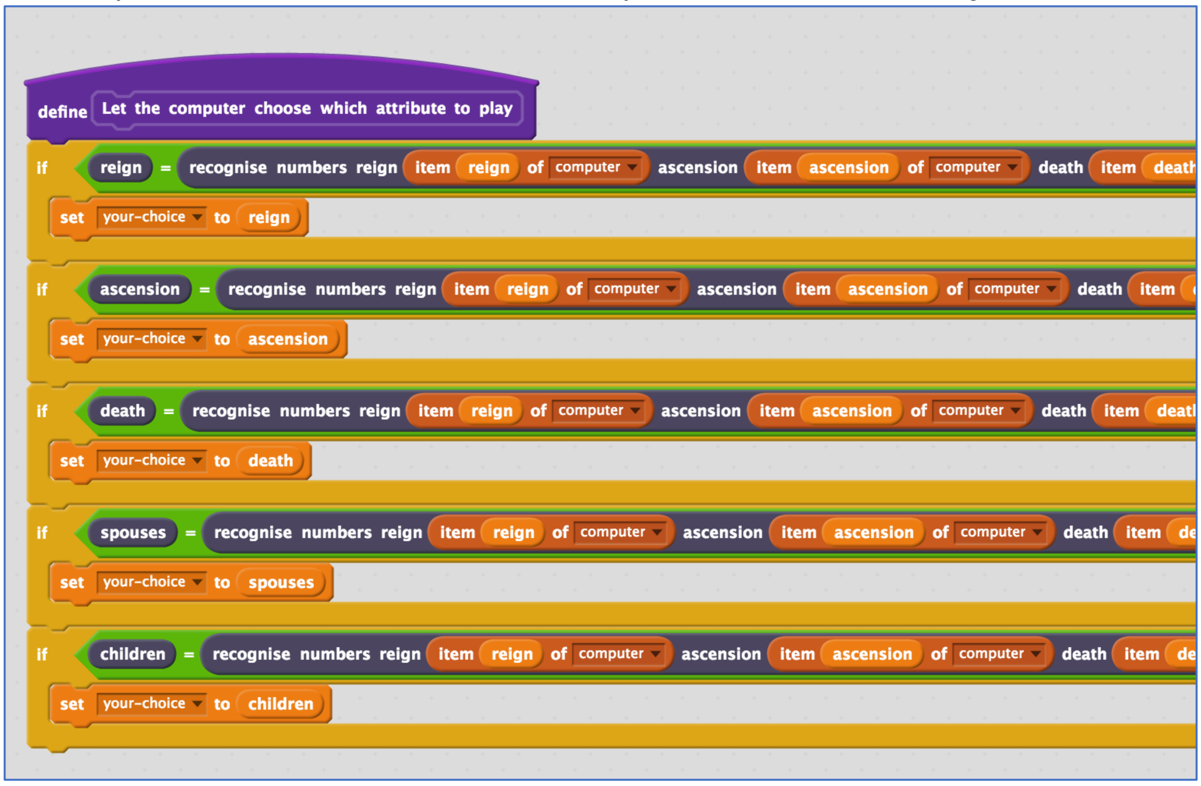
44. Add it to an “if” block to get a script snippet that looks like this:



```
if < [ ] = recognise numbers reign item reign of computer ascension item ascension of compu
    set your-choice to [ ]
```

45. Duplicate the snippet 5 times, and use it to make a new script for “**Let the computer choose which attribute to play**”

This replaces the random choice script that was there before



```
define Let the computer choose which attribute to play
if < reign = recognise numbers reign item reign of computer ascension item ascension of computer death item death
    set your-choice to reign
if < ascension = recognise numbers reign item reign of computer ascension item ascension of computer death item
    set your-choice to ascension
if < death = recognise numbers reign item reign of computer ascension item ascension of computer death item deatl
    set your-choice to death
if < spouses = recognise numbers reign item reign of computer ascension item ascension of computer death item de
    set your-choice to spouses
if < children = recognise numbers reign item reign of computer ascension item ascension of computer death item de
    set your-choice to children
```

Last updated: 31 May 2018

46. Save your project
*Click "**File**" -> "**Save Project**"*

47. Play the game until you reach the score of **20**

What have you done so far?

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

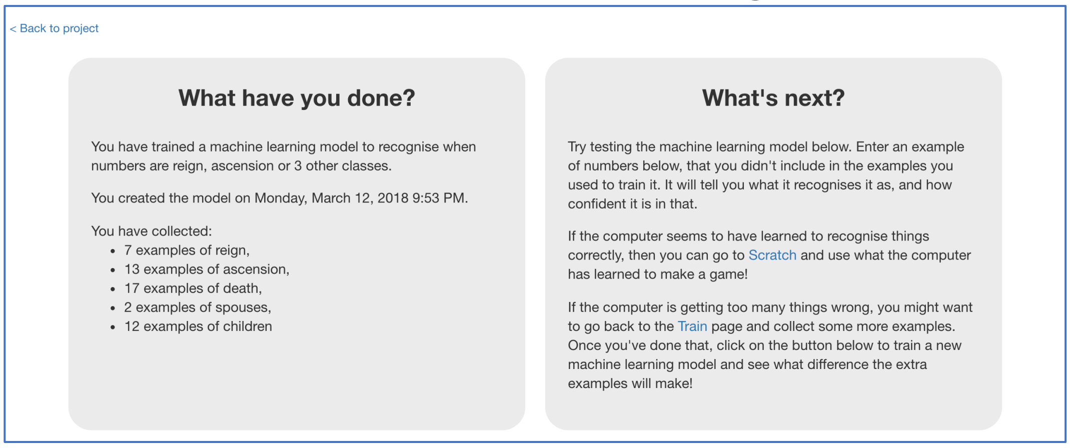
You haven't collected nearly enough examples to train a good model yet. The computer won't have seen enough examples of the game being played to have learned the types of values to expect, or the values that are more likely to win. Its predictions will often be wrong.

To get better, it needs more examples. Lots more examples.

48. Go back to the training tool window

49. Click the **"< Back to project**" link, then back to "**Learn & Test**"

50. Click the "**Train new machine learning model**" button


< Back to project

What have you done?

You have trained a machine learning model to recognise when numbers are reign, ascension or 3 other classes.

You created the model on Monday, March 12, 2018 9:53 PM.

You have collected:
- 7 examples of reign,
- 13 examples of ascension,
- 17 examples of death,
- 2 examples of spouses,
- 12 examples of children

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 difference the extra examples will make!

51. Switch back to the Scratch window.
If you accidentally closed it, you can get back to it by doing this:
** Click the "**< Back to project**" link*
** Click the "**Scratch**" button*
** Click the "**Open in Scratch**" button*
** Open the Scratch project you saved before, with "**File**" -> "**Load Project**"*

52. Play the game again.
Is it getting any better? Does the computer win more often now?

53. Repeat steps 48 – 52 to collect more examples, and then train a new machine learning model with them. Do this a few times.

What have you done so far?

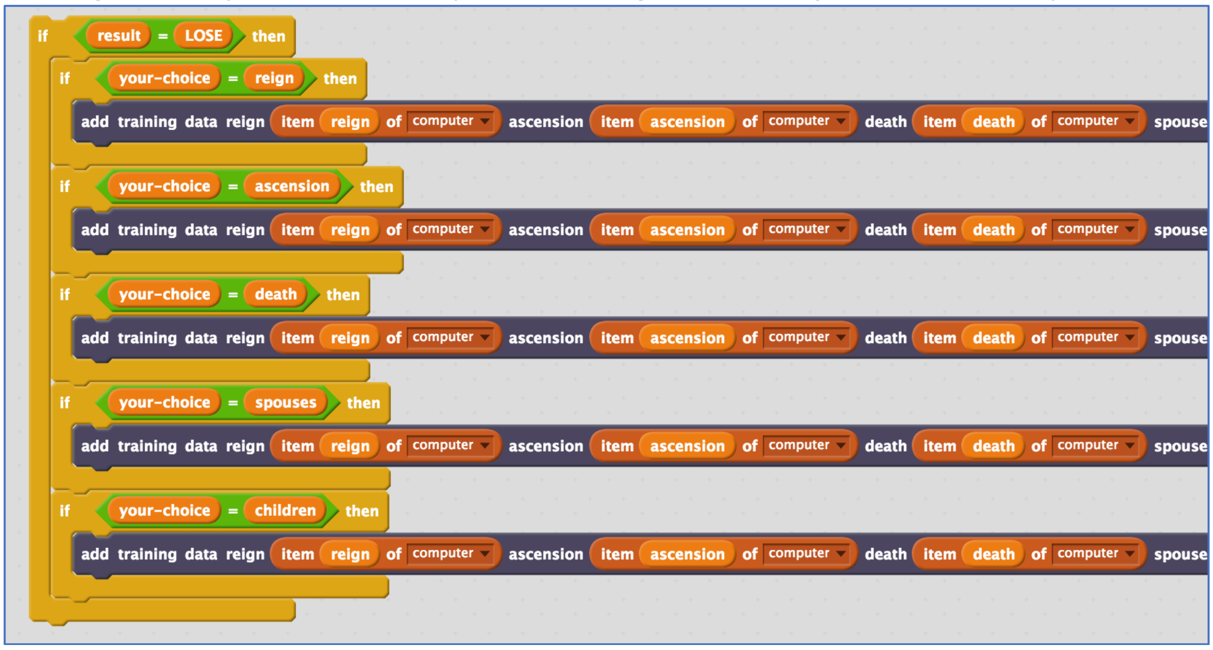
The computer is only learning from the decisions that you make. To speed up the collection of training examples, next you'll let the computer learn from it's own moves as well.

54. Open the game in Scratch

55. Find the "When I receive 'results-state'" script in the Stage that you made earlier

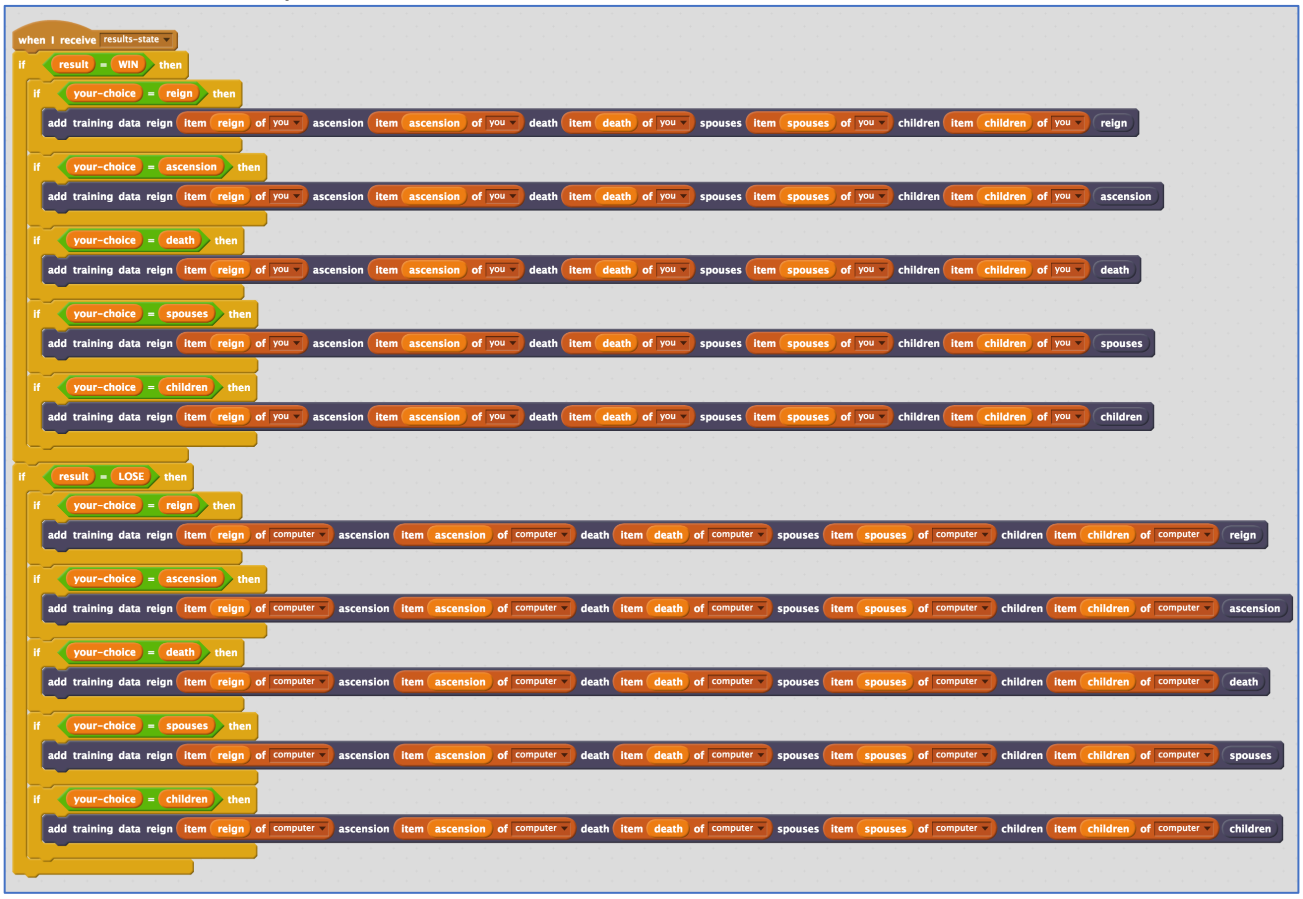
56. Duplicate the "if result = WIN" block

57. Modify it so that the computer's moves can be added to the training examples when the computer scores a point.
Change "WIN" to "LOSE" (you lose when the computer scores a point)
Change the "you" to "computer" (to get values from the computer card)



```
if result = LOSE then
    if your-choice = reign then
        add training data reign item reign of computer ascension item ascension of computer death item death of computer spouse
    if your-choice = ascension then
        add training data reign item reign of computer ascension item ascension of computer death item death of computer spouse
    if your-choice = death then
        add training data reign item reign of computer ascension item ascension of computer death item death of computer spouse
    if your-choice = spouses then
        add training data reign item reign of computer ascension item ascension of computer death item death of computer spouse
    if your-choice = children then
        add training data reign item reign of computer ascension item ascension of computer death item death of computer spouse
```

58. Join it all up



```
when I receive results-state
if result = WIN then
    if your-choice = reign then
        add training data reign item reign of you ascension item ascension of you death item death of you spouses item spouses of you children item children of you reign
    if your-choice = ascension then
        add training data reign item reign of you ascension item ascension of you death item death of you spouses item spouses of you children item children of you ascension
    if your-choice = death then
        add training data reign item reign of you ascension item ascension of you death item death of you spouses item spouses of you children item children of you death
    if your-choice = spouses then
        add training data reign item reign of you ascension item ascension of you death item death of you spouses item spouses of you children item children of you spouses
    if your-choice = children then
        add training data reign item reign of you ascension item ascension of you death item death of you spouses item spouses of you children item children of you children
if result = LOSE then
    if your-choice = reign then
        add training data reign item reign of computer ascension item ascension of computer death item death of computer spouses item spouses of computer children item children of computer reign
    if your-choice = ascension then
        add training data reign item reign of computer ascension item ascension of computer death item death of computer spouses item spouses of computer children item children of computer ascension
    if your-choice = death then
        add training data reign item reign of computer ascension item ascension of computer death item death of computer spouses item spouses of computer children item children of computer death
    if your-choice = spouses then
        add training data reign item reign of computer ascension item ascension of computer death item death of computer spouses item spouses of computer children item children of computer spouses
    if your-choice = children then
        add training data reign item reign of computer ascension item ascension of computer death item death of computer spouses item spouses of computer children item children of computer children
```

What have you done?

You've made a Top Trumps bot that can 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?