Creative AI: Creative Applications of Machine Learning

Simple can be powerful

- This session is about familiarising yourself with the basics of machine learning
- It’s also about getting a better understanding of what machine learning systems do, and how they can be manipulated for creative purposes
- As a result, we are going to focus on simple machine learning algorithms, rather than more complex ones.
- However, what we will learn is just how powerful even simple machine learning systems can be!

Machine Learning

- Some of these approaches are a little bit complicated when you’re starting out, and it’s easy to not fully understand what you’re doing.
- Also, some machine learning approaches require some extra hardware.
- We’re not going to cover some of these generative approaches in this session, as it’s just an introduction.

Getting Comfortable

- First, we’re going to concentrate of getting comfortable with basic concepts.
- These are surprisingly easy to get started with using a little bit of JavaScript, and some starter code.
- We’re going to have a look at some code examples that make it easier to get started, without introducing too many complex ideas.

Supervised learning using RAPID Lib

- We are mostly going to be using what we call "supervised" learning approaches.

- Supervised learning uses **examples** to train models

- The supervised learning algorithms that will be covered in the course can be broadly grouped into two types: Classification and Regression.

Supervised learning using RAPID Lib

- We will be using a javascript machine learning library called RAPIDLib

- We will also get ‘control data’ from sliders, the mouse and keyboard and other places.

Supervised learning using RAPID Lib

- For this first part we are just going to get comfortable with using a machine learning system called RapidLib.

- RapidLib takes an input, for example the mouse, and records it alongside some output.

- We will then train a model and use this to map 2 dimensional mouse input (x,y) to control 4 synthesis parameters.

rapid-lib time!

- https://mimicproject.com/guides/RAPIDMIX
- This explains how you can start to create your own machine learning system very quickly without any prior experience.

How it works

- Use the example below to randomise synth parameters until you find a sound you like.
    - You can also use the sliders to control the sound in more detail
- Move the mouse to somewhere on the screen, then click and hold to record samples.
- Find more sounds and place them elsewhere on the screen and record again.

How it works

- When you have a few, hit “t" to train.
- Now when you move the mouse around you will be able to explore your newly trained sound space!

How it works

- Try clearing the dataset (“x")
- change the training examples to see how this changes the behaviour of the system after it is re-trained.
- Spend some time experimenting with the trained models to start to understand how the models are changed.

let’s take a look

- Open Project

Recap

- This week, we learned how to create a basic dataset, and train a simple regression model to explore and control a range of sounds created by a synthesiser.
- We didn't need to worry that much about the actual synthesiser controls (the parameters), or what they did - instead we used machine learning to make up a range of sounds based on some random examples.
- We then moved through the sound world produced by the machine learning system using some simple inputs.

Classification

- We’re going to look in more detail at Classification
- We’ll explore how classification models work using a simple visualisation
- We’ll also use machine listening technology and simple classification model to detect different types of sounds automatically.

Classification

- When relationship between your inputs and your outputs is complex, classification algorithms can allow you to build things that would be challenging or impossible to program yourself by hand.

- This is what makes machine learning so exciting - you can create highly complex systems without having to program them manually.

- This is really powerful, and also pretty simple.

Classification

- Now we're going to create a very simple classification model.
- We are going to use something called "K-Nearest Neighbour" to classify some inputs.
- So what is K-Nearest Neighbour?

K-Nearest Neighbour

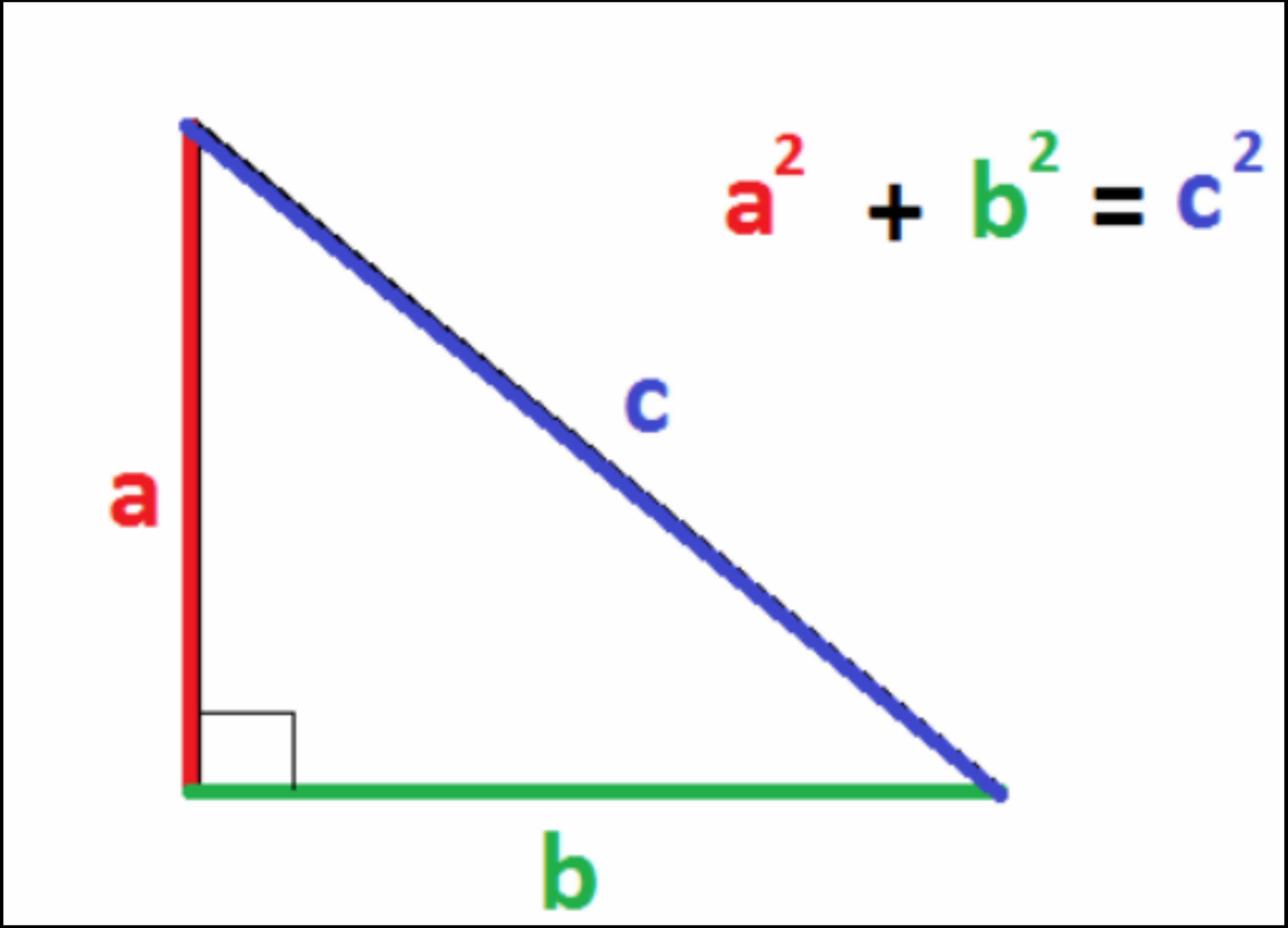
- K-Nearest Neighbour, or KNN, can be used for Classification or Regression.
- We’re going to use it for Classification.
- This means it’s going to tell us what kind of input we have, based on its similarity to other inputs we’ve told the model about.

K-Nearest Neighbour

- KNN does this by comparing the current input with a previous stored inputs using a distance measurement.
- One simple example of a distance measurement is **the euclidean distance.**
- You definitely already know what this is, but possibly haven’t thought about it this way.

Distance

- Getting the distance between two points in 2D is identical to working out the length of the hypotenuse of a right-angled triangle, with the two points being at each end.
- Square the two straight sides, add them, and the square root of this is the length of the diagonal side.
- Remember??????
- This works no matter how many dimensions your shape has. It's a fundamental concept that is used all the time.


$a^2 + b^2 = c^2$
c
a
b

K-Nearest Neighbour

- The KNN gives a new input the same ‘class’ or label as a certain number of its nearest neighbours.
- The number of neighbours that it checks against is a small value, represented by the letter K.

K-Nearest Neighbour

- If K = 1, then the new input is given the same class as its nearest neighbour.

- If K is > 1, say 5, then the 5 nearest neighbours are checked. The new input gets the most common class of those nearest neighbours. Often they are weighted based on how similar they are.

K-Nearest Neighbour

- K-Nearest Neighbour isn’t that great, but it is super simple and pretty effective for the stuff we’re doing today.
- When deciding on an approach to take, we compared algorithm's sensitivity to noise and their capacity to learn complexity. Often this choice can be a payoff between the two.
- Yes we are using the English spelling of Neighbour, because we are in England.

Classification Explorer

- To evaluate a model, we can test it on new inputs ourselves, or we can investigate its decision boundary.

- The task we are going to do will explore this further, allowing you to provide data and see the decision boundaries that are made.

- To improve a model, we can try and provide more data, or better data, or we can change the features that we use. We should aim to pick features that are relevant to phenomena we are trying to model.

Classification Explorer

- We’ve already given you a brief feel for how machine learning can allow you to map inputs, such as a mouse, to control an output, such as music synthesis parameters.

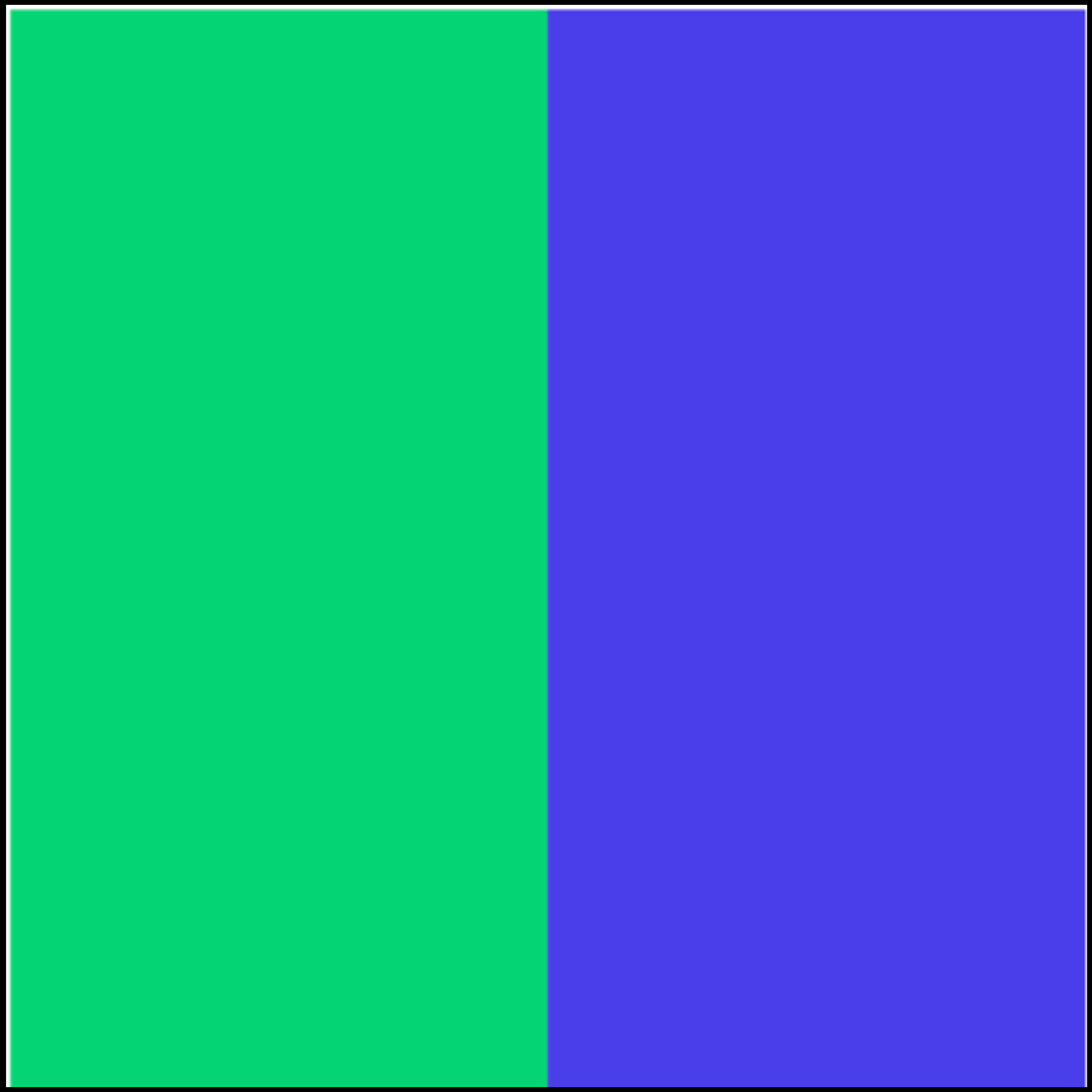
- But we can also sculpt the decision boundaries in classification tasks. This allows us to understand better how the models work by incrementally building up our own datasets.

Classification Explorer

- Check out the Classification Explorer on the MIMIC platform.
- https://mimicproject.com/code/7f92bd4e-6d2b-181c-559f-4add766f2095
- In order to input training examples, hold down a number key (e.g. 1) and move the mouse (you may have to click inside the example first to bring it into focus). This will input examples of that class as long as you are holding down that number key. The decision boundary will then be displayed.

Classification Explorer

- Try to choose a set of training examples that will draw the boundary on screen, with Class 1 on the left in green and Class 2 on the right in blue.



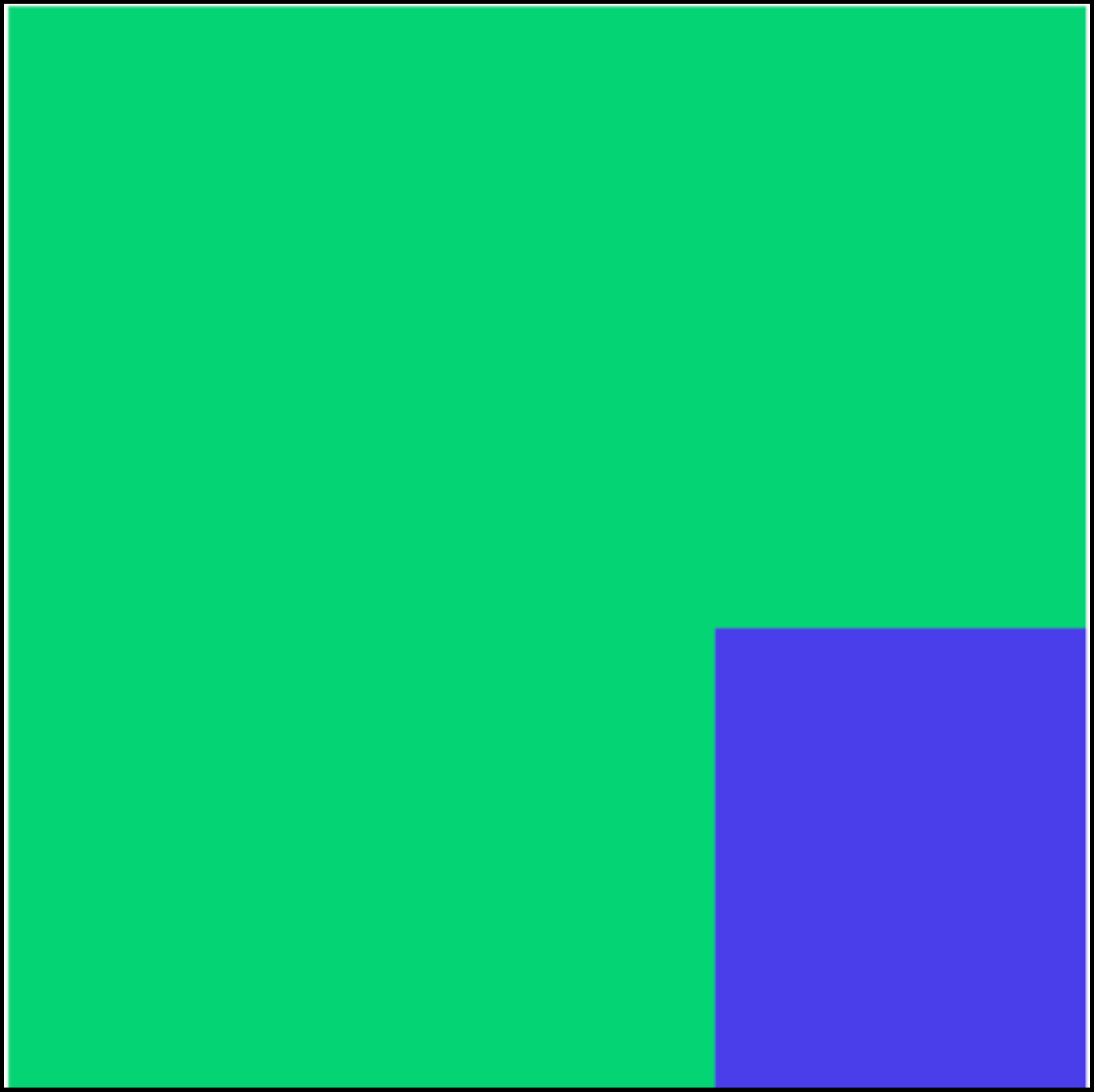
- Can you make it look like this?

Classification Explorer

- Keep improving your boundary by changing your training examples, re-training, and re-drawing the decision boundary until you are happy with how it looks. If you need to start again, just pause and play the example.
- When actually making classifiers for a specific purpose, how close you need the decision boundary to match will depend on your intentions. We realise that it may not be possible, or even desirable, for it to be perfect!

Classification Explorer

- Now try to make this decision boundary, with Class 1 in green and Class 2 in purple.
- This one is harder and you will not be able to make it perfectly using the K-Nearest Neighbour algorithm implemented in RapidLib.
- What do you think it is about the KNN algorithm that makes this decision boundary hard to implement?



Classification Explorer

- Have you had to manufacture examples close to the decision boundary to make it fit the shape?
- How might this effect the make-up of your datasets when working on an actual project?
- Will it just consist of representative examples of thing you are modelling?

Classification Explorer

- It may be the case that it contains edge cases included to directly influence the behaviour of your trained model.
- These issues don’t necessarily go away when we use more complex classification systems
- We often find that we have to be very specific when providing data. This is why creating data sets is one of the things Machine Learning professionals spend most of their time doing!!!

What can we do with Classification?

- So now we have a better idea of what classification is.
- We’ve also had a good think about the issues that can impact on classifiers, causing them to make errors.
- What can we do creatively with classification?

Machine Listening with Classification

- Take a look at this simple classification example.

- It takes audio from the microphone input and uses it to create classifiers that can discriminate between different types of sounds.

- We’re going to use old youtube videos for this, but you could use absolutely anything.

Machine Listening with Classification

- https://mimicproject.com/code/3864f3e5-8263-b70e-5ef9-1037c724d4ec
- Let’s talk through this example

Machine Listening with Classification

- This example uses a representation of sound called a “Mel Frequency Cepstrum Coefficient”
- Also know as the MFCC

Machine Listening with Classification

- The MFCC is made up of a small list of numbers that represents the timbre, or ‘sound texture’ of the sound.
- There are around 20 of these MFCC lists (or ‘frames’) every second.
- We store examples of them in the KNN with a label.

Machine Listening with Classification

- When a new input MFCC frame comes in, we check to see if it is close to any MFCCs that we have labelled.
- We give the new MFCC input the same label as its nearest neighbour.
- This is surprisingly accurate.

Recap

- So we looked at Classification in more detail
- We also thought about the problems that can occur with classification
- We then looked at how we can use a simple form of classification to detect different types of sounds
- We found out that this was pretty impressive

Regression

- Let’s think about Regression in more detail.
- We can use this using the MIMIC regression explorer.
- This is a tool that we’ve built for you so that you can get a more intuitive grasp of how regression works, and also reason about the technique a little better.

Regression

- Regression models can be used to map continuous inputs.
- As with Classification, different algorithms produce different types of lines or curves given the same data.
- As such, it’s important to pick an algorithm that can model the appropriate complexity. As with Classification, there is currently no choice of regression algorithm in Rapidlib.

Regression

- “Many to many mappings” are when we map lots of inputs to lots of outputs
- They can allow us to quickly build a complicated system. With this, you can design interactions that react to people’s movements or actions without directly having to consider the exact mathematical relationship between input and output.
- Later on, we will see how you can use regression to control 3 parameters of a granular synth with just 1 slider. Although we could literally use loads.

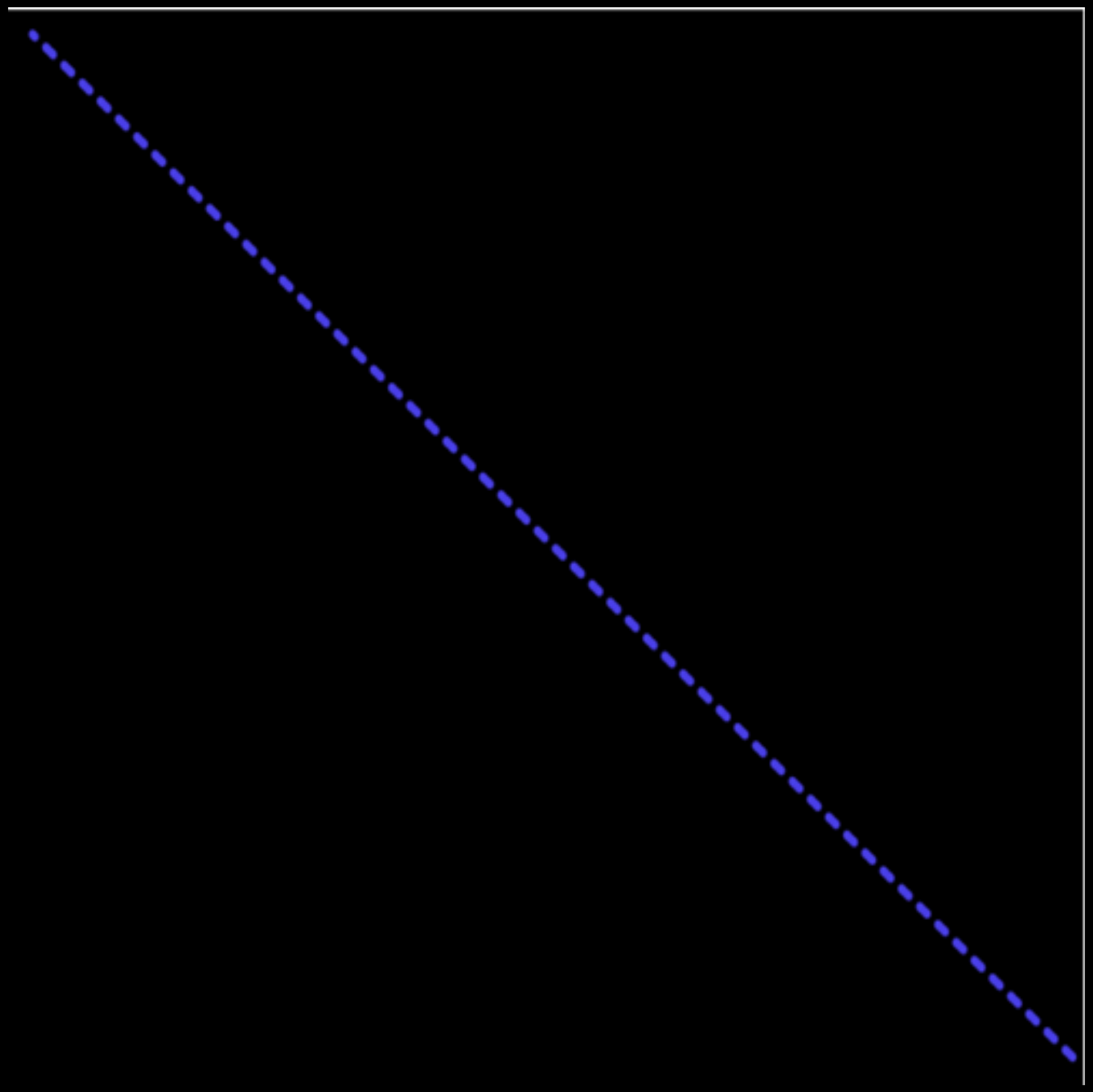
Regression Explorer

- First we're going to look at the Regression Explorer.
- This will help us get a more intuitive grasp of what is going on when we create and use regression models.
- You can use the Regression Explorer on the MIMIC platform.
- It's here: https://mimicproject.com/code/26ab5507-0d25-07eb-cb03-aaa93883765d

Regression Explorer

- In order to input training examples, click onto screen at any point.
- The X value denotes the input value, whereas the y value denotes the output value. You will then see the regression line drawn as you add values.
- Try to get a feel for what types of lines are capable and how they’re influenced by the training data.

- Create a training dataset this way to produce the line below.



Regression Explorer

- Keep editing your data and/or algorithm until the regression line drawn by the Regression Explorer matches the line as closely as possible.

- Now, think about whether you might be able to create this same line using even fewer training examples. If you think it's possible to create this line using fewer examples, delete your training dataset and give it a try!

- As with the Classification Explorer, feel free to fork the code use RapidLib yourself to use your own inputs and outputs. In this case you will need to have a dataset including one input and one output only.

Regression Example

- Now we are going to try and see if we can train a model with 3 outputs to behave consistently.
- We’re going to one single input to control EXACTLY 3 output parameters.

Regression Example

- Below we have an example of using a slider as input to control a granular synthesiser (borrowed from Zya)

http://www.zya.cc/granular

- Granular synths play lots of small fragments (grains) of a soundfile at various positions.

Regression Example

- You can smooth the start and end of each grain (the envelope), and have more or fewer grains (the density)
- The outputs of the regression model control these three things - position, envelope and density.

Regression Example

- Play around with the two parameters and click on different parts of the waveform to find some sounds you like.

- When you are ready to record, select the "Record" checkbox.

- From now on, whenever you play the synthesiser, the position in the sample, the release and density will be recorded, along with the input slider value. Record in some sounds you like, remembering to map each one to a different value on the input slider.

Regression Example

- When you are ready to play, select the "Run" checkbox.
- This will train your model with the recorded dataset and now all 3 synthesiser parameters will be controlled by just the one value from the input slider.
- Keep recording examples until you can reliably control the output.

Regression Example

- Next, train model to reliably access the following values. To clear your dataset, just pause and play the embedded code.
- (0.0, 0.0, 0.0) (all models output 0 simultaneously)
- (1.0, 1.0, 1.0) (all models output 1 simultaneously)
- (0.5, 0.0, 1.0) (model 1 outputs 0.5, model 2 outputs 0, and model 3 outputs 1)

Regression Example

- This should give you a pretty good idea of how efficient regression models can be, or otherwise.

- This regression model uses a simple neural network. You can combine this network with others and stack them together to create more complex networks.

- This allows you to train more complex models, but issues around training these kinds of models still occur all the time!

Classification 2

- For this next part we are going to revisit the Classification Explorer.
- This time we are going to experiment with more than 2 classes, as well as more complicated decision boundaries.

Classification 2

- Sometimes if your data is noisy, or the feature representation is not sufficient, you will not be able to create the right decision boundary. In this situation, you should consider providing more, or better, data.

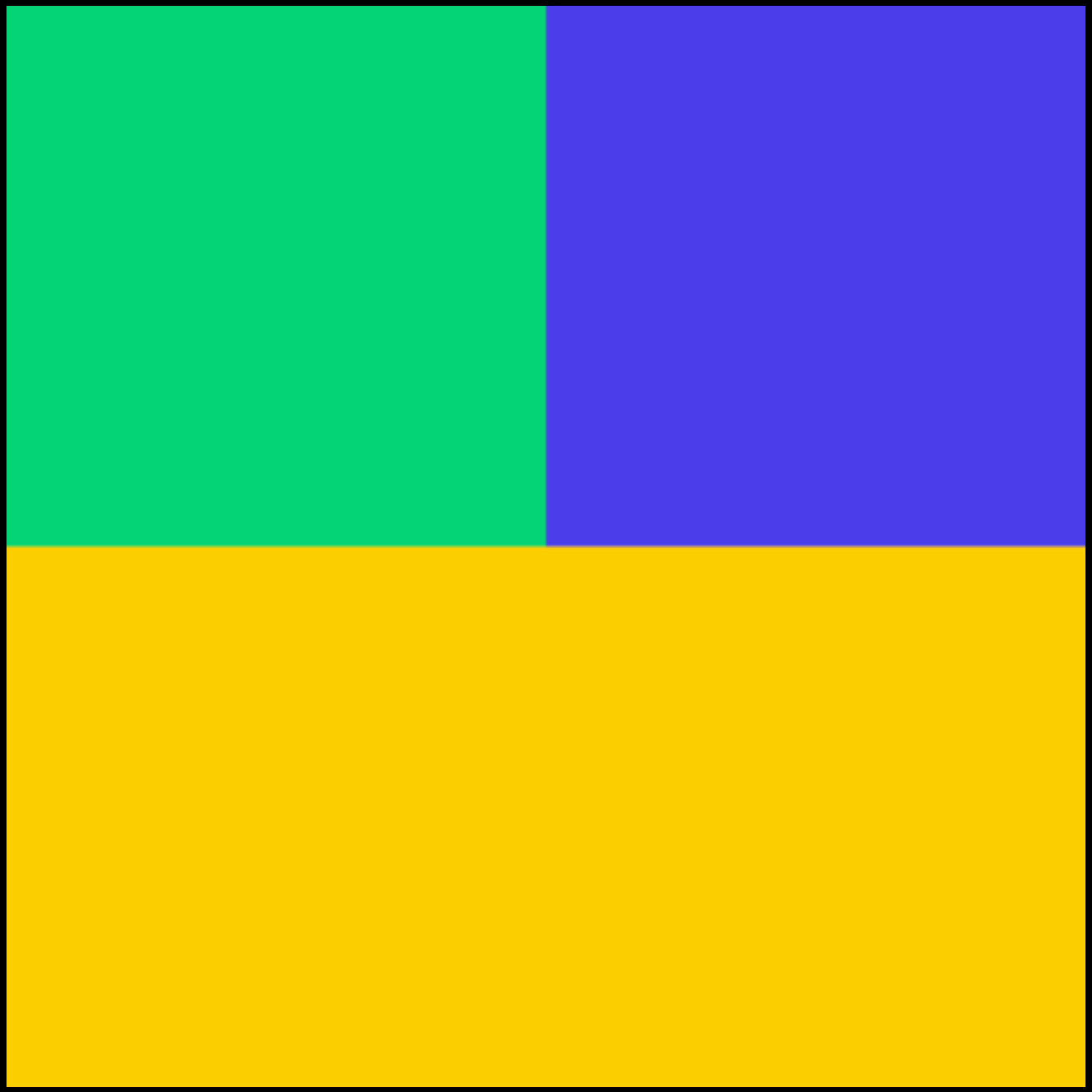
- Alternatively, if you are sure your data is correct, you may have to spend more time picking the algorithm that will be able to capture the complexity of the phenomena you are trying to model.

- However, picking an overly complex model can lead to overfitting of data, meaning your model will not generalise well to new data not present in the training set.

Classification 2

- By adding new training data, try to recreate the decision boundary below using the Classification Explorer. As you are doing this, think about
  - How the model improves (or not) as you add more examples.
  - How well the shape of the decision boundary the algorithm seems to want to make matches the boundary you are trying to make.

Classification 2



Classification 2

- Think about how the challenge of classification changes when you start to include non-binary problems.
- Now try creating some different boundaries by changing your training data using 3 or more classes.