Episode 1: Core AI Concepts

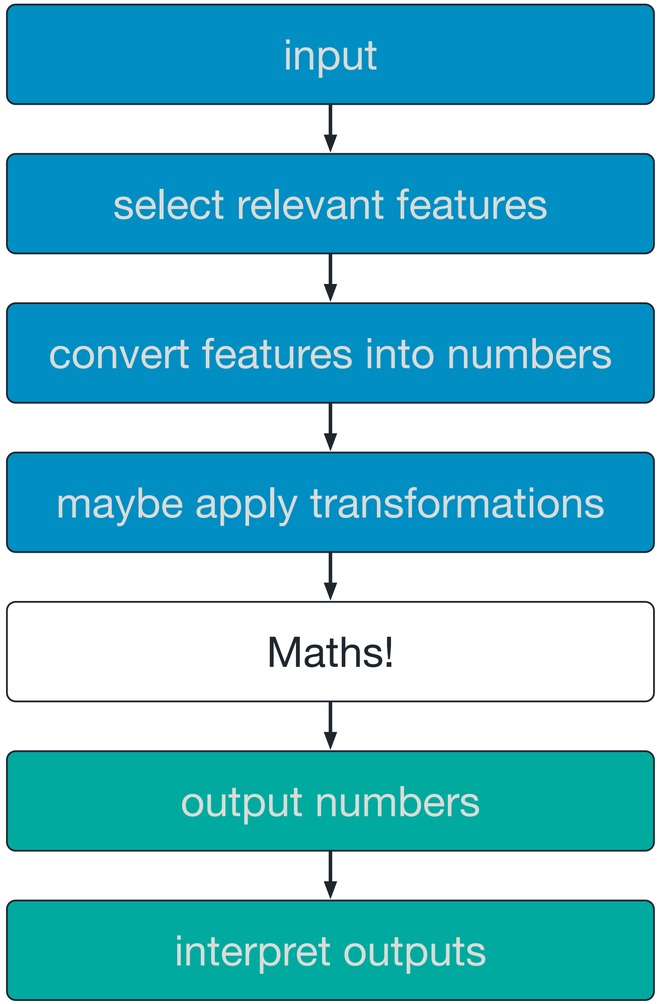
Alastair Droop

AI & Machine Learning: From Theory to Practice

“The capacity of computers or other machines to exhibit or simulate intelligent behaviour; the field of study concerned with this.”

https://www.oed.com/viewdictionaryentry/Entry/271625

Some Terminology…



The inputs to an AI process can be pretty much anything we can measure

Features are the things (called variables) that we want to measure (eg. Blood pressure, disease status, *etc…*)

We encode features into numbers so we can use maths to process them. Different kinds of features will use different encodings to do this

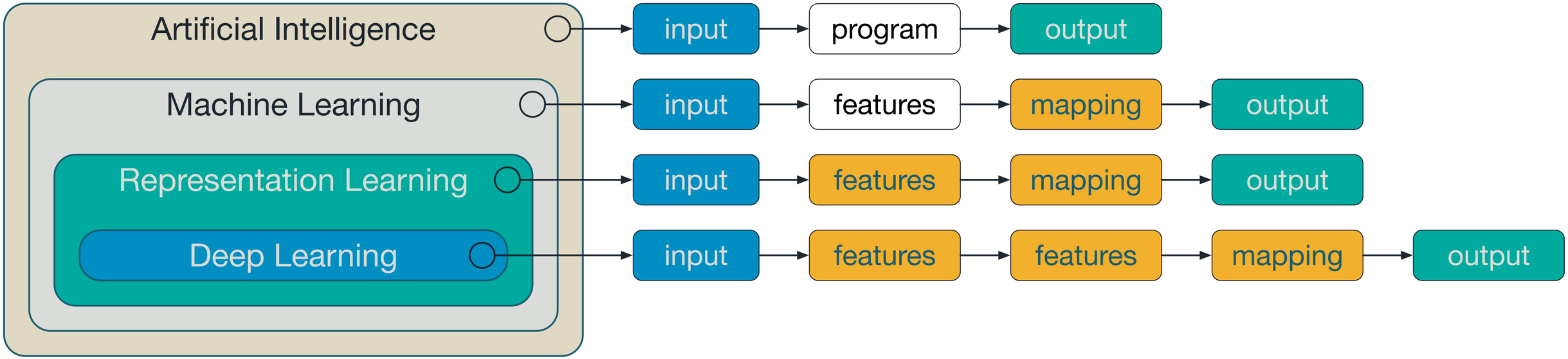
Transformations change the values of the features in defined ways to make them easier to use. Whatever transformations we do, we're changing all the values in the same way

We input the transformed input features into a mathematical function (an algorithm) that generates the outputs for us

The AI function generates numerical values as outputs

We need to interpret the numerical values to work out what the AI has told us

Types of AI



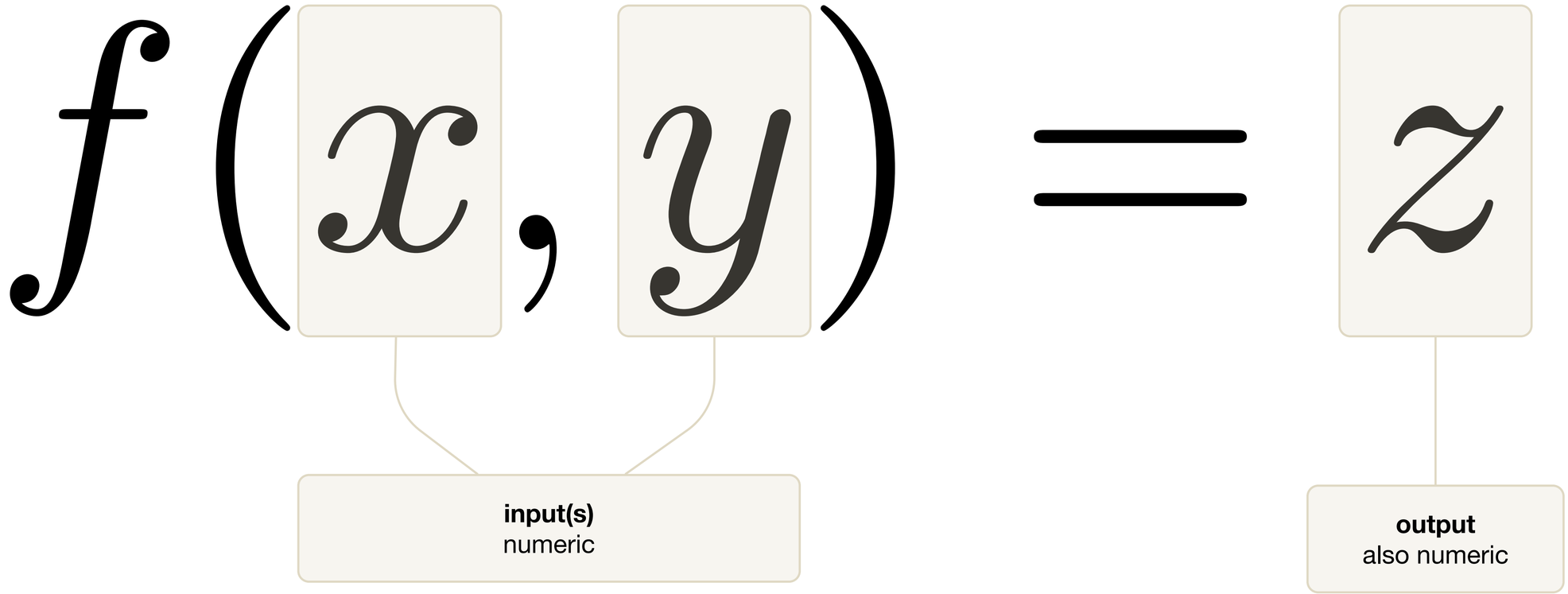
The boxes in yellow are where we allow the computer to learn

Machine learning: the computer learns how to modify & combine (*ie*. map) the inputs

Representation learning: the computer learns the features to extract as well as the mappings

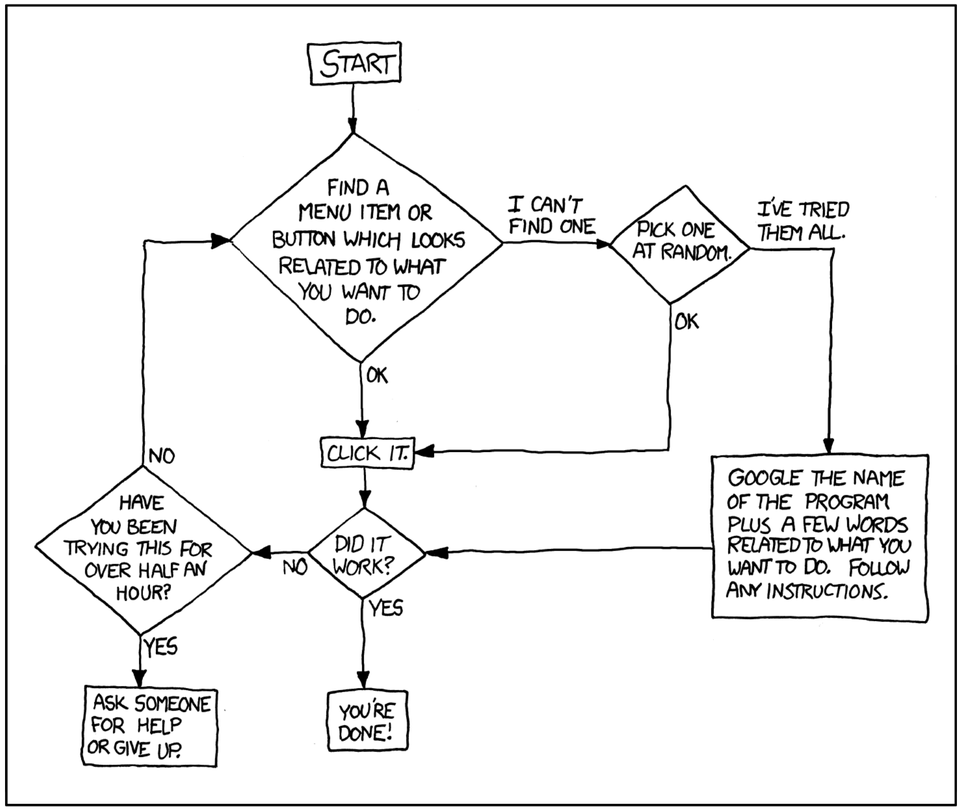
Functions

A function takes zero or more inputs and returns an output

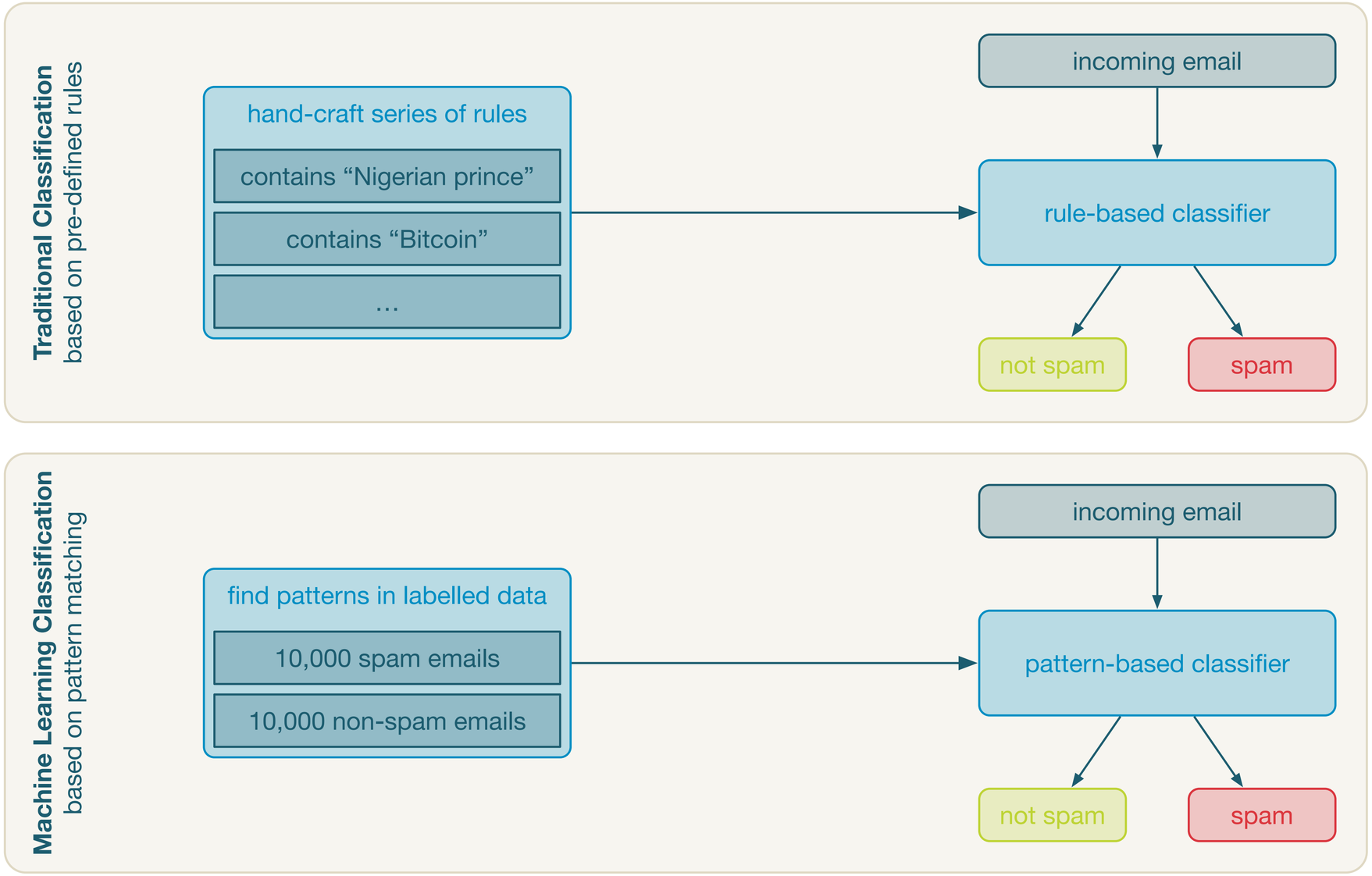


Algorithms

A series of specific & reproducible step-by-step instructions to perform a specific task



https://xkcd.com/627/


Traditional Classification
based on pre-defined rules
hand-craft series of rules
contains “Nigerian prince”
contains “Bitcoin”
…
incoming email
rule-based classifier
not spam
spam
Machine Learning Classification
based on pattern matching
find patterns in labelled data
10,000 spam emails
10,000 non-spam emails
incoming email
pattern-based classifier
not spam
spam

Classification

A very common problem in data science is classification

- Given a set of data, place each datapoint into one of several categories

Unsupervised Learning (eg. k-means & hierarchical clustering)

- No requirement to train on known data before use

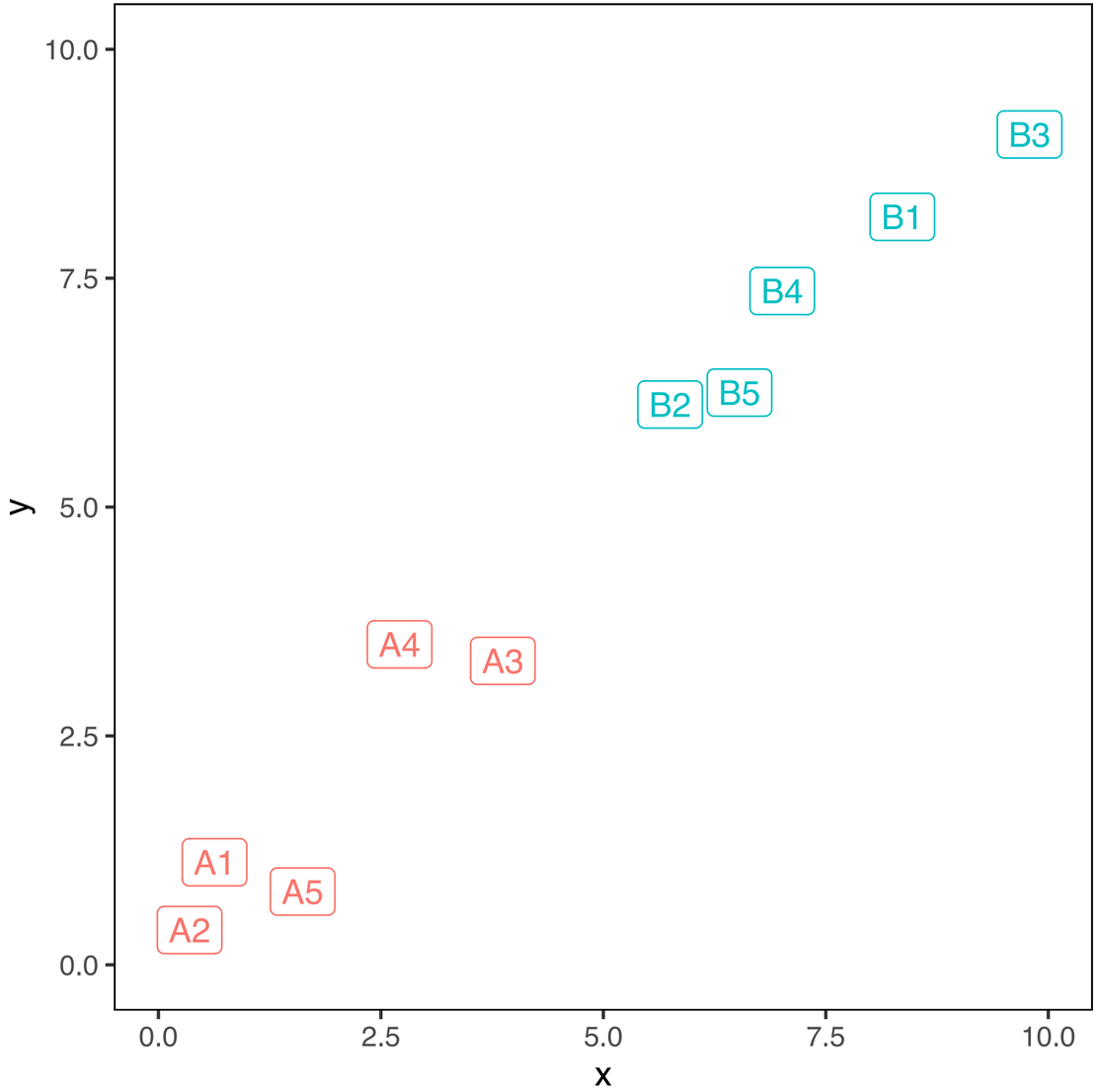
Supervised Learning (eg. Linear regression, SVM, Decision Trees, Neural Networks)

- Trained on known, labelled data before being used to classify unknown data

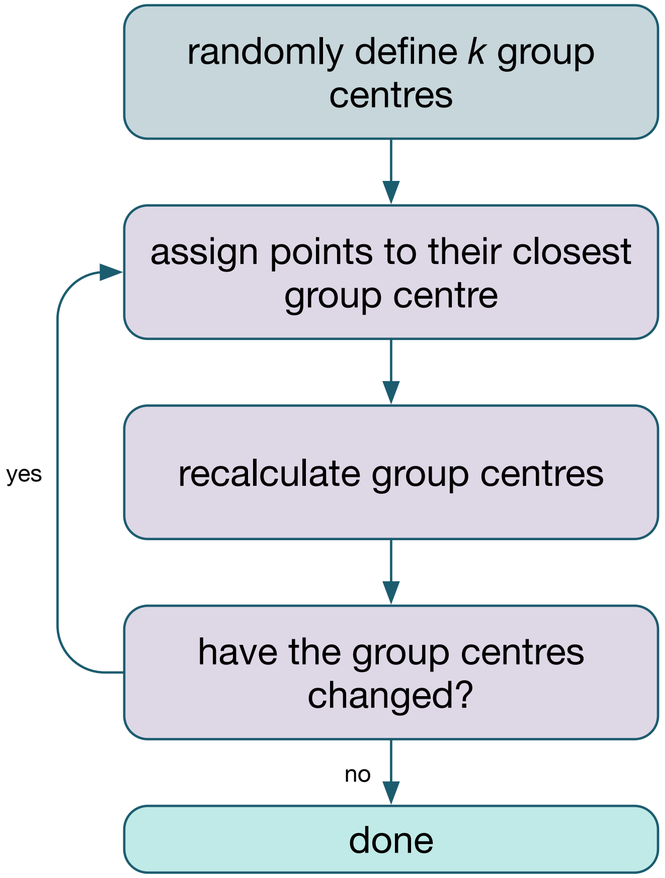
Imagine we have some 2D data

- Dimensions are simply variables that we measure
- Each data point has a value for each dimension

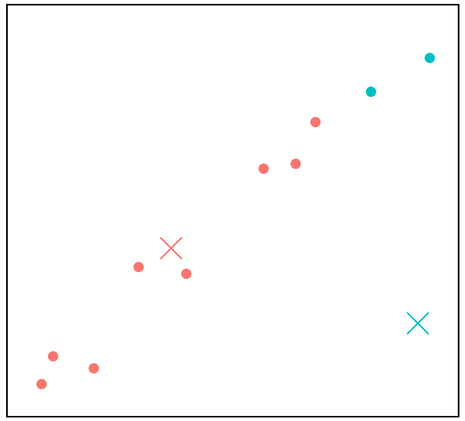
We wish to place each datapoint into a group



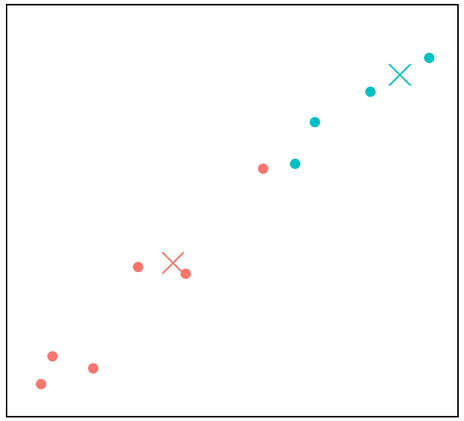
k-Means Clustering



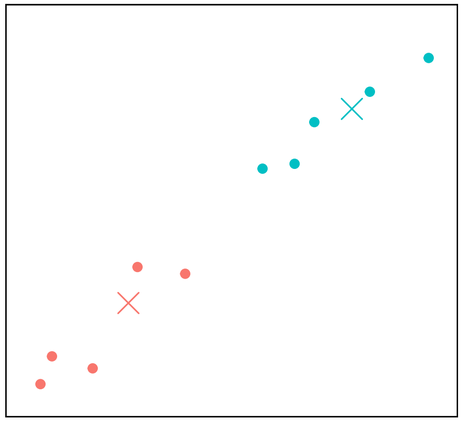
Initial



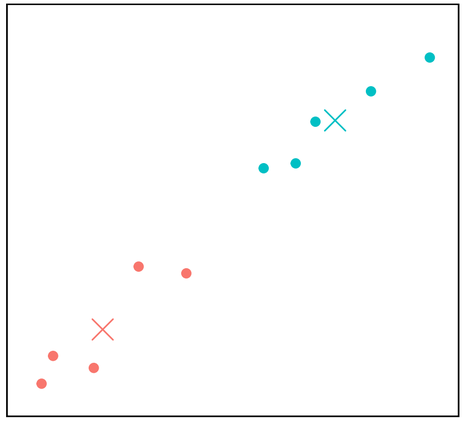
Iteration 1



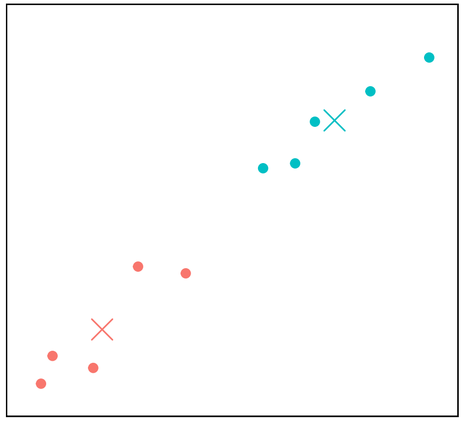
Iteration 2



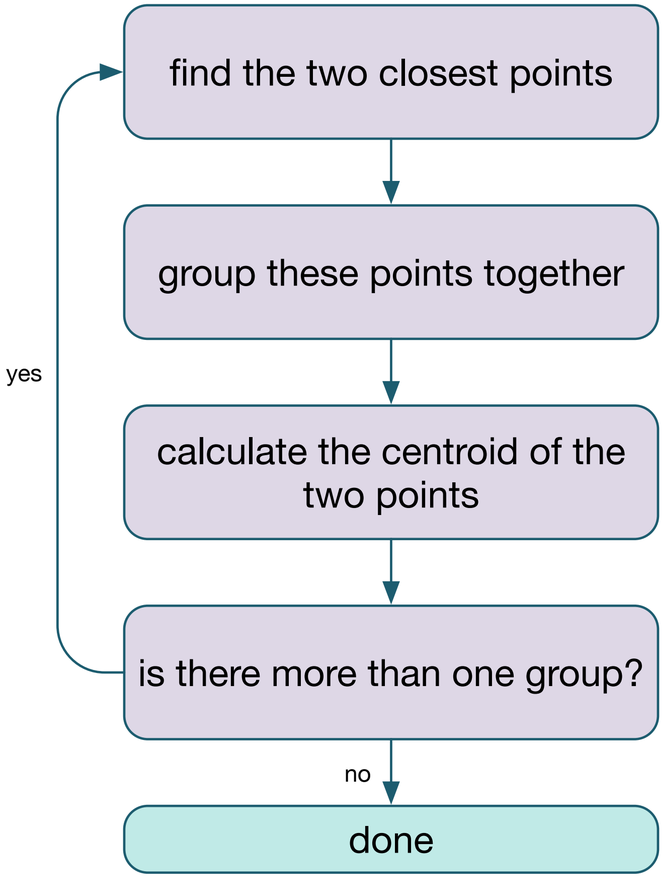
Iteration 3

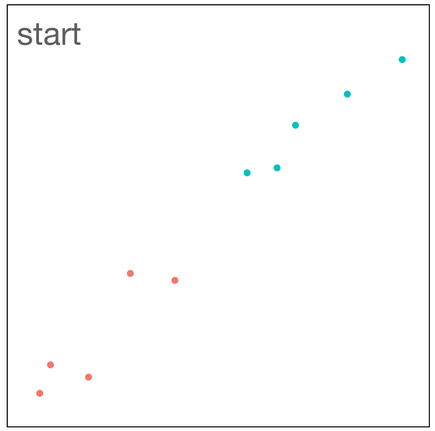


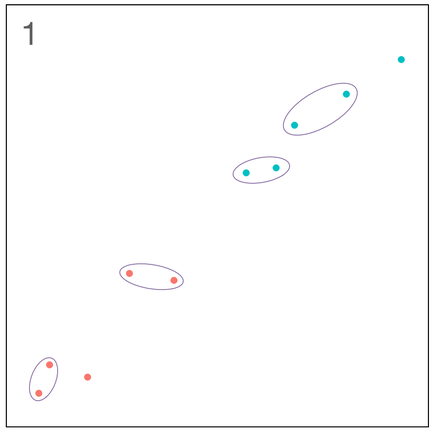
Iteration 4

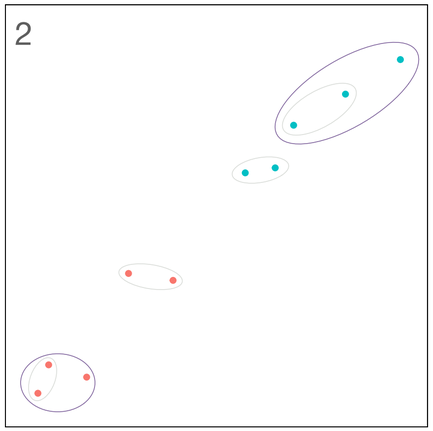


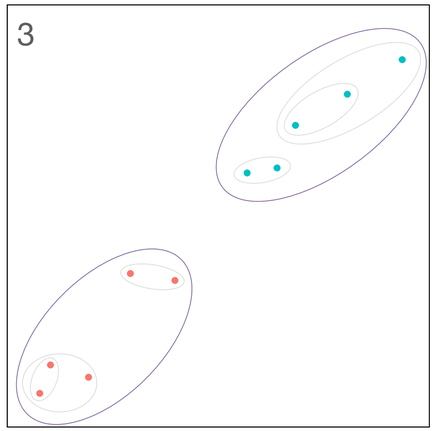
Hierarchical Clustering

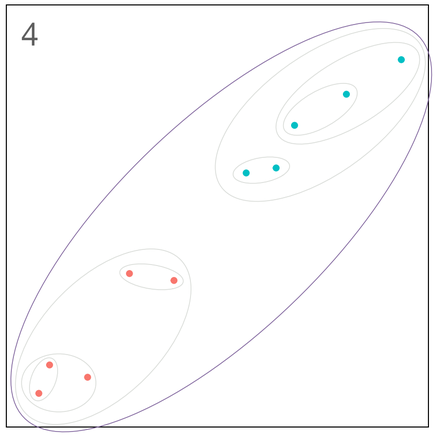


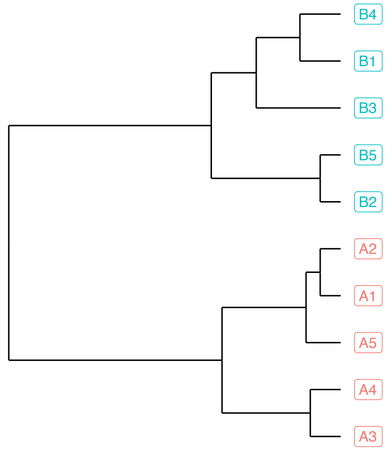












Encoding

Encoding data changes the labels to make it easier for computers to work with

Encodings need to be unambiguous, consistent and (as much as possible) complete

Karyotype	Name	Incidence (live births)	Encoding
XX	Female	1 in 2	1
XY	Male	1 in 2	2
X	Turner syndrome	1 in 4,000	3
XXX	Trisomy X	1 in 2,000	4
XXY	Klinefelter syndrome	1 in 1,000 – 1 in 2,000	5
XYY	Jacobs syndrome	1 in 1,000	6
XXXY	48,XXXY syndrome	1 in 100,000	7
XXYY	48,XXYY syndrome	1 in 80,000	8
XXXXY	49,XXXXY syndrome	1 in 200,000	9
other	–	–	10
unknown	–	–	0

https://medlineplus.gov/genetics/condition

Real Data Are Messy

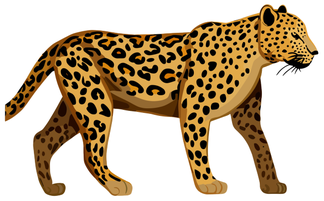
Many problems are not simple to solve using the above techniques

Which of these images show leopards?










Dimensionality Reduction

When performing experiments, we measure multiple variables for each of multiple samples

- Expression of multiple genes
- Fluorescence of multiple markers

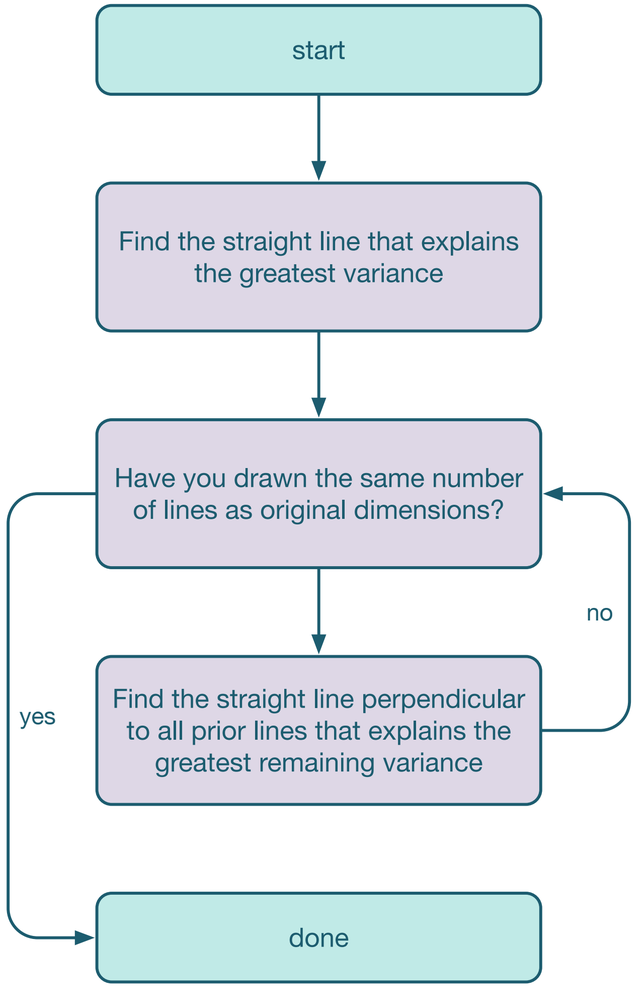
We wish to make statements about sample groups, taking into account all of the variables

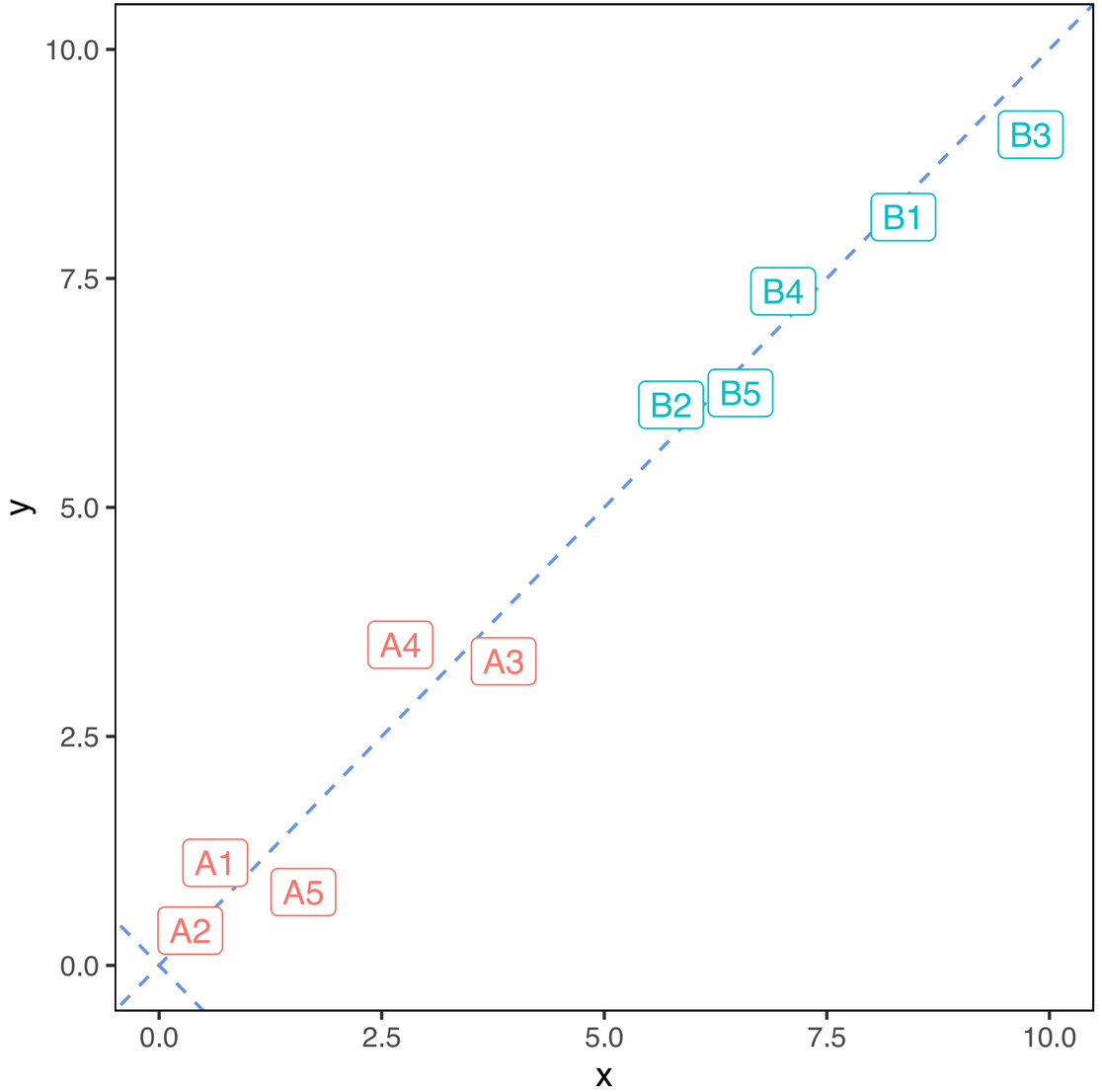
The variables we've measured often don't correspond to the features we care about

For example:

- If we wish to identify UK cities, we might record latitude and longitude.
- We've used two variables to describe a single hidden variable (which city)

PCA


start
Find the straight line that explains the greatest variance
Have you drawn the same number of lines as original dimensions?
no
yes
Find the straight line perpendicular to all prior lines that explains the greatest remaining variance
done

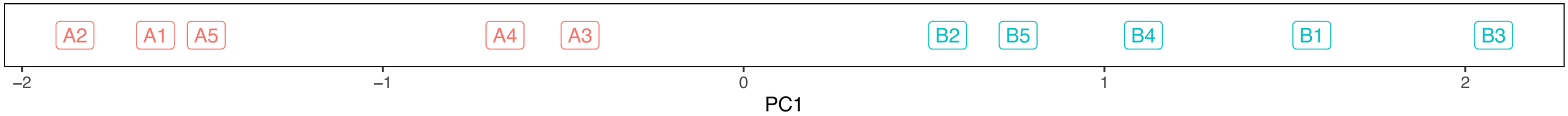

10.0
7.5
5.0
2.5
0.0
y
0.0
2.5
5.0
7.5
10.0
x
B3
B1
B4
B2
B5
A4
A3
A1
A5
A2

Each new dimension we get from PCA is a transformation of the original coordinate space

But, PCA ranks these new axes by the amount of variance they explain

This means that we can decide how many principle components to use

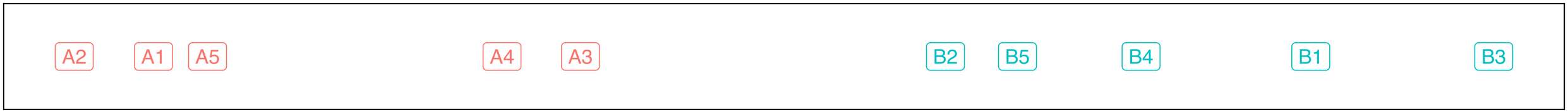
In our case, one dimension will do



Note that we've lost some information in doing this, but often this is OK (or even desired)

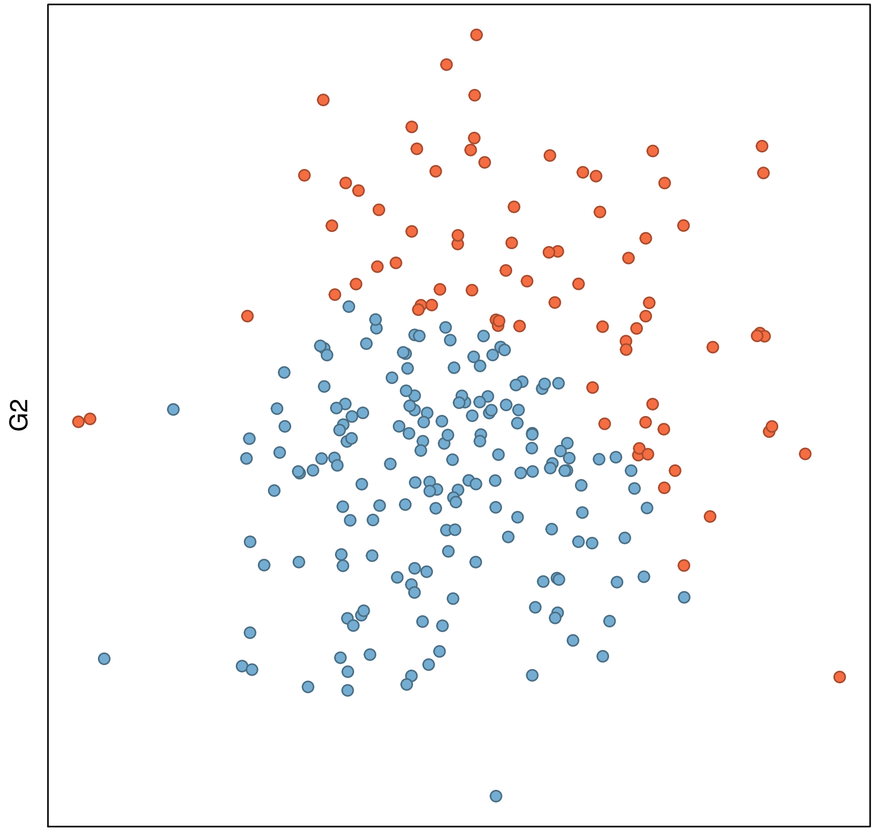
Another approach is to rescale our data so that the distances in a lower dimension are the same as those in the higher-dimension space

This is Multidimensional scaling (MDS)

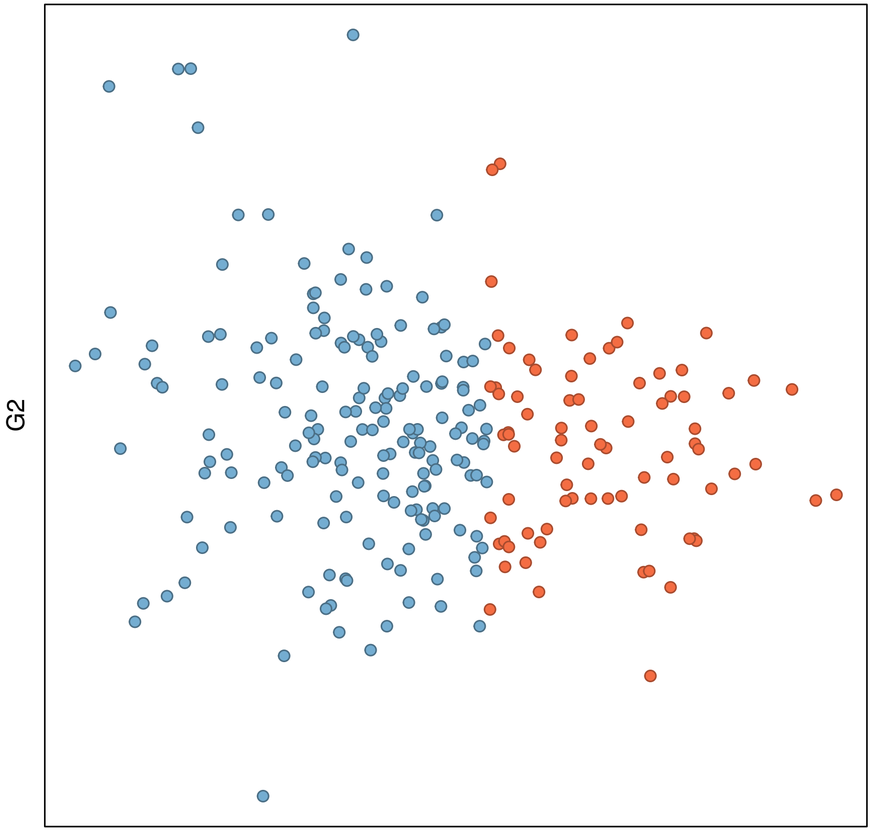


Transformation

We can transform our data before starting to analyse it to make the maths easier



Polar coordinate transform



Neural Networks

Neural Networks are a family of methods for classifying complex datasets

Allow us to make sense of complex and messy data

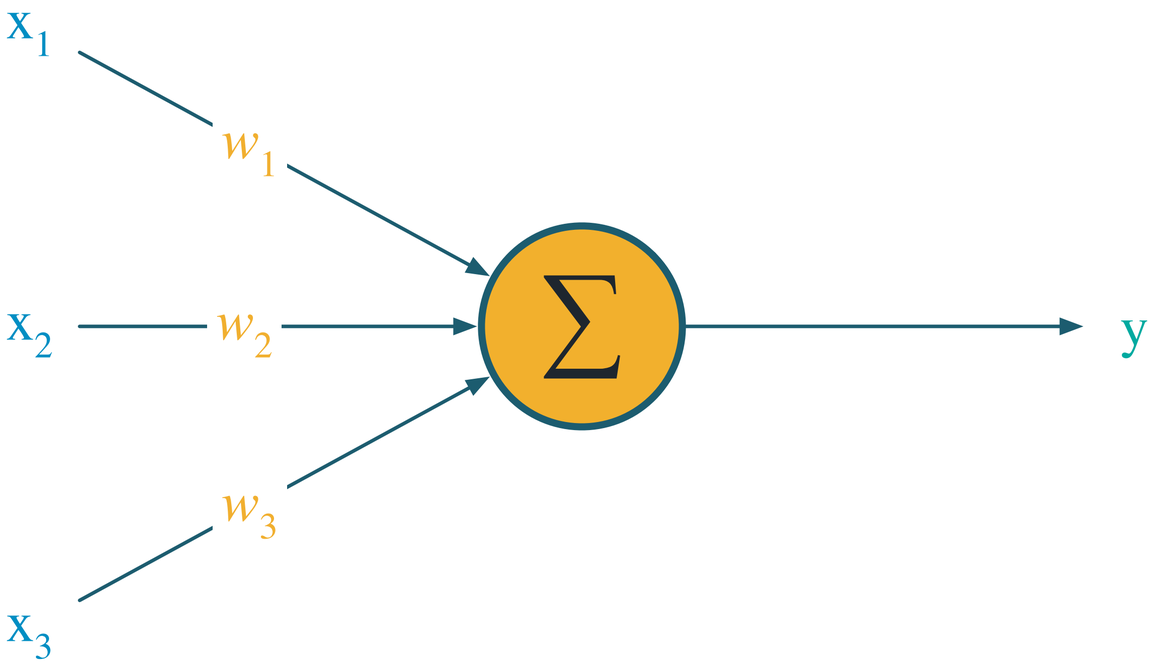
Originally inspired by biological neural networks

Individual neurones (*perceptrons*) process their individual inputs based on learned weights

Allows us to train classifiers without knowing how the classifiers work

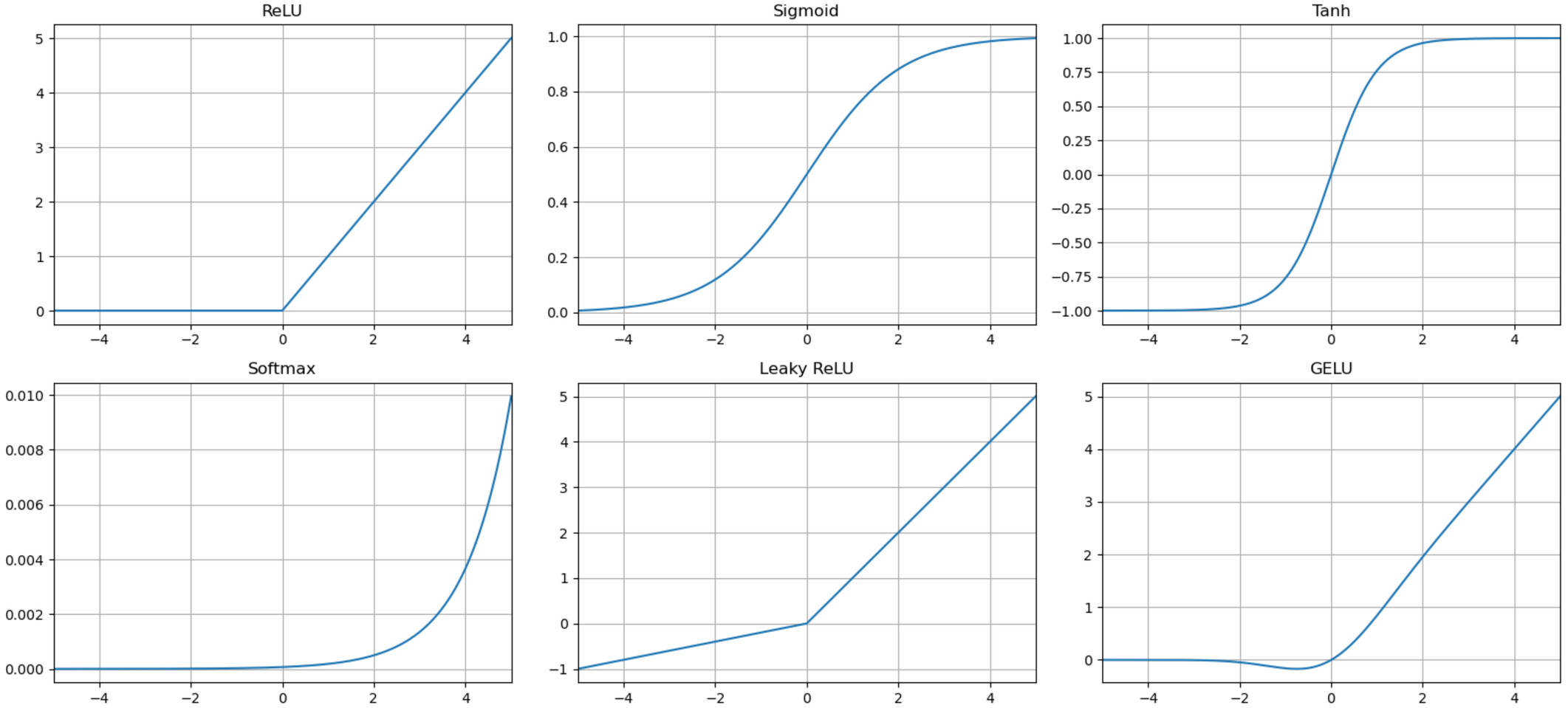
Perceptrons

A perceptron is a function that sums its weighted inputs



$$y = f\left(\sum_{i=1}^{n} w_i x_i + b\right)$$

Activation Functions



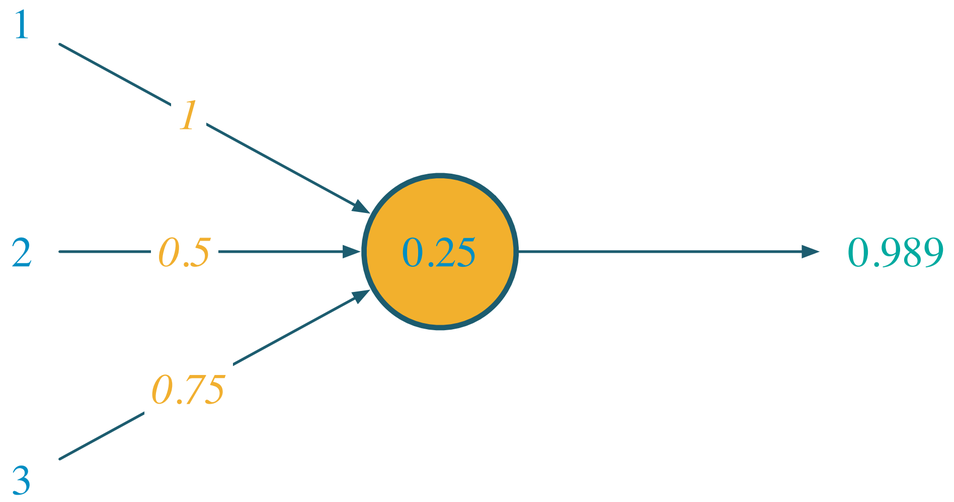
Examples

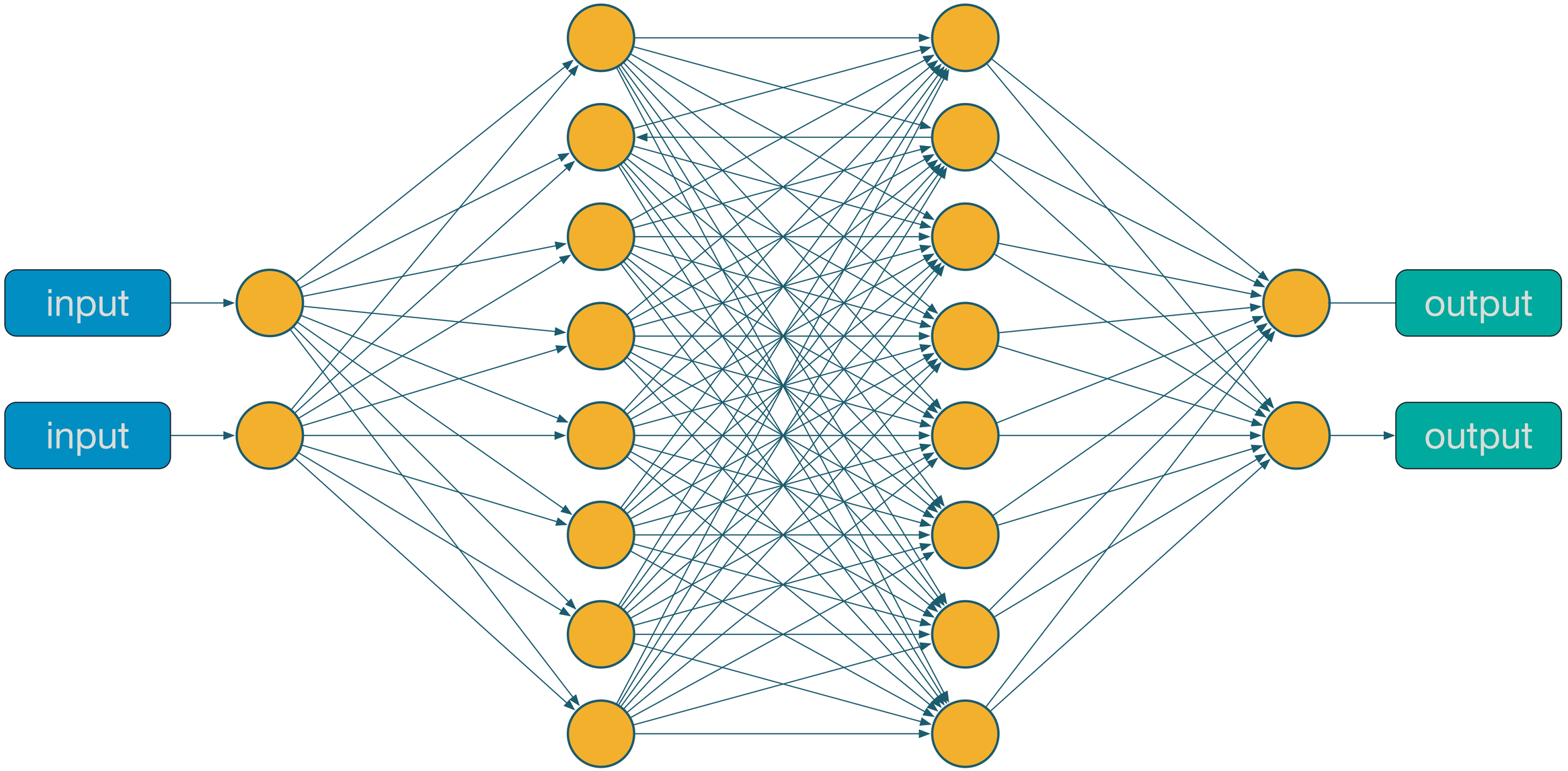
3 weighted inputs

Bias $b = 0.25$

Activation function is sigmoid

Resulting value is 0.989




input
input
output
output

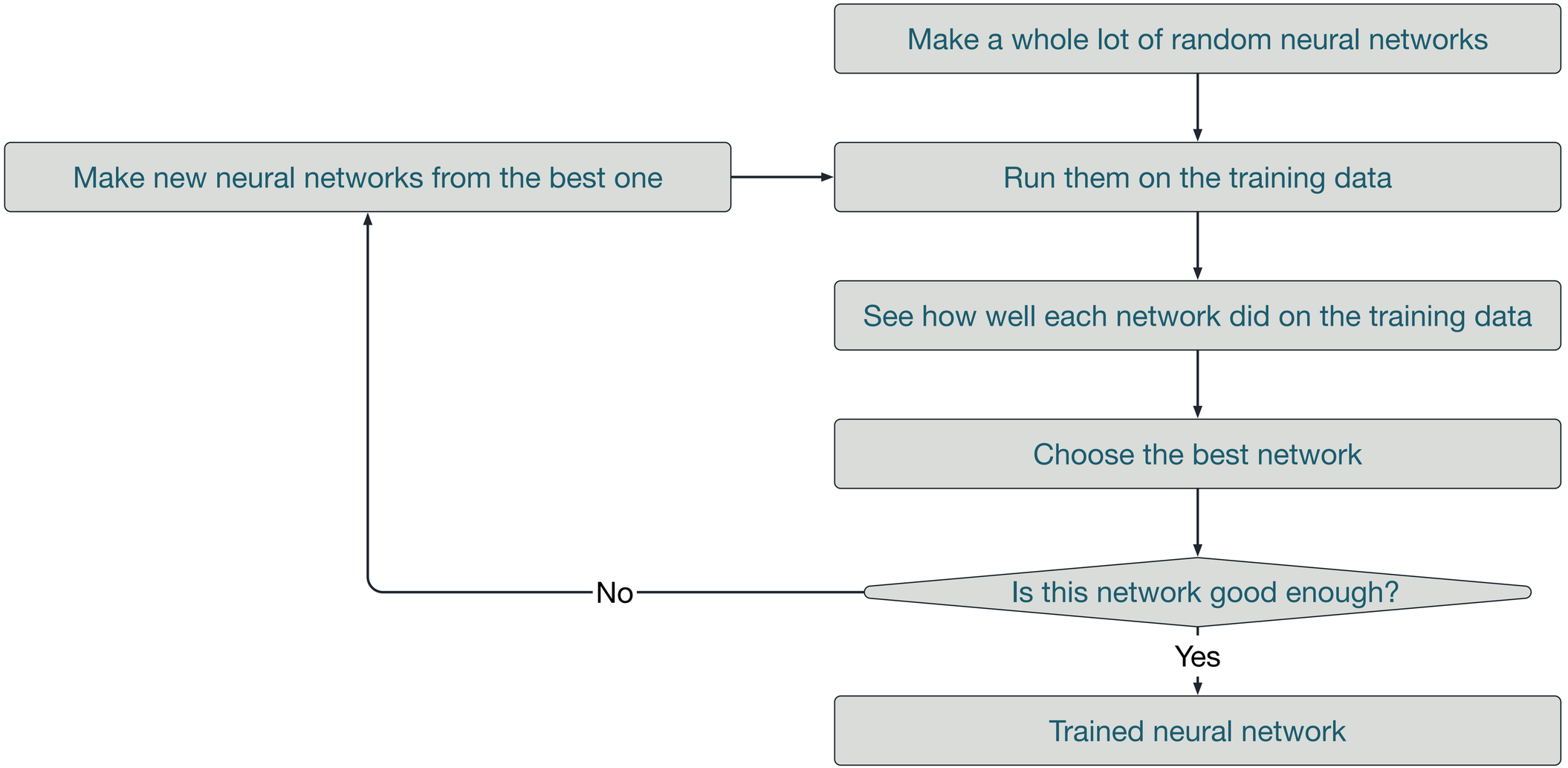
Training Neural Networks 1

To train a neural network, we use training data

Training data are simply data where we know the inputs and also the outputs we want to get

Training involves randomly changing weights and then hoping the new weights are better than the old ones

There is a risk of overfitting


Make a whole lot of random neural networks
Run them on the training data
See how well each network did on the training data
Choose the best network
Is this network good enough?
No
Yes
Make new neural networks from the best one
Trained neural network

Tweak the perceptron weights until the neural network gives the right answer

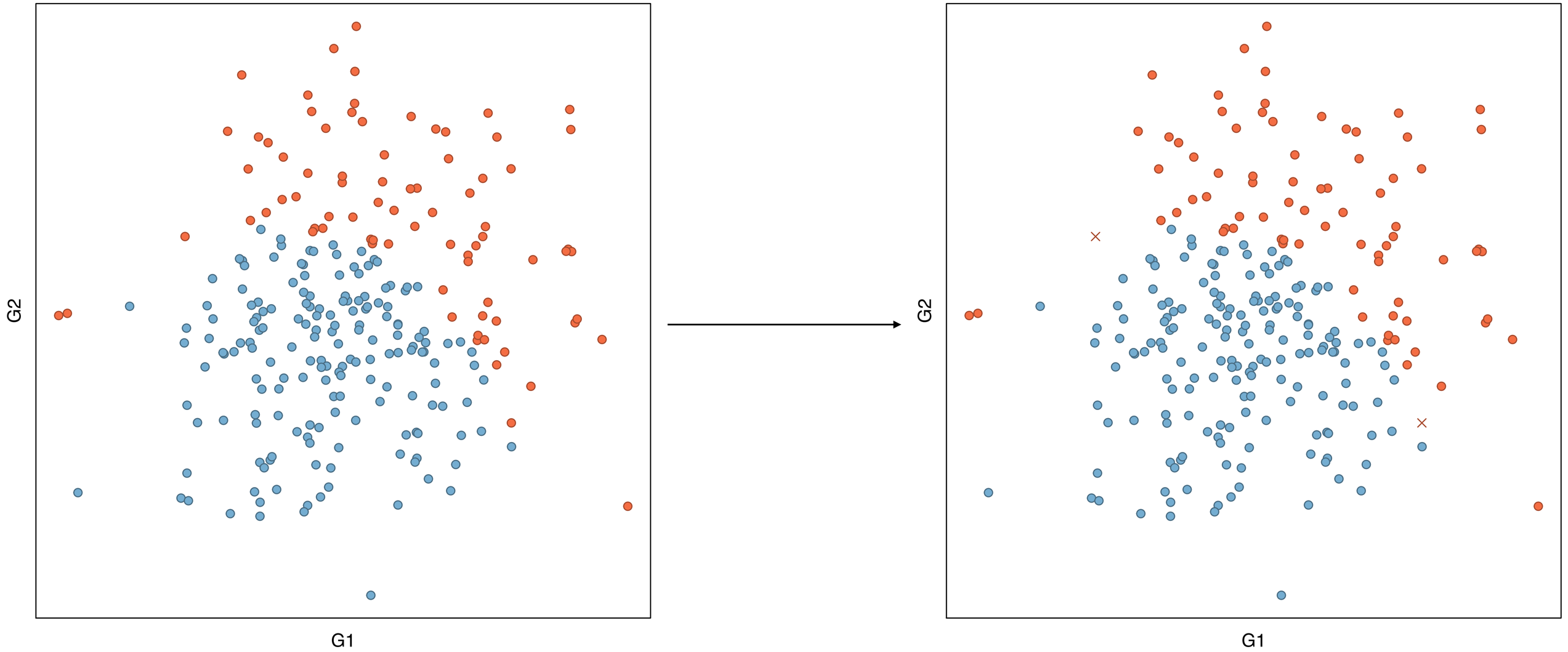
The trouble is, there are *a lot of weights*

In our example, there are 96 different weights, each of which can be any number

As we don't control what the neural network decides to learn, it is crucial that we:

- Use appropriate training data
- Use enough (lots) training data

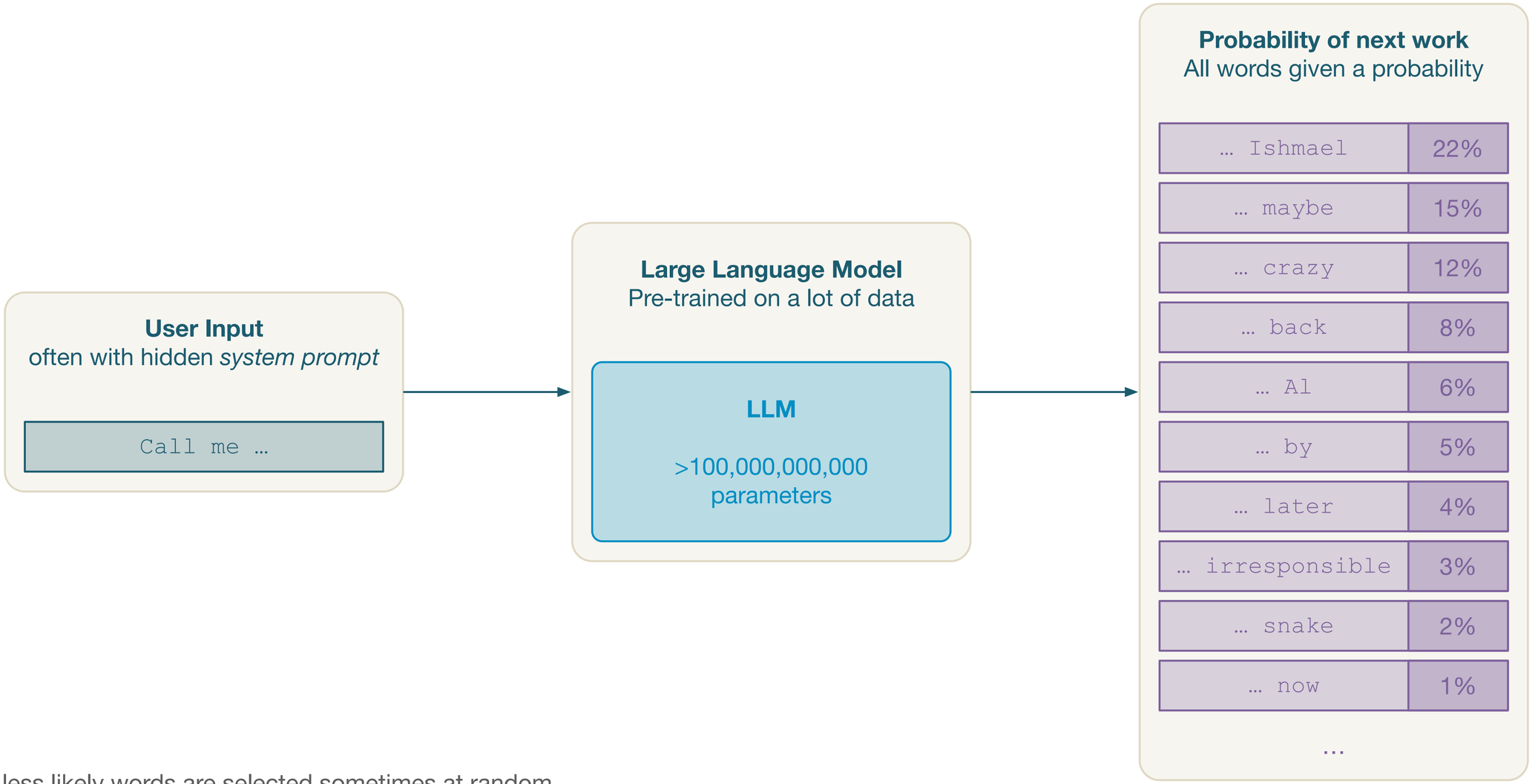
It is very easy to underestimate how many training data points we need



We train our neural network on 10,000 training points, then make it guess on our real data

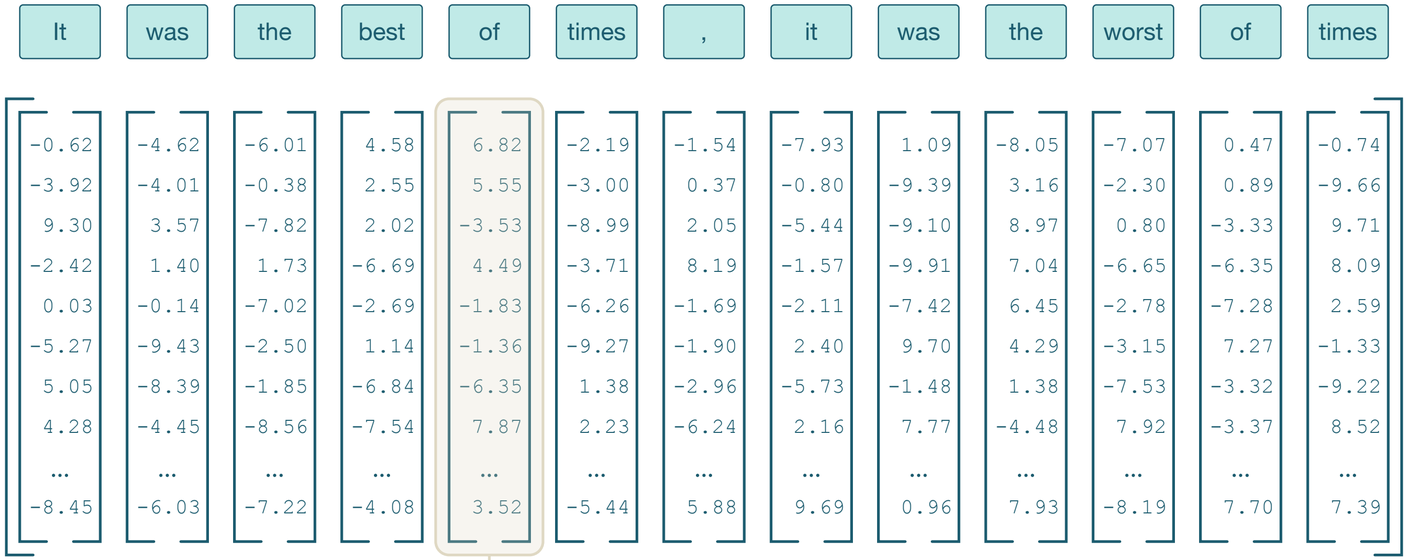
After all that, it still gets two points wrong!

Large Language Models



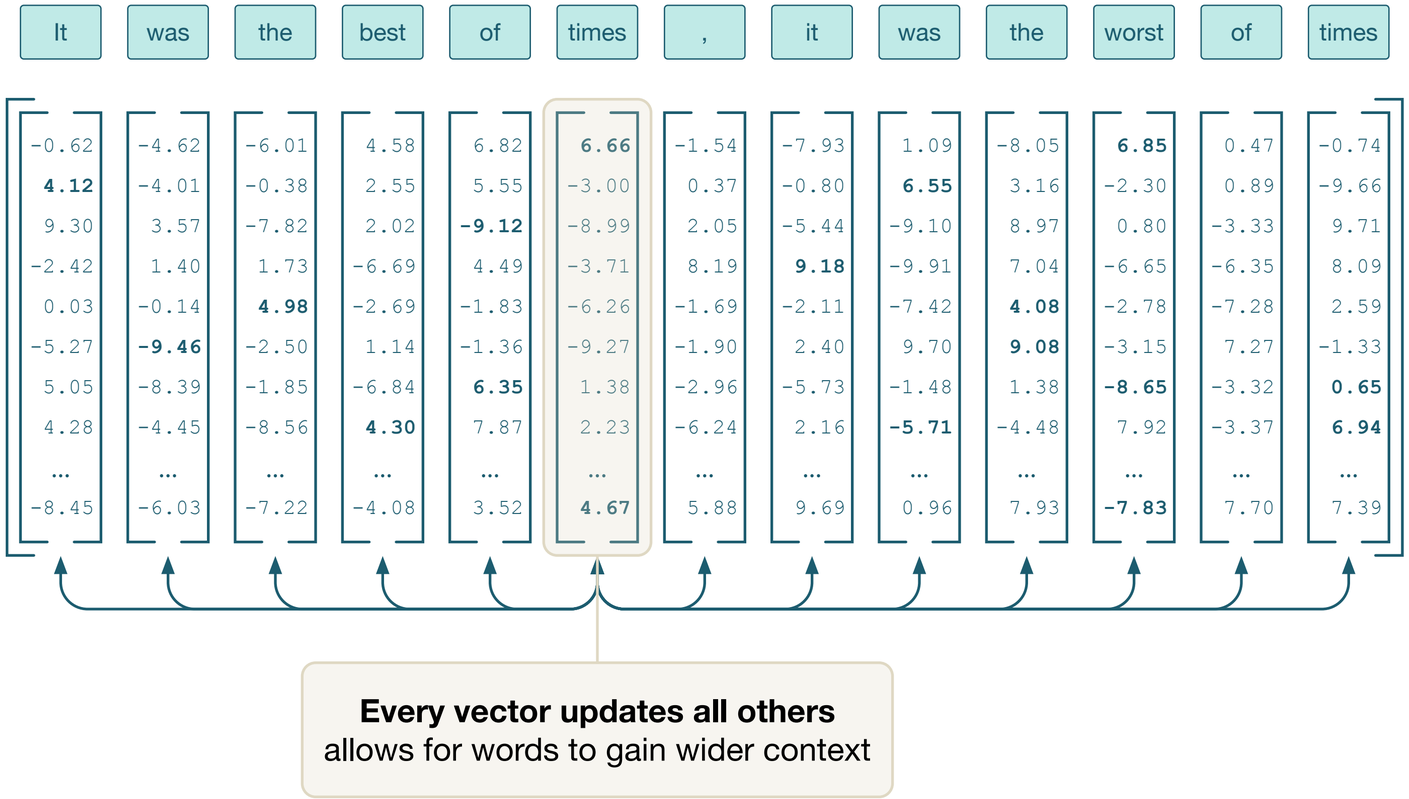
NB: Usually, less likely words are selected sometimes at random

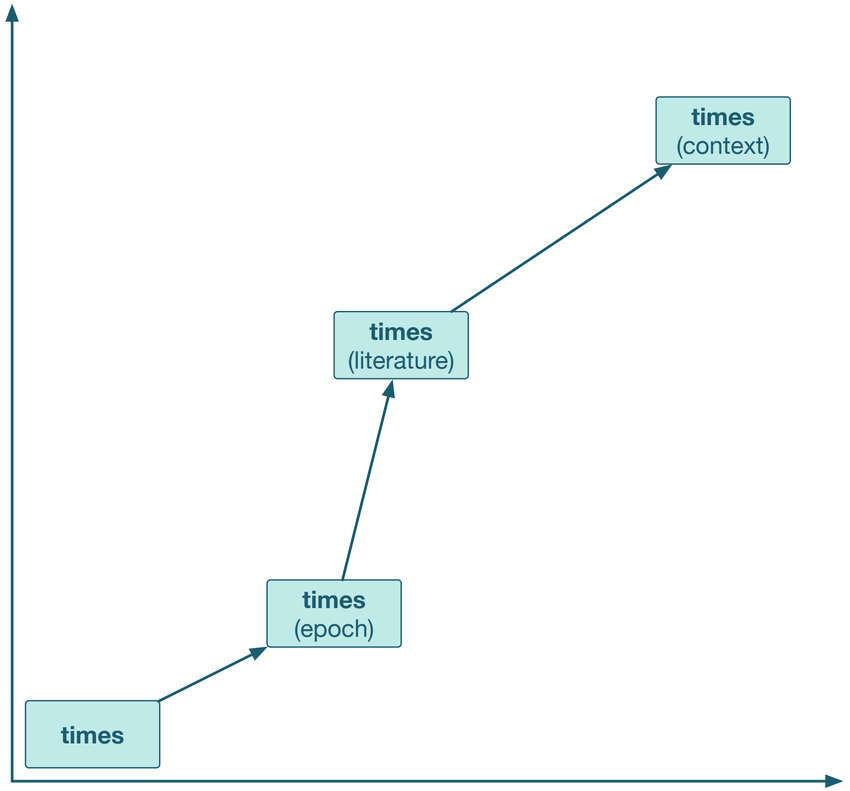
Transformers



NB: "Transformer" is the **T** in GP**T**

https://papers.nips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf





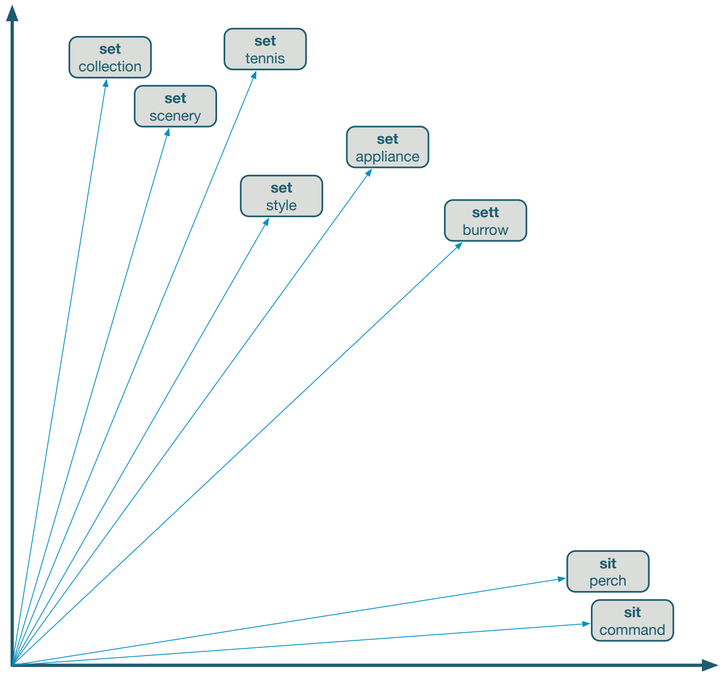
NB: Remember that the

NB: This process is termed *attention*

The mapping of a word to a vector is an *Embedding*

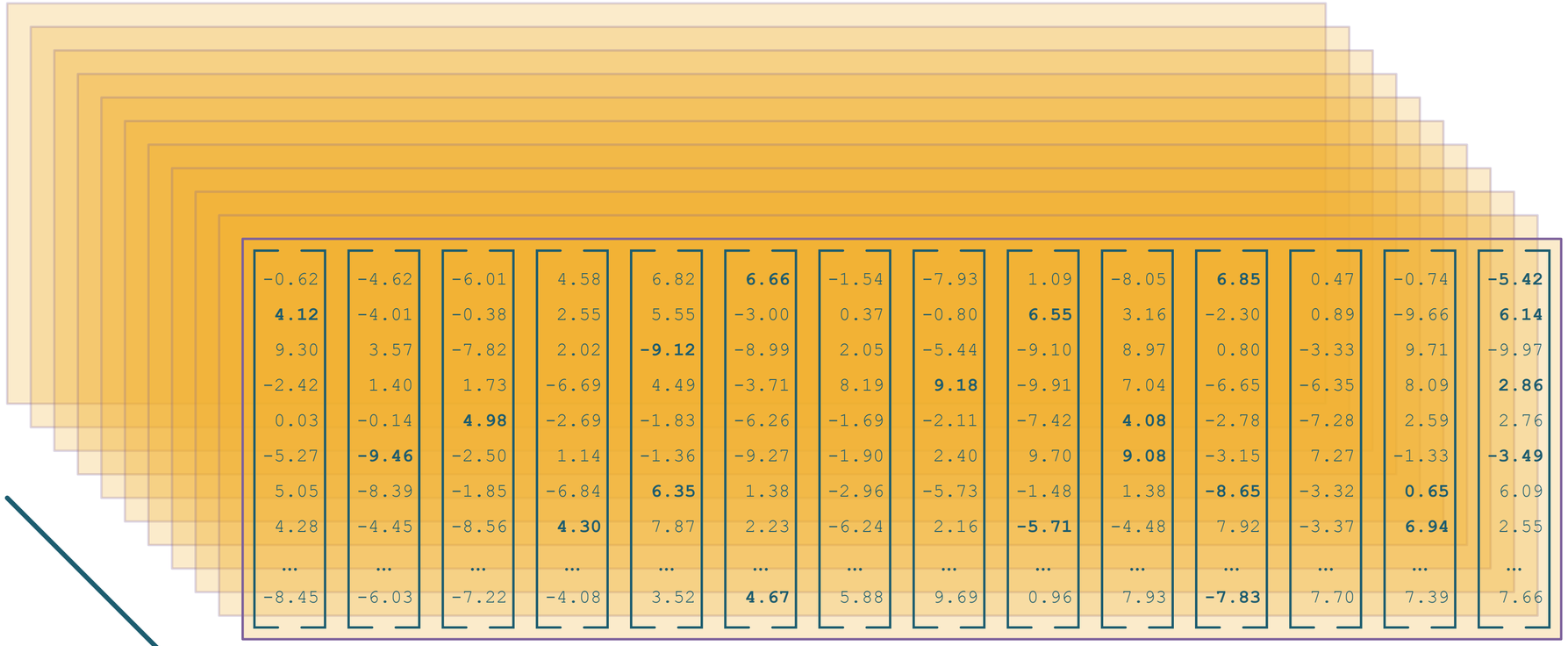
Similar terms will appear close in higher-dimension space

Embeddings *are* mathematical vectors



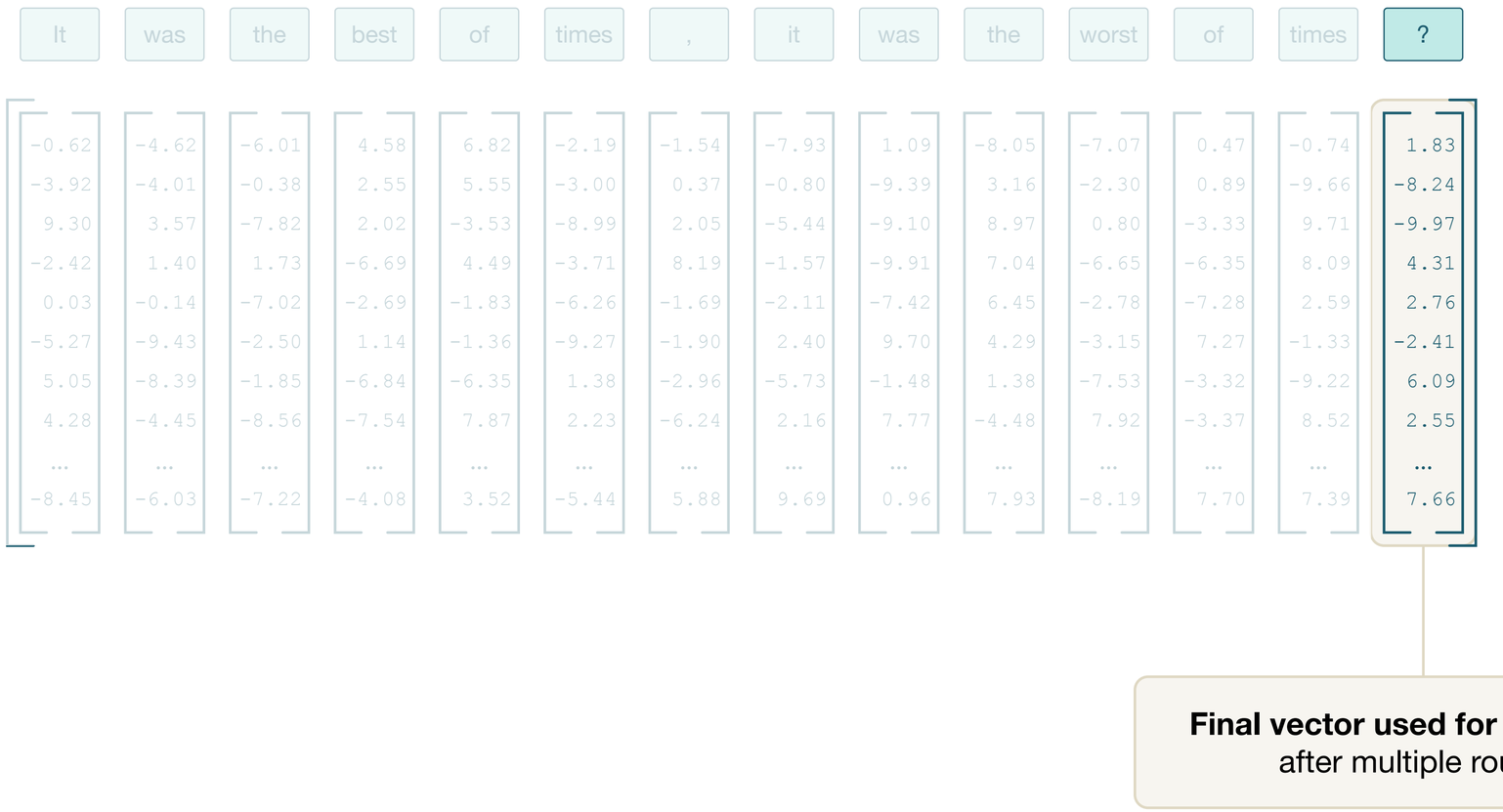
$$E(\text{man}) - E(\text{woman}) + E(\text{uncle}) \approx E(\text{aunt})$$

https://youtu.be/wjZofJX0v4M



Vectors fed into multiple layers
each layer takes input from the last

NB: These are *feed forward layers*

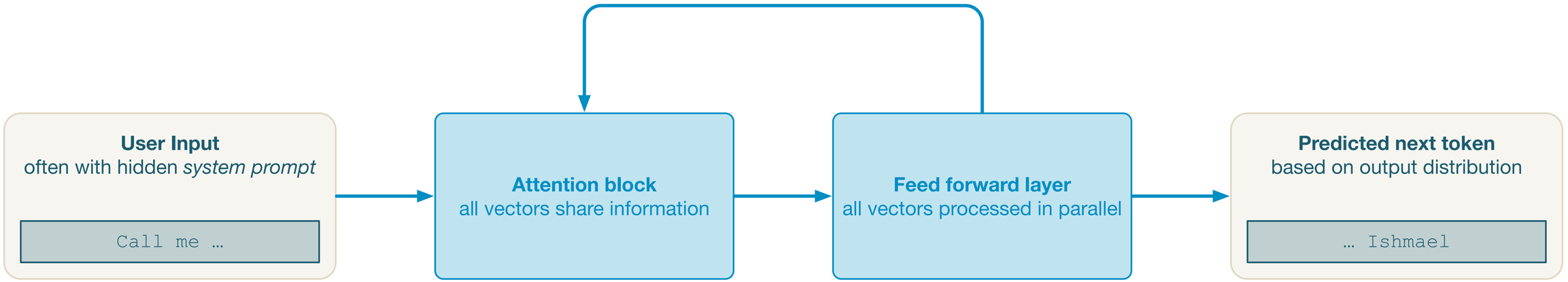


Final vector used for prediction
after multiple rounds

Probability of next work
All words given a probability

… ,	85%
… .	8%
… ;	3%
… –	1%
… in	<1%
… for	<1%
… and	<1%
… during	<1%
… but	<1%
… when	<1%

…


User Input
often with hidden system prompt
Call me …
Attention block
all vectors share information
Feed forward layer
all vectors processed in parallel
Predicted next token
based on output distribution
… Ishmael