W2. **Machine learning theory**

Last week: defined machine learning (Tom Mitchell): algorithm improves at some task with experience

Data (input) —> guess = output (prediction) —-> feedback —> new guess (= repeat) etc.

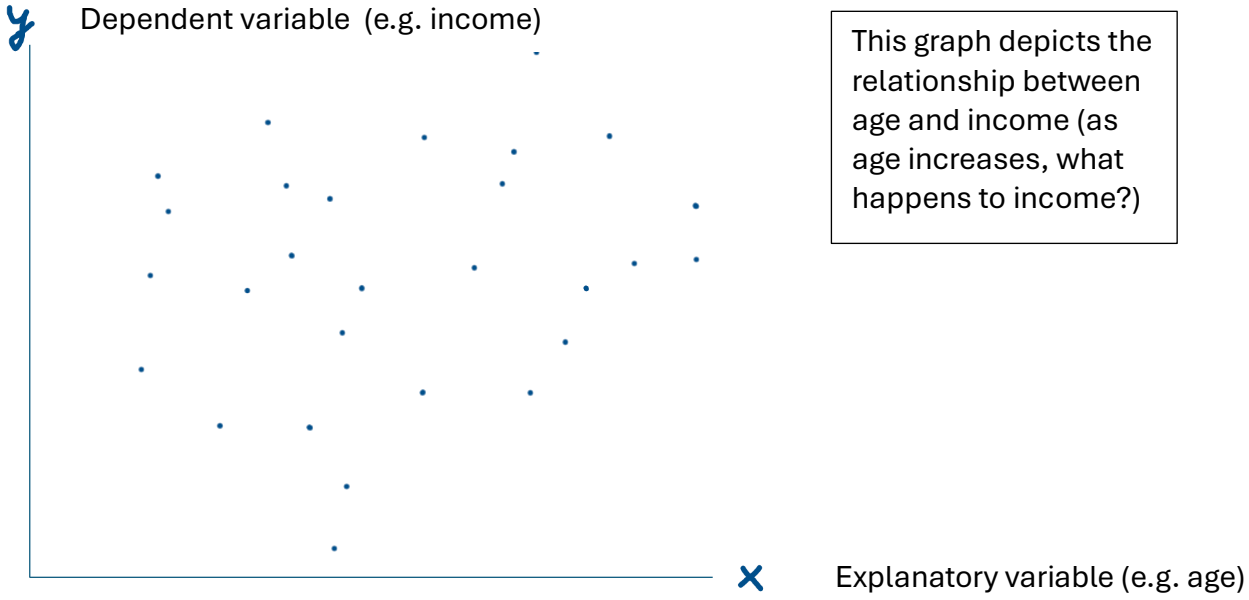
—problematize: what task (=prediction), what improve, what experience
—> against calling this “learning”

- bc of what we discussed last week
- bc of narrative, and logic
- and because of a technical reason having to do with math = “we can do it all at once” (i.e. without “experience”)—for this, let’s compare with statistics.

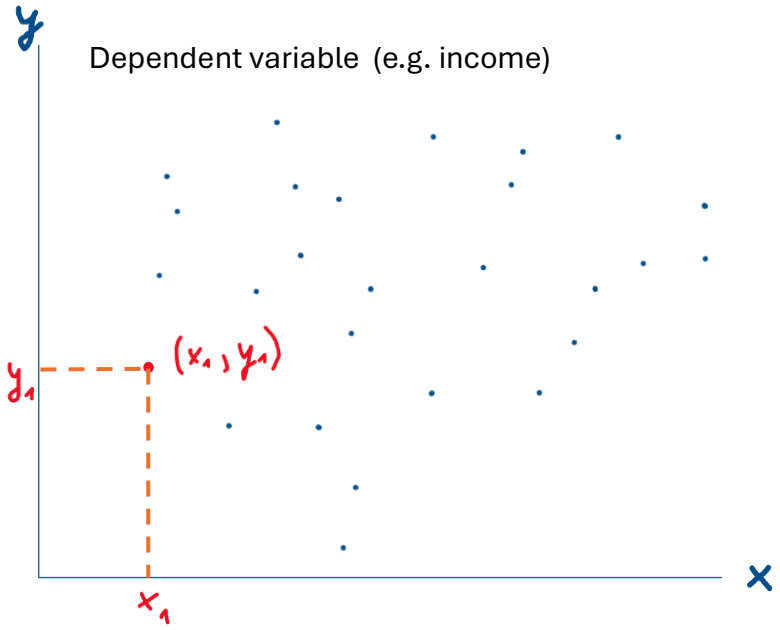
Revisit linear regression: Same process! But all at once, instead of through iteration

We want to recall and understand the following basic things about linear regression:

- what it means to “capture the patterns in the data”
- what it means to capture those patterns all in one go vs. through iteration, i.e. through “experience”
- what it means to make a *prediction*
- (maybe: reminder about omitted variable bias)
- A quick reminder about the mathematical way to write the line = capture the pattern



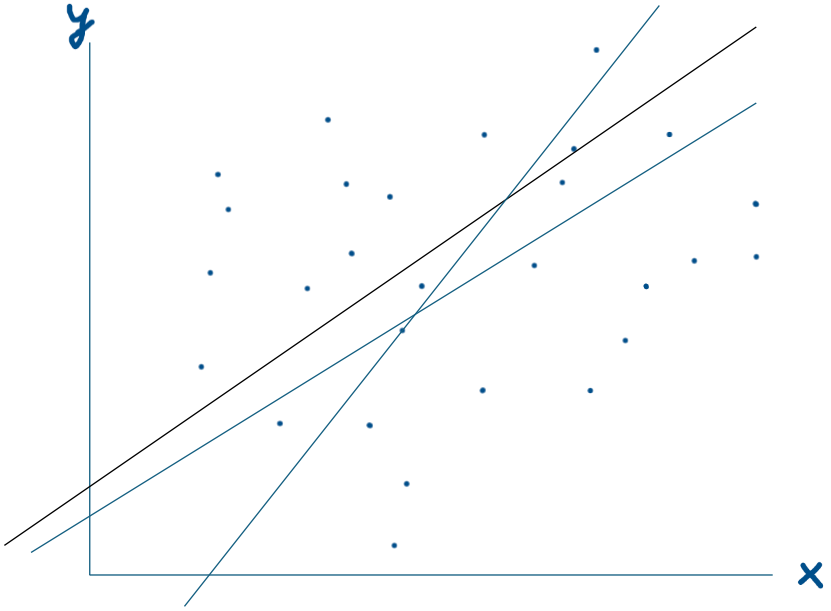
This graph depicts the relationship between age and income (as age increases, what happens to income?)



Any point (x_1, y_1) is an observation: individual 1 has x_1 age and y_1 income

Explanatory variable (e.g. age)

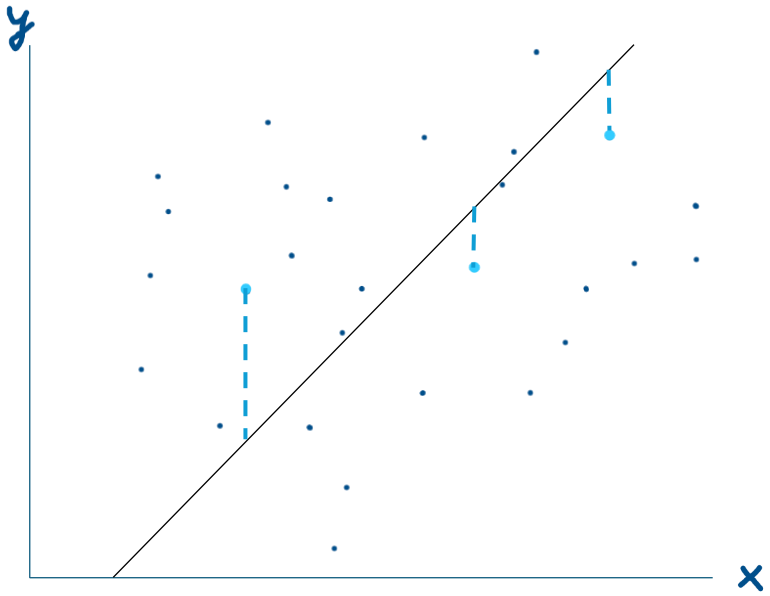
If we were able to "summarize" or "simplify" this data cloud into a simple shape that we can easily describe, then we could tell a story about the relationship between x (age) and y (income)…



Which line does the best job of capturing this data cloud…?

Explanatory variable (e.g. age)

→ the task is to find the best line that "captures the direction" of this data cloud. Which means: a line whose every point tries to minimize the distance between the line and the data at that point:

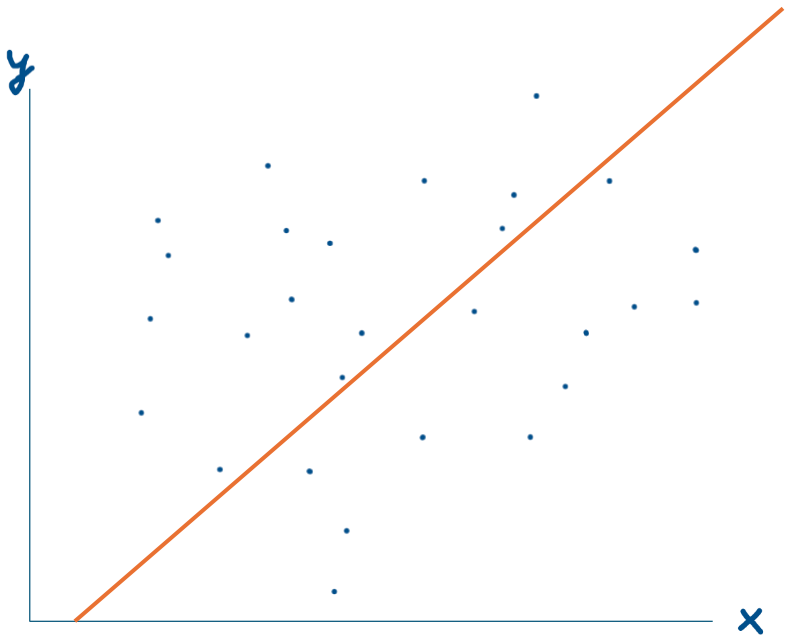


For this line, I've illustrated that "distance from the data" for 3 randomly chosen points.

The ideal line calculates this distance for ALL data points, sums them up, and then chooses the points on the line *in such a way* that the SUM OF DISTANCES is as small as possible.

Note: this is one calculation.

The result of this SINGLE calculation is the one line that minimizes the distance to the data:



This line is the regression line. It is the outcome of a single calculation. Mathematically, it is:
$y = w * x + w_0$

where ***w*** = ***regression coefficient*** (which we were looking for)

and w_0 is the intercept

To repeat, this regression line (⇔ regression coefficients) is the outcome of a single calculation: the minimization of the sum of all distances between the line and the data points.

That minimization is also the objective of machine learning – we'll compare that in a second and then we'll review what **prediction** is, and **how to interpret the regression equation**. Then we'll go into the technicalities of machine learning.

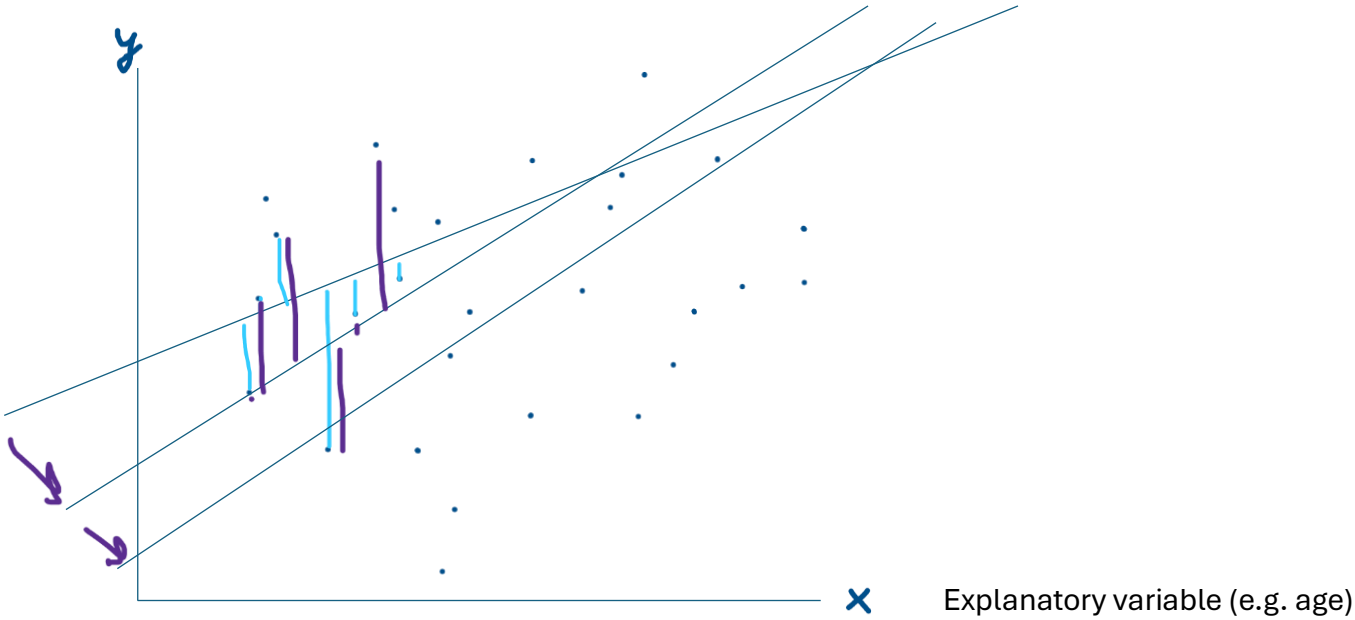
Machine learning's iteration ("experience") vs. the single calculation

Pushing back against the notion of learning:

What machine learning does is:

1. Start with a random guess of a line ("just chuck a random line on the graph")
2. Calculate all the distances between that line and the data points (= "the cost")
3. Perform gradient descent on this cost, to move it toward a direction where it decreases ==> use GD to determine how to adjust the regression coefficient *w*
4. Draw a new line (= new output) based on the updated *w,* and repeat the whole process (for which this new line is now step 1).
5. Do this until gradient descent produces no further decrease of the cost (= the distance from the data), take that *w* as the best one, and use that *w* to draw the final line.

Let's draw this ourselves:



This iteration is what is understood as "learning." You see why I say this has little relationship to how humans learn.

The outcome ***is the same as for good old fashioned inferential statistics.***

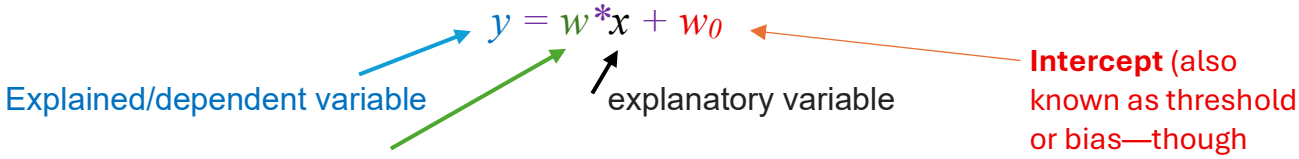
➔ **Machine learning is just statistics, but in iterated form** instead of calculated all at once.

This means:

- There is no actual learning there
- But it can solve calculations that are too complicated to be calculated all at once (because the data is too large, and too complex)
- The conceptual underpinnings and interpretation of good old fashioned inferential statistics really matter for understanding ML—this is what we turn to now.

Interpretation of regression line / regression equation

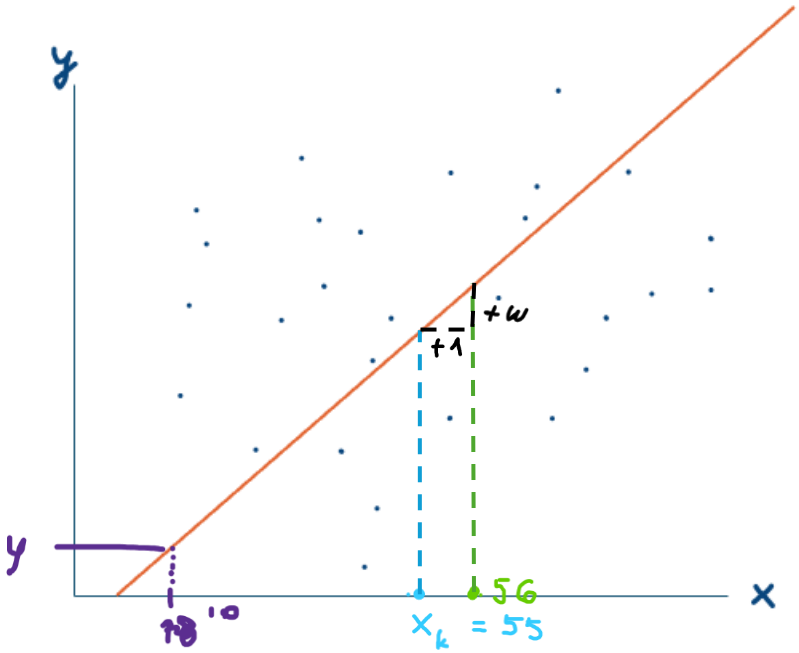
Regression equation with one explanatory variable (e.g. age):



Regression coefficient (aka "weight")
relationship between explanatory variable and dependent variable
= impact of x on y ?? (= cause..???? → "correlation does not equal causation")
= "if x increases marginally (i.e. by 1), how much does y increase or decrease?"
= (geometrically) slope

So the interpretation of the regression equation / regression line is:

"The regression tells us what happens to y (income) with one additional unit of x (= one additional year of age)."



This is called **inference:** we are able to describe the relationship of these two variables (age and income), and we are able to make **predictions—**that is, "educated guesses" about what we expect income (y) to be for an age (x) for which we don't have data. Let's take a look at prediction:

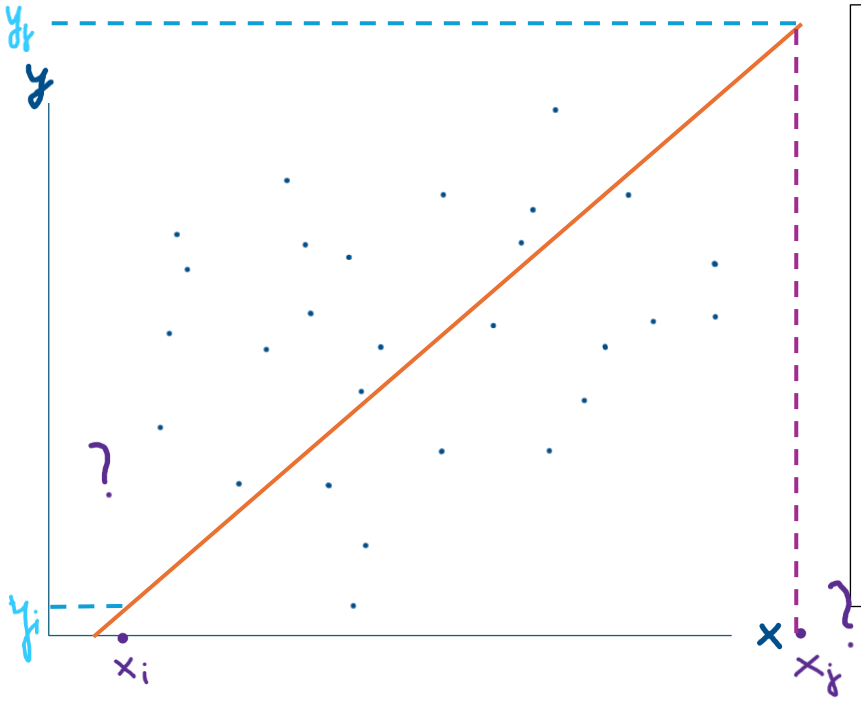
Prediction

Prediction just means calculating an estimated y (income) for a value of x for which we don't have data (= an age that is "outside the dataset").

How can we see what is in the data set? Well, it's just the data points…

So, for example, in this data we don't have data points for x_i and x_j. (Read off the chart: at x_i and x_j, if you go up vertically, do you see a point?)
Perhaps these values of x are age 10 and 99, so that's why there's no data on anyone of those ages earning an income. But perhaps we want to find out how much we'd expect someone at these ages to earn? To do that, we would make a "prediction", i.e. an educated guess.

Intuitively, what we do for a prediction is just read off what value of y the line gives at x_i and x_j, respectively (see graph). Mathematically, we plug the x values into the regression equation (see text box)



Suppose we want to know how much income a person with the age of x_i and x_j would have—but no such person is in our data set!

The regression line gives us an answer: an "educated guess" would be
$y_i = w^*x_i + w_0$ and
$y_j = w^*x_j + w_0$

which just means, which point on the line is associated with x_i and x_j respectively?

The result is y_i and y_j, which are perhaps £10 and £1b. We can say: "*A person of age 10 is predicted by the model to earn £10 a year, and a person of age 99 is predicted to earn £1b a year.*"

A prediction just means: assuming that the relationship captured by the regression line is (a) correct, (b) not too general, and not too specific, and (c) stable, I can infer the value of y pertaining to any value of x by just plugging that x into the equation.

(Note, this hinges on knowing the correct *w…!* This means: prediction depends on capturing the patterns in the data well.)

As seen from this example, **prediction** is pretty much always feasible (for example, I could extend the line into infinity) **but *it doesn't necessarily make sense***. *("A kid of 10 years of age is expected to earn on average £10/year?!?")*

So always be critical of predictions.

Any output of a machine learning process is a prediction. (When ChatGPT "says" something, it "predicts" which sequences of letters is the most likely given your prompt.)

<u>Some important caveats</u> (aka, some reasons why predictions or inference wouldn't make sense)**: omitted variable bias , and functional form (overfitting, underfitting)**

We've been looking at a regression with one explanatory variable (because it's easy to draw). But—we typically have multiple explanatory variables (for example, other than age, what might contribute to income? Work experience, education level, biases of employers, etc… so let's just look at an example with multiple explanatory variables (also called regressors):

Regression equation with multiple explanatory variables (e.g. age and work experience):

$$y = w_1 * x_1 + w_2 * x_2 + w_0$$

Interpretation:

"If I increase x_1 by 1, but hold x_2 constant, how much does y change?" (or vice versa)

"With one additional year of work experience, and *everything else being held constant,* how much does income go up?" (or the same with age)

The idea is that any phenomenon (such as income) is too complex to be explained by just one explanatory variable.

What could go wrong if we don't include enough explanatory variables: Omitted variable bias

OVB: = when we omit an explanatory variable that is correlated **both** with one that we have included **and** with the dependent variable. In this case, the explanatory variable that we have included will be the one that (wrongly!) expresses the relationship between the omitted variable and the dependent variable.

Examples:

<u>OVB 1: Age and work experience</u>
Work experience (i.e. years worked) is probably a better cause of higher income than age, but it is also correlated with age. If we omit work experience and use only age as an explanatory variable for income, the age variable will (wrongly!) express the impact of work experience on income, and give us the misleading impression that age *causes* income to go up.

Note: the prediction could still be right (i.e. with 1 additional year of age, income goes up by say £1000), but the reason for it (the explanation) is wrong. It's not age causing the income to go up.

<u>OVB 2: Race and poverty</u>
A person's income level is typically adversely affected by the condition of poverty, and poverty in the UK (and elsewhere) is typically correlated with race, due to historic reasons such as segregation, Islamophobia, racism, and so on, which have made housing, education, and access to jobs much more difficult for people of color. If I want to explain income by using only race as an explanatory factor and omitting poverty, I could rightly conclude that "a person of color earns on average £1000 less than a white person." But if I say that "this proves that people of color are inferior" /

"race explains income", that would be wrong (not just ethically, but mathematically). In this case, the variable "race" channels the impact of the omitted variable poverty. (And it can also channel the impact of other omitted variables such as employers' bias/discrimination, etc etc.) A more correct statement would be:

- On average, people of color earn £1000 less than white people, but we don't know from this why this is the case (= correlation exists, but no assumption about causation)
- On average, people of color earn £1000 less than white people, but this could be caused by many factors correlating with race, such as a higher likelihood of poverty, lower access to education, or employers' bias (=correlation exists, but we only speculate about causes, without making affirmative statements on them)

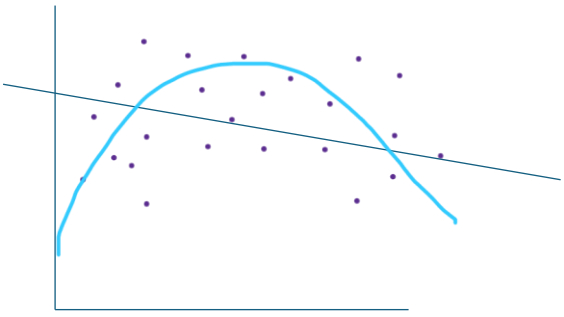
<u>OVB 3: ER doctors and death</u>
In a dataset of 100 road accidents, the likelihood of the death of the people involved is investigated. The variable considered is whether an ER doctor arrived to the scene or not. It turns out that if a doctor was present, the person in the accident is more likely to die. Can we conclude from this that we shouldn't ever call doctors to the scene, because we don't want to die?

Of course not. The omitted variable is *severity of accident.* The more severe the accident, the more likely the person is to die, but it's also likelier that a doctor will be called (= the omitted variable is correlated both with the included variable and with the dependent variable). The only included variable ("was doctor called") therefore wrongly expresses the influence of the omitted variable.

→ **Conclusion on omitted variable bias:** Predictions using correlation may still be right, but the explanations underlying them are wrong. So whenever an AI model makes a prediction (for example, it generates an image or some text), that prediction may be "correct" (e.g. the human in the image has 5 fingers on each hand, or the generated sentence makes sense grammatically), *but this is not because the underlying rules /causes have been understood.* The algorithm doesn't "understand" that human bodies frequently have 10 fingers, or that an English sentence typically needs a Subject and a Verb—the algorithm doesn't even "know" what a hand, a subject, or a verb are! All that it understands is that "when there are pixels in this and this shape, they're also likely to be here", or, "when there is a "th" sequence it is likely to be followed by "e" (*the*), "at" (*that*), "ere" *(there)* or "eir" *(their)*.

Prediction is purely about correlation, not causation.

underfitting and overfitting



The dark line **underfits** the data bc it doesn't capture the direction well (isn't always minimizing the distance from the data).
A different functional form **fits** the data better.

The light blue line **overfits** the data bc it doesn't captures this specific dataset too closely.
→ for a different dataset, it would perform poorly

A more general functional form **would perform better**, could be generalizable.

MACHINE LEARNING AND NEURAL NETWORK ARCHITECTURE

You can use ML to calculate a simple regression line. Or, you can use it for other tasks:

- Predictive models: like linear regression, what is someone's income likely to be, what is the likelihood that someone has committed welfare fraud / will get a disease / will reoffend etc.
- Classification models: which digit is depicted in the image (see below), what is the gender/race of the person on the image (:/), what object is on the image
- Generative models: what is the next likely sequence of letters/pixels (to create text or an image)
- Recommendation models: what content (song on Spotify, video on TikTok, post on X, news item on Facebook) should be presented to this user given their past engagement?

They all:

- boil down to making predictions
- based on capturing patterns in the data and expressing that mathematically as a function (finding the correct *w*'s)
- which has been found through iteration
- and through minimizing the cost function (=minimizing the distance from the data)

In the below, we go through this using a **classification model that classifies images of handwritten digits** into one of the options of (0, 1, 2, …. 9).

The two videos that will be relevant for us (you should watch these at home!) are:

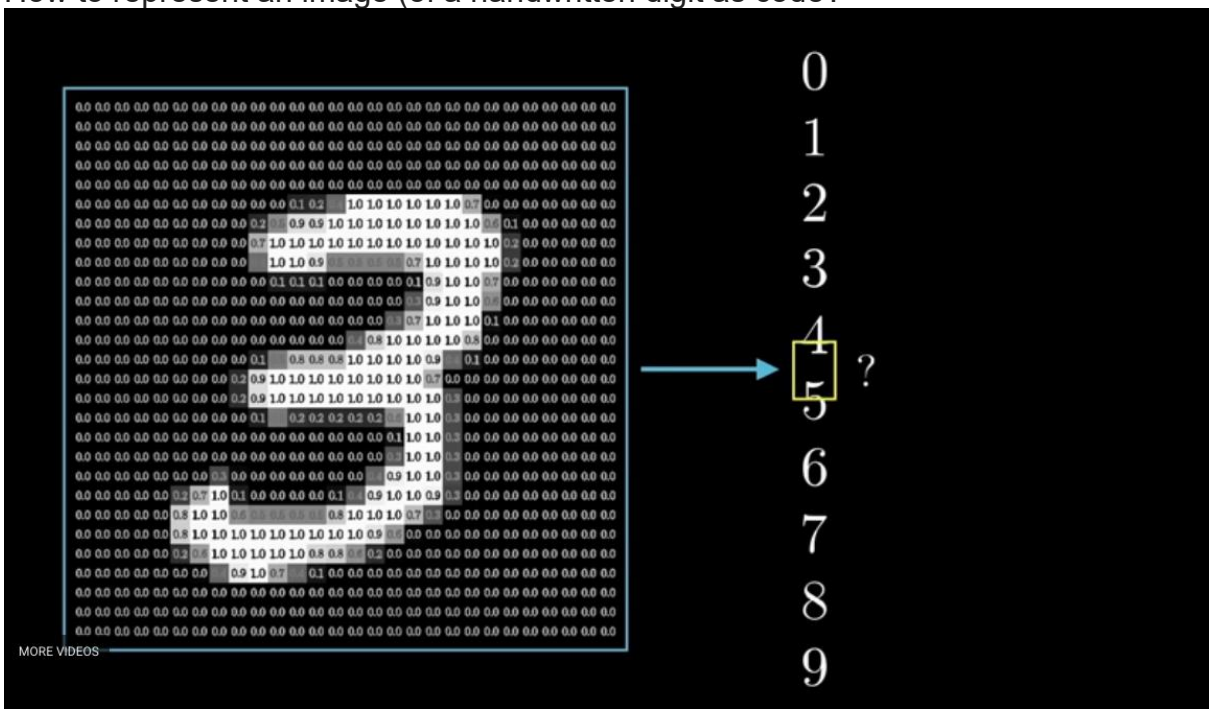
https://www.3blue1brown.com/?v=neural-networks

https://www.3blue1brown.com/?v=gradient-descent

Our structure is:

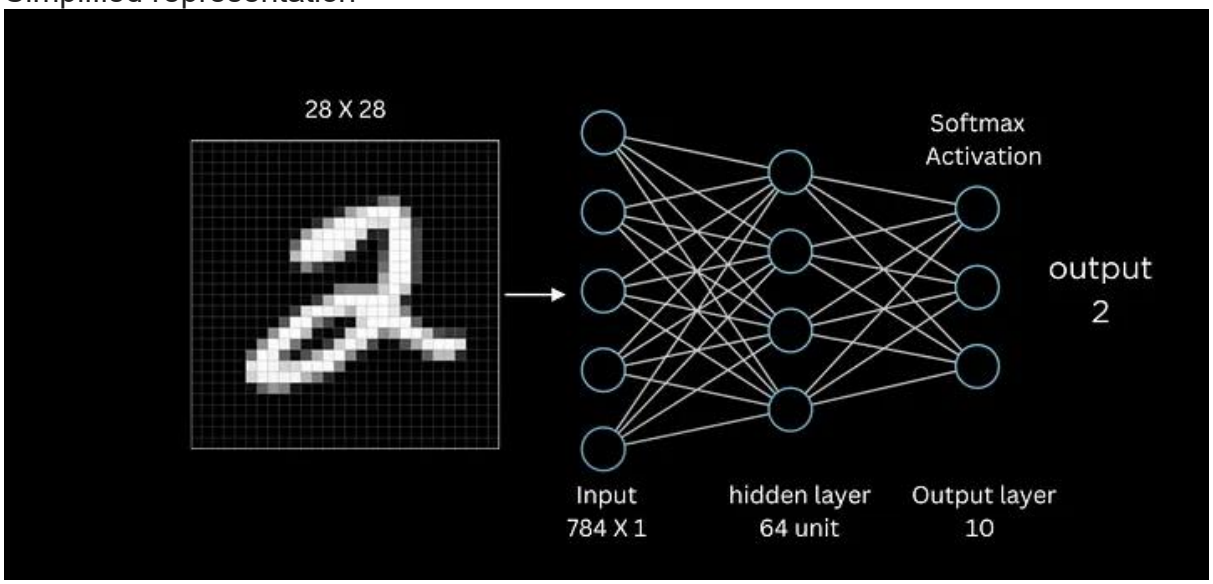
- how to capture an image into code
- basic structure of neural network
- what is a neuron
- how do neurons “communicate”

How to represent an image (of a handwritten digit as code?



What is a neural network?

Simplified representation



Source: https://github.com/mctripp10/neural-network-handwritten-digit-recognition?tab=readme-ov-file

- it's a layered structure – but it's a representation (nothing that "is" actually like this exists), and what it represents really are equation systems
- where each connection (edge) is a particular equation that communicates something
- but what tf is a neuron?? – just ooooone more moment please…

Compare a Neural Network (NN) to linear regression:

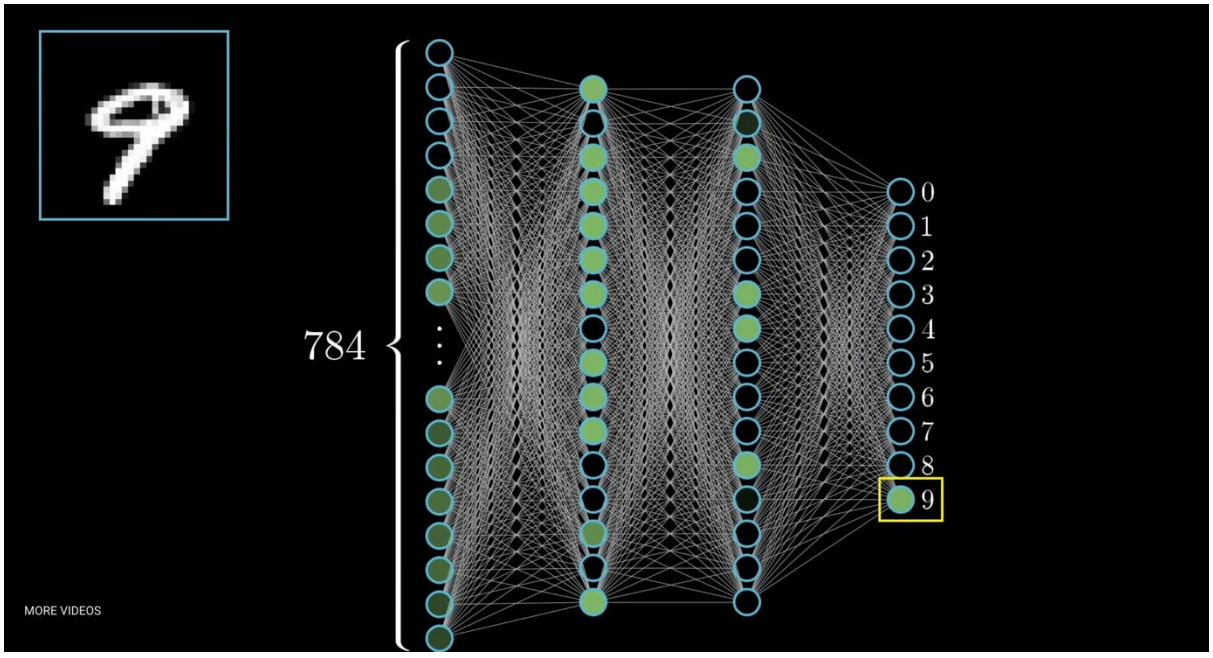
Linear regression: data cloud → regression equation → regression line (output)

NN (digit recog.): image of digit → neural network → classification output ("2")

In both cases, the objective is:
- find the coefficients *w* so that the distance from the data is minimized
- but the functional form in an NN is not linear, it takes the form of a Neural Network = equation system (complicated)
- the regression coefficients *w* are called, in NN, *weights or parameters.* In linear regression, there are usually 12 of them max. In NNs, there could be millions, even billions. But.. where are they? We'll find out!

"What the heck are neurons" + "where are the weights" ctd.:

Consider again our "3", or "2", and, below, the image of a "9":



This "9" image has 28 pixels by 28 pixels. 28 * 28 = … 784
→ the neurons in the first layer of the neural network simply represents each and

every one of the pixels of the input image.
→ this is called the **input layer**

There are 10 neurons at the end, in the last layer (**the output layer**). They each represent one of the 10 digits. Each of these neurons produces a score between 0 and 1, which translates to how likely that digit is considered to be depicted. Only the neuron with the highest likelihood score "fires": this gives us our output, which is a classification: "This image depicts a 9."

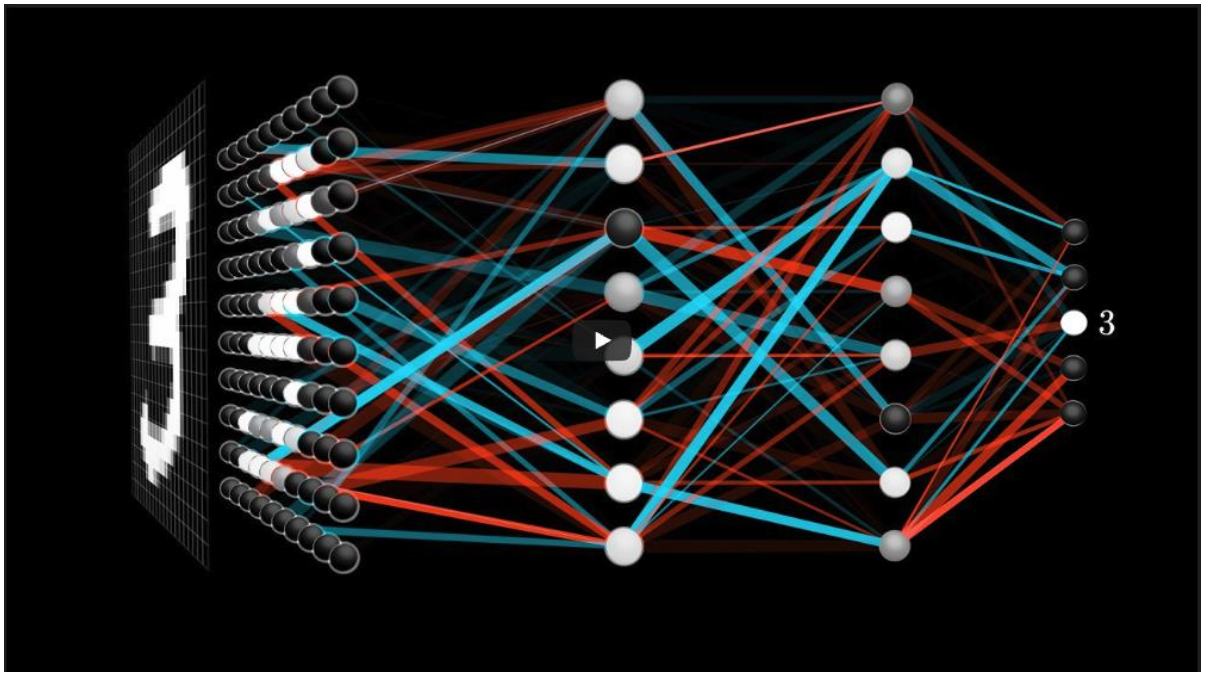
Neurons are a metaphor to the synapses of the brain—but only a metaphor: they are a point in the system that "fires" (or activates) if the information they pick up passes a certain threshold.

Analogy: the synapses in the brain fire if the nerves have picked up stimuli, so there is an electric current passing through the nervous system.

But neurons are not nerves or synapses. They are merely **nodes in an equation system,** where there are equations with particular threshold criteria. And that node can be understood as simply containing a number, which is called that node's **activation.**

To understand this, let's look at the output and input layers (and then we'll look at the hidden layers, too).

- Output layer: the threshold is determined by the probability score of all of the nodes; the node with the highest probability score activates ("fires"): this is the node that yields the output ("This is a 9.")
    - What these final nodes communicate, then, is the model's output = the classification of the image as depicting a particular digit.
- Input layer: for a 28*28 pixel image, each of the 784 nodes represents the value of the pixel. In this example, where the image is B/W and the digit is written in white, the higher the number (e.g. RGB code) of a particular pixel, the stronger the shape right there in the image.
    - What these nodes communicate, then, is "where in the image do we recognize a shape/writing"
    - In other words, the input layer *breaks down the image into bite-size parts* as illustrated below:



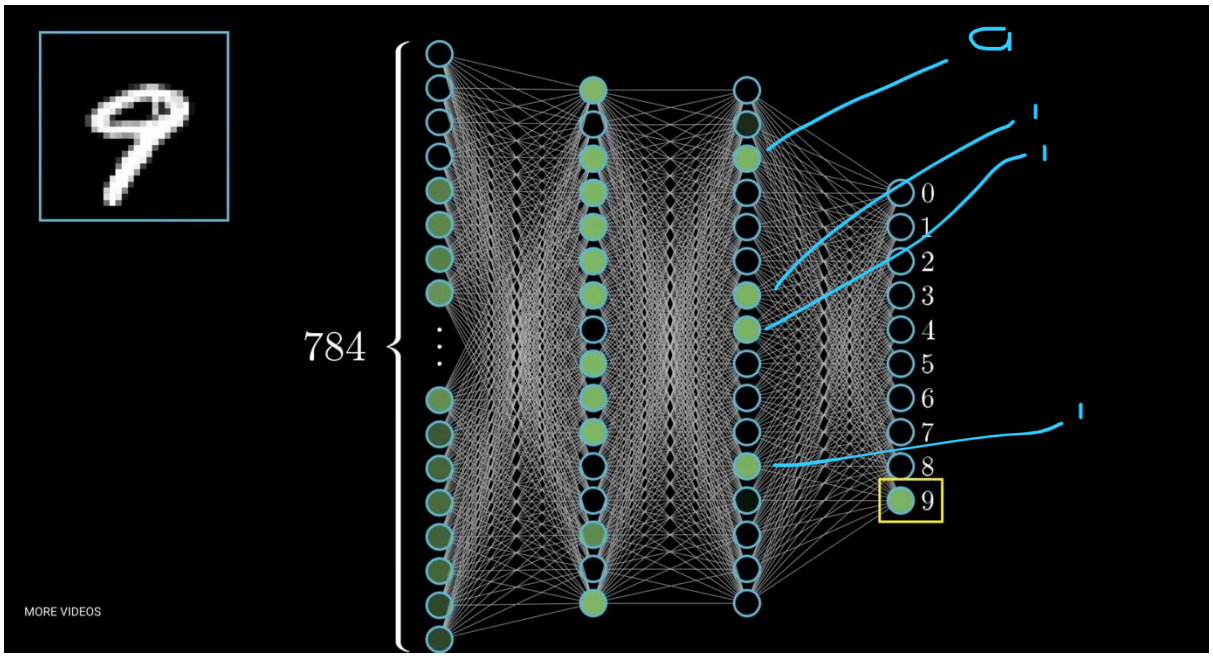
Now the question is: what happens in between the input and output layers?

These are called **hidden layers**.

What are these layers doing?

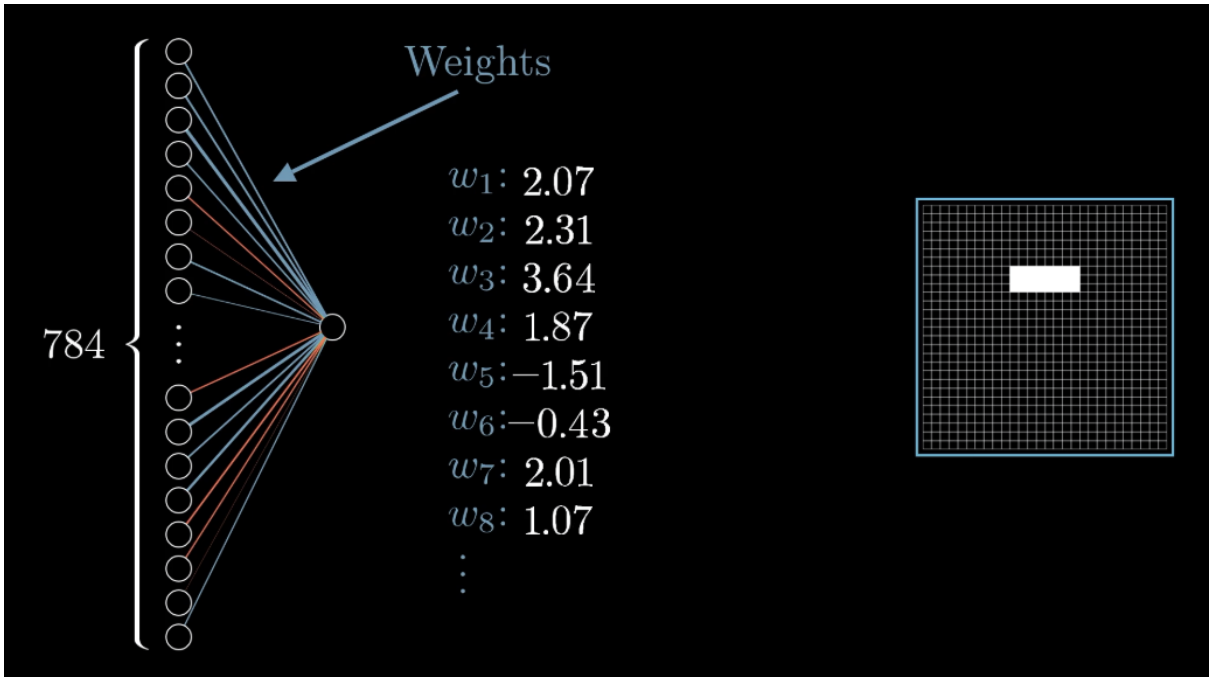
To understand this, let's go backwards from the output.

E.g. our node "9":



Though this is not 100% exact/correct, but we might think of them communicating in more or less this way.

So let's think about how this is communicated:



So, the way to think about weights is: **weights express how weighty/significant/impactful the information of any single node is for the activation of a next node.**

So for this particular neuron/node in the second layer, the information it receives is…:

$$w_1 a_1 + w_2 a_2 + w_3 a_3 + w_4 a_4 + \cdots + w_n a_n$$

Where the a's are each neuron's activations (number representing the color of that pixel, between 0.0 and 1.0).

What does this remind us of…?

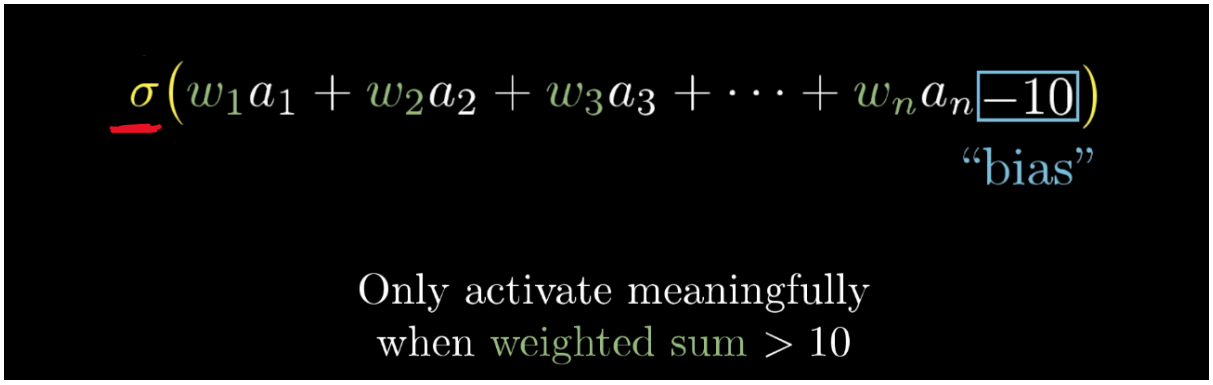
Each node in the second layer receives something like a regression equation.

What does this mean?

The logic/intuition is: we're breaking down the problem into many-many-many "mini-regressions."

Two other notes:

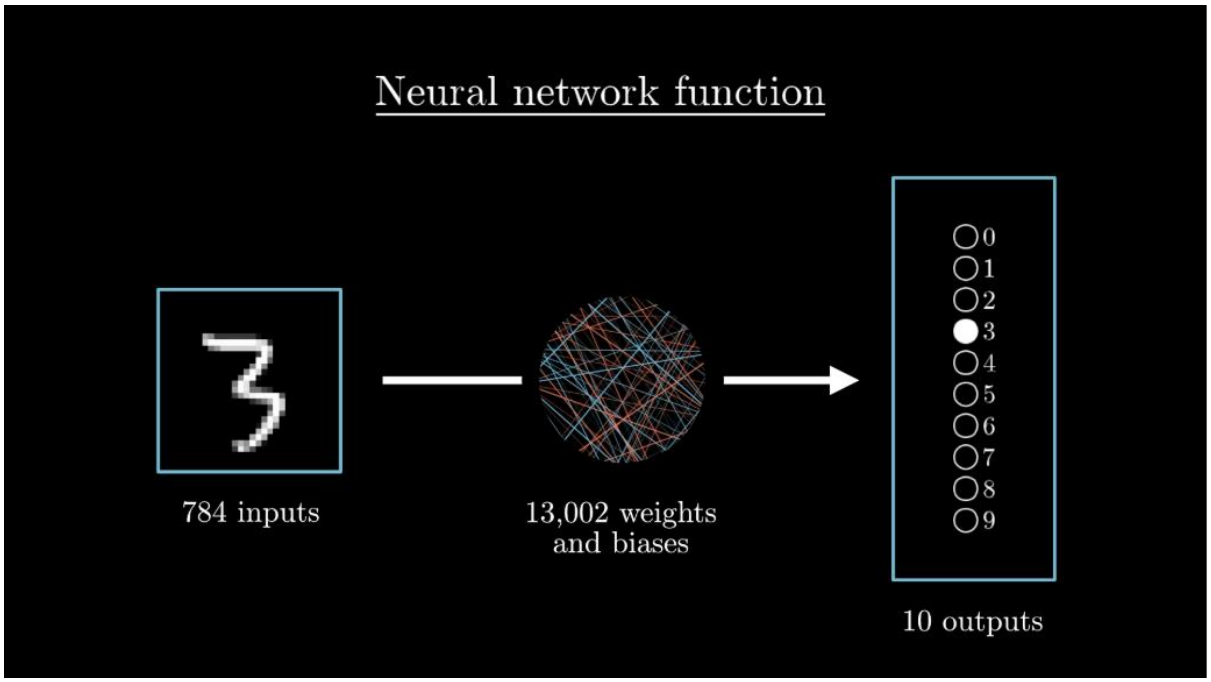
- “squishification”: Squeezing the result of the equations into the range of between 0 and 1 (does the neuron “fire” or not) → we apply a function to them that works like a filter, and this function is called a Sigmoid function
- Adding “bias” or “threshold” for firing



So you can think of the bias or threshold **as a condition (and if-statement) that the weighted sum has to meet in order for this neuron to fire.** If this condition isn’t met, the weighted sum “isn’t significant enough” for this neuron to fire.

This happens for every neuron… so, With this hidden layer of (say) 16 neurons, that’s 784x16 weights and 16 biases. = total of 13,002 parameters…

Then we can write this whole thing as matrix notation, but I won’t do that to you…



The training and learning process:

Cost function and gradient descent—minimizing distance

Activation functions

Supervised and other learning types—discuss importance of input data

e. Problems—ethical, power, truth, environment