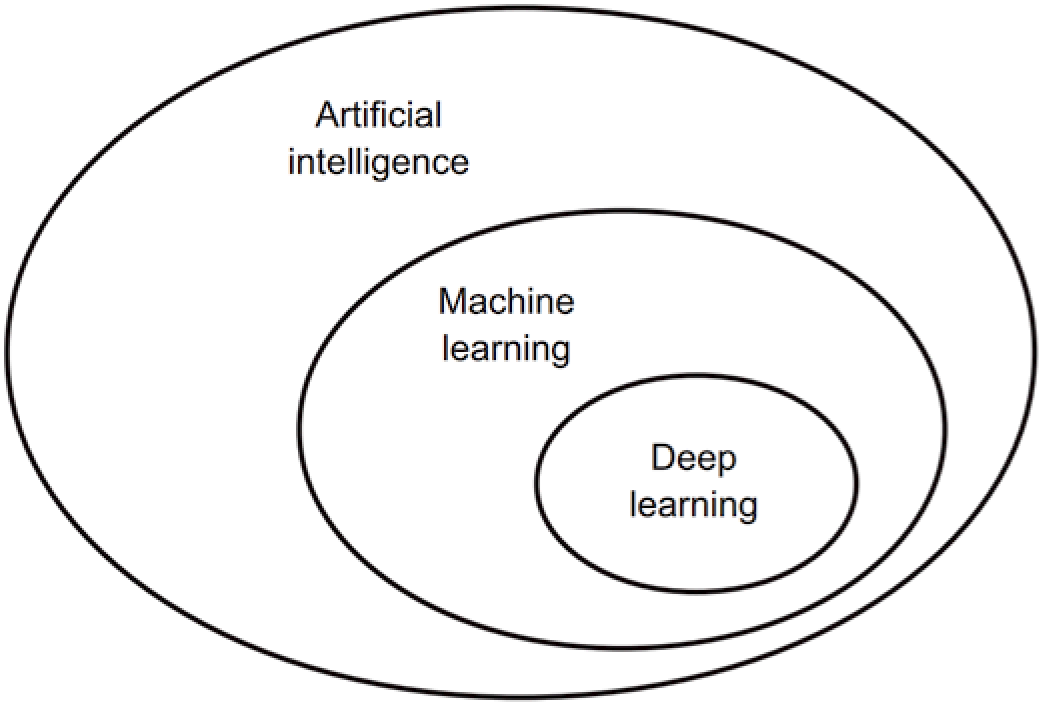
AI Hierarchy



Machine Learning

Computer algorithms that improve automatically through experience. (Once an LLM is put into production, they aren't improving or learning)

This is an inverse to the way we have been trained to write code (below) we are used to writing the rules and then receiving data. In this case we consume the data and then create the rules.

We provide the data, the algorithm develops the rules, and machine learning is often equated as AI, but is actually a subfield.

Deep Learning Basics

- Based on neural networks
- “Backpropagation” (Ability of a neural network to correct its errors)
- “Curve fitting”
- “Pattern Matching”
- Most of the current success in machine learning is based on this approach

Data Transformation

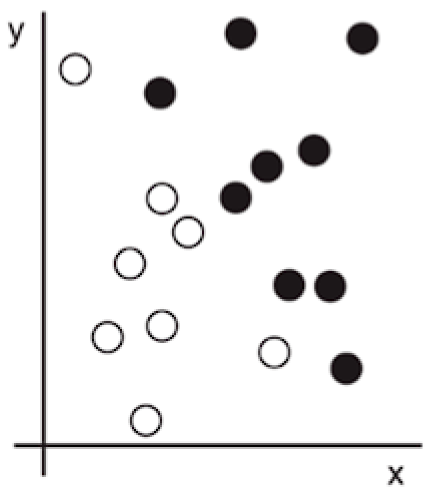
- The central problem in deep learning is to meaningfully transform data. IE learn useful representations of the input data at hand. Representations that get us closer to the expected output

  What is a representation?

  Example: A color image can be encoded in the RGB format or the HSV format. Two different representations of the same data.

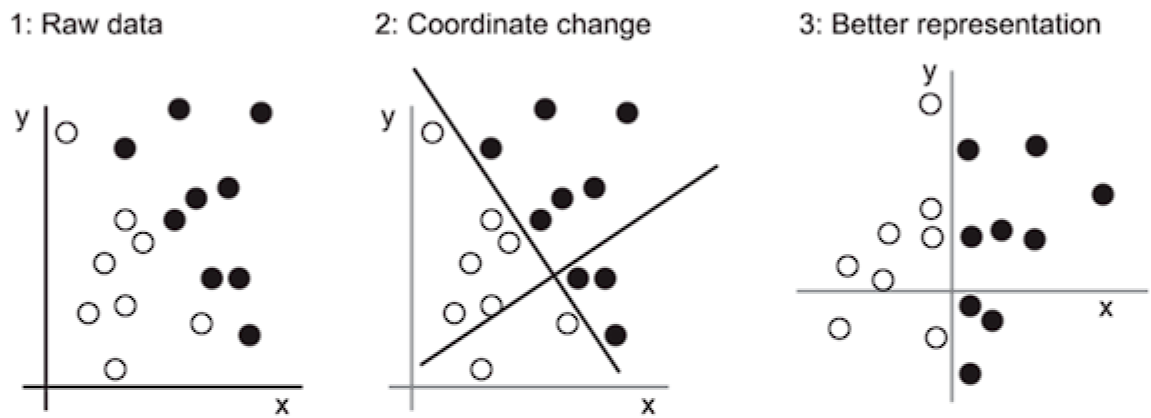
Example Transformation

Consider an x axis and a y axis and some points represented by their coordinates in the X Y system.



As you can see there's a few white points and a few black points. Let's say we want to develop an algorithm that can take the coordinates (x,y) of a point and output whether the point is likely to be black or white. The inputs are the coordinates of our points (data) and the expected outputs are the colors of our points. A way to measure if our algorithm is doing a good job could be for example the percentage of points that are being correctly classified.

What we need to do is have a new representation of our data that cleanly separates the white points form the black points, a coordinate change in this case would work.



In this case, we defined the coordinate change by hand: We used our human intelligence to discover our own representation of the data. This is fine for such an extremely simple problem, but can you still do the same task if you were classifying images of handwritten digits?

Could you write down explicit, computer executable image transformations to show the difference between a 6 and an 8, between a 1 and a 7, and across all kinds of different handwriting?

This is possible to an extent, rules based on representations of digits such as “number of closed loops”, or vertical and horizontal pixel histograms, can do a decent job at telling apart handwritten digits. But finding such useful representations by hand is hard work, and as you can imagine the resulting rule based system would be brittle.

Every time you would come across a new example of handwriting that would break your rules, you would have to add new data transformations and new rules while taking into account their interactions with every previous rule.

Why don't we just automate this? What if we tried systematically searching for different sets of automatically generated representations of the data and rules based on them, identifying good ones by using as feedback the percentage of digits being correctly classified in some development dataset. We would then be doing machine learning. Learning, in the context of machine learning, describes an automatic search process for learning data transformations that produce useful representations of some data, guided by some feedback signal. IE representations that are amenable to simpler rules solving the task at hand.

In summary, that's what machine learning is

Searching for useful representations and rules based on some input data, within a predefined space of possibilities, using guidance from a feedback signal. We need a feedback signal to know if we are making progress. This simple idea allows for solving a remarkable range of intellectual tasks, from speech recognition to autonomous driving.