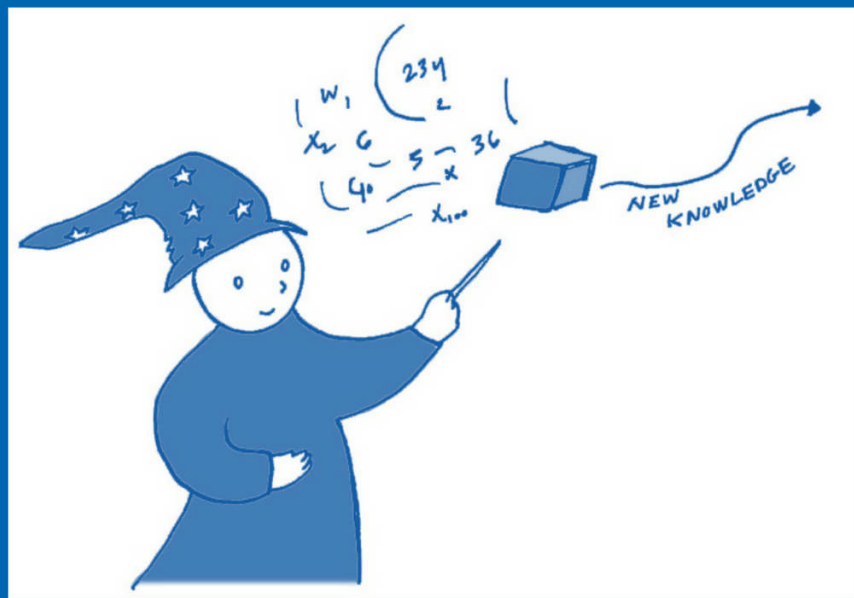


# HOW MACHINES LEARN

AN ILLUSTRATED GUIDE TO MACHINE LEARNING

by Helen Edwards



## ABOUT INTELLIGENTSIA RESEARCH

Intelligentsia Research is a market and investment research firm focused on artificial intelligence and machine learning.

We aim to distill the complex. To separate the reality from the hype. And to celebrate the advancements of today in the context of AI's history and long-term vision.

We bring a unique angle to research leveraging our backgrounds in large-scale IT, consumer product marketing, and Wall Street equity research to understand both the consumer experience as well as the underlying IT infrastructure required for delivery.

Our analysis is human. We develop ideas and models to help our clients understand why this era of AI is different and what matters for machine learning to be successfully adopted and developed, escaping the booms and busts of the past and bringing the considerable benefits of intelligent machines to society as a whole.

At the same time, maybe it's our grey hair, but we are constantly skeptical of exaggerated claims - both positive and negative - and bring you research that demystifies and critiques the inner workings of the technology.

Copyright © 2016 by Koru Ventures, LLC, owner of Intelligentsia Research

All rights reserved. No part of this publication may be reproduced, distributed, or transmitted in any form or by any means without prior written permission of the publisher, except in the case of brief quotations embodied in reviews, press coverage, and certain other noncommercial uses permitted by copyright law.

# TABLE OF CONTENTS

Forward	p. 4
What is Artificial Intelligence	p. 6
- A Brief History of Artificial Intelligence	p. 9
What Kinds of Artificial Intelligence Are There?	p. 10
Machine Learning	p. 13
- Supervised Learning	p. 14
- Unsupervised Learning	p. 16
- Reinforcement Learning	p. 17
Teaching a Machine to Learn	p. 19
- Inverse Deduction	p. 20
- Neural Networks and Deep Learning	p. 22
- How Do Machines See?	p. 29
- Evolutionary and Genetic Algorithms	p. 33
- Bayesian Algorithms	p. 36
- How Do Machines Converse?	p. 45
- Analogizers	p. 47
Human Guidance for Machines that Learn	p. 51
- Human Input #1: Knowledge of the Domain	p. 52
- Human Input #2: Engineering the Features	p. 53
- Human Input #3: Dealing with Whether the Model Works Well Enough	p. 54
Six Tips Before You Get Started	p. 55
- #1 Know What Problem You Are Trying to Solve	p. 55
- #2 Good Data > Big Data, Still	p. 56
- #3 It's More Science Than Software	p. 56
- #4 It's More Wisdom Than Hack-Dom	p. 57
- #5 AI Tends to Polarize People, the Message Matters	p. 58
- #6 Values and Transparency, Bias and Ethics	p. 58
The Last Word	p. 62
Acknowledgements	p. 63

# FORWARD

Artificial intelligence is changing our lives in ways we need to understand. Algorithms govern how we find information, how we learn, how we move, how we buy, what we buy, how we stay healthy, how we meet, whom we meet, how we are treated and what we are treated with. Marketing, analytics, diagnostics, manufacturing, driving, searching, speaking, seeing, hearing are all being disrupted and reshaped by machines that learn. Algorithms that can operate at the speed and scale that data is now generated are now making, what once was impossible, a practical reality. For example, algorithms that can:

Detect subtle patterns in transactions; across massive volumes of text, financial and location information to accurately predict fraud.

Process millions of data points across disparate data sources to reliably predict maintenance requirements of complex equipment, such as jet engines.

Fuse the large repositories of disparate data found in today's social sources, web logs, transactions systems, geo-spacial systems and individuals' personal devices to create actionable insights for marketers.

And as we fuse more artificial intelligence into our everyday lives, we learn more about ourselves; our preferences, how we choose, what we want to control, how we may be manipulated by machines.

But how do machines learn?

*"It's just math."*

- Oren Etzioni

Which implies that to understand artificial intelligence, you have to understand how machines learn, and, that to understand how machines learn, you have to understand the math. Schooling up on complex math is not something everyone wants to do so I created this book to show you the math that matters.

Why does it matter?

*"Computer says no."*

- Carol Beer, Little Britain

This catchphrase, made popular by the comedy show *Little Britain*, underlies a common concern in artificial intelligence; the lack of transparency and disempowerment that automated systems can create. Now that computer programs can adjust themselves, operate over larger and larger user bases, be physically embodied in robots or vehicles and do not require human input to keep

producing results, this concern becomes, not simply limited to unhelpfulness or poor customer service, but is potentially far more serious. Bias, ethics, discrimination and inaccuracy are all inevitable consequences of poorly designed and implemented artificially intelligent systems. To understand what decisions a human makes and how the mathematics drives an algorithm is an important part of understanding artificial intelligence.

The goal of this book is to get you up to speed on what drives the artificial intelligence you encounter today so you can understand what makes this field of computer science different from the software engineering of the past. It is aimed at executives who would like to use machine learning in their business and want to understand the underlying mechanics, and for anyone else who wants to understand more about the architectures driving artificial intelligence and machine learning.

# WHAT IS ARTIFICIAL INTELLIGENCE?

First, what is intelligence? There are multiple attributes we associate with intelligent behavior, including:

Having the ability to build, maintain and utilize a large store of knowledge

*Snails are related to shellfish, they live everywhere on earth, some can be used as musical instruments, some can shoot “love darts.”*

Commonsense reasoning

*If you stick a needle into someone’s arm, does it make a hole in the arm or the needle?*

Determining relationships between facts

*Tiddles is a cat. Cats are mortal. Tiddles is mortal.*

Being analytical

*Problem: How many planes are in the air right now?*

*Approach: How many airports? How many flights per airport? How long is each flight?*

Having the capacity to learn

*First steps, first word, first prize*

Communicating ideas to others and understanding their communication

*...----...    --- --    (This is Morse code where person 1 says SOS and person 2 says Ok)*

Perceiving the world

*The sky is blue*

Making sense of the world

*The sky is grey. It might rain.*

Artificial intelligence is machines doing these things. It has primarily been used for automation of labor. Many repetitive tasks have been automated – telephone operators, bank tellers, factory assembly workers – and many tasks lend themselves to computerization because they are made up of well-defined, repetitive procedures that can be easily encoded in software.

Artificial intelligence has mostly been about routine tasks, for example, removing a human from a repetitive spot welding sequence in vehicle fabrication. Over the last half century, increasing computing power has enabled more complex and sophisticated automation. But non-routine tasks, such as driving a car, recognizing objects in images, and anything involving interaction in natural language, have resisted automation because machines have not easily achieved many things that we find intuitive.

For example, take this simple statement:

Pat heard the alarm go off

For us it's obvious that an alarm going off is the same as an alarm ringing, but not so for a machine. We also intuitively understand that "Pat" is a person in this context. But a machine would also view "Pat" as potentially being a verb, at which point the sentence makes no sense at all.

As algorithms tackle vast streams of data in new ways, the way machines understand the world is changing. New data, such as images of every house in every street, countless photos of friends, pets and places, sensor data from fitness wearables or from connected cars allows machines to learn to distinguish a cat from a dog, a friend from a foe and what you don't want from what you will probably buy. With a little help from humans, machines take data and teach themselves about the world, updating their programs and improving on their own.

So, yes, while artificial intelligence is about automation, the explosion of data sources and the power of computing now means that what we can automate is rapidly changing. We are now able to automate tasks we've previously thought of as being uniquely human. Automation has become personalization.

Should we be worried?

Artificial intelligence futures are often presented on some continuum of utopia or dystopia. Elon Musk and Stephen Hawking are among experts who raise legitimate concerns about the invention of sentient artificial intelligence that could think and respond faster and more effectively than humans. An Oxford University philosopher, Nick Bostrom, talks about the "control problem" of how to set up an artificial intelligence that does not work against us once it achieves super-human intelligence.

Near term, there are important control considerations such as safe design of self-driving cars, international rules of engagement for autonomous warfare, ethics in statistical algorithms. There are also valid concerns over increasing technological unemployment as automation increases. Some of these I touch on later but my focus in this book is on a simple introduction to the science, math, and engineering that defines how machines learn.

The bottom line is that artificial intelligence is designed to improve our lives.

We routinely use the results of machine learning algorithms multiple times a day. Whether we talk to Siri or click on a recommendation from Amazon or become aware of how our newsfeed on Facebook is personalized, our everyday consumer



experiences are being transformed.

Natural language understanding is giving us personal assistants that truly understand our needs.

Faster and more accurate analysis and prediction is making us more confident in the professional advice we receive.

Smarter devices are making our homes more comfortable and secure. And more entertaining.

Better choices are making us more efficient. Whether it's in transportation, commerce, social, finance, entertainment or education, we have more control over how we use our time.

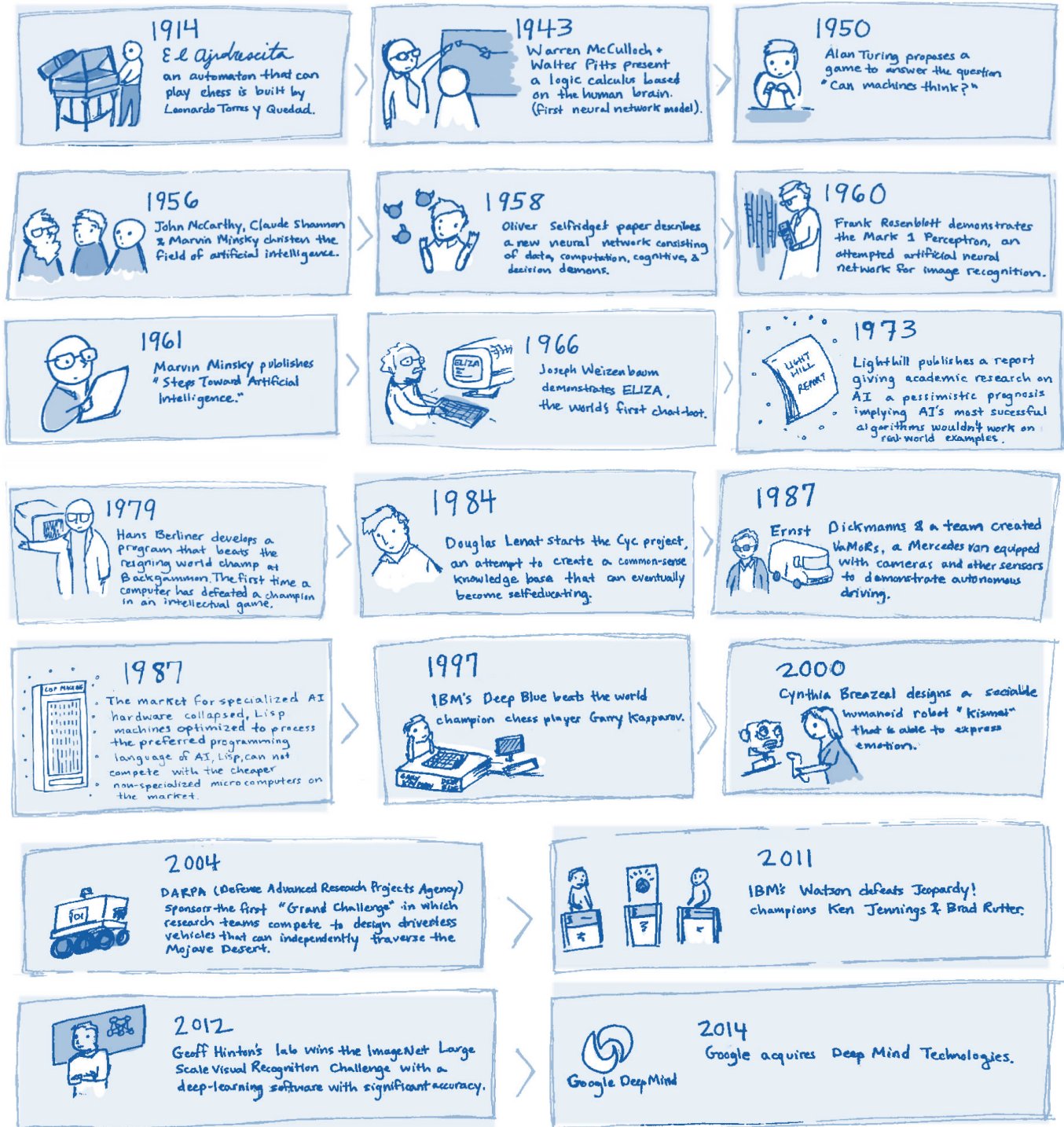
Computers are exceptionally good at tasks where the goal can be specified, such as drawing a graph from a table of numbers. Because so much human interaction and creative endeavor resides online and can be captured in data and shared and analyzed by algorithms, even more human-oriented tasks that involve social skills and creativity can now be learned by machines. Complex webs of human interaction and behavior are now routinely analyzed with the outputs used to personalize things for you.

In more and more instances, there is now no longer a technical barrier to a machine achieving any given goal without human guidance. This is a fundamentally different situation than how we have traditionally interacted with computers and is the principle reason why it's important to demystify how today's machines learn.

But first, a brief history of artificial intelligence; from Alan Turing's famous Turing test to Alpha Go, from artificial intelligence's promise through its multiple winters.

A 100-year journey...

## A BRIEF HISTORY OF AI



# WHAT KINDS OF ARTIFICIAL INTELLIGENCE ARE THERE?

There are numerous ways to categorize artificial intelligence. With its multiple crossovers and interrelationships it can, at times, be confusing.

One way to subdivide artificial intelligence is into two broad fields:

- Knowledge or Expert systems
- Machine Learning systems

**Knowledge or expert systems** provide a structure for capturing and encoding the knowledge of a human expert in a particular domain. Knowledge systems underpin almost all of what we consider traditional “if-then” rule systems and were the first truly successful artificial intelligence. They were invented in the 1970s and proliferated in the 1980s. For instance...

IF THIS, THEN THAT

If weather forecast app shows rain tomorrow,  
then alert me with a mobile notification.



While the vast majority of technology today remains rule-based, these systems have limited capacity and flexibility in the modern, internet connected world. They simply cannot change fast enough to be useful for long and they ultimately limit what people can do with information. For example, expert systems have been used for years to help doctors make the right treatment decisions. At the extreme, for a given set of inputs, an expert system will give *one* result that applies to everyone in the world. But now the goal of personalized medicine is to have as many results as there are *people* in the world.

A step change has happened in scale, speed and scope with the new artificial intelligence that is machine learning.

**Machine learning systems** provide computers with the ability to learn without explicitly being programmed. Algorithms build models, which then update and change as they are exposed to new data. Machine learning’s history began in the 1950s but recent progress is a result of convergence between the development of probabilistic tools in the 1990s and access to large datasets and

powerful chips used for graphics processing in recent years.

One example of machine learning in use is web search. Google runs sophisticated algorithms that personalize based on what Google knows about you as well as a multitude of other factors that Google knows about people like you and knows about the world. As a result, different people get different results when searching on the same thing. We even get different automatically generated prompts in our searches that can vary based on where and when we are searching. Try an experiment: type in “what is the best...” on different devices. Do you get the same suggestion each time?

My phone: “what is the best...way to retrieve an anchor.”

My laptop: “what is the best...pokemon in pokemon go.”

Another example is predicting prices. Say you want to predict the sale price of a home. You can create a model that predicts this based on a set of characteristics (square feet, number of bedrooms, number of bathrooms, and so on). If you use a static model, the algorithm runs only once and gives a one-time correlation, likely only applicable in a local area. But houses are constantly being bought and sold and there are many patterns seen in one location that can inform prices in all areas. Patterns we can't see because there are simply too many variables and too many hidden relationships. A machine learning algorithm can build a model in many, many variables, look at each new house sale, compare it against the model's predicted price for that house, and then adjust the model automatically to make it more accurate. Machine learning systems can also handle a lot more data, in different forms and at speed.

Just how important is this data deluge anyway?

### **Data growth is exponentially exponential...**

Worldwide data growth is on phenomenal growth curve, widely forecast to grow from 4.4 zettabytes in 2013 to 44 zettabytes by 2020 (a zettabyte = 1 trillion gigabytes). Even this number is regularly revised up. It's difficult to visualize this amount of data; all it says to most of us is “really, really big.”

### **...And lots of that data growth is coming from new sources.**

Global mobile data traffic will increase nearly eightfold between 2015 and 2020. Mobile network connection speeds will increase more than threefold by 2020. By 2020, more than three-fifths of all devices connected to the mobile network will be “smart” devices. This isn't limited to smartphones but also includes autonomous vehicles, smart home sensors, smart cities and a vast range of industrial sensors and controllers.

### **Computing power is also growing at an exponential rate...**

Even as the physical limit of chip speed approaches and Moore's Law reaches its limit,

supercomputers are still forecast to be around 30 times more powerful in 2020 than 2015. Many of the recent advances in artificial intelligence have been due to the use of graphics processing units (GPUs, more powerful chips primarily designed for graphics processing). Chips are now being designed specifically for machine learning applications, where speed and power consumption are optimized.

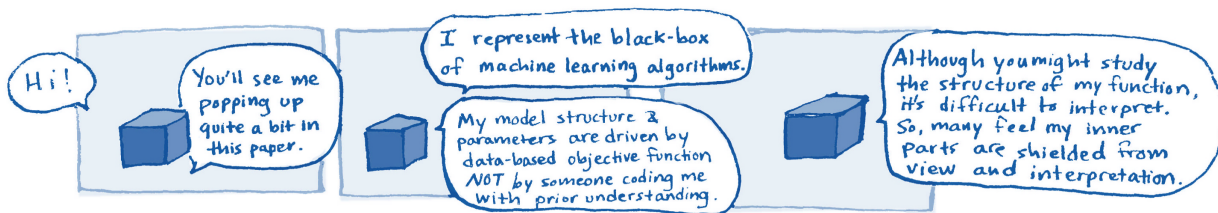
### **...Allowing data scientists to handle even the most complex “unstructured” data.**

Up until recently, data has required structure to be useful to computers but algorithms are increasingly able to process this data. Unstructured data is data that is unlabeled and not organized, often filled with ambiguities and irregularities. The vast majority of the world’s data is unstructured – images, video, text contained in documents.

Next up: machine learning, what it is, how it works and how it’s used.

But just before we do...

In the next section you’ll notice a small black box included in the illustrations. Many of the steps that machines use to learn contain processes that aren’t especially transparent. There may be steps where calculations are deeply embedded and difficult to visualize or extremely complex math that not even the experts completely understand. Whenever the black box appears, suffice to say there’s some “magic” that makes the algorithm work. This opacity makes learning algorithms very different from traditional code and is one of the core reasons it’s important to understand the math behind machine learning.



# MACHINE LEARNING

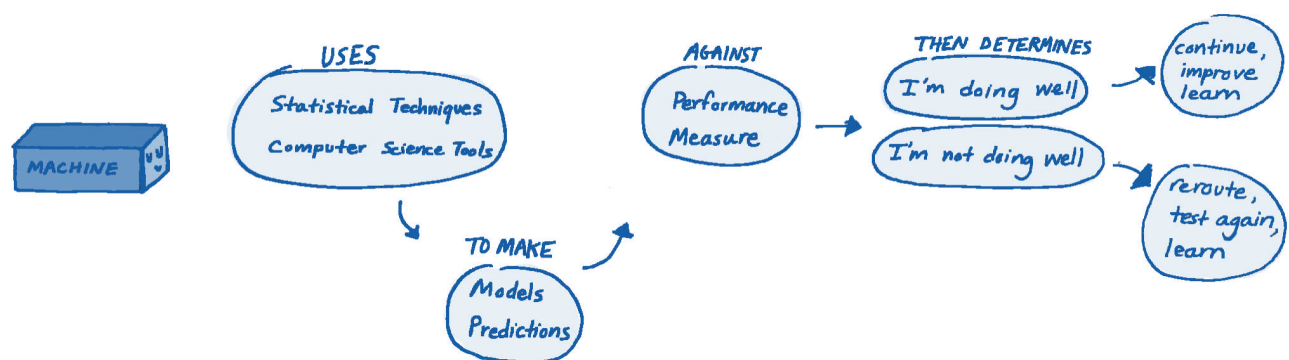
$$Y = f(x)$$

LEARN FROM X (RELEVANT DATA) TO MAKE Y (ACCURATE PREDICTIONS)

This simple mathematical function is at the heart of machine learning. We want to make predictions from input variables but we don't know the function that mathematically relates the inputs to the outputs. If we did we would use it, but because we don't we have to learn it from the data. A machine learns the relationship between inputs and outputs and then automatically improves its programs to make new predictions from fresh data.

There are many ways that machines can rewrite their own programs. Most rely on some form of calculus or statistical technique. All machine learning algorithms use data to train a model that can then make predictions or decisions against some performance measure. Data is the raw material; learning style depends on the type of data, the type of problem and the type of algorithm.

In the mathematical expression above, performance is measured based on how accurately the program predicts a value for an output that was already known. The difference between the predicted output and the actual output is the error. There are various methods for programming a computer to adjust its function based on this error.

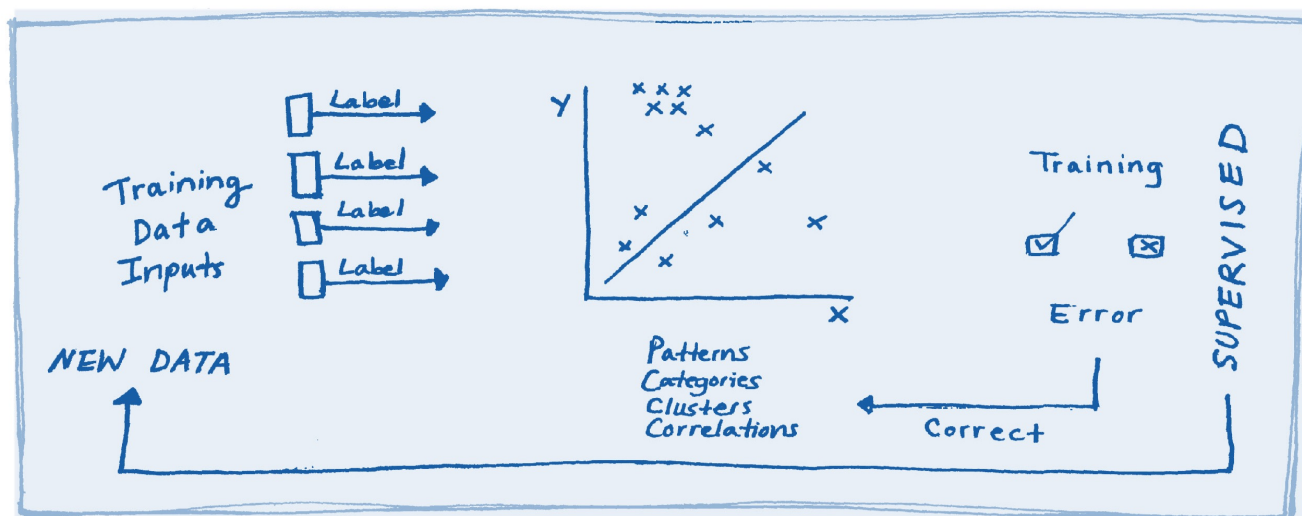


Before getting into how this is done, let's first cover how the machine knows there's an error to act on. There are three general classes of machine learning: supervised learning, unsupervised learning, and reinforcement learning.

## SUPERVISED LEARNING

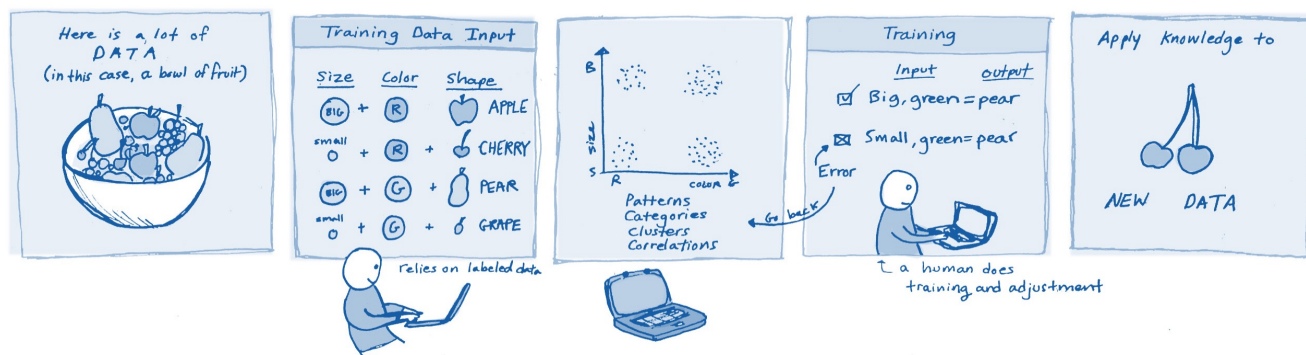
Most machine learning algorithms are supervised learning algorithms. Supervised learning algorithms make predictions based on a set of examples that are provided and labeled by humans. This means a human decides what data to use (called feature engineering), provides data to train the model (called training data), and provides data to test the model against (called test data).

In supervised learning, data is labeled with a value of interest, say, a particular stock price. The algorithm looks for patterns in the values. It can use any information that may be relevant (financial results, weather, world events) to find a pattern. Once it finds a pattern it makes a prediction for, say, tomorrow's price. When the model is being trained, it is corrected until the desired level of accuracy is reached.



This illustration shows data, which has all been labeled based on the characteristics of  $x$  and  $y$ . These labeled data points are plotted and a line has been drawn that fits the data best. We test the error by finding the point on the line of a known  $x$  and  $y$  (the prediction) and comparing it to the actual result (test). The program reshapes the line to produce a better fit, until some acceptable level of error is reached. This is an example of simple linear regression. Machine learning takes this technique and automated it based on a test, predict, adjust, and retest cycle.

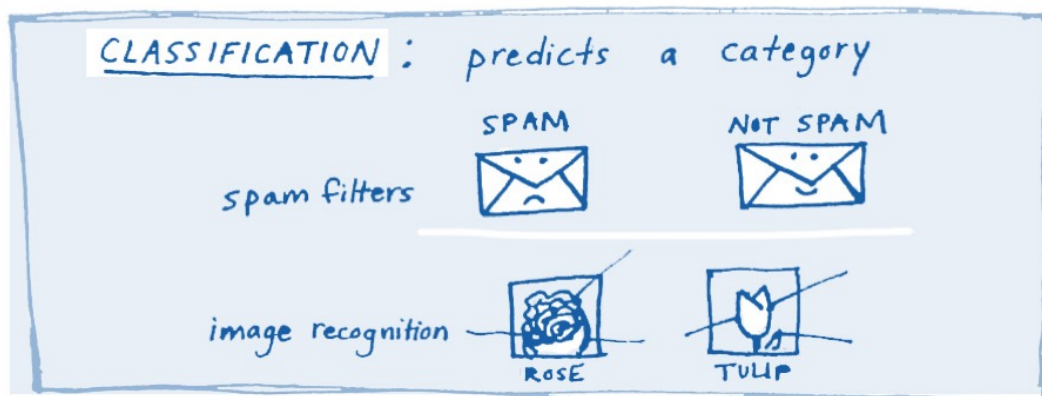
Let's look at classifying a bowl of fruit into different classes based on supervised training.



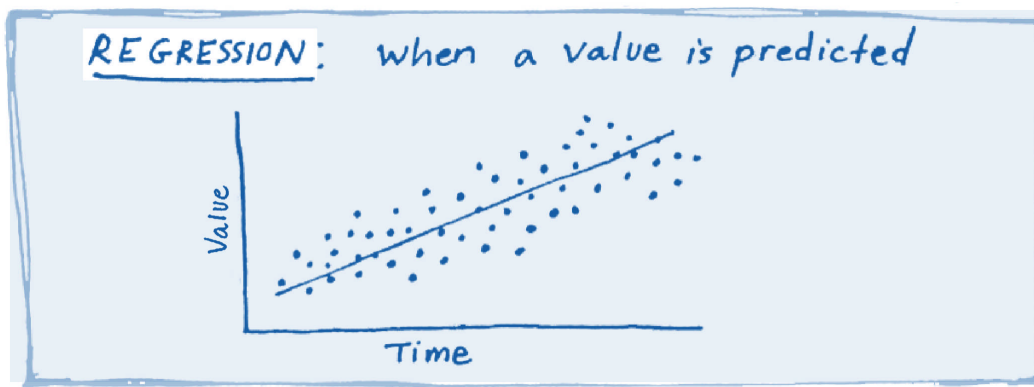
In this example, a bowl contains four types of fruit. A human labels the input data (size and color) and output data (shape). The algorithm uses the training data set to find clusters of similar shapes then a human tests these results by using a different set of data, the set that was held back for testing purposes. The human tunes the model for optimal performance, gathers new data and uses the model to make a prediction. A cherry is not a pear but two cherries whose stalks are still connected are recognized as cherries.

There are three common applications of supervised learning:

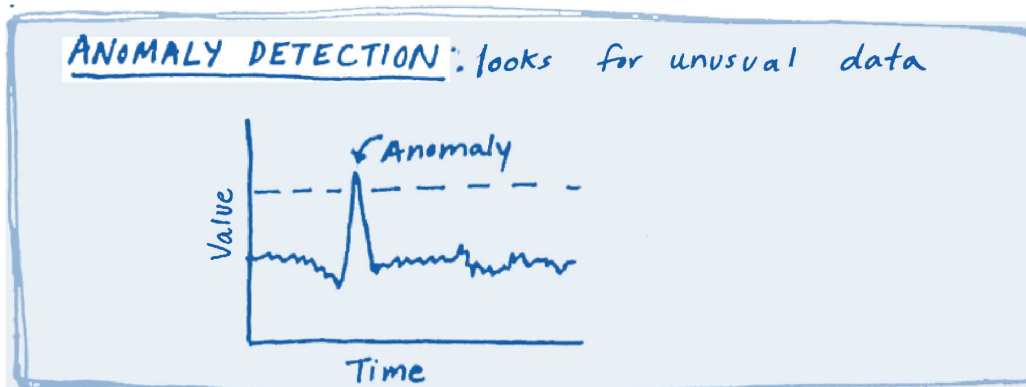
**Classification:** when data is used to predict a category, e.g. spam filters and image recognition such as distinguishing between “tulip” or “rose”



**Regression:** when a value is being predicted, e.g. a stock price



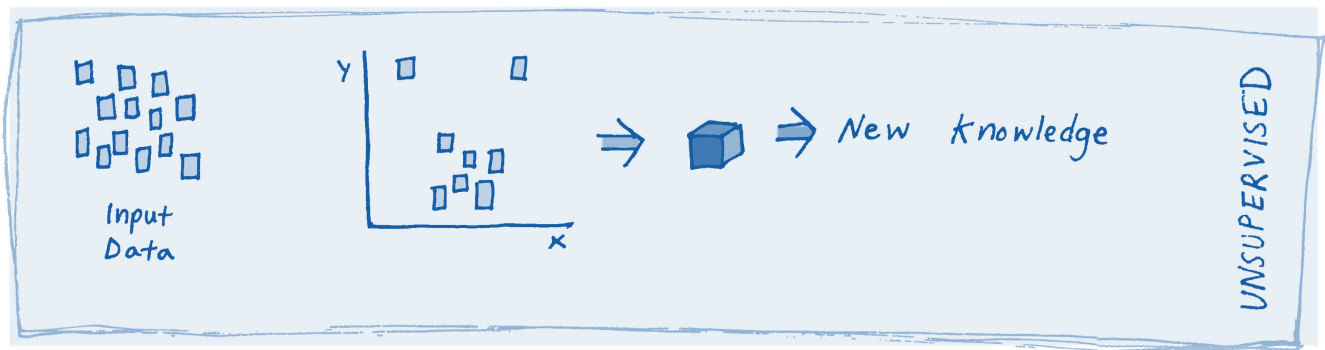
**Anomaly detection:** when looking for unusual data e.g. fraud detection



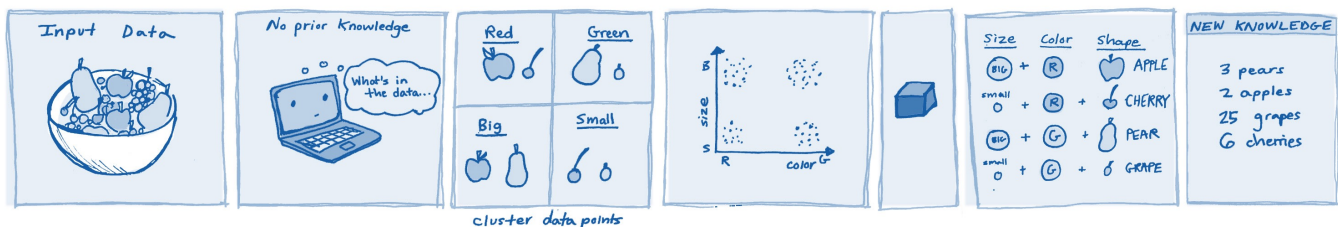
## UNSUPERVISED LEARNING

Unsupervised learning uses different learning approaches and different mathematical processes to manipulate the data and provide a result.

In unsupervised learning, data points have no labels and the goal instead is to organize the data by similarity and understand its structure. There isn't a known result or "correct answer" and a model is prepared by deducing structures in the data rather than testing against correct answers. An important goal of unsupervised learning is to get the machine to find data patterns that humans don't know about.



Let's go back to the bowl of fruit. In unsupervised learning, the machine discovers characteristics, such as size, color and shape. The machine clusters the data according to the characteristics it has found and, as output, is able to identify four different types of fruit as well as count how many in each class.

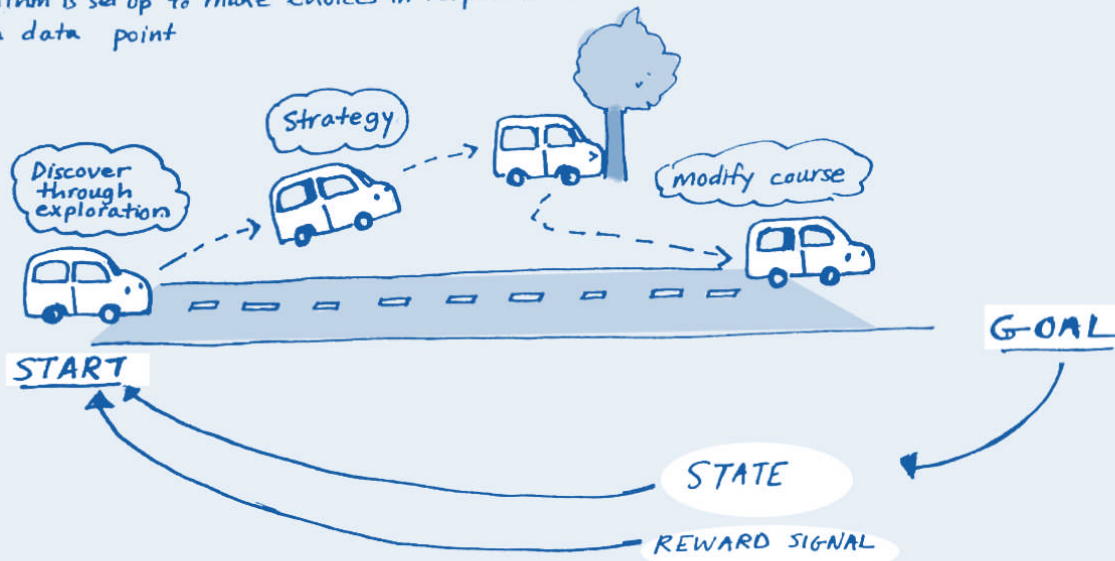


## REINFORCEMENT LEARNING

Reinforcement learning is used in situations where the optimal outcomes are not known or are more difficult to define and where there is less feedback. The feedback can be much later in the process and may not provide any information about the steps themselves. The steps have to be discovered through exploration – testing out a lot of different strategies and measuring performance against the long-term goal.

In reinforcement learning, the algorithm is set up to make choices in response to each data point. A short time later the algorithm receives information about the value of the decision it made (a “reward”) that it then uses to modify its strategy. Reinforcement learning is commonly used in robotics, including the software for autonomous or self-driving vehicles.

- optimal outcome unknown or difficult to define
- less feedback
- algorithm is setup to make choices in response to each data point



It isn't possible to define the optimal path for a vehicle to drive. There are too many roads and too many obstacles in the world to define the positive result in every case. But we can define that success is staying on the road and failure is going off the road. Using reinforcement learning, the car will get a reward when it stays on the road but will not when it goes off the road. The car incorporates the feedback loop given the circumstances and builds its own model for how to drive to reach the defined success of staying on the road.

# TEACHING A MACHINE TO LEARN

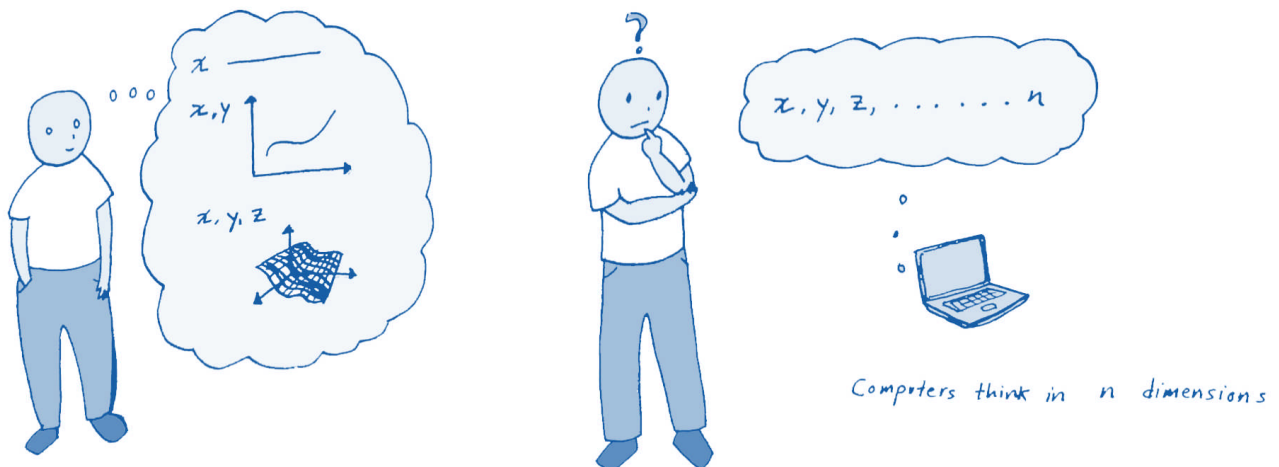
Human intuition and experience is powerful. We are adept at transferring knowledge and concepts from one domain to another. When comparing energy use with processing power, we are efficient thinkers. The fastest supercomputer in the world can process 93 quadrillion calculations per second while the equivalent measure for a human brain is 3.5 quadrillion calculations per second. But when it comes to energy, the supercomputer takes 15.3 million watts while our brains consume a tiny fraction of this, just 20 watts. When comparing connections, the human brain has around 90 billion cells, which are linked together with trillions of connections. When this was mimicked digitally, scientists needed more than 82,000 processors running on one of the world's fastest supercomputers to mirror just 1 second of a normal human's brain activity. That means that the fastest supercomputer in the world with 10.66 million processors can only mimic the brain activity of 130 people at a time. But we suffer from a host of cognitive biases; tendencies to think in certain ways that can lead to systematic, and repeatable, deviation from what the math would say. A couple of common examples:

**Anchoring effect:** the tendency to rely too heavily on one piece of information when making a decision. Usually this is the first piece of information. Retailers use this effect when an artificially inflated “original” price is marked down, making us feel like we get a deal.

**Availability heuristic:** the tendency to overestimate likelihood because memories are recent, unusual or emotionally charged. If you have a preoccupation with car accidents it's likely this will increase your perception of the likelihood of getting in an accident.

**Clustering illusion:** the tendency to overestimate the importance of streaks or clusters in large samples of random data and hence, seeing phantom patterns.

Many more mental shortcuts contribute to distortions in how we evaluate data. One important limitation is that we can't hold more than a few variables in our minds at any given time.



*“If people could see in high dimensions machine learning would not be necessary”*

- Pedro Domingos

Let's go back to our house price prediction example and conduct a thought experiment. Imagine your next-door neighbor sells her house for \$500,000. Her house has one bathroom and two bedrooms; your house has two bathrooms and four bedrooms and is double the size. What's your prediction of how much your house is worth, given this information? \$800,000, give or take? Do you find it's relatively easy to hold three dimensions in your head and at least come up with a range? But try adding more: your house is ten years older and a bit tired while hers is recently renovated, you have a lawn while her yard has contemporary minimalist landscaping, yours has a nice deck and a better view but hers has a lap pool and an outdoor fireplace. Do you find it's now not possible to keep track of all the variables, much less compute a new prediction?

Machine learning is the field of science where mathematical algorithms calculate the specific relationships between multiple variables. Many of these algorithms are based on statistical methods and all of them have some kind of iterative or automatic sorting method that allows the algorithm to converge to an optimal solution.

There are literally hundreds of different algorithms, variations on variations, and the field is changing fast as people find clever ways to combine different approaches. I have chosen to use a simple categorization developed by Pedro Domingos, one of the world's top machine learning researchers. In his book, *The Master Algorithm*, he categorizes algorithms into five classes.

**Inverse deduction** – where the algorithm iterates to fill in gaps in existing knowledge

**Neural networks and deep learning** – where a mathematical network mimics the human brain

**Evolutionary** – where an algorithm sorts and culls outcomes in a way that simulates DNA replication

**Bayesian** – where the algorithm uses a type of probability theory to solve for reducing uncertainty

**Analogizers** – where the algorithm reduces contrast between old and new sets of information

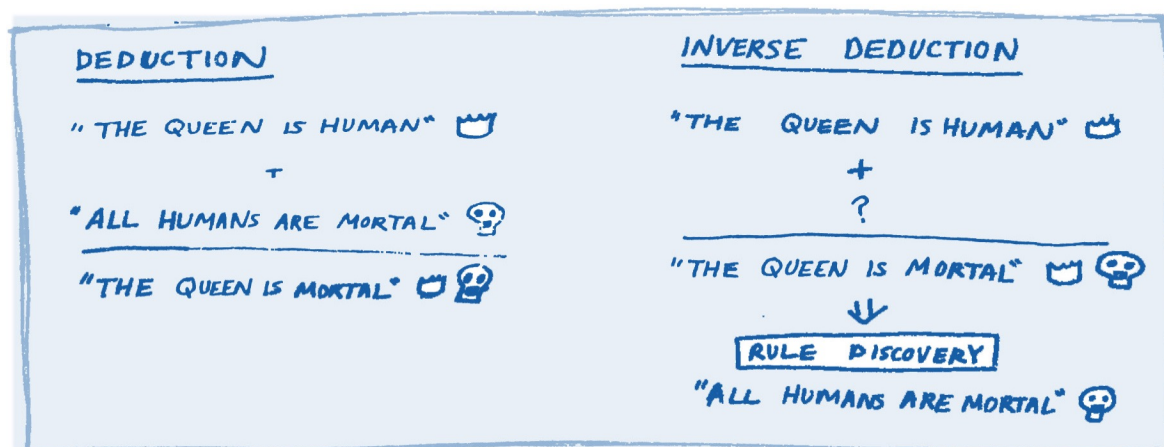
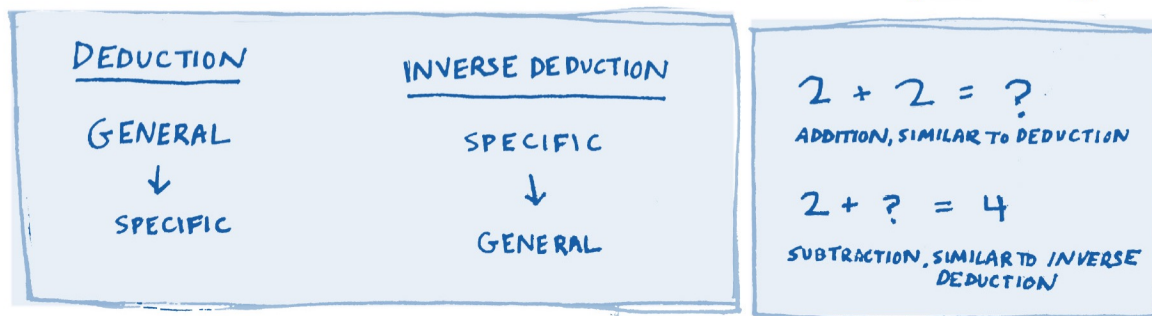
## INVERSE DEDUCTION

Inverse deduction essentially works backwards from conclusions or “learning by example.” An algorithm includes some known or presumed premises and asks the system “what knowledge is missing?”

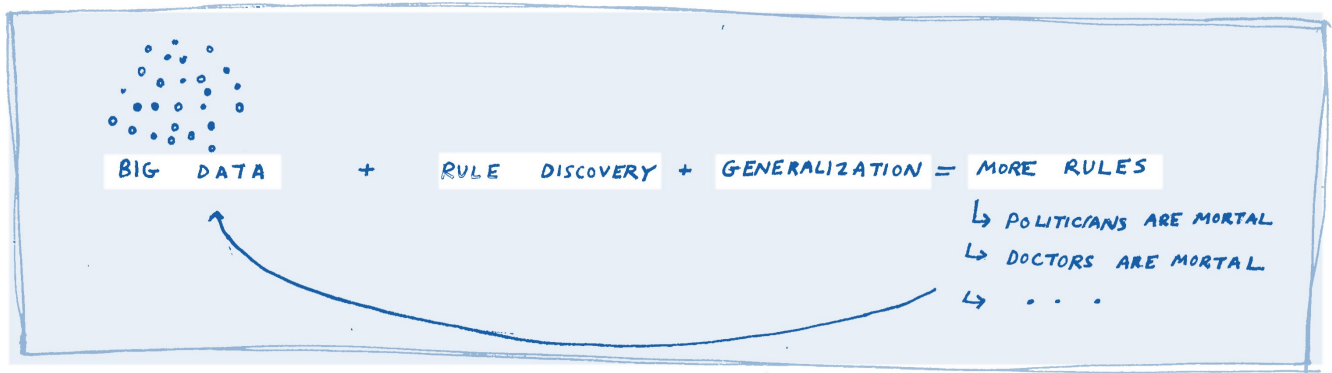
These algorithms can automatically construct rules to account for observations by attempting to find general patterns and then inferring a rule. Inputs are selected or identified that do the best job of dividing the dataset into similar parts. In many ways, inverse deduction mirrors the scientific method as specific observations are linked to a more general rule by way of testing and iterating against a

hypothesis.

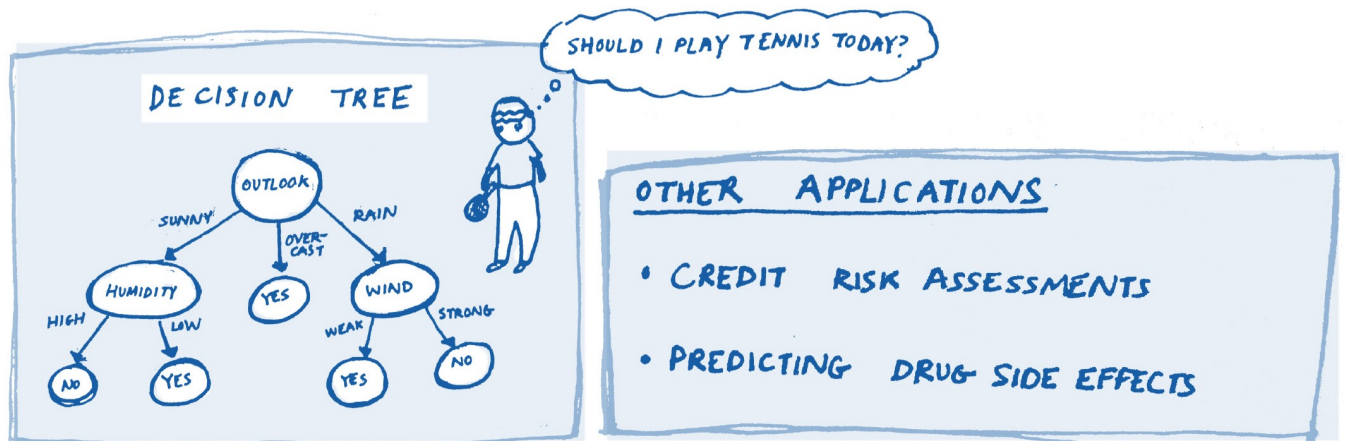
INVERSE DEDUCTION - fills in gaps in existing knowledge



In inverse deduction, very large data sets can yield a large amount of new knowledge by generalizing each rule across the data set. In many scientific areas, data is generated much faster than it can be analyzed. Machines are able to automatically generate hypotheses, which explain the observations, and then initiate experiments to generate more data, thus testing the hypothesis, and then repeat the cycle. This process has been used to automate the scientific process and has practical application in many areas. For example, in determining gene function and speeding up drug discovery by automating the "short list" of safe, effective molecular structures. This can cut years from the development timeline by eliminating many potential options from being produced then going through the process of animal, then human, testing.



Many artificial intelligence systems are variations of deduction. Decision trees are embedded in many common systems, such as how a bank decides to grant a loan or how a call is routed through a customer service center. Their biggest limitation is flexibility; rules are rules and there are too many to compute and apply using these methods.



## NEURAL NETWORKS AND DEEP LEARNING

**Neural networks** are inspired by the structure and function of the brain. There are many varieties of neural networks, utilizing different “neuron” designs, embedded statistical functions and computational tricks. While they were invented a long time ago, there has been remarkable progress in the past few years. Neural networks are the foundation for deep learning, currently the most active and diverse branch of machine learning development.

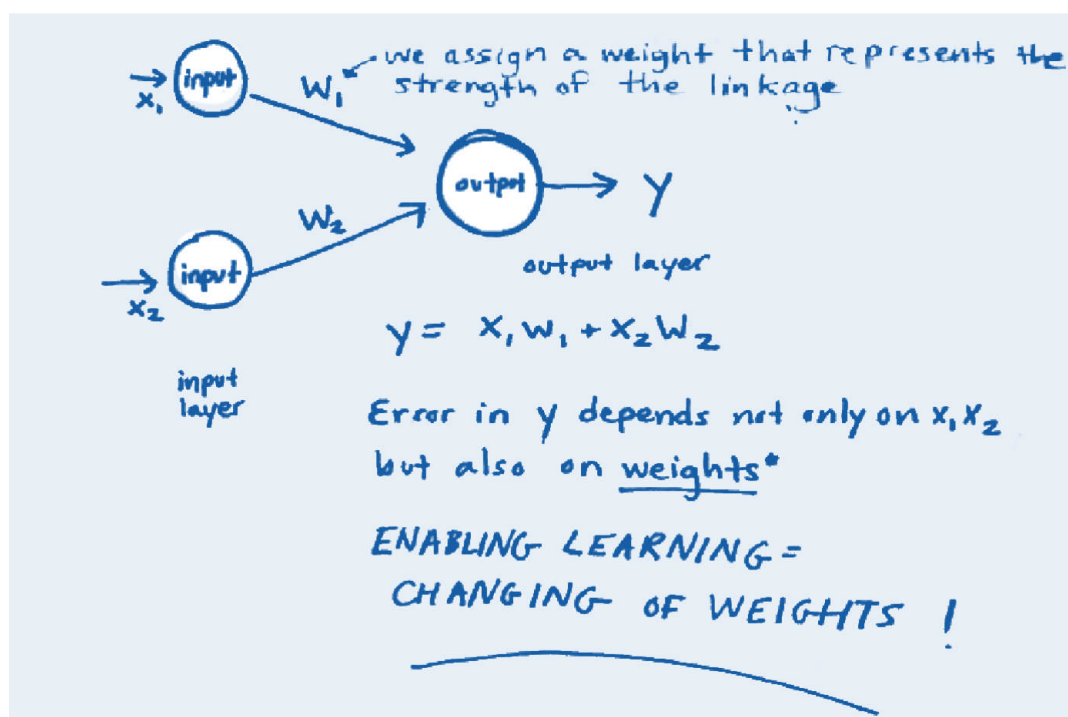
What are the components of a neural network?

*“What fires together, wires together.”*

A biological neuron is a cell that is electrically excitable and can transmit signals to other cells. The

signals it receives from other cells (via a synapse) may be excitatory or inhibitory. If the excitatory signal a neuron receives over a short period of time is large enough, the neuron generates a brief pulse, called an action potential, which then activates other synapses and other neurons. What happens in response to some stimulus, therefore, depends on the strength of the signals and the configuration of the network.

A mathematical neuron also receives inputs. Each input is assigned a weight. A higher weight is excitatory and a lower weight is inhibitory, relatively speaking. Each neuron can receive multiple inputs, each with a different weight. The strength of the connections is called “activation” and it is calculated as a weighted sum of the activations of all the neurons that feed into it. The activation function within the neuron takes the input and calculates the output activation. This output is used as input for other neurons further up the network.

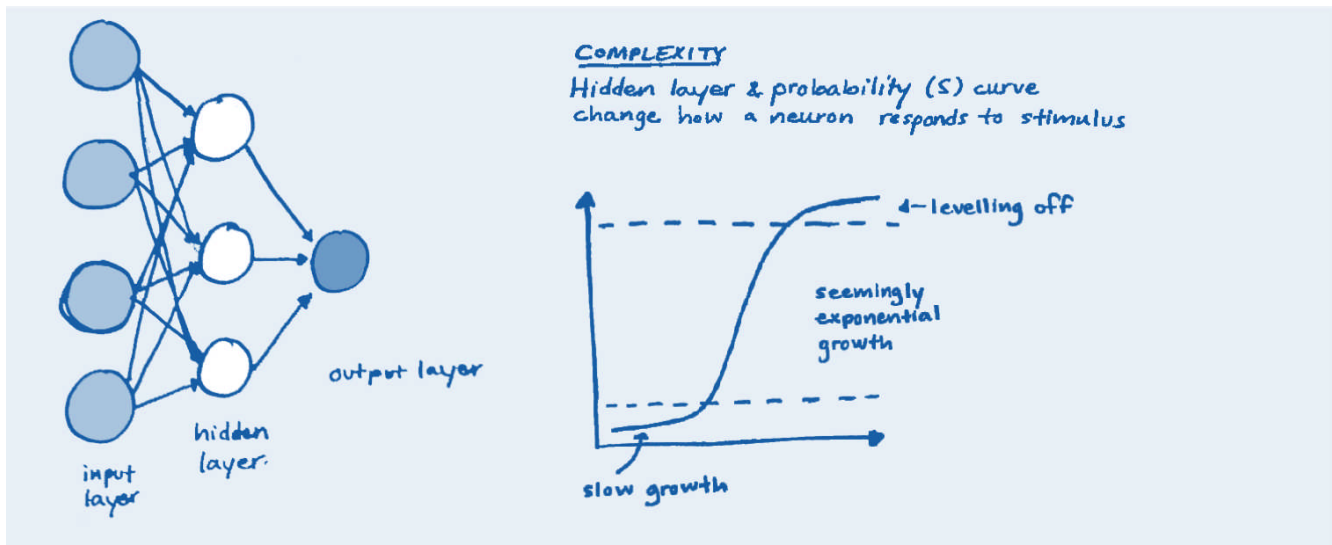


To visualize the inputs and outputs, imagine you want to mix a pot of purple paint from a pot of blue paint and a pot of red paint. How much of each color you choose are the weights and the difference between the desired shade and the actual output is the error. If the desired shade is redder, you will increase the weight of red, if the desired shade is more violet, you will increase the weight of blue.

Neurons themselves contain mathematical functions that can change the nature of the output. They can contain step functions with simple on/off behaviors or statistical functions that allow for more gradual change as the neuron responds to inputs. Imagine your finger over the end of a garden hose and how it's possible to adjust the spray from off to on, to a full-on squirt, as you adjust the pressure and angle of your finger.

Neurons are linked together in layers to form a network. There are no connections within the layers, only connections between the layers. Layers whose output is only used as an input to other neurons are called hidden layers. Activations in higher layers are calculated by inputs from lower layers, all the way up to the output layer. The simplest neural network is a feed forward network.

FEED FORWARD - output of lower layer influences the one above but connections can only run from lower to upper  
Network has no memory



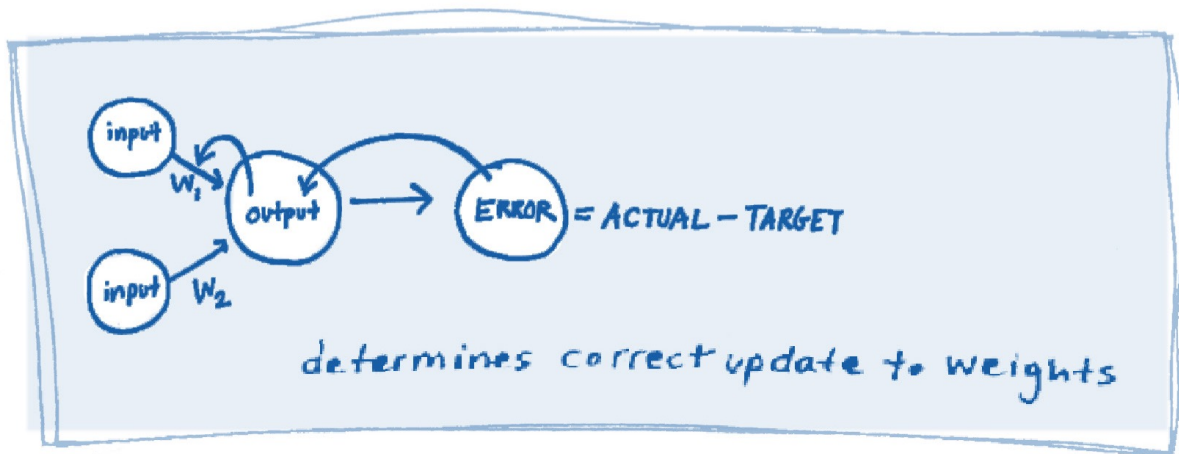
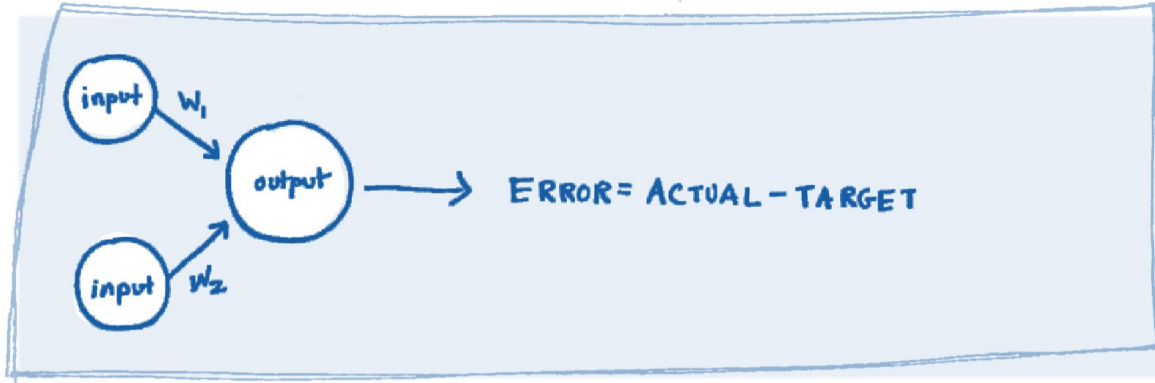
How does the network learn?

An important invention in the field of neural networks was backpropagation or “backprop.” Backprop is an abbreviation of “backward propagation of errors” and relies on some calculus to get there.

When neural networks are fed data, the output can generate an error measure. In neural networks this is in the form of a mathematical expression that can be used to adjust weights in the network by progressively seeking a lower error. This error is called a loss function and it is used to adjust the weights by going back through the network. At the end of the process, the network will have optimized all the weights and can correctly map inputs to outputs.

Backpropagation is an important technique as it means that the algorithm automatically alters model weights, thereby “learning” through self-modification and feedback.

## BACKPROPAGATION - backward propagation of errors



You may have noticed something...backprop needs a known output, something to solve for. As a generalization, backprop is used in supervised learning.

So how do neural networks learn in an unsupervised way?

Can't see the forest for the trees?  
Searching for needles in too many haystacks?  
T. M. I...

Sometimes there's just too much information. Sometimes reducing the amount of information, simplifying the data, can reveal more.

Let's say we are trying to predict whether someone has the flu from a checklist of three symptoms; cough, high temperature, aching joints. We assign the value "1" to "yes" and consider them sick when the person has at least two of these.

But we also want to know other things about the patient that might mean they are less likely to be sick, for example, that they had a flu shot or that they have taken Vitamin D pills or that they recently

started doing extreme crossfit (which would explain the aches and pains). There we label the counter symptoms in the same way – “1” for “yes,” “0” for “no.” We consider a patient to be healthy when they have at least two of these.

Now let’s put this idea into a neural network. There is one input layer and one output layer, each with the 6 data features; cough, high temperature, aching joints, flu shot, Vitamin D, extreme crossfit. If we only have two neurons in the hidden layer, the hidden layer can only feed forward “sick” or “healthy.” Over successive iterations, the two hidden units will be forced to have different sensitivities (activations) to the inputs. The end result is that we have created a network that has learnt a compressed representation of the data while making accurate predictions about whether someone is sick or healthy based on their symptoms and history.

## FLU SYMPTOMS

6 binary input features [a, b, c, d, e, f]

first 3 refer to symptoms of the flu

last 3 refer to counter symptoms

for example

100000 refers to a patient with a high temperature 🌡️  
 010000 refers to a patient with a cough 🤧  
 110000 refers to a patient with a high temperature AND cough 🌡️🤧

000100 refers to a patient who got a flu shot 💉  
 000010 refers to a patient who takes vitamin D pills 🍬  
 000110 refers to a patient who got a flu shot AND takes vitamin D pills 💉🍬

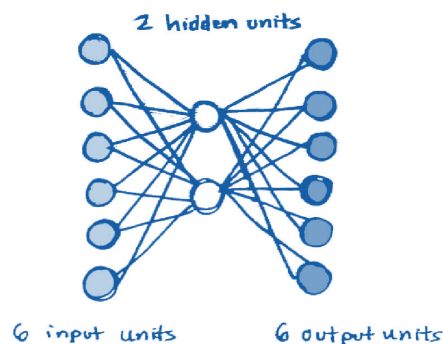
a patient can have symptoms and counter symptoms

010100 refers to a patient with a cough who also got a flu shot

BUT a patient is considered sick when s/he has at least two of the first three features and healthy if s/he has at least two of the second three

11000, 101000, 110000, 011000, 011100 - has the flu 😞

000111, 001110, 000101, 000011, 000110 - is healthy 😊



- hidden units refer to either the label "has the flu" or "is healthy"
- one unit will exhibit a higher activation rate for "has the flu," the other will exhibit a higher activation rate for "is healthy"
- a compact representation of the flu data set

In machine learning there are many tricks that are used to get a computer to discover a pattern or a relationship that a human can't see. Many techniques aim to break down something complex into the right little pieces to find the meaning. But there are other tricks, where hidden rules are found by

forcing the computer to compress information, break it up and discard parts of it, then forcing the computer to reconstruct the original form.

When we do this inside a neural network, the resulting structure contains autoencoders - an unsupervised neural network that works backward – deconstructing an output and then reconstructing the right inputs. By forcing the machine to deconstruct information and then reconstruct it, the network finds correlations in the simplified structure. The advantage of doing this isn't just about processing power or memory, it's fundamental to learning: the network figures out how to compress things on its own, making a clean image out of a noisy and distorted one.

AUTOENCODER an unsupervised neural network used for learning efficient codings  
(output layer has same number of nodes as input layer)  
not trained to predict target value  $Y$  given inputs  $X$ , but to  
reconstruct their own inputs  $x$

## A Recap...

We now know how the output of a neuron is affected by the weights of the inputs,  
And, we know how the output of neurons can feed forward and be the inputs for higher-level neurons,  
And, we know how the output of a neuron can be tested, evaluated, and corrected by backpropagation,  
And, we know that hidden layers can learn the data at a conceptual level, capturing core features of the data that we don't see.  
So, what happens if you stack more and more of these layers?  
That's called deep learning.

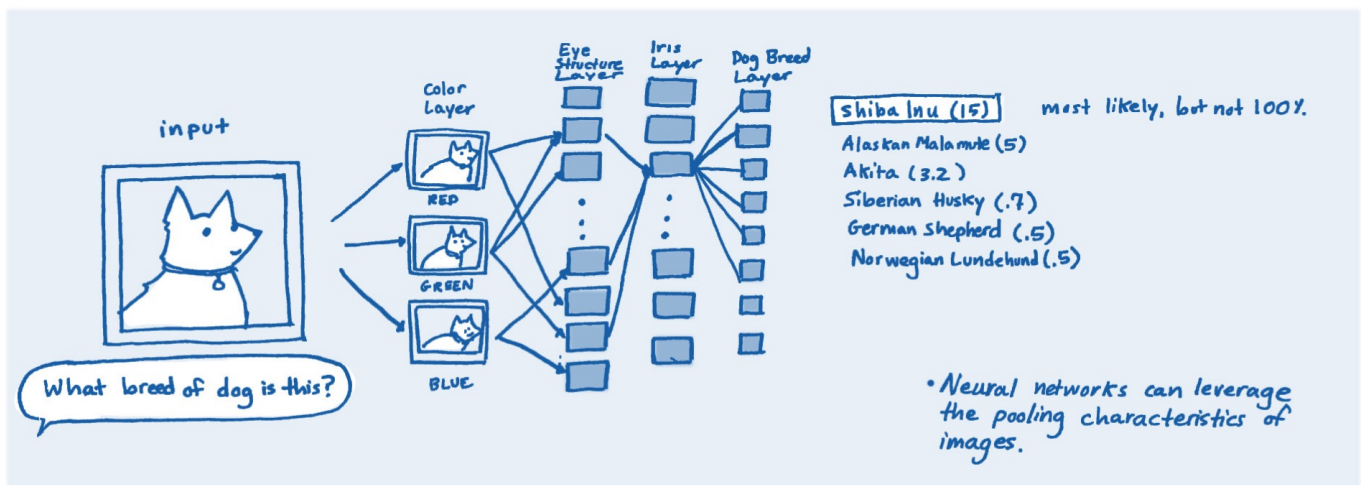
**Deep learning** is a recent update to neural networks. In deep learning, many more layers of neurons are connected up and each layer can be trained in turn. Each hidden layer gets more and more sophisticated and can encode more complex or abstract representations. For example, if a network receives a large set of facial images, the first autoencoder learns to encode small features in one local area of the image, such as a corner or a line or a dot. The next takes this as input and learns the next level feature such as the edge of an eye. Finally, the top layer can put this all together and recognize the image as a picture of The Queen, say.

## DEEP LEARNING

self-driving cars, image recognition, speech recognition  
ex: Google mapped every location in France in 2 hours using street view images and a deep learning algorithm that learned to recognize and read street numbers.



Deep learning can be supervised or unsupervised. In supervised learning, the network is trained and the network learns by adjusting weights and honing in on the correct answer. In unsupervised learning, the model takes vast amounts of rich input data and uses all the tricks it has to encode for what it sees. While the biggest performance gains have been made with supervised learning, the ability that deep learning has to discover new features for itself is at the forefront of artificial intelligence today.



neural network activated by pixels of an image > neurons are weighted and transformed by a function  $f(x)$  > activations of these neurons are passed onto other neurons (layers) > output neuron that determines breed is activated

## HOW DO MACHINES SEE?

We can transform the scope of what we can do with machines if we have machines that can process visual information. Just think of the vast store of visual information that we now have access to online and what more can be created; medical images, images from space, any number of images of people, places, things. Machines are faster, cheaper and, in many cases, more accurate than humans at processing and classifying images.

Deep learning has revolutionized this field, primarily because it is very good at discovering complicated structures in high-dimensional data. Images contain many features with subtle yet important differences that have been impossible to code in the traditional way.

One way machines learn to see is with a type of neural network called a convolutional neural network (CNNs or ConvNets). A convolutional neural network is a type of feed forward neural network inspired by the structure of the visual cortex. These networks are like huge filter banks, processing input information through progressively more complex filters and layers.

**Convolutional neural networks** have important design elements that make them particularly useful in computer vision. They uniquely have:

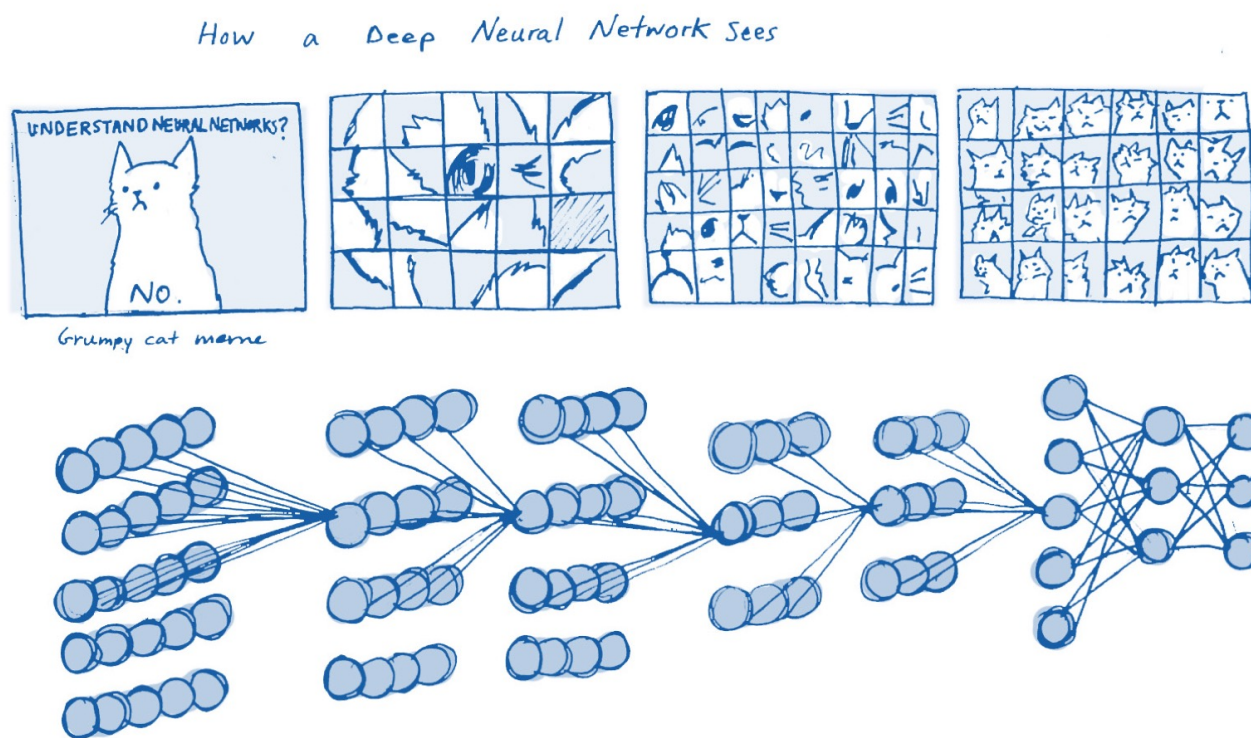
*Local connections*, which enable them to easily detect highly correlated values and form distinctive local motifs such as the edge of an eye.

*Shared weights*, so that a pattern can be detected more easily once it's been detected in another part of the array.

*Pooling layers*, which can mathematically simplify things thereby reducing the number of dimensions and helping to make the network less sensitive to small distortions.

*Convolution steps*, a sub-sampling process that increases the efficiency of the network.

Any image is fed into the network as an array of pixel values. In the first layer, the network will learn the lowest order features, typically the presence or absence of edges. Then the second layer may put these edges, and their orientations and locations, into a higher order combination to correspond to parts of the objects. The network keeps “feeding forward” so the subsequent layers build the representation of the objects in the image to a higher-order, more abstract level: a color boundary to the edge of an eye to an eye.



The big breakthrough in convolutional neural networks came in 2012 when, with the availability of powerful processing chips, they were applied to a much bigger data set than had been used before, about a million images. The result was astounding: a halving of the error rate of any previous approach. They are now the dominant approach for recognition and detection. Convolutional neural networks are huge; they can contain 10 to 20 layers, hundreds of millions of weights and billions of connections.

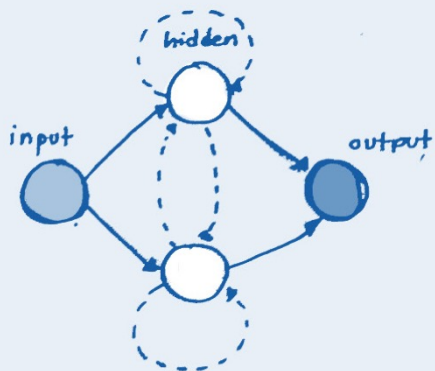
### **A Recap...**

We now know how the output of a neuron is affected by the weights of the inputs,  
And, we know how the output of neurons can feed forward and be the inputs for higher-level neurons,  
And, we know how the output of a neuron can be tested, evaluated, and corrected by backpropagation,  
And, we know that hidden layers can learn the data at a conceptual level, capturing core features of the data that we don't see,  
And, we know how we can stack many layers together and create deep learning networks, which can handle complex tasks like helping a machine see.  
So, what happens if the machine can remember the output from multiple iterations and use this knowledge to do better?  
That's called Recurrent Neural Networks.

**Recurrent Neural Networks** (RNNs) can take multiple inputs and deliver multiple outputs rather than simply a single classification. They are able to operate with sequences, where the neuron "remembers" its previous activation, a kind of artificial short-term memory, as well as able to then receive a fresh input.

In other types of neural networks, the inputs and outputs are fixed, that is, put in an input and get out a set of probabilities of different classes. For example, input a million photos of brown birds and get an output of the probabilities of different breeds of brown bird. Recurrent neural networks can handle sequences of inputs and outputs, which make them very flexible and adaptive. Instead of being constrained to fixed inputs and outputs, the networks can learn to provide a different output or refine down the input. Clever!

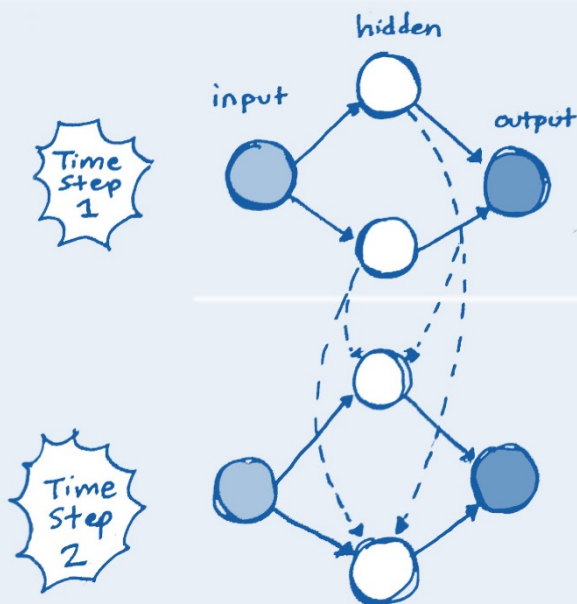
Remember how I said there were no connections between the neurons in the same layer in a neural network? Recurrent neural networks create a kind of layer by adding a loop in time. This is clever way of giving a network "memory," where it remembers the previous iteration and uses that to update its output.



### Recurrent neural networks

add another set of connections between artificial neurons

Activations feed back into themselves  
at the next step in the sequence



• at every step, the hidden layer receives activation from the layer below it and its own activation from the previous step in the sequence

• added connections link one time step with the next  
seen more clearly when network is unfolded in time

In practical terms, this makes them able to do very intelligent things such as automatically generate captions to an image and translate language in real time.

Where is deep learning heading?

There are three areas of deep learning development that are particularly exciting.

**Sparse data.** Deep learning breakthroughs happened because of access to massive data sets. It turns out that bigger has been better for neural networks. But this is a “one size fits all” approach and may not always be the most efficient. New approaches are being developed where deep learning can work when there is less data.

**Generative Adversarial Networks.** This idea hinges on the principle “if you can’t create it, you don’t understand it.” In this field, the idea is to simultaneously train two neural networks with one network

being trained on real input while another network tries to create similar images for the first one to train on. It's like a machine learning arm wrestle where the network "creating" the image is trying to better the network "learning" the images.

**Hardware.** All of this processing is expensive, in computational terms. A renewed focus on energy efficient computation and optimizing chip design for artificial intelligence means deep learning is now being embedded in silicon. Google's TPU (Tensor Processing Unit) is specified for deep learning and is tailored for Google's deep learning product, TensorFlow. Google cites that this is roughly equivalent to fast-forwarding the technology seven years into the future or three generations of Moore's Law.

### A Final Recap...

We now know how the output of a neuron is affected by the weights of the inputs,  
And, we know how the output of neurons can feed forward and be the inputs for higher-level neurons,  
And, we know how the output of a neuron can be tested, evaluated, and corrected by backpropagation,  
And, we know that hidden layers can learn the data at a conceptual level, capturing core features of the data that we don't see,  
And, we know how we can stack many layers together and create deep learning networks, which can handle complex tasks like helping a machine see.  
And, we know that when machines can remember the output from multiple iterations and use this knowledge to do better, we can build some very intelligent and human-like interfaces that have the capacity to improve themselves without human intervention.

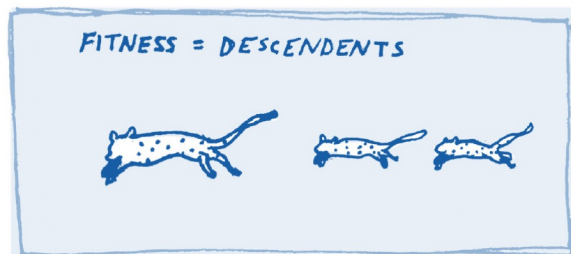
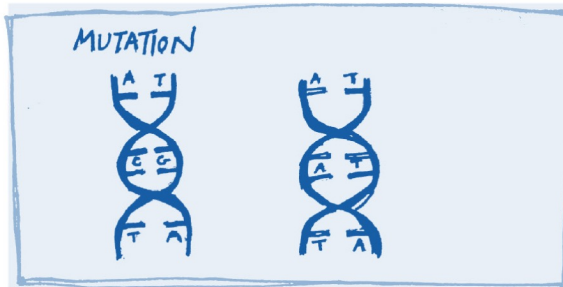
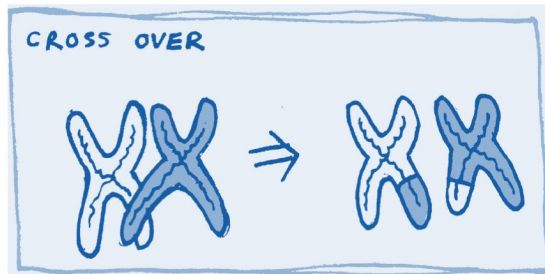
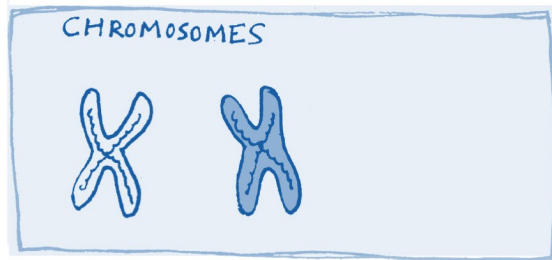
## EVOLUTIONARY AND GENETIC ALGORITHMS

Evolutionary algorithms mimic the process of DNA replication in a computerized "survival of the fittest." The algorithms improve by testing against an outcome, making these algorithms a form of reinforcement learning, combining trial and error with:

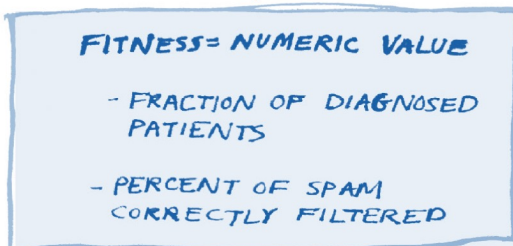
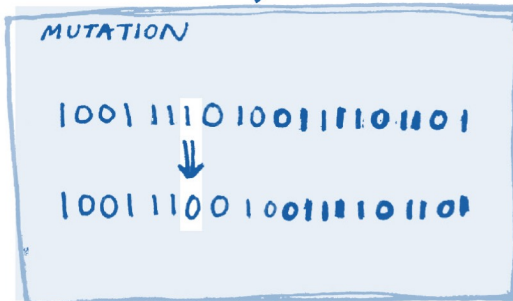
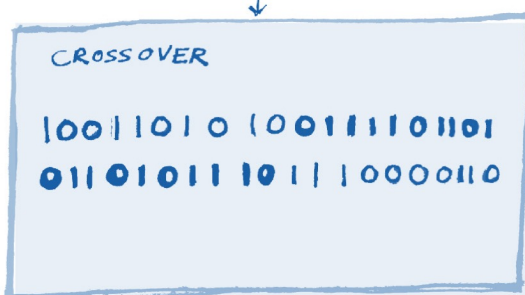
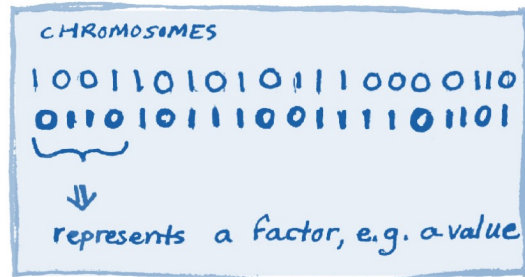
*Selection* - where better outcomes thrive,  
*Crossover* - where parts of these good outcomes are combined,  
*Mutation* - where occasionally random new combinations occur.

Evolutionary algorithms optimize against a fitness function that is usually designed by a human, for example the percentage of spam that is correctly filtered.

## BIOLOGY



## AI



As they can find truly novel solutions, evolutionary algorithms tend to work well when there is no model for finding the right answer. For instance, since there is no way to program a robot to walk across snow, the robot can experiment using a genetic algorithm and find the “fittest” method to walk without falling. One disadvantage of evolutionary algorithms is that they can take too much computation to converge – get to a solution – which is expensive both in terms of time and money. And just like the robot, they can get lost in never-ending iterations, stumbling around in a hyperspace of variation.

Let’s go back to our house price example and consider a key advantage of evolutionary algorithms, discovery of a novel solution from many variables. So far, we know that we can hold a few variables in our head and we know that traditional statistics is a tried-and-true way of making a price prediction

across multiple variables based on techniques like regression. We also know that we can build a learning algorithm that can automatically update the regression analysis in response to new data. But what about if we want to find an equation to represent much more complex relationships, particularly those relationships where time is a factor? Time series data, where data is represented in time order, can be difficult to model. Delay effects, overlapping time series data, irregular intervals between variables, and competing effects are all hard to unpick with linear modeling. Evolutionary algorithms are able to handle this type of data and could be used to find a relationship as complex as:

$$\begin{aligned} \text{Our house price prediction} = & (\text{Recent house sale prices within one mile})^3 \\ & / (\text{Recent house sale prices within 100 miles})^3 \\ & + 2.55 \times (\text{Marketing done three weeks ago}) \\ & + 3 \times (\text{Number of color photos in flyer})^2 \\ & + 0.0992 \times (\text{Hits on website within last two weeks}) \\ & + \log_2(\text{Frequency of open house}) \\ & \times 1/(1+(\text{Rain on the day of the open house})^2). \end{aligned}$$

Clearly this is a fictional example with made up variables, but evolutionary algorithms have discovered these types of complex, non-linear formulas. They are being used to do everything from logistics management in retail operations to optimization in large industrial processes.

Recent breakthroughs in performance have revived interest in evolutionary algorithms. Convergence problems have been solved by use of a machine learning version of “Occam’s Razor” (the principle stating, among competing hypotheses, the one with the fewest assumptions should be selected), where solutions are strongly biased towards the simplest outcome and more complex outcomes are “killed off” early in the computation. Efficiency has been improved with the use of parallel computing, algorithms are now used at very large scale, processing variables that have traditionally been difficult to use, such as time series data. They are also used effectively in hybrid approaches where evolutionary algorithms are used to find to initial weights to seed a related neural network.

## BAYESIAN ALGORITHMS

Most of the analytical processes we've talked about so far use values as inputs, for instance, the color of a pixel, the price of a house, or the body temperature of a patient. What if the input is a probability instead of value? How do we use an input like "10% of happy people are rich" in an algorithm? For problems like this we can use Bayesian algorithms.

We normally think of probability in terms of frequency or propensity. For example, the probability of getting heads when flipping a coin is  $1/2$  and the probability of getting a 1 when rolling a dice is  $1/6$ . Frequency works well for completely random situations like coin tosses but less so for situations where there is some sort of pattern. In these situations, the probability of an event is based on the things that might be related to that event.

Imagine you live in a place where storms arrive in fronts. One minute you're sitting on the porch enjoying the view of a perfect blue sky, then you notice a band of dark clouds. Next minute, you're in a howling gale, fighting horizontal rain and running to the washing line to salvage the laundry. You're used to this pattern and you know that when you see a mass of cloud in a low-lying line, the chances of rain are close to 100%. Now imagine you take a vacation in San Francisco in the summer time. You arrive at SFO mid-afternoon to cornflower blue sky and a comfortable 75 degrees. As your Uber approaches downtown, you glance west and see a band of grey gloom, ominously cloaking the towers of Twin Peaks. When the Uber driver drops you at your hotel, it's now 62 degrees with a rather chilly wind and it feels like 56!

"Looks like rain," you say. "Ninety five percent chance would be my guess." Your Uber driver simply shrugs.

"You know what they say, the coldest winter I ever had was summer in San Francisco."



No rain appears but the next day, there's the same pattern. This time you get to experience San Francisco's famous fog for yourself as you hastily walk across the Golden Gate Bridge in the biting, damp wind with many other hapless, cold tourists. This time your guess at the probability of rain is 50/50. By the end of your week long vacation, you've come to expect the fog and know that there's pretty much no chance of rain. You started out the week with a probability of rain of 95%, adjusted this to 50% and settled on something like 10%.

So it is with Bayes'. We refer to the "before" probability as the prior probability, and the "after" probability as the posterior probability.

**BAYES THEOREM** : Probability is described in terms of conditions that relate to the event, NOT in terms of the frequency of an event

probability - the extent to which an event is likely to occur, measured by the ratio of favorable cases to the whole number of cases possible

FAVORABLE CASES  
WHOLE NUMBER OF  
CASES POSSIBLE

roll a 5	 $\frac{1}{6} = 16.67\%$
flip tails	 $\frac{1}{2} = 50\%$

EVENT(S)  
# OF OUTCOMES

\* Works well for completely random situations, but not for situations where there is some pattern (like the weather)

→ In Bayesian interpretation, probability is an abstract value that we assign a hypothesis, our "degree of belief" in the truth of the hypothesis in question

HYPOTHESIS: it will rain tomorrow 

higher the probability → the more you think you'll need your umbrella

adjust the value based on evidence you acquire about the situation

INITIAL VALUE = PRIOR PROBABILITY & use evidence to arrive at

NEW VALUE FOR PROBABILITY = POSTERIOR PROBABILITY ← hopefully closer to the truth than prior

Bayesian statistics treats the probability of belief (in some knowledge) as a starting point. This prior probability represents how much a model is believed to be true before any data is actually observed. After the data is gathered, another probability distribution (the posterior probability) is generated. So, we start with an initial value (the prior probability) and then adjust the value as data is generated (the posterior probability). As iterations progress, the hypothesis is progressively updated and the uncertainty is reduced.

Before we discuss Bayesian algorithms we have to get to grips with Bayes' Theorem, a way to describe the probability of an event based on things that might be related to the event. Bayesian thinking isn't necessarily intuitive and it's best explained with an example.

Your friend Max receives a positive test for cancer and wants to know what the chance is of him actually having cancer. He knows that the test is not the event, that is, having a positive test is not the same as having cancer. He also realizes that tests are not perfect, that they can detect cancer when it's not there (false positive) and fail to detect cancer when it is there (false negative). After receiving a positive cancer test result, the likelihood of having cancer isn't just the accuracy of the test, it's the chance of a *true positive* result divided by the chance of *any positive* result. What is the chance that Max has cancer, given that 1% of people have cancer, 80% of tests detect cancer when it is there and 9.6% of tests detect cancer when it is not there?



\*ALREADY HAVE →      ← \*DON'T HAVE

	CANCER 1%	NO CANCER 99%
TEST +	80%	9.6%
TEST -	20%	90.4%

1% of people have cancer

\* ALREADY HAVE you are in the first column

- there's an 80% chance you'll test positive
- there's a 20% chance you'll test negative

\* DON'T HAVE you are in the second column

- there's a 9.6% chance you'll test positive
- there's a 90.4% chance you'll test negative



- How accurate is the test?
- What are the chances you have cancer?

POSITIVE TEST you are in the top row of the table

but don't assume anything

- could be a true positive
- could be a false positive

My test result is positive, but what are the odds I have cancer?



$$\text{CHANCE OF A TRUE POSITIVE} = \frac{\text{CHANCE YOU HAVE CANCER}}{\text{CHANCE THE TEST CAUGHT IT}} = \frac{1\%}{80\%} = 0.008$$

$$\text{CHANCE OF A FALSE POSITIVE} = \frac{\text{CHANCE YOU DIDN'T HAVE CANCER}}{\text{CHANCE THE TEST CAUGHT IT ANYWAY}} = \frac{99\%}{9.6\%} = 0.09509$$

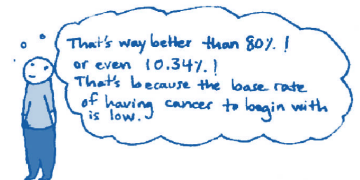
	CANCER 1%	NOT CANCER 99%
TEST +	TRUE POSITIVE $1\% \times 80\% = 0.008$	FALSE POSITIVE $99\% \times 9.6\% = 0.09509$
TEST -	FALSE NEGATIVE $1\% \times 20\% = 0.0002$	TRUE NEGATIVE $99\% \times 90.4\% = 0.88601$

THE CHANCE OF GETTING ANY POSITIVE RESULT (TRUE OR FALSE) =

$$0.008 + 0.09509 = .10304$$

$$\text{CHANCE OF HAVING CANCER} = \frac{\text{CHANCE OF A TRUE POSITIVE RESULT}}{\text{CHANCE OF ANY POSITIVE RESULT}}$$

$$= \frac{0.008}{.10304} = 0.0776 \approx \underline{7.8\%}$$



Interesting! If Max had only taken into account the test accuracy, he would have assumed that there was an 80% likelihood that he has cancer, when in fact his chances are only 7.8%. The base rate of having cancer was low to begin with and the rate of false positives is so high that there will be a whole lot of false positives in any given population.

Bayes' Theorem is the mathematical expression of this idea:

*The probability of the model given the data is the probability of the data given the model, times the prior probability of the model, divided by the probability of the data.*

So, if we're going to look at the cancer example in a mathematical formula...

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

where A and B are events and  $P(B) \neq 0$

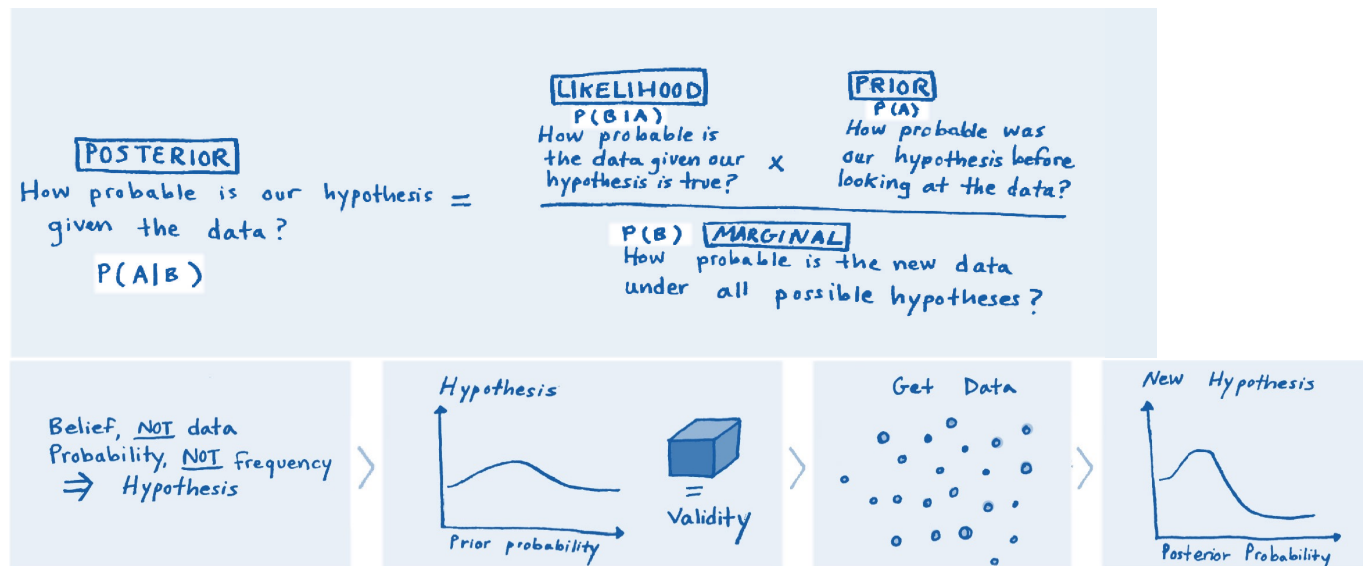
$P(A)$  and  $P(B)$  are the probabilities of observing A and B without regard to each other.

$P(A|B)$  a conditional probability, is the probability of observing event A given that B is true.

$P(B|A)$  the probability of observing event B given A is true.

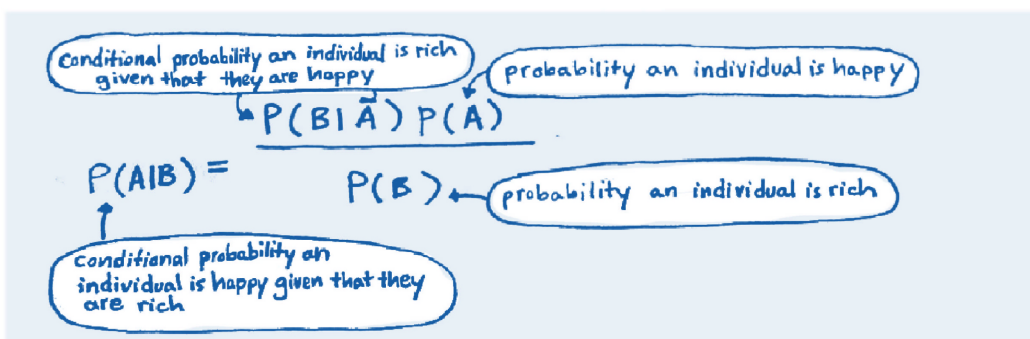
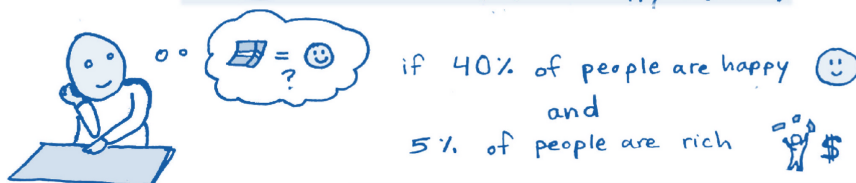
$$\begin{aligned} P(A|B) &= \frac{P(B|A)P(A)}{P(B)} \\ &= \frac{80\% \times 1\%}{10.34\%} = \frac{0.008}{.10304} = 0.0776 \approx \underline{7.8\%} \end{aligned}$$

Or more generally...



Here's another example...

You've seen a study "only 10% of happy people are rich"  $P(B|A)$ ,  
but you want to know what percent of rich people are happy  $P(A|B)$



$$\begin{aligned}
 P(\$|\text{smiley}) &= \frac{P(\$|\text{smiley}) P(\text{smiley})}{P(\$)} \\
 &= \frac{10\% \cdot 40\%}{5\%} = 80\%
 \end{aligned}$$



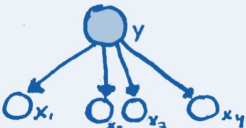
From theory to practice...how does this idea get converted into learning algorithms?

Bayesian algorithms use probabilistic inference, a statistical procedure to estimate parameters of an underlying or hidden distribution based on a prior distribution. So while, in a neural network we apply weights to inputs, in a Bayesian algorithm, we assign probabilities instead. Going back to the example of mixing red and blue paints, a neural network would apply weights to each color creating some shade of purple while a Bayesian algorithm would apply a probability of *either* red or blue. “Learning” is the process of continually updating the probabilities of the inputs based on the new output, which in turn updates the probability of the output.

One application in machine learning is Naïve Bayes algorithms...

Naïve Bayes algorithms take this idea and apply it in situations where there are multiple classes and we need a quick and easy way to build a predictive model based on the individual probability of these classes. It’s called “Naïve” because the model makes the assumption that a particular feature in a class is not related to any other features. For example, a fruit might be considered an apple if it is red, round and the size of a baseball. Even if these features depend on each other, or other features, all of them independently contribute to the probability that this fruit is an apple.


Naive Bayes



All input variables are independent

(ex: a fruit may be considered an apple if it is red, round, and  $\cong$  10cm in diameter.)

A naive Bayes classifier considers how each of these features contribute independently to the probability the fruit is an apple, regardless of possible correlation between color, roundness, & diameter.

 = PROBABILITY MODEL  $\Rightarrow$  NEW PREDICTIONS

Naïve Bayes algorithms will calculate the posterior probability for each class in the data set. The class with the highest posterior probability becomes the outcome of the prediction. For example, if you want to make a prediction about whether your daughter’s soccer match will be cancelled given rain, a Naïve Bayes approach involves calculating the probabilities of individual combinations, such as the

probability of it being overcast, raining or sunny, followed by calculating individual posterior probabilities for the probability of cancellation given rain.

Naïve Bayes is used a lot in text classification and spam filtering where it works well because of the assumption of independence between the features. It is also popular in social media sentiment analysis where it can efficiently identify positive and negative customer sentiment.

Another way to use Bayes Theorem is in Bayesian networks...

Bayesian networks are a union of graph theory (mathematical structures used to model pairwise relations between objects) and probability theory. They work well when both uncertainty and complexity occur at the same time. In this partnership, graph theory brings modularity, where a complex system can be built up by combining simpler parts, and probability theory acts as the glue, ensuring that the system as a whole is consistent. Probability also dictates the interface, providing ways for the data to reach the models. They are used for many applications, especially where there's a need to model highly interacting sets of variables.

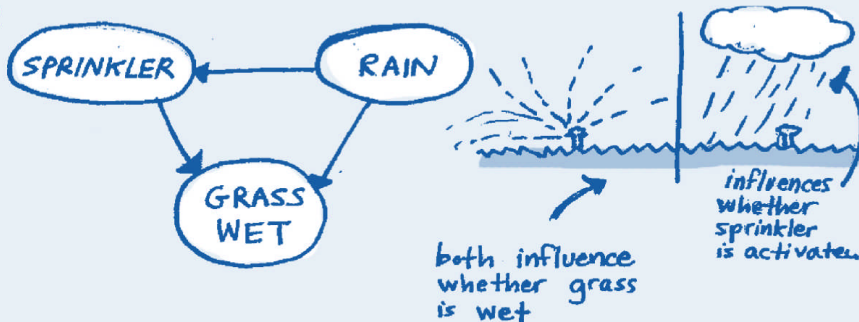
The most common problem we want to solve using Bayesian networks is probabilistic inference. For example, take a water sprinkler network and suppose we can see that the grass is wet. There are two possible causes for this: either it is raining, or the sprinkler is on. We can use Bayes' rule to compute the posterior probability of each explanation and predict whether it's more likely that the grass is wet because of the sprinkler or because of rain.

## Bayesian Networks

a probabilistic graphical model that represents random variables and their conditional dependencies via a directed acyclic graph (DAG)

Ex1: could represent probabilistic relationship between diseases and symptoms. Given symptoms, network can be used to compute probabilities of the presences of various diseases

EX2:



RANDOM VARIABLES + CONDITIONAL DEPENDENCIES

$$\text{Cube} = P(G, S, R) = P(G | S, R) P(S | R) P(R)$$

$G$  = Grass wet ( $T$ ,  $F$ ) (yes, no)

$S$  = Sprinkler turned on ( $T$ ,  $F$ ) (yes, no)

$R$  = Raining ( $T$ ,  $F$ ) (yes, no)

What is the probability it is raining given that the grass is wet?

$$P(R=T | G=T) = \frac{P(G=T, R=T)}{P(G=T)}$$

## Bayesian Algorithms

are used in :

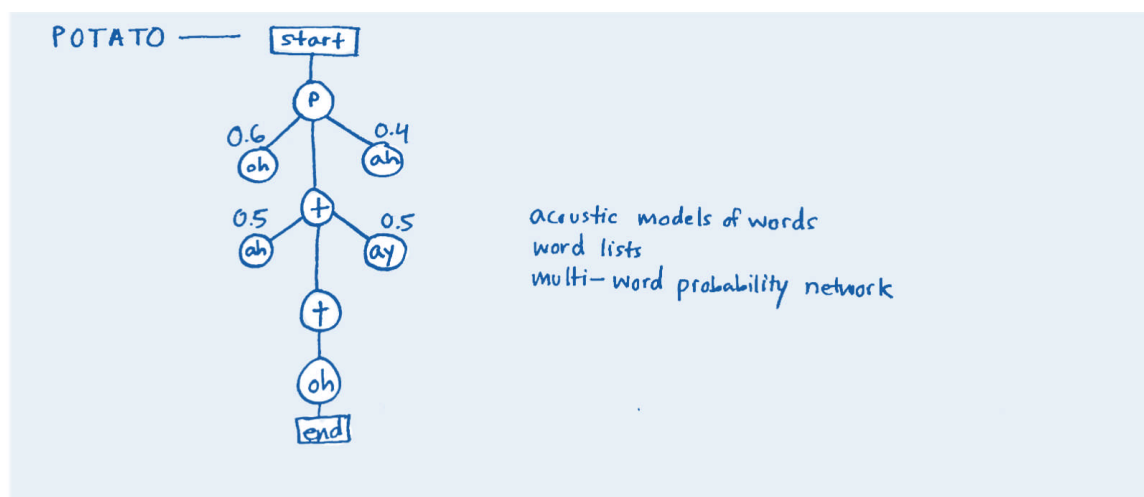
- computational biology
- semantic search
- speech recognition
- image processing
- document classification
- sports betting

## HOW DO MACHINES CONVERSE?

Having machines that can intelligently converse with humans represents a major breakthrough in our interface with technology. But good conversation requires an understanding of context and intent as much as it does the ability to hear, see and translate words.

Voice assistance is growing fast, now representing more than 15% of all search traffic. As it grows in accuracy, conversational artificial intelligence, such as chatbots, is becoming ubiquitous.

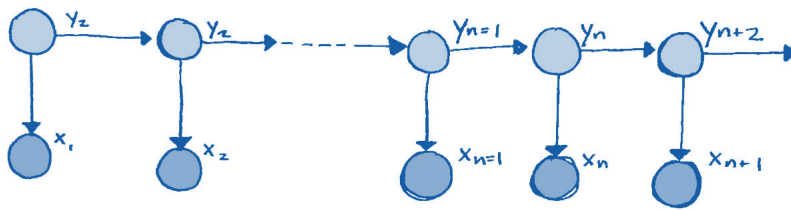
Human speech is complex, with many branches, twists and turns so the training data set for natural language is very large. The selection, compilation and preparation of training data in natural language processing remains somewhat of an art and the details can make the difference between a well-performing system and a poorly performing system. And in language, accuracy matters. An accuracy of greater than 95% fundamentally alters the user experience and is the point where conversational artificial intelligence becomes the preferred interface for many interactions.



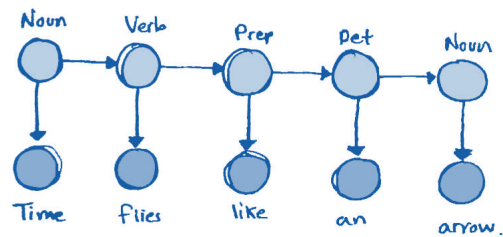
One way that machines can learn speech is with a special case of Bayesian networks called a Hidden Markov Model. Hidden Markov Models can model time series data and represent probability distributions over sequences of observations. The assumption that an event can cause another event in the future, but not vice versa, allows the model to “flow forward” in time.

In the next figure, there is some state ( $x$ ) that changes with time that we want to estimate or track, in this case words in a sentence. The words are hidden so we cannot directly observe them. Instead we want to predict them. What we can observe is something correlated with the actual words: the state ( $y$ ), which represents, say, grammatical structure. Each state applies a probability distribution over the output. We may not know the words (hidden) but we know that they correlate with the way words fit into the structure of a sentence.

## HIDDEN MARKOV MODEL



GOAL: given a sentence determine the part-of-speech tag of each word.



- system has to figure out where word starts & stops

r eh k ao gn ay z s p iy ch

"recognize speech"



r eh k ay n ays b iy ch

"Wreck a nice beach"



An intelligent assistant can predict the next word in the sentence (the hidden element) based on the network structure and the prior probabilities of the non-hidden elements. Speech recognition systems figure out how to find hidden words, build them into phrases and turn phrases into sentences. They rely heavily on being trained on relevant vocabulary and on as much data as they can get about which word is more likely to follow any other word. As more data piles in, the system calculates new probabilities and its accuracy improves. Talk to Siri, she'll only get better.

## ANALOGIZERS

*“Sword is to warrior as pen is to ?”*

*“Go is to green as red is to ?”*

*“Meow is to cat as bark is to ?”*

These are simple analogies. It’s a trivial task to complete the sentence with the correct word: writer, stop, dog. In many ways, analogies are the oldest kind of intelligence and it’s something we humans do intuitively all the time.

In machine learning, analogizers match bits of data using mathematical functions to ask, “what things do I see that are most like things I’ve seen before?”

### ANALOGIES

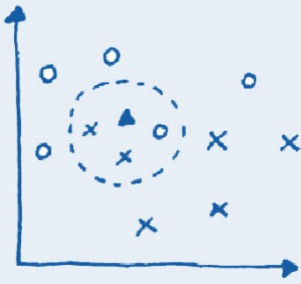
- Learning by analogy is recognizing similarities between situations and inferring other similarities
- Key problem is judging how similar two things are

Many recommendation and classification systems are based on analogous algorithms. At a high level they all work on a common principle and are all trying to find or construct what’s called a “decision boundary,” a line (or surface or area) that separates one class from another.

The simplest algorithm is called k-nearest neighbor. The k refers to the k (number) of closest training examples. The graphic below shows using k-nearest neighbor to classify the blue triangle (the test sample) as closest to either the Xs or Os (the two classes). In this example using  $k=3$ , the blue triangle’s three nearest neighbors are two Xs and one O which means the blue triangle is assigned to the X class, its closest class.

This is a kind of “majority vote” and has its drawbacks if the class distribution is skewed with a lot more data of one class than another.

## K-NEAREST NEIGHBOR finding the closest (or most similar) points



The test sample (blue triangle) should be classified either to the first class of circles or the second class of xs

$K$  = user-defined constant (in this case the area within the hyphenated circle)

thus the sample is assigned to the second class of xs given  $K$ .

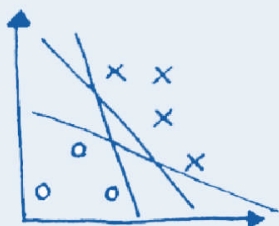
MAJORITY VOTE OF  $K$  NEAREST POINTS

$$d(x, x_j) = \sum_{k=1}^K (x - x_{jk})^2$$

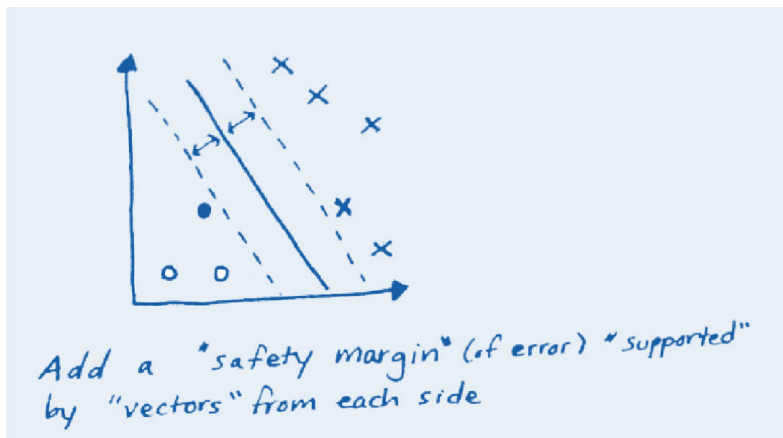
Another type of analyzer is Support Vector Machines...

Instead of setting a decision boundary by setting a value for  $k$  and computing distances between the test sample and the data points – the center of mass of the data – it is possible to separate data with lines (or surfaces in higher dimensions). The first step is to calculate the line that best separates the data – the decision boundary. But what if you want to be specific about the rate of false positives and false negatives?

Support vector machines are more complex algorithms where the data has been transformed into vectors, which are essentially compressed representations of raw data. Vectors can be used to numerically represent all sorts of data, such as words in a document. For example, the short sentence “I hate rabbits, but rabbits love me” could be represented in a word count vector (1, 1, 2, 1, 1, 1) for (I, hate, rabbits, but, love, me).

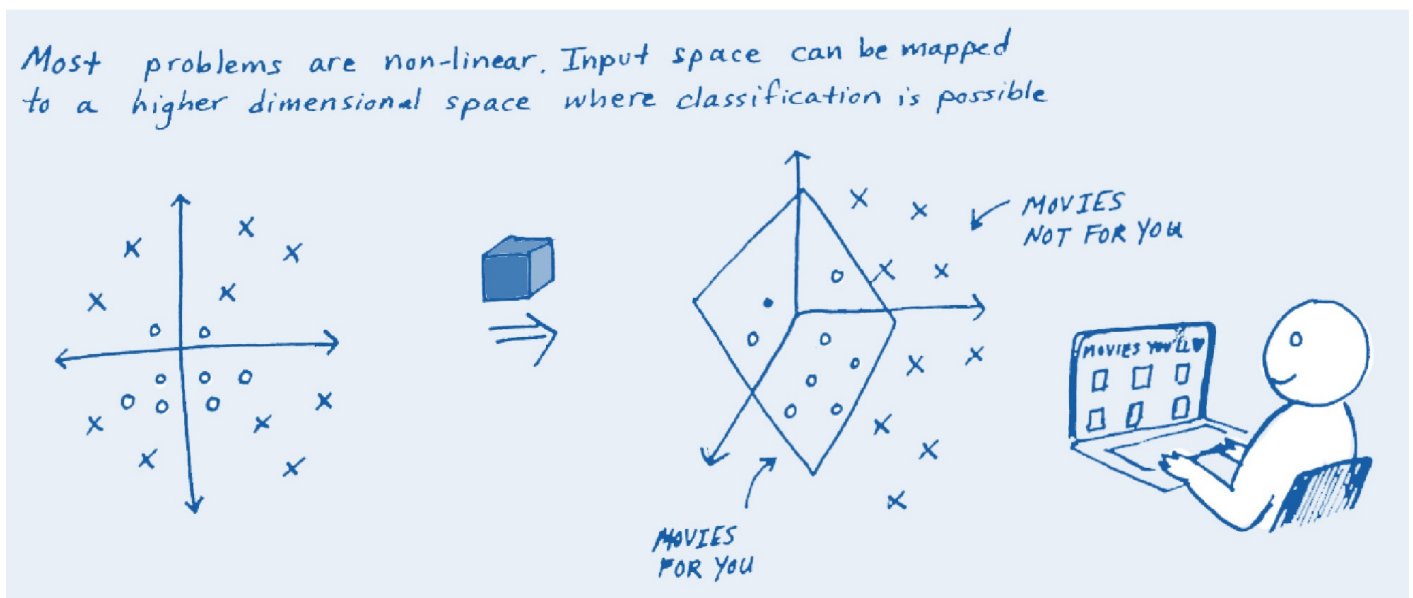


Which line is the best classifier?



Support vector machines solve for the decision boundary based on a support vector (literally, what the line leans on for "support") on either side acting as a "safety margin." This smoother boundary helps make them very accurate classifiers. It also makes them very fast and efficient on large data sets as once the support vectors are identified, a lot of other data can be thrown away. This might not be a big deal with small data sets but a lot of time can be saved when millions of data points can be discarded.

Analogizers are powerful when it comes to dealing with non-linear features in the data because they can handle data in many dimensions. When there are non-linear features, these algorithms can use mathematical techniques that create enough dimensions to be able to separate the data into distinct classes. Which means they can easily and rapidly sort through a lot of changing input, sending inputs to distinct classes. Imagine this being a little like rugby players in a ruck. When the whistle blows, they separate from their tangle and reform as two distinct groups inside their team line. A neat trick.



Analogizers are extremely common in our everyday life. They provide the technology behind many

common artificial intelligence applications in e-commerce. They learn through new data, recalculating new decision boundaries in a split second.

Say you prefer suspense movies. Your streaming content provider knows this and constantly sorts through new releases to present you with movies of this genre that it thinks you will want to watch. It predicts that because you watched *Memento*, you'll enjoy *Argo*. But instead of choosing *Argo* you choose *The Good, the Bad and the Ugly*. So the algorithm resorts and next time adds a recommendation for *No Country for Old Men*. Then someone uses your account to watch *Thelma and Louise* and next time you logon you have a suggestion for *Gone Girl*.

Next time you look for a movie, consider that there's an algorithm adjusting to your preferences, what you've watched before, what's new and what you have searched. Imagine if the algorithm had access to more data about you – who you are with, where you've been that day, what travel you are planning, what you've news you've been reading – and adjust its selection against far more personal information.

Analogizers power personalization.

# HUMAN GUIDANCE FOR MACHINES THAT LEARN

*“Machine learning works spectacularly well, but mathematicians aren’t quite sure why.”*

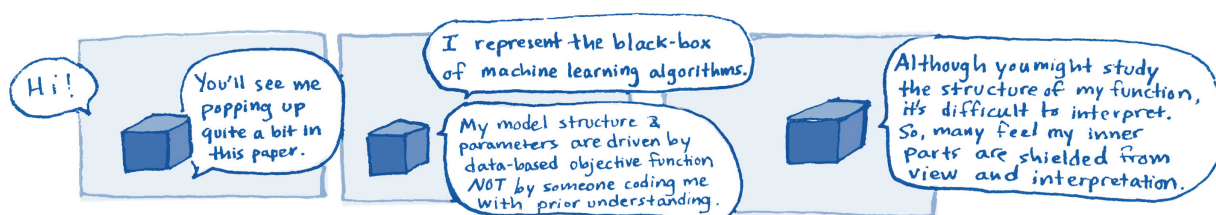
- Ingrid Daubechies

Versus

*“Machine learning is 99% human work.”*

- Oren Etzioni

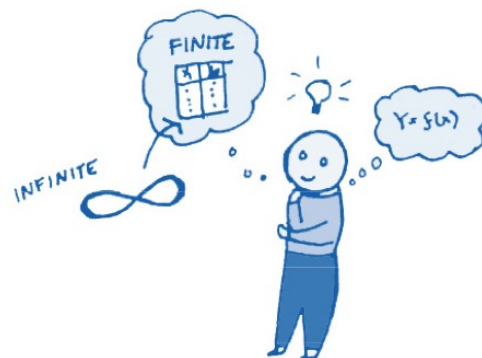
I’ve talked about the math in the black box: deep learning, support vectors, posterior probabilities. Now it’s time to talk about the human conditions.



Setting up a good machine learning algorithm has been described as requiring a lot of intuition and being something of a “black art.” Having an insight into the levers humans play with and what judgments are made in machine intelligence will deepen your understanding of just what it takes to make a machine “intelligent.”

There are many important and subtle decisions that a human makes when it comes to building a useful learning algorithm.

Something that makes machine learning different from the hard logic of an expert system is that the number of possible combinations of inputs and outputs is vast, sometimes not even finite. It’s vital to have a clever way to test and validate performance. Whether it’s personalizing webpages, where algorithms can be customizing billions of pages a day in real time, or identifying new molecular structures at the rate of millions a second for drug discovery, being assured that the machine is looking at the right data, accurately running the correct algorithms and making good predictions is key.



Bigger is not necessarily better, simplicity can beat complexity...

As algorithms become more sophisticated it can be tempting to adopt more complex models. There

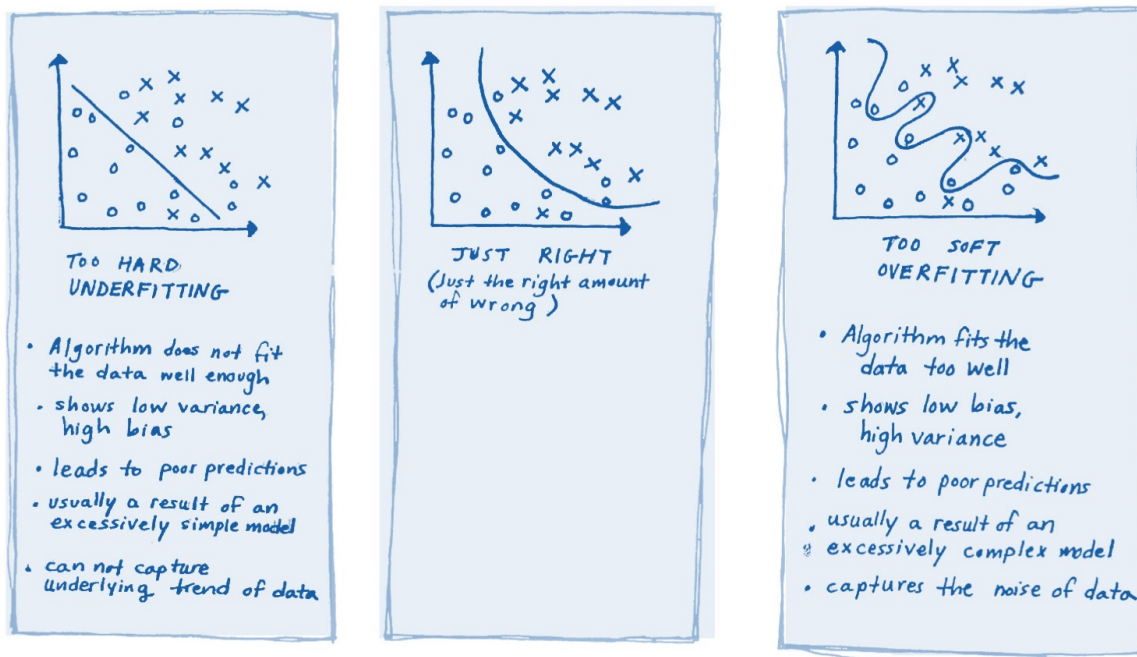
are many knobs and dials to adjust in any algorithm so complexity doesn't necessarily win out over more simple models executed well by clever and imaginative humans. While some machines are successfully learning and discovering without being directly controlled by a human, the vast majority of machine learning applications have a lot of human involvement.

## HUMAN INPUT #1: KNOWLEDGE OF THE DOMAIN

Machine learning is a knowledge lever, a way to get more from less. This means that a key criteria for choosing an approach is to have good understanding of what kinds of knowledge are a good fit.

Sometimes there isn't enough knowledge and data to choose a good algorithm at the outset. The algorithm could simply be encoding random oddities. This is called "overfitting" and Domingos describes this as "the bugbear of machine learning," it's counterintuitive and means that using a more powerful learner is not necessarily better than using a less powerful one.

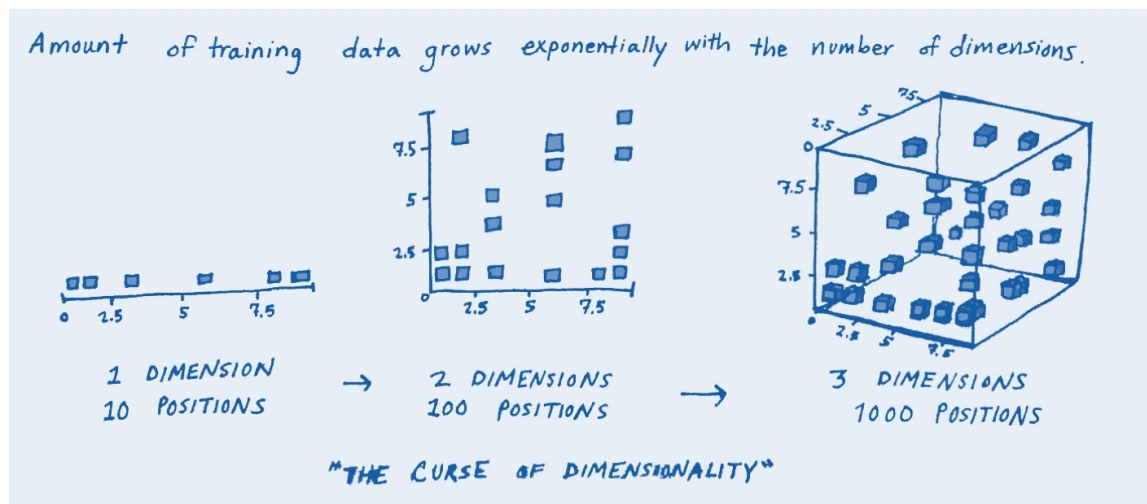
This is important to know about if you want to assess a machine-learning based product from a vendor who has taken a bunch of your data and gives you trial demo. In the world of machine learning, this may not be a custom fit, it could simply be an overfit. Seeing how the application performs making predictions on new data is the only thing that matters.



The next biggest problem is "the curse of dimensionality." As the number of features (therefore dimensions) grows, the amount of data needed to "fill up" the space grows exponentially. In some problems, there may never be enough data and just adding more features to simply add more data

may cause more problems than it solves.

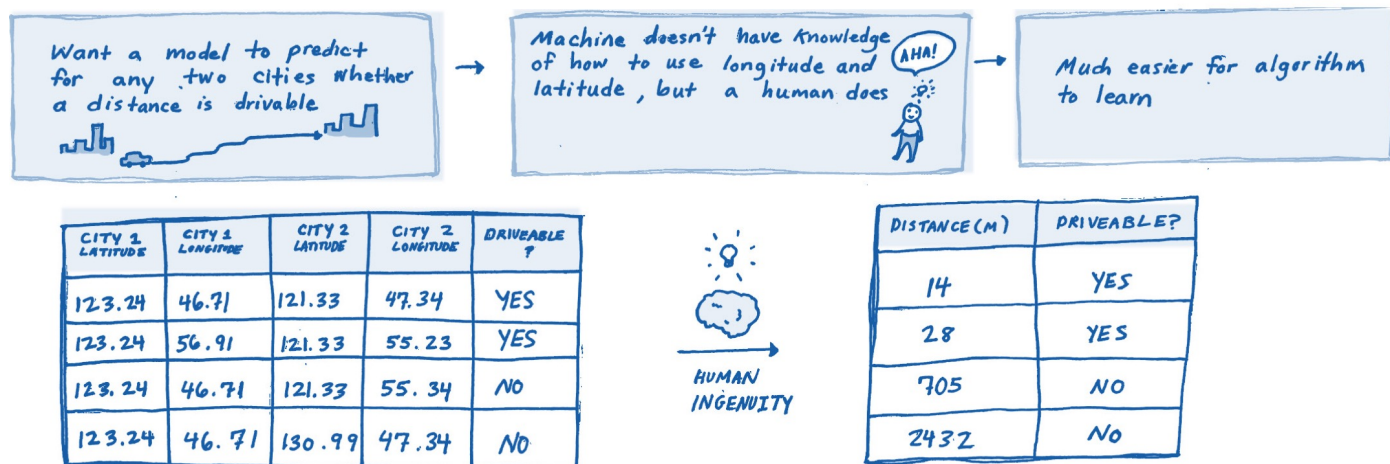
Data is the raw material of machine learning and generally more data is better. The trick is to know when it's not. For that, you need people who understand the problem you are trying to solve.



## HUMAN INPUT #2: ENGINEERING THE FEATURES

Feature engineering is where most of the effort goes in a machine learning project, an average of 70% of the project's time. And feature engineering is where humans excel, where innate skills of creativity and intuition win out. Humans can see how to put information together, craft data and inputs that enable the algorithms to more easily find the right function.

An elegant example is engineering features so that a computer can predict whether it's possible to drive between two cities based on the latitude and longitude of the city. In this example, the computer doesn't have knowledge of how to use latitude and longitude because neither is closely correlated with "drivable." But a human can see that changing the features will help the algorithm build a good model.



Raw data is messy. It's extremely time consuming to gather, clean, integrate and pre-process data and even then, it's often not in a form where it's able to be learned from. Feature engineering is how raw data is transformed from one form into another form that is better correlated with the problem being tackled.

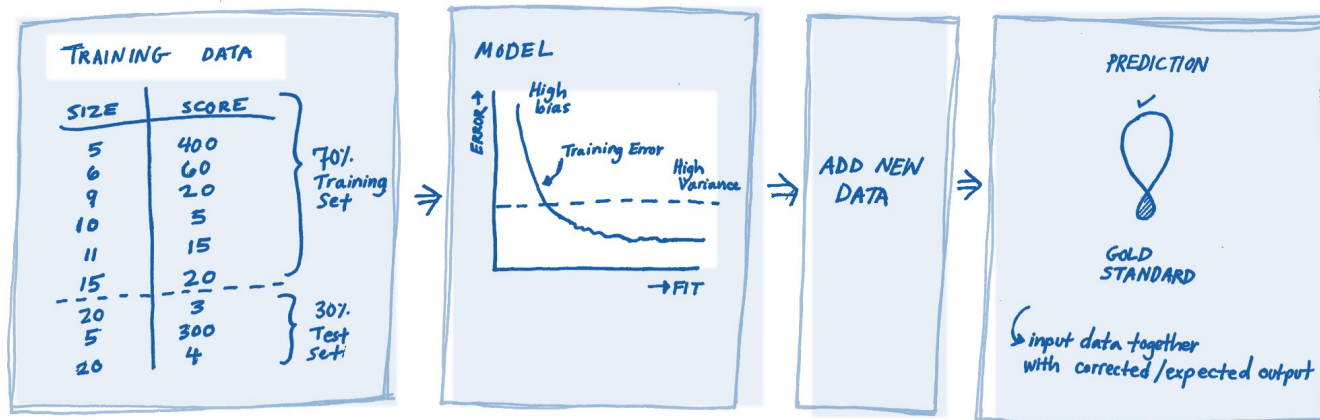
Data labeling and feature engineering will become less of an issue as machines improve processing of unstructured data. There are many hybrid approaches being researched that involve crowd sourcing or deep learning methods to extract features. Over time, feature engineering may reduce in importance, but for now, for most machine intelligence, this is an important place for human guidance.

## HUMAN INPUT #3: DEALING WITH WHETHER THE MODEL WORKS WELL ENOUGH

Once the data is ready to go, the learner is trained on a partial set of the data, the "training set." Some data is held back, the "testing set." It's not uncommon for humans to shortcut this step and get great results from the training set without adequately testing against the withheld data. As soon as new data is presented to the learner, the results then end up being no better than random guessing.

If effort has gone into constructing good features and the algorithm has been trained and tested against the appropriate data but the output isn't accurate, humans have two choices: design a better learning algorithm or gather more data. In general, more data wins out but there are limits.

Now that the world is awash with data, the constraint has moved to time. There isn't enough time to process all the available data and a lot goes unused. The clever role of the human is to pick the optimal balance between algorithm complexity, data complexity and computing power. As always, the accuracy of the prediction is what matters.



## SIX TIPS BEFORE YOU GET STARTED

### To recap...

You now know how artificial intelligence today is about using machines that learn,  
 You know that machines that learn are driven by a set of mathematical algorithms,  
 You know that they use data to continuously update their programs without human input,  
 You also know that their inner workings are not necessarily transparent,  
 And that people are key to getting the right outcome.  
 Finally, you now know what kinds of applications these algorithms are driving.

It's becoming more apparent that machine learning is a new driver of competitive advantage. Those companies that use the speed and scale of machine learning to offer, for example, better recommendations or make breakthroughs in medicine, will get ahead. Getting ahead will drive more data creation, more data will drive more learning. Rather like cyclists who can draft together drop those who can't, machine learning creates an advantage that will make it hard to keep up if you don't have it.

Some tips for success...

### #1 KNOW WHAT PROBLEM YOU ARE TRYING TO SOLVE

It sounds obvious but it's surprising how often people will breeze through this step, assuming that the problem is well defined or that everyone agrees. All the upcoming complexity and key decisions – what data, what scale, how accurate you need the prediction – will hinge on the clarity of the problem. If you decide to play with the technology but lack a clear view of the problem you are trying to solve, you will be lost in the wilderness of hyperspace, as will your algorithms.

I won't belabor the point; you've probably got your own method for checking this box. Just make sure it's done.

## **#2 GOOD DATA > BIG DATA, STILL**

Some people may believe that data quality is no longer of concern, that as long as it's big, it's good. While these algorithms certainly perform better on larger data sets, it still has to be good data. Deep learning is not a data cleaning process. In fact, the biggest gains in deep learning have been made in supervised learning with structured data.

Many companies have struggled to meet the ROI and other, intangible expectations for their Big Data projects. Whether it's a shortage of data scientists, the real-life difficulties of governing data lakes or the complexities of managing multiple vendors, advisors and service providers, data management is hard.

There's an important, and logical, reason why machine learning and big data are not in a perfect long-term relationship. There's an extra subtlety with non-stationary data and adaptive models. Once models are built they tend to only work well on the data (the world) they were trained on. If the world changes, they can stop working and yield inaccurate results. If valid Big Data is eroded by a changing world, essentially it's just small data again. And it's very hard to know whether it's just the ups and downs of the normal world or if the world has changed. In fact, there are many researchers and startups actively working on this (called sparse data) because the economics of big, non-stationary data in machine learning will limit progress in some important fields.

In any machine learning project it's important to recognize that much of the effort will be driven by the very unsexy process of selecting, cleaning and sorting data. Make sure your Big Data is Good Big Data.

## **#3 IT'S MORE SCIENCE THAN SOFTWARE**

Facebook's machine learning research group talks about "experiments." This is very deliberate language – machine learning is trial and error, domain specific feature engineering and a lot of experimental design. It doesn't follow standard software development processes and it doesn't yield a comfortable answer in the Boolean logic that is reflective of its creator, the software engineer. Machine learning projects require an even tighter problem definition than software projects because the effect of making the prediction and contributing to the environment alters the problem itself. It's game theory on a grand scale.

There are gaps that exist between the old software models with programmers and traceable logic and

the new coding of machine intelligence. There is no version control, there are no specific releases, and there is no testing. As machine learning diffuses across systems, companies must figure out what it means to de-bug something that is not modular, constantly changing and may unwittingly embed biases based on the nature of the statistical algorithms and training data.

Machine learning is not tidy, staged or certain. It's a process of prototyping, evaluating, testing and scaling. And just as pharmaceutical companies can't afford to spend all their resources at the pre-clinical trial stage, neither can you spend all your budget on the first iteration. Many experiments are required and maintenance remains an unknown quantity. Many problems need to be reframed in less-than-absolute terms, say by setting acceptability rates for accuracy or false positive / negative results. This world of uncertainty isn't a natural fit for companies that think in absolutes.

Get comfortable thinking in probabilities.

## **#4 IT'S MORE WISDOM THAN HACK-DOM**

There's a reason many of the celebrities in artificial intelligence have their share of grey hair. Machine learning artisanry takes years and usually a PhD, or two. Over the last twenty to thirty years, as the foundations for today's machine learning have been laid in applied mathematics and statistical artificial intelligence, people who persisted had time to try numerous approaches. Today, as a new generation of data scientists and engineers craft modern artificial intelligence, the earlier experience plays an important and active role. There is a long history in the fundamentals that modern computer science development relies heavily upon.

The big players in artificial intelligence and machine learning all have research groups because R&D is an important investment strategy – Facebook, Google, Baidu, Microsoft, IBM, Uber, Toyota. Facebook developed their own natural language translation system. The language of Facebook is quite different – full of colloquialisms, abbreviations and localizations. OpenAI's mission is to build safe artificial intelligence, available to all. The machine learning community publishes findings and often values know-how over patents. Many start-ups are founded by academics. It's a way to get recruited and a way to see research applied.

Modern IT tools have lent themselves to a hack mentality. In companies, hack-a-thons and skunk work developments seem almost to mock the inertia of the installed IT architecture. Some of this sentiment exists in machine learning. Google's TensorFlow and AWS's machine learning as well as host of open source machine learning libraries may give the impression that this is simply an easy add-on for a software engineer to master. Even better, it's free.

But there is no free.

The process is one of science – hypothesis driven, solid experimental design, deep understanding of the principles, a facile knowledge of the subtleties of statistical multi-dimensional analysis and innate

skills for creative ways to test results against a control. It can be difficult to forecast the time and cost of a project and, as in science, taking an experimental, hypothesis-driven, proof-of-concept approach is the best way to start.

## **#5 AI TENDS TO POLARIZE PEOPLE; THE MESSAGE MATTERS**

Technology disrupts labor. Everyone you talk to has an opinion about robots taking jobs or about the future of professional life. When Oxford University scholars, Frey and Osborne, make claims that 47% of jobs in the USA are threatened by technology, people get nervous. Deloitte economists traced technological-driven employment changes in the UK over 150 years and concluded that the technology debate is skewed towards destruction. They found that technology has many creative effects but that they are harder to define and therefore to predict.

The problem for individuals is that technology moves a lot faster than people. There's a direct relationship between enterprise status (job level) and enthusiasm for artificial intelligence. One study shows how big this gap is. AI excites around 42% of senior executives while a measly 15% of front line workers share this sentiment.

There is a prediction asymmetry in technology. We can't envision what will be created nearly as easily as we can determine what will be destroyed, and we fear losses more than we desire gains. People are far more likely to fear artificial intelligence and resent those who have it if they believe it will result in losses for them.

You can't envisage all the gains and losses from artificial intelligence but you can be thoughtful in how you talk about them.

## **#6 VALUES AND TRANSPARENCY, BIAS AND ETHICS**

The word "algorithm" is getting lodged in our collective consciousness as some mysterious entity, operating outside of our control and being used by unseen others to manipulate us. Whether it's how Facebook delivers news or how Google gives us our search results, algorithms are somehow seen as having morality, in and of themselves.

By now you know that algorithms are simply sets of rules or routines that we've had for a long time. Before big data and machine learning algorithms, algorithms were hand-coded by programmers, with bias handled with policy. Policy was informed by statistics. People made decisions on policies, which determined such things as who would get a loan approved. Embedded in any insurance company's actuarial tables are data on whether men or women are safer drivers or what should be the difference in relative premiums for low income versus high income households. We didn't refer to the algorithms as having bias. We referred to statistics, risk and fair-treatment policies or laws preventing discrimination.



We can use machine learning to create a lot of new knowledge from a relatively small amount of prior knowledge. This makes these algorithms very powerful engines of discovery. But it can also result in propagation of any bias when they are scaled up and used across many problems, computers or social networks.

So what new sources of unfairness should we be worried about with the diffusion of machine learning algorithms?

## **INEQUALITY**

Developed economies, with sophisticated technology infrastructure and well-developed data systems, will be able to access the benefits of machine learning artificial intelligence at a much faster rate and to much greater effect than developing countries. Sophisticated stores of digitally structured data abound in wealthy countries. On the other hand, with breakthroughs in machine processing of unstructured data as well as computer vision and natural language processing, developing economies may have a chance to “leap frog” a lot of traditional and costly data infrastructure. But it's impossible to ignore the head start that developed economies have and the risk of further economic inequality between nations.

## **DISCRIMINATION**

Google offensively labeling black people as gorillas? Staples discriminating against poor people?

Gender stereotyping in search results based on what job is queried? Discriminatory on-line advertising that associates low paying work with women? All of these happened.

Programmers routinely incorporate user data into complex algorithms, heuristics, and applications. Most of the time what we get is beneficial, giving us more finely grained information. But it can have unintended consequences such as discrimination, especially if information on minorities is underrepresented in the data that was used to train the algorithm.

We should think of these unwarranted associations as bugs. And do what we used to do with policy - check we had it right - with new forms of debugging software. Discrimination in this context is machine bias. And machine bias is a bug.

## **LACK OF TRANSPARENCY**

It's true, we don't know quite how machine learning works. But it does. Just consider that for a moment. When the output of an algorithm is no longer traceable, when the answer is instead bound in complex, multidimensional probability, and the person who is accountable can't explain why a particular decision was made, what are we to think?

Much of what we consider fairness relies on trust as much as it does on an explicit demonstration of right over wrong. Decisions and actions aren't always black and white. Our sense of justice being done or a decision made fairly is as much based on how we were treated and whether someone acted in good faith as it is on a categorical boundary. If people defer to an algorithm, shrug their shoulders and say "well I don't understand it either" trust will be mortally wounded.

## **ETHICS**

Name any thorny ethical issue and chances are there's no known, much less simple, solution. The law is a hazy code of a society's ethics, based on context and prior cases. Intuition fails us when we are presented with a dilemma. We can hold two competing views in our head but a machine cannot.

There are also hidden ethical issues that affect our daily lives. In any algorithm design there is necessarily a certain number of false positives and false negatives. The designer of the algorithm has to make tradeoffs between these two. If this is an algorithm for identifying pictures of cats the tradeoff may not be especially critical but, if it's an algorithm for medical imaging for a dangerous condition, the design of this tradeoff has profound consequences. Such an algorithm embeds ethical considerations for the diagnosing doctor. The designer of the algorithm, not the doctor, makes the tradeoff between the risk of missing cancer in a small number of cases versus the risk of unnecessarily alarming a group of people who are, in fact, not sick.

## PERSONALIZATION

A core goal of personalization is to create an experience that's completely oriented to your individual situation, whether it be in medicine, e-commerce or in a physical-world experience. The dilemma is in privacy and security and in creating an experience that's trusted. Too personal can be profoundly uncomfortable. Designing truly human-centric artificial intelligence is harder than it looks and, outside of science fiction, hasn't been achieved very often.

In summary, many things that are being revolutionized by artificial intelligence are unchanged. They still function according to the rules that are already in place. But pay particular attention to the new places the rules need to be applied – the way data is collected and used, what proportion of false or incorrect results algorithms generate and who sees those results, the attitude of people to receiving automatically generated decisions, the way customers react to accurate personalization. These are all new areas of oversight and governance in the age of machine intelligence.

### To Recap...

- Know your problem

- Know your data, what new data you need and what it will cost to get it

- Define success in statistical terms and think in probabilities

- Take an experimental approach, start with a hypothesis and a proof of concept

- Focus on the new capabilities you will need and develop a conversation around those

- Update your governance and policy to account for machines that learn.

# THE LAST WORD

*“AI is the new electricity.”*

- Andrew Ng, Stanford University

Machine learning is as big a breakthrough as the invention of electricity. Machine learning is the fundamental technology that will diffuse through everything we touch. Computers no longer learn through programming, they learn through experience. Trading your stocks, identifying the specific drug that will work best for you, driving you to work, writing your personal news articles, predicting what services you need and when, offering you a set of curated products; these are all being done progressively faster and more accurately because of machine learning.

Now you understand the mechanics of the math and the scope of the engineering decisions that go into these algorithms. Now you can see that these algorithms aren't yet ready to assume a human-level general intelligence without many other advances in computer science, social sciences, neuroscience and a host of other technology-related disciplines. Consider how a child learns to recognize a cat. With somewhere less than a dozen images, a child can not only recognize the cat, but generalize this knowledge to be able to recognize it from the side, or behind or in poor light. Today, even the best deep learning systems require orders of magnitude more images. Or consider too, how while each of these algorithms can be very powerful, researchers are still working on putting them together in a way that maximizes the advantages of them all in one algorithm.

Machine learning hasn't yet solved the hard problems of common sense; the background knowledge we acquire through our experience of the world and the ability to appreciate context and therefore, to reason. The ability to know that a toilet and a chair, whilst visually similar, have quite different purposes most of the time. But this is changing fast as machine learning accelerates discovery of new ways to solve the big challenges of artificial intelligence.

Machine learning is at an exciting point. Now is the time to learn about what it is (and consequently isn't) so you are equipped to follow the hype with a critical eye, to ask the right questions and to be consciously aware of its potential for bias. Now is the time to learn how machines learn so you can understand the impact on your daily life and, in future, be able to make your own predictions of how it changes our personal connections, our education, our professions and our shared experience of human creativity.

# ACKNOWLEDGEMENTS

Intelligentsia Research acknowledges:

- The “giants” of AI whose work has both informed us and been instrumental in making this primer both accurate, yet accessible
- Celeste Knudsen for her skilled and entertaining illustrations of complex machine learning concepts
- Pedro Domingos, whose book “The Master Algorithm” we highly recommend. The development of the “Five Tribes” structure has made machine learning far more accessible for everyone.
- BigML for their ideas on feature engineering in particular
- Jason@machinelearningmastery for helping us step through the basics
- Zachary Lipton at UCSD for explaining how time works in a neural net
- Bob Van den Hoek for his Quora answers
- Norm Jouppi, Google
- Alex Gray, Skytree
- Yann LeCun, Director of AI Research at Facebook
- Kalid Azad, BetterExplained, for perhaps the best ever Bayes explanation (and we studied many)
- Ivan Vasilev for the flu
- And all the other individuals we talked to, at conferences and on the phone, who are working to make machine learning great.