What is Machine Learning Anyway

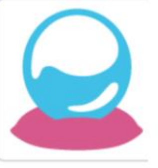
Presenter: Michael Weir

26 February 2020

Objective

- Give you a sense of what Machine Learning (ML) is about - what it is, what it isn't
- Show you what ML looks like as a workflow
- Highlight three representative scenarios for different uses
- Provide examples of good and bad uses
- Provide some ideas for how to think about ML
- Future directions and discussion/questions

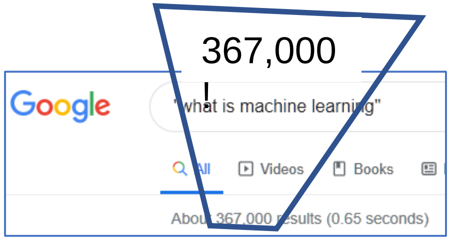
First, a quiz…

- Which is the closest match to a deployed ML capability:
    - A  B  C 

- Which of the following is the truest statement:
    - It is always better to have more data
    - It is always better to have the smallest amount of good data to solve your problem
    - It depends
- True or False (for yourself): I personally don’t use ML
- ML has been deployed in the field since the:

  1950’s 1970’s 1990’s 2000’s

So … What Is it?



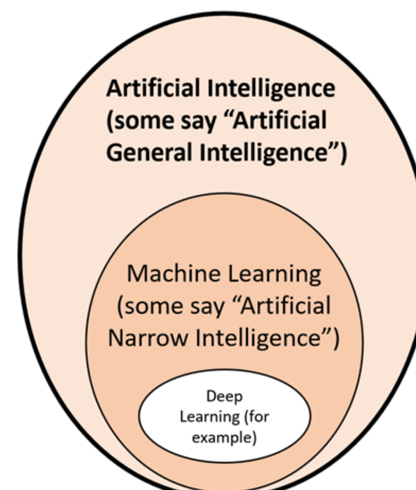
- Google (4 days ago) 367,000 hits
- Industry (2015) - 4 types of ML
- Academia (2012-2019) - Variously 3, 4, 5, 7, … types
- Why so many different views
  - Algorithms versus learning approach versus methods
    - Terminology differences/conflict – what's a "type?"
  - More recently, "fame and fortune…"
- We will stick to basic architecture/general consensus
  - Base ML family and two offshoots (NN, CNN, RNN)
    - Neural Network, Convolutional NN, Recurrent NN
  - Supervised Learning to get grounded in the mechanics

ML - what it is, what it isn't

- ML is a subset of Artificial Intelligence (generally accepted)
- It takes in data (numbers) and finds patterns
- It is made up of data handlers and mathematical algorithms
- It is software that people build
- It is closely related to, and includes many

ideas from, Data Mining

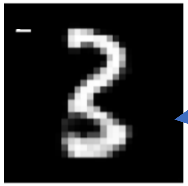
  - and sometimes, you only need DM, not ML



ML - what it is, **what it isn’t**

- ML is not equivalent to AI
- ML is not intelligent (but it is artificial…)
- ML doesn’t understand things like you do
  - It does not see things, hear things, read things…
  - It uses math to transform data and seek patterns in numbers
- ML does not know when it is wrong.
  - That’s the scary part for me…
  - And why we need more understanding

Trust me, this is a Manhole Cover!



And this is a “2”! *

* From “Generating Natural Adversarial Examples”, 2018, here: https://arxiv.org/pdf/1710.11342.pdf

Machine Learning in Practice

- It is important to consider two very different aspects of ML in practice – Learning and Inferencing
  - Learning is what most people think of when picturing "doing" ML – the data, training, testing, validating, etc.
    - This is really the "build it" phase of an ML capability
  - Inferencing is what it is actually used for in practice
    - *Deploying* a capability – the reason you build it
  - This brief emphasizes mostly Learning, with some nods to when/where Inferencing constraints need to be considered in the Learning phase

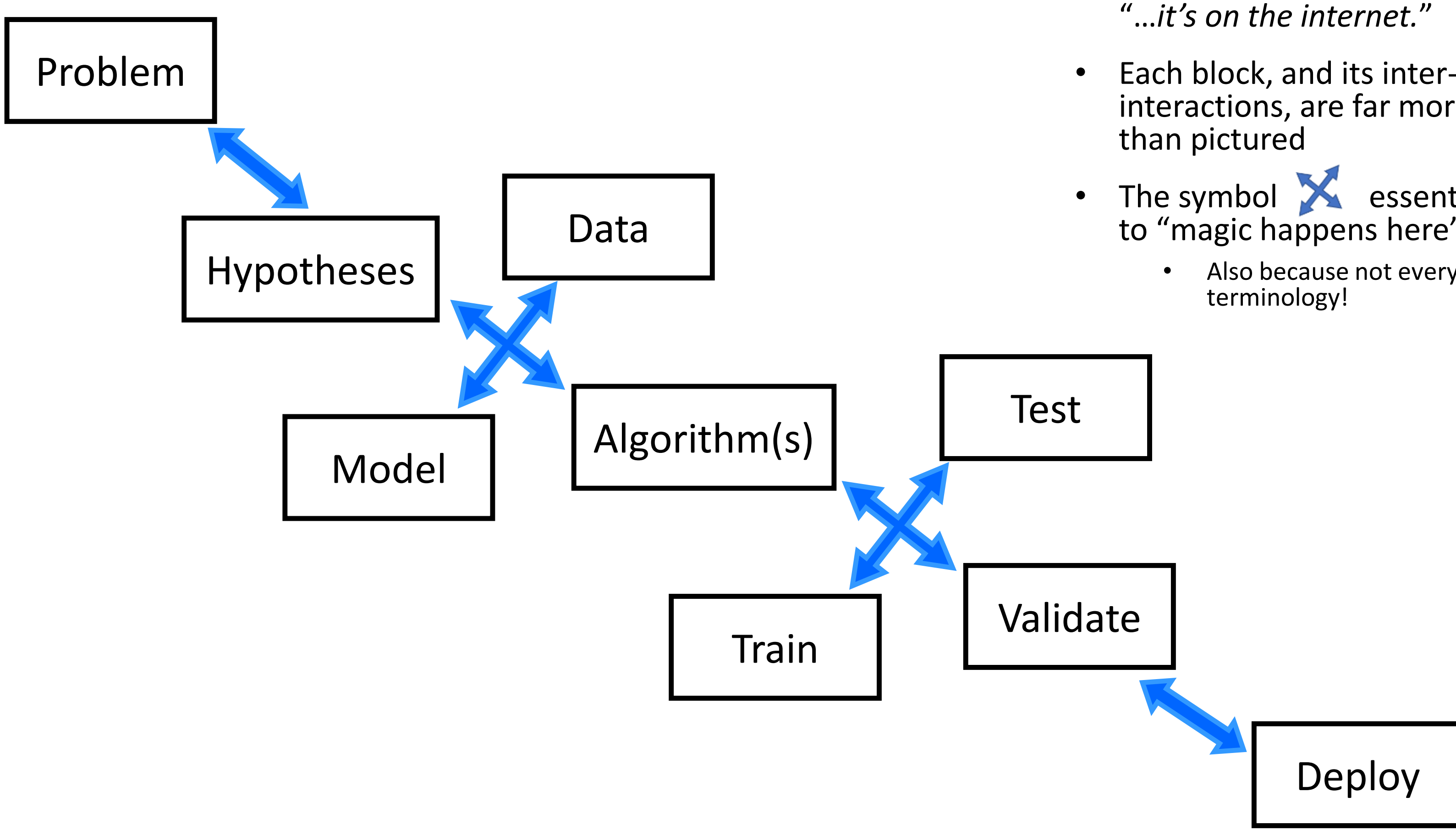
Starting position–Why use it?

- Three characteristics are required to effectively apply ML as an asset to solve your problem
  - There is existing/generated “good” data (usually lots of it)
    - No/minimal data, no ML
  - There is (there might be) a pattern in the data
    - No pattern, no ML*
  - The problem is likely intractable using other tools
    - If there are other methods, they are probably less resource-intensive
    - If there are other methods, they may be more robust/accurate (no “guessing”)

Framework for the ML Discussion

- With preliminaries out of the way, following is one possible holistic picture of an ML “pipeline”
- It starts with the Problem, and flows to the Deployment of a capability
- The “pipeline” here is a (not strictly sequential) process that represents a problem-solving activity, and where ML can be applied…

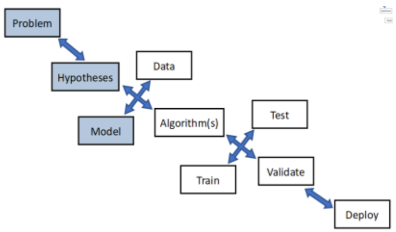
General ML Pipeline



- Each of these blocks is at the same level of generality as the statement *"…it's on the internet."*
- Each block, and its inter-block interactions, are far more complex than pictured
- The symbol essentially translates to "magic happens here"
  - Also because not everyone agrees on terminology!

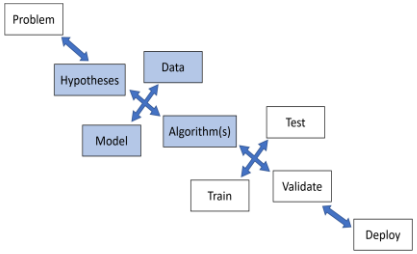
Problem/Hypothesis/Model

- From a few slides back, you need to satisfy a few conditions to effectively use ML
  - Data, pattern, context (is ML best use)
- Then, you as the problem author need to define the hypothesis/model to define boundaries
  - There is no substitute for domain knowledge
  - A weak model (non-representative of real problem) will almost always guarantee weak ML capability
  - This "think first" aspect is missed by many ML'ers
    - There is no magic in ML; GIGO applies! (garbage in, garbage out)
- Definitions (Problem/Hypothesis/Model) vary in different ML camps, but the ideas are the same

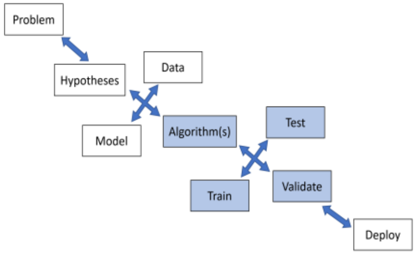


Where the Wild Things Are

- Hypotheses/Data/Model/Algorithm
  - Creativity playground, grunt work, background research
  - Data mangling

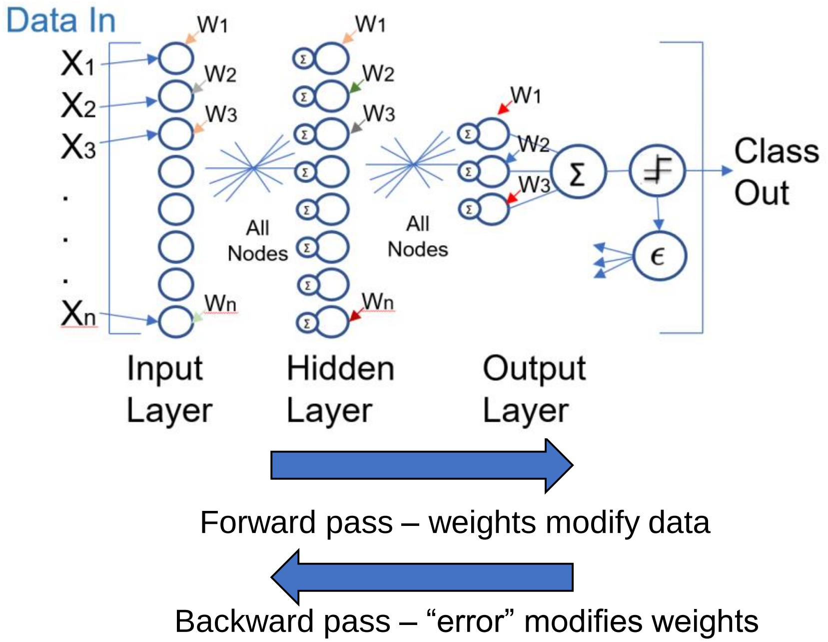


- Algorithms/Train/Test/Validate
  - Refining parameters, using expertise, making guesses
  - Interpreting outputs, reflecting on model
  - If frustration gets too high, back to H/D/M/A, else…
  - Prepare for transfer to real-world



ML - Simple Neural Net

- The diagram shows the basic flow of information through a typical (simple) Neural Net for 3 Classes (i.e., was the input data sample from class 1, 2, or 3)
- On the forward pass, weights are used to modify the values of the input data to each node in a layer, passing that value to all of the nodes in the next layer (we are only showing two)
- On the backward pass (“Backpropagation”), corrections are fed back through the network to change/update the weights (based on a measure of whether or not the “answer” is correct)
- Those new weights are used to modify the values of the next input data sample, which are then passed to all nodes in the successive layer, to get a new “answer” (etc…)
- When weights stop changing, the “learning” is done



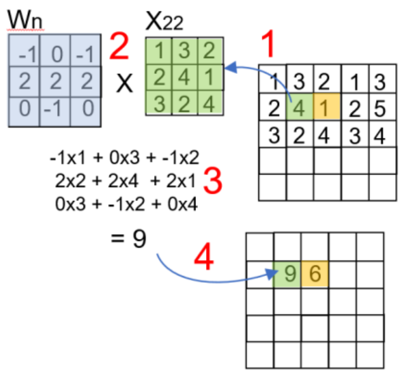
Key idea – “weights”, “error” are not explicitly programmed; they depend on sample input and each other – it “learns.” It is the foundational basis for most all ML approaches.

ML - Convolutional Neural Net

- The CNN operates on input values not as a column vector of sample values, but a 2-D representation of the target class (like a picture of a hand-written digit of 1, 2, or 3)
- Blocks of sample space (i.e., 3x3) are multiplied by a block of weights to arrive at a 3x3 matrix of values; those values are summed to represent the new "value" for the center sample
- The weight block is moved across the 2D space with overlap and the products are summed for each sample in that 2D layer (here, 9 values added to get one new sample value for each sample)
- The resulting 2D sample space is the same size as the original, with new values at each location (simple CNN)
- Real-world CNN layers are more complex

Basic 4-step calculation of weights times data block, for location [2,2] (value 4) and output location [2,2] (value 9)

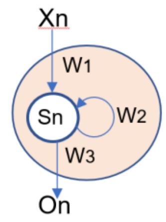
At next step, location [2,3] (value 1) would become 6



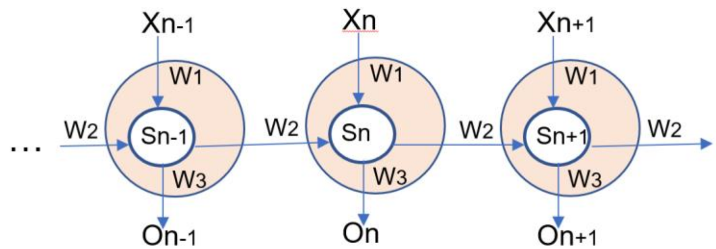
Key idea – Convolutional layers build up 2D patterns in successively more complex ways to represent higher-level abstractions of the 2D space (like "cat"). That is, they can persist relation information over spatial extent

ML - Recurrent Neural Net

- The RNN operates on an input as a sequence of values, and tries to find relationships between a given value and those before/after it within a given window. It has to have memory to accomplish that as data moves through it
- For a single RNN node, there are three weights and a “hidden” state/value that represents the current and previous (or zero) condition
- When viewed in sequence, the propagation of information is visible in the updating of the successive “S” values with previous information
- This is simple RNN; real-world RNN implementations use Long-Short-Term Memory (LSTN) nodes, which are more complex and can retain longer time/distance relationships with better learning performance



Single RNN node; “hidden” value Sn is “memory”; weights are shared across RNN layer



Laid out across time, you can see the propagation of information to following/future nodes; backpropagation needs to account for both layer sequence and time sequence to update weights

Key idea – Recurrent layers build up before-after relationships in successively more complex ways to represent higher-level abstractions of the sequential (1D) space (like phrases and sentences). That is, they can persist relation information over time/distance

Examples of Good/Bad Uses (1)

- Lots of great successes
  - Image analysis to support medical diagnosis, screening; live surgery support (link)
  - Data analysis to support weather forecasting (link)
  - Chemoinformatics for new drug discovery (link)
  - Learning molecular electronic properties for new materials (link)
  - Handling the classification portion of smart disposal and waste management for better recycling (link)

Examples of Good/Bad Uses (2)

- Middle of the road "successes"
  - ML to name new ice cream flavors (link)
  - Finding animals in a supermarket trip (video)
    - Applying Google's DeepDream to video to "find" animals – it will do what you tell it to do. Hypnotically entertaining and instructive
  - Autonomous cars (so far)
  - Autonomous "X" of various types (ML plus other tech)
    - Household helpers, "really smart" devices
    - Hotel guest assistance (robotics and ML)
    - Eldercare support (link)
  - Recommendations on various web sites (link)

Examples of Good/Bad Uses (3)

- “Bad” and “Arguably Bad” uses
  - De-anonymization of public data via ML across terabytes of data (coders) (public) (yahoo)
  - COMPAS ML program for determining criminal offense and jail time (link)
  - Insurance companies mining your health data (link)

How to think about Machine Learning

- The Good news and Bad news –
  - ML will give you an answer
  - Neither you nor the algorithm really know if it is right
    - Valiant, Probably Approximately Correct (PAC)*
    - (but lots of folks are working on it)
- For modern (i.e., real) problems, you need lots of data (sample sizes of 10^6, 7, 8, 9…)
- Use frameworks to get to work quickly, but know that doing ML well means understanding the math
  - "Rules of thumb" are baked into most of them but…
  - It is very, very, very easy to get "wrong" answers

*Valiant, Leslie - https://en.wikipedia.org/wiki/Probably_approximately_correct_learning

What We Didn't Talk About

- Security, Policy, Privacy…. (all big discussions!)
- Current or trending techniques/ideas
    - Deep Learning, Generative/Adversarial, Genetic/Evolutionary, State-based/Q Networks, Reinforcement Learning, Inverse RL, Geometric methods…
- Data handling/cleaning
    - A very large part of a working ML capability,
    - Some estimate 80% of the job is data mangling
- Ethics (data bias, ethical use,…), especially for autonomous uses
    - A driver and a pedestrian have very different views of what should happen…
    - Dept of Defense, 24Feb20 Ethical Principles for Artificial Intelligence
- "What's hot in ML right now"
    - "Explainable AI", automated/autonomous data provisioning, etc…
- All are valid topics on their own, lots to explore

Questions?