Machine Learning: Evolution and Perspectives

José R. Dorronsoro

July, 2021

The Mechanization of Reasoning

The Origins

- Starting points:
  - Aristotelian logic: rules for general reasoning
  - Euclid axioms: basis from which all geometric truths can be logically derived
- Natural ground: Mathematics
- First try: Gottfried Leibniz with his *characteristica universalis* and *calculus raciocinator*
- Great impulse by the increasing formalization of mathematical reasoning on the second half of the XIX century
  - Big names: Karl Weierstrass, Georg Cantor, Richard Dedekind, Leopold Kronecker
- Culmination in *Grundgesetze der Arithmetik* by Gottlob Frege (1893, 1903)
- But just before Frege's second volume, Bertrand Russell points out that paradoxes can follow from Frege's axioms

Hilbert's Program

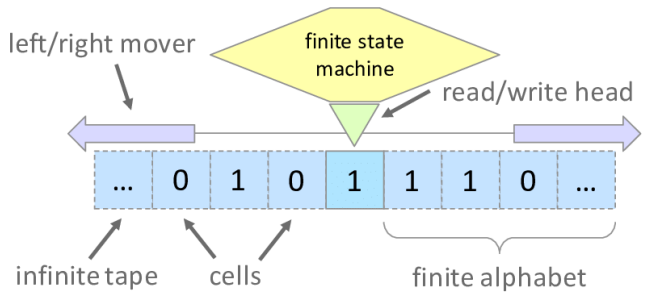
- Problem: Frege's system allowed for **inconsistencies** derived from definitions such as the set made up of *all sets which do not have themselves as members*
  - The barber's paradox: if the barber is the person who shaves anyone that doesn't shave himself, who shaves the barber?
- Consequence: strong sentiment of crisis but also a new impulse to have a solid foundation of mathematical systems
- Formalization and direction in **David Hilbert's Program**, with two key requirements
  - **Consistency**: to free mathematical reasoning from paradoxes
  - **Completeness**: to ensure that any true statement in such a system can be proved to be true applying the system rules

The Great Shock

- However Gödel proves (1931) his **Incompleteness Theorem**: *There are consistent mathematical systems that contain true statements that, however, cannot be proved within the system itself*
  - Example: any consistent formal system which includes Peano's basic arithmetic axioms is incomplete
- But what does it mean to prove something?
- Another Hilbert's requirement is **decidability**: there should be an **algorithm** for deciding the truth or falsity of any mathematical statement
  - Underlying belief: the solution of any mathematical problem is susceptible of being "mechanized"
- To go on this requires to make precise what do we mean by an algorithm

Turing Machines

- They formalize the intuitive concept of an algorithm
- They consist of an infinite tape where a head can read or write ones and zeros plus
  - A finite set of states that include the start and end of its execution
  - A finite look up table to decide actions



- And there are **Universal Turing machines** that can replicate the working of any other Turing machine
  - They can be seen as the equivalent of modern general purpose computers able to execute any program
- Next question: can the brain workings (and hence human reasoning) can be captured by a Turing machine?

The Church-Turing Thesis

- Essentially, it states that, indeed, the preceding is true
  - It is more of an hypothesis than of a formal statement susceptible of mathematical proof
- Refutation has been tried by proposing other computation models and show they are not equivalent to Turing's one
  - But this has not been achieved
- In fact, modern computers fall within Turing's model
  - Small wonder: to a great extent Turing invented modern computers!!
- The **Physical Church Thesis** states that any physical computing procedure can be replicated by a Turing machine
- If true and given that the brain is obviously a physical computing system and intelligence derives from its workings, it would follow that **intelligence can be implemented on Turing machines** (and, hence, on computers)

The Rise of Artificial Intelligence

Towards Computer Intelligence

- New factor after 1945: the modern computer
  - The previous mostly theoretical discussions can be recasted as statements on computer program executions
  - This opens the question of whether computers can achieve “general" intelligence
  - And leads to the question on how to check whether a computer behavior can be considered intelligent
- The initial question **can machines think?** is reinterpreted in Alan Turing’s *Computing Machinery and Intelligence* (1950) as deciding **whether machines can imitate what thinking entities can do**
- Famous example: **Turing’s Test** on whether a machine can win an **Imitation Game**
- Turing appears to have believed (in 1950) that by 2000 his test would have been passed but it is still undecided
  - The yearly Loebner Prize declares the top performer at the moment
  - Winner in the last five years: the Mitsuku-Kuki chatbot
  - Try to have a chat with her ;-)

The Start of Artificial Intelligence

- Term coined by John McCarthy with the *1956 Dartmouth Summer Conference on Artificial Intelligence*
- Overall goal: **Strong AI**, i.e., machines having the same intellectual capabilities of humans
- Impressing first achievements:
  - Chequers players (intelligent in the 1950s but now automated chess players that beat everybody are not considered as such)
  - The General Problem Solver (could have saved time to Russell and Whitehead when writing their *Principia Mathematica*)
  - The first robots (work still in progress)
  - The Automated Mathematician (said to have rediscovered the **Goldbach conjecture**)
- In parallel: great and constant progress of **computer technology**
  - In the hardware and software industry but also in research and education with the birth of computer science

Rise and Fall

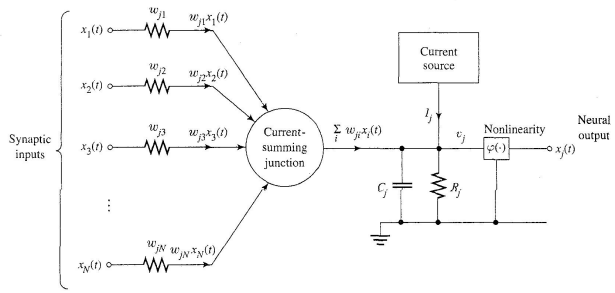
- Overall paradigm: **Symbolic Artificial Intelligence**, rooted on logic
- Main technological achievement: **Expert Systems**, made of
  - Large factual knowledge data bases
  - Production rules as condition-action pairs
  - A control system that selects and executes the appropriate rules
- But their limitations soon are clear:
  - To keep large knowledge databases and rules contradiction free becomes more and more difficult
  - They are brittle systems when facing small changes
  - Don't work when facing uncertainty or can't adapt to similar but different circumstances
  - Tedious and very ad hoc work needed to build them
- Consequence: the cold and very long **AI winter**

Connectionist AI

- At the beginning Symbolic IA was only half of the game
- The other half was **Connectionist IA**, the approach to explain cognitive and mental phenomena in terms of artificial variants of biological neural networks
- Key idea: intelligence is based on the **joint working of very large assemblies of artificial neural networks (AANs)**
- Alan Turing was one of the first proposers in his report *Intelligent Machinery*
  - He envisaged neuron-like objects connected in such a way that the **interconnection weights can be changed**
  - These adaptative weight changes would be the way the system learns
- Starting point in electronic simulations of biological neurons

The McCulloch–Pitts Neuron

- Idealized electronic version of a neuron's working (1943)



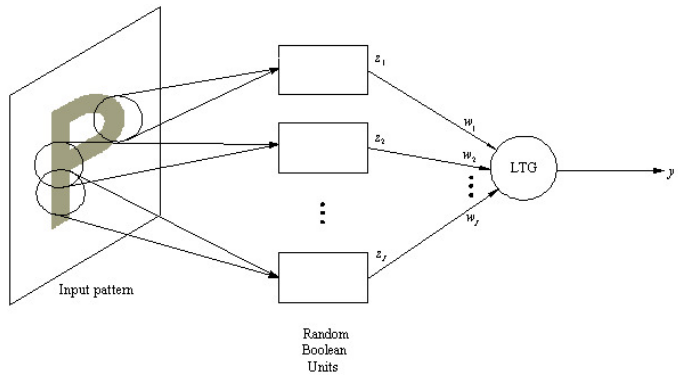
- Taking weights $w = 1/R$ as conductances, input and output potentials as x^I, x^O and $wx = \frac{x}{R}$ as intensities, the McCulloch–Pitts neuron outputs a potential x^O as

$$\begin{aligned} x^O &= 1 \text{ if } \sum_{j=1}^{N} w_j x_j^I = w \cdot x^I > I^O \\ &= 0 \text{ if not} \end{aligned}$$

- Thus, the neuron fires only if the aggregated intensities it receives are bigger than a threshold

The Perceptron

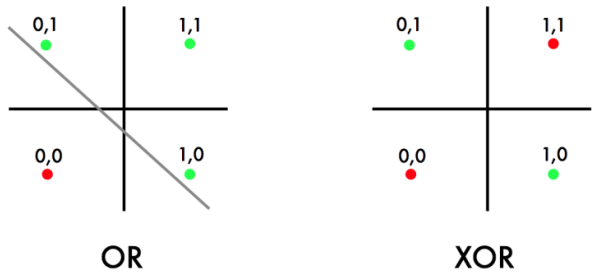
- Proposed by **Frank Rosenblatt** (1957) as a first step to build an artificial retina



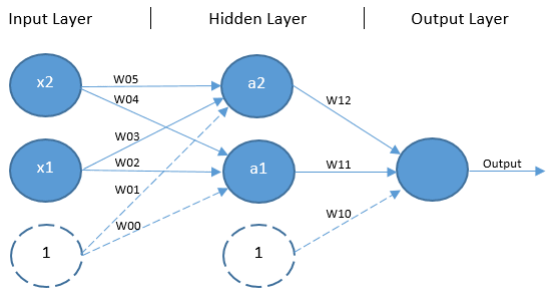
- Key step: linearly separate two classes of vectors x with labels $y = \pm 1$
  - We want a weight vector w such that $w \cdot x > 0$ if $y = 1$ and $w \cdot x < 0$ if $y = -1$
- First (and big) success: this can be done by an iterative algorithm, the **Delta Rule**, that **learns**
  - If the patterns can be separated, the Delta iterations arrive at a w that separates them **after a finite number of steps**

The Connectionist Winter

- Perceptron learning caused a lot of excitement but could not solve simple separation problems such as the XOR



- XOR (and essentially any classification/regression problem) can be solved working with **Multilayer Perceptrons** (MLPs)



- But **no algorithm to make them learn** was found, and interest in PCPs decayed

Machine Learning

First Steps

- Term coined in 1959 by Arthur Samuel (IBM), a pioneer in the field of computer gaming and artificial intelligence
- Soon veers towards algorithms for pattern classification, as in
  - Nils Nilsson's *Learning Machines* (1965), trying to make MLPs learn (no success)
  - Richard Duda and Peter Hart's *Pattern Recognition and Scene Analysis* (1973), with a large presence of statistical analysis
- T. M. Mitchell's book definition (1997): *Machine Learning is the study of computer algorithms that improve automatically through experience*
- Subtle shift: ML is no longer seen from a cognitive perspective but, instead, from an operational one
- Great impulse: the **second generation** of MLPs

Connectionism Rebirth

- New proposal around 1985 to define MLP learning as a form of **error minimization**
- This leads to the modern redefinition of **Machine Learning**
  - The machines: **mathematical input–output processes** that lend themselves to some form of **parameterization**
  - The learning process: **adjust the machine's parameters** until a goal is reached
  - The goal: to **minimize some error measure**
- Summing things up: a ML process tries to find a concrete mathematical/algorithmic **input–output, parameterized transformation**

  $$f(x, w)$$

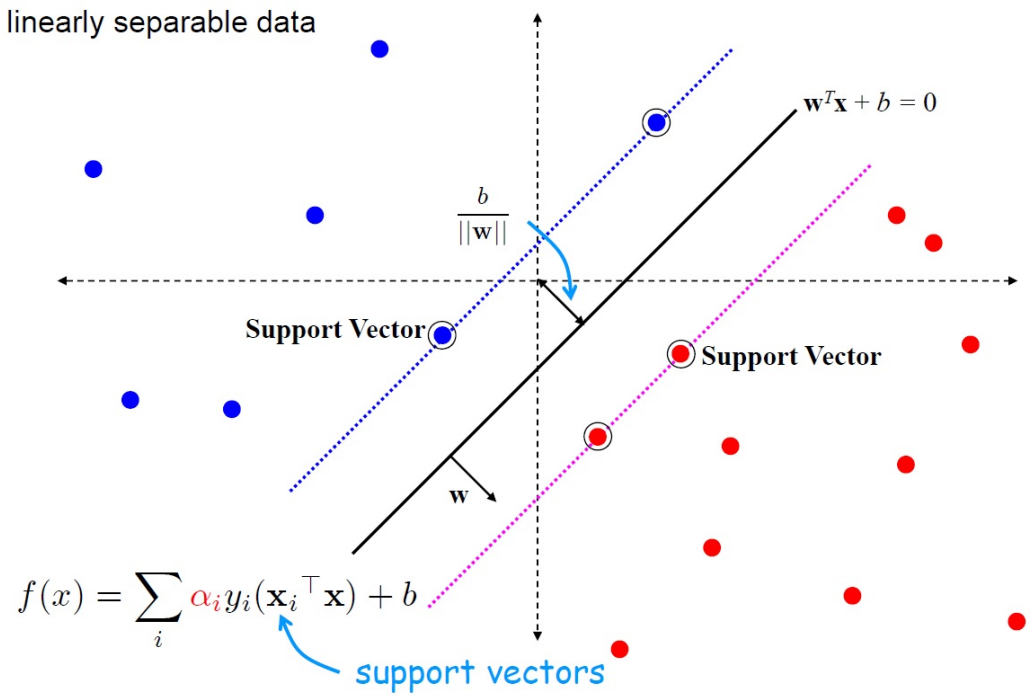
  that **minimizes an error measure** by iteratively **adjusting the transformation's parameters** w

The Evolution of ML

- MLPs have an enormous success as they provide a way of building useful **supervised** models in a **semi automatic** fashion
- Key ingredient: **backpropagation** computation of error gradients
- Although they only had one or (rarely) two layers, they were applied to many problems in different fields
- In my humble experience:
  - Credit card fraud detection (classification)
  - Manuscript character recognition (classification)
  - Renewable energy prediction (regression)
- After a while, MLPs slacked a bit and new, competing proposals appeared:
  - **Support Vector Machines** (around 1995)
  - **Random Forests** (around 2000)
  - **Gradient Boosting** (around 2000)

Support Vector Machines

- Goal: to achieve better generalization by maximizing separation margins

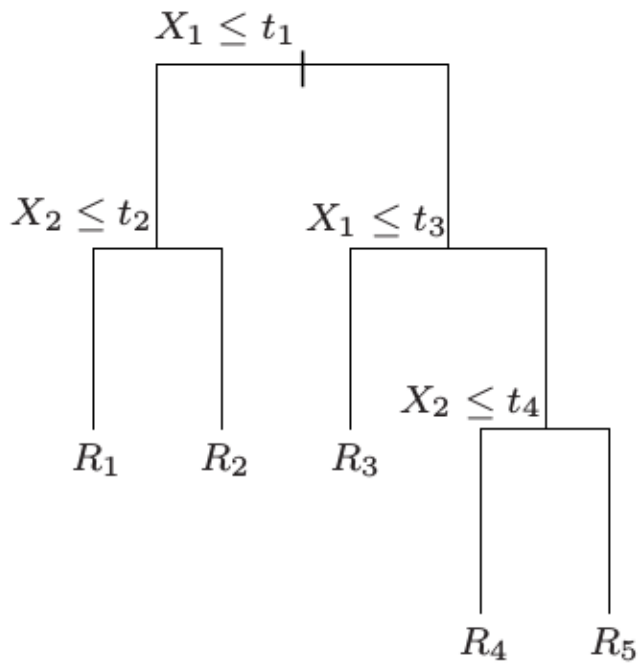


From A. Zisserman, C19 Machine Learning, Oxford University

- Key non linear extension through the use of **kernels**

Decision Trees

- Random Forests and Gradient Boosting rely on **decision trees**
- They are built following a sample split procedure along selected variables

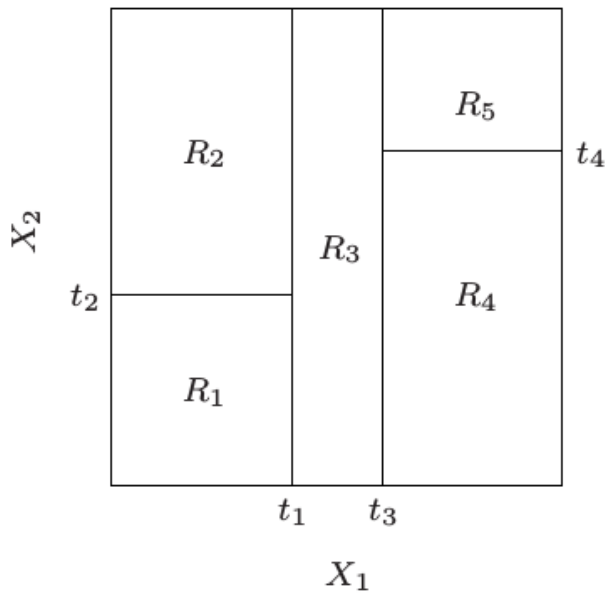


Taken from James et al., *An Introduction to Statistical Learning*

- Variables and split values are selected according to some **gain criterion**

Regression Trees

- As a result, these splits divide feature space into rectangular regions



Taken from James et al., *An Introduction to Statistical Learning*

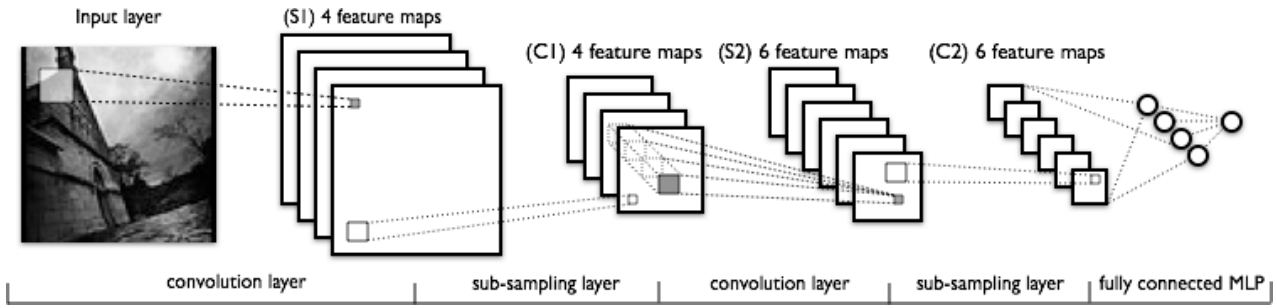
- In regression trees a **single prediction value** is assigned to each region R
  - Usual choices: the average value of y^p or the output of a linear regression model over the samples $x^p \in R$

Random Forest Regression

- Single trees may not yield good models
- In Random Forest Regression
  - Many trees T_k are **randomly** built (the **forest**)
  - The final model is their average $\mathcal{T}_M(x) = \frac{1}{M}\sum_1^M T_k(x)$
- The forest's trees must also be built **independently**
- Randomness and independence are achieved by
  - Randomly and independently select a subsample when building each tree
  - Randomly and independently select a subset of features for each split
- Pros: good, easy to build models which allow some interpretation of variable relevance; can handle categorical features
- Cons: random final model, several hyper–parameters to be tuned

The Boom of Deep Networks

- But a **third generation** of MLPs reappeared around 2010 stronger than ever as **Deep Neural Networks**
- An example: the LeNet architecture for image processing



From Convolutional Neural Networks (LeNet) tutorial

- And the floodgates opened:
  - Starting in 2012, major breakthroughs were achieved in significant problems in computer vision and speech recognition
  - A **new mood** arose: what was impossible before is now fairly easy and leads to state of the art results

Why?

- First, a much better understanding of operational issues: weight initialization, unit activations, optimization algorithms, regularization
- Also the use of GPUs (and TPUs) for faster matrix-vector multiplications
- But also the great flexibility allowed by **automatic differentiation**: backpropagation can be applied to essentially any network architecture and error function
  - The 1985 MLPs were essentially classical **linear and logistic regression algorithms** applied to the (hopefully better) **last hidden layer representations**
  - Modern NNs extend this enormously
- And learning cost is essentially **linear with respect to sample size**
  - Indispensable on the current Big Data era
  - But there is a ton of very interesting middle/small data problems

A Comparison

- Evolution of problem sizes and memory and hardware requirements

Decade	Dataset	Memory	FPC/second
1970	100 (Iris)	1 KB	100 KF (Intel 8080)
1980	1 K (Boston housing)	100 KB	1 MF (Intel 80186)
1990	10 K (MNIST)	10 MB	10 MF (Intel 80486)
2000	10 M (web pages)	100 MB	1 GF (Intel Core)
2010	10 G (advertising)	1 GB	1 TF (Nvidia C2050; GPU)
2020	1 T (social network)	100 GB	1 PF (Nvidia DGX-2; GPGPU)

Taken from Dive into Deep Learning, Zhang et al.

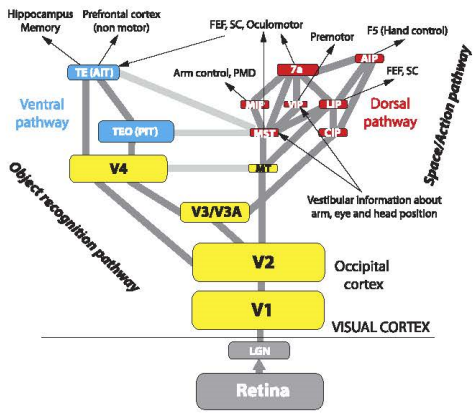
- Consequence: the top ML performers evolve from linear models to kernel methods to deep learning

The Latest Triumphs

- **Generative Pre-trained Transformer 3 (GPT-3)**, a language model introduced by Open AI that uses deep learning to produce human-like text
  - Huge size: 175 billion weights (according to Wikipedia)
  - According to The New York Times, it is "amazing", "spooky", and "humbling", but also "more than a little terrifying"
- **Alpha Go**, developed by Deep Mind-Google, beat the world's best Go player
  - It self-learned by playing against itself
  - Deep Mind then released Alpha Zero that not only beat Alpha Go at Go but then learned to play chess and beat the leading computer programs (and humans too, of course)
- **Self driving cars**, by Waymo and the top car makers, involving deep learning but many other technologies
  - Levels from 0 to 5 considered, with Level 0 the standard human driving of today
  - Level 4: human supervised self driving; considered as achievable
  - Level 5: full autonomous self driving; not so close

The Ideal Deep Net

- Plausible goal: train (teach?) networks to process information in a hierarchical way
- Model to pursue: the layered information processing and actuation in the human visual cortex



From Kruger et al., PAMI 35, 2013

- Ultimate goal: to replicate the cortex's workings to
  - Decompose a complex tasks in elementary subtasks
  - Solve each one separately and
  - Merge these subsolutions on a complex and rich representation

ML Everywhere!!

- Right now, AI is often seen as reduced to ML, which is neither exact nor fair, and we are still far away from general human intelligence
- But the current “agnostic" approach is resulting in increasingly complex, powerful and very useful systems
  - There seems to be an ML-based solution for any problem!!
- The current ML success has produced three important but difficult new requirements:
  - **Accountability**: indispensable
  - **Transparency**: perhaps not so easy with the current leading algorithms
  - **Fairness**, where the burden lies not only on algorithms but on data and, ultimately, on people
- These are very much open questions where the concrete issues have yet to be properly identified and, more so, addressed

Accountability, Transparency, Fairness

- Accountability: must an algorithm be **held accountable** on its decisions?
  - At first sight, possibly no, but think about self driving cars causing an accident
- Transparency: try to explain **how** an algorithm works and **why** it has made a concrete decision
  - The how may be easy in some algorithms (decision trees), but much harder in others (DNNs)
  - And the why is always hard: why did you choose the tree splits in a concrete way?
- Fairness is desirable always but essential in some applications
  - A clear example is the COMPAS recidivism risk score
  - It touches on almost all aspects of algorithm design, construction and exploitation
  - As well as on accountability and transparency!

What Next?

Renewing The AI Conversation

- Old thinking: pile up a large number of decision rules (in symbolic AI) or artificial neurons (in connectionist AI) and general artificial intelligence will just ignite
  - This has been essentially abandoned by now
- New thinking: build intelligent systems by building first specialized ML models for layered modules and develop then supervising programs that make them work together
- Perhaps the clearest example is autonomous driving systems which are made up of
  - Intelligent sensors
  - Intelligent data interpretation models
  - Overall supervising and decision modules
- That is, systems with a predominance of diverse ML modules coupled with logic decision subsystems
  - This is certainly disruptive, but much more so is the non stop and very wide automation of industrial and (increasingly) service processes
  - Very likely with important economic and social disruptions

What Next?

- Hard to tell, but we may look at two examples: OpenAI, NIPS 2015 Symposium
- OpenAI: *... to advance digital intelligence ... to benefit humanity ... unconstrained by ... financial return ...*
  - Chief scientist: I. Sutskever (U. Toronto, U. Stanford, Google)
  - Sponsors: Elon Musk (Tesla), Reid Hoffman (LinkedIn), Peter Thiel (PayPal)
  - Up to 1 billion dollars pledged
- NIPS 2015 symposium Algorithms Among Us: The Societal Impacts of Machine Learning, with among others
  - Nick Bostrom, Future of Humanity Institute–Oxford U.
  - Andrew Ng, Stanford–Coursera–Baidu, The Economic Impact of Machine Learning (podcast)
  - Erik Brynjolfsson, MIT, The Second Machine Age
- We'll see!!!