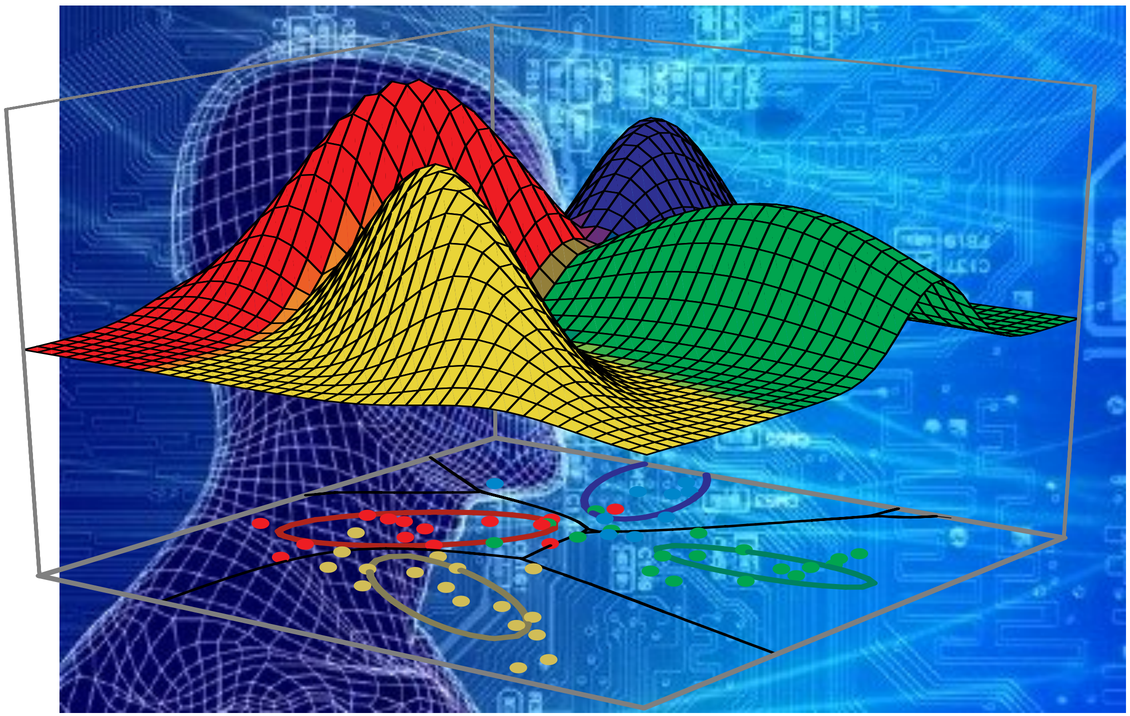
SDN and Machine Learning

A (Very) Brief Introduction to Machine Learning and an SDN Application



Agenda

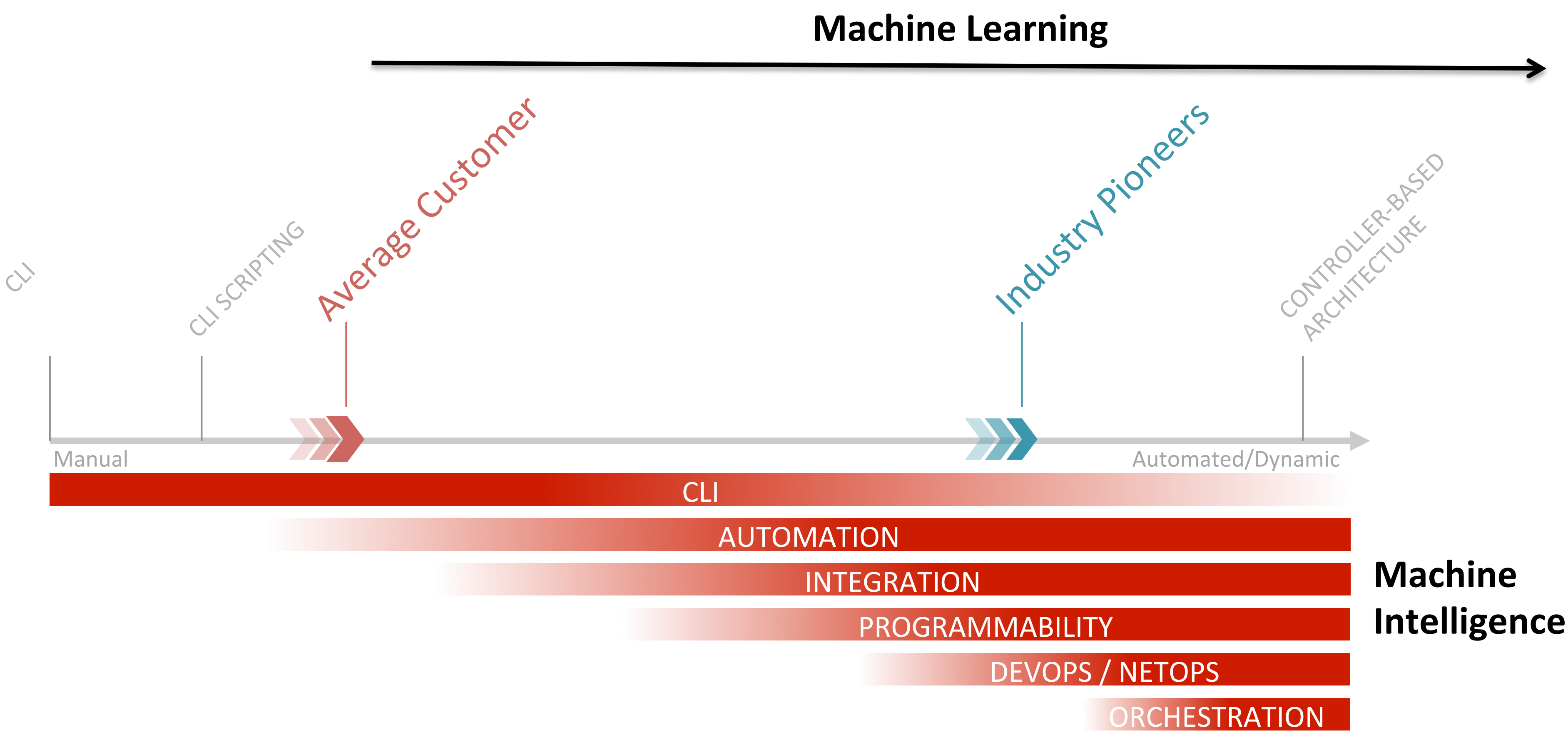
- Goals for this Session
- What is Machine Learning?
  - And how can it possibly work?
- Shallow Dive Into Deep Neural Net Technology
- PCE?
- Q&A

Goals for this Session

To cut through some of the Machine Learning (ML) hype and give us a basic common understanding of ML so that we can discuss its application to our use cases of interest.

So, remembering our architecture…

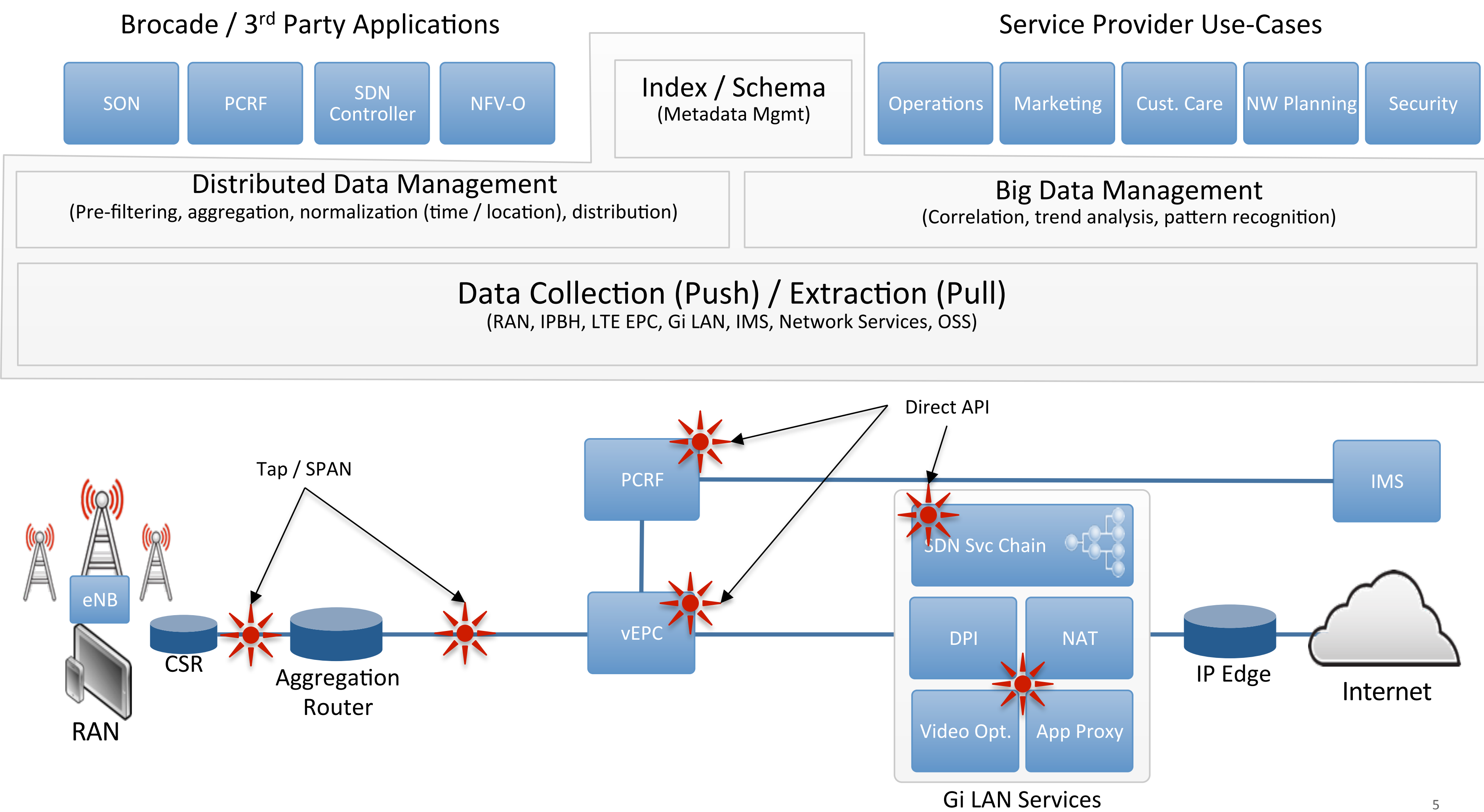
Another Way To Think About This Automation Continuum



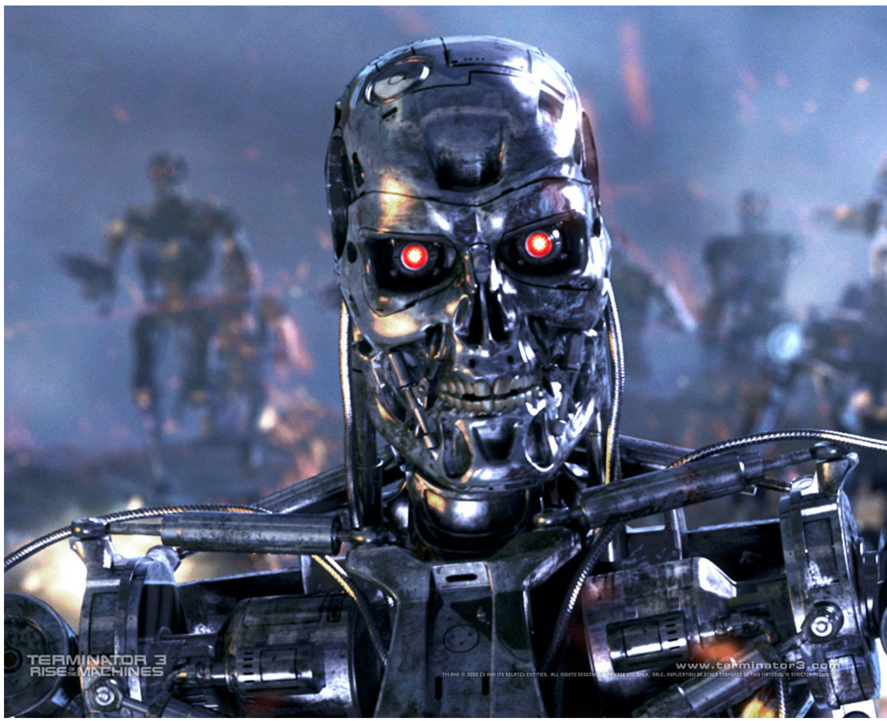
Original slide courtesy Mike Bushong and Joshua Soto

What Might a Analytics Platform Look Like? (Mobile)

Think “Platform”, not Applications, Algorithm, Visualization

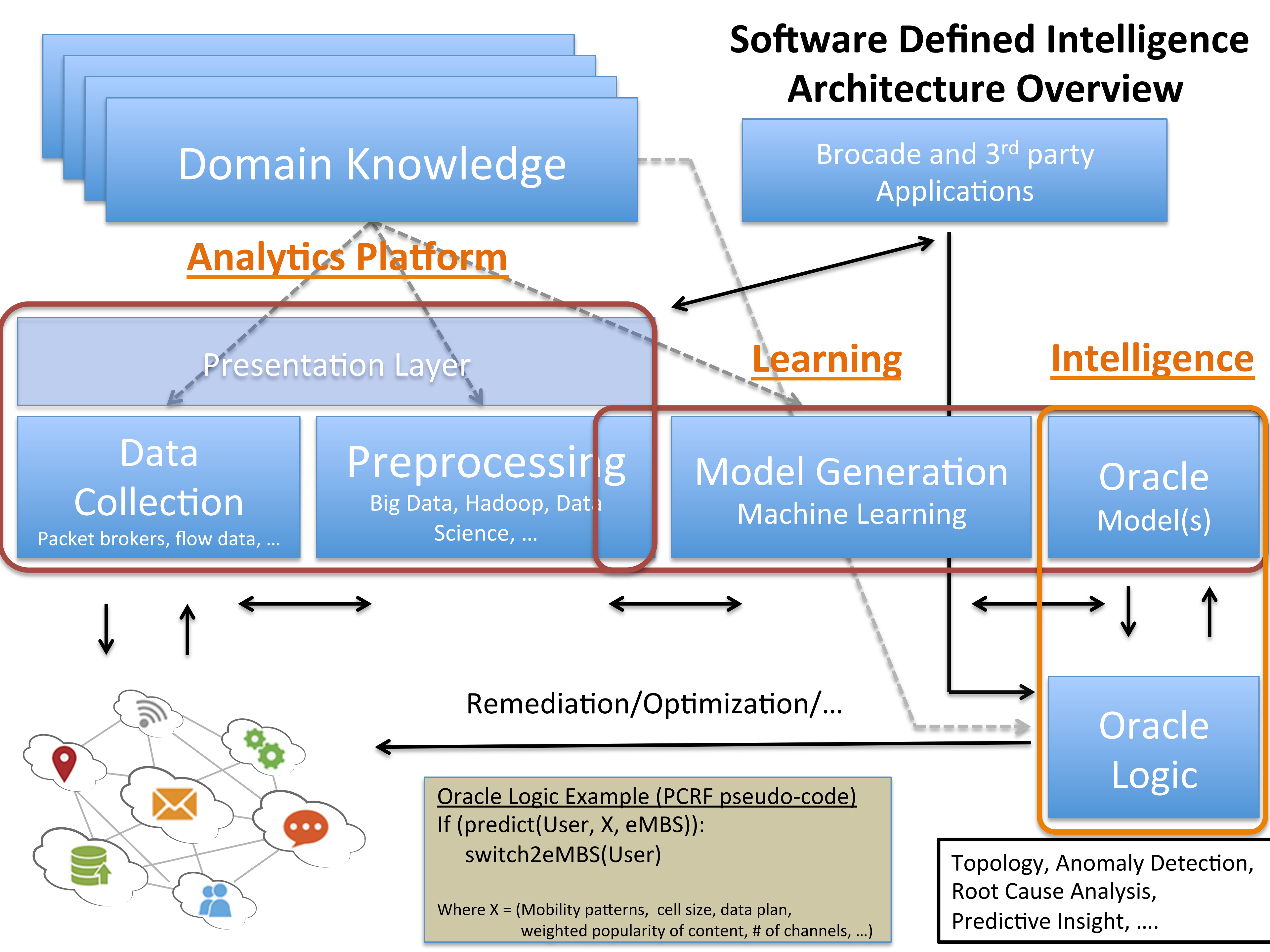


Where I Want To Go With This



Not This

What Might an Architecture for this Look Like?


Software Defined Intelligence
Architecture Overview
Domain Knowledge
Brocade and 3rd party
Applications
Analytics Platform
Presentation Layer
Learning
Intelligence
Data
Collection
Packet brokers, flow data, …
Preprocessing
Big Data, Hadoop, Data
Science, …
Model Generation
Machine Learning
Oracle
Model(s)
Remediation/Optimization/…
Oracle
Logic
Oracle Logic Example (PCRF pseudo-code)
If (predict(User, X, eMBS)):
switch2eMBS(User)
Where X = (Mobility patterns, cell size, data plan,
weighted popularity of content, # of channels, …)
Topology, Anomaly Detection,
Root Cause Analysis,
Predictive Insight, ….

Machine Intelligence LANDSCAPE

CORE TECHNOLOGIES

ARTIFICIAL INTELLIGENCE
IBM WATSON, MetaMind, Numenta, ai-one, Cycorp, Microsoft Research, nara, Reactor, Scaled Inference

DEEP LEARNING
vicarious, Vision Factory, facebook, Baidu IDL, Google, ersatz, SKYMIND, SignalSense

MACHINE LEARNING
rapidminer, context relevant, Oxdata H_2O, DATARPM, Liftignite, Spark Beyond, Azure ML, yhat, Wise.io, Sense, GraphLab, Alpine, Ayasdi

NLP PLATFORMS
cortical.io, idibon, LUMINOSO, wit.ai, Maluuba

PREDICTIVE APIS
AlchemyAPI, MINDOPS, Google, bigml, indico, ALGORITHMIA, Expect Labs, PredictionIO

IMAGE RECOGNITION
clarifai, MADBITS, DNNresearch, DEXTRO, VISENZE, lookflow

SPEECH RECOGNITION
GRIDSPACE, popup archive, NUANCE

RETHINKING ENTERPRISE

SALES
Preact, AVISO, RelateIQ, NGDATA, Clarabridge, FRAMED, infer, Attensity, causata

SECURITY / AUTHENTICATION
CROSSMATCH, conjur, EYEVERIFY, BITSIGHT, CYLANCE, bionym

FRAUD DETECTION
sift science, socure, ThreatMetrix, feedzai, Brighterion, verafin

HR / RECRUITING
TalentBin, entelo, predikt, Connectifier, gild, hiQ, Concept Node

MARKETING
brightfunnel, bloomreach, CommandIQ, AIRPR, RADIUS, TellApart, people pattern, Freshplum

PERSONAL ASSISTANT
Siri, x.ai, Google now, Cortana, cleversense, tempo, Robinlabs, KASISTO, fuse machines, VIV, CLARA LABS

INTELLIGENCE TOOLS
ADATAO, Palantir, Quid, Digital Reasoning, FirstRain

RETHINKING INDUSTRIES

ADTECH
METAMARKETS, dstillery, rocketfuel, YieldMo, ADBRAIN

AGRICULTURE
BLUE RIVER, TerrAvion, ceresimaging, HONEYCOMB, THE CLIMATE CORPORATION, tule

EDUCATION
declara, coursera, KNEWTON, kidaptive

FINANCE
Bloomberg, FinGenius, alphasense, KENSHO, Dataminr, minetta brook, BINATIX

LEGAL
Lex Machina, brightleaf, COUNSELYTICS, RAVEL, JUDICATA, Brevia, DiligenceEngine

MANUFACTURING
SIGHT MACHINE, MICROSCAN, BOULDER IMAGING, IVISYS

MEDICAL
Parzival, transcriptic, Genescient, ZEPHYR HEALTH, grand round table, bina, TUTE GENOMICS

OIL AND GAS
kaggle, AYASDI, TACHYUS, biota, Flutura

MEDIA / CONTENT
Outbrain, newsle, ARRIA, SAILTHRU, wavii, Owlin, NarrativeScience, yseop, Summly, Prismatic, ai

CONSUMER FINANCE
Affirm, inVenture, zestfinance, BILLGUARD, LendUp, LendingClub, Kabbage

PHILANTHROPIES
DataKind, thorn, DATA GUILD

AUTOMOTIVE
Google, Continental, TESLA, MOBILEYE, CRUISE

DIAGNOSTICS
enlitic, 3SCAN, lumiata, ENTOPSIS

RETAIL
BAY SENSORS, PRISM SKYLABS, celect, euclid

RETHINKING HUMANS / HCI

AUGMENTED REALITY
wearable intelligence, APX, blippar, META, layar

GESTURAL COMPUTING
THALMIC LABS, omek, FLUTTER, LEAP MOTION, eyeSight, 3Gear Systems, GestureTek, nod

ROBOTICS
intel, LIQUID ROBOTICS, SoftBank, iRobot, Boston Dynamics, jibo, anki, evolution robotics

EMOTIONAL RECOGNITION
affectiva, BEYOND VERBAL, EMOTIENT, cogito

SUPPORTING TECHNOLOGIES

HARDWARE
NVIDIA, XILINX, QUALCOMM, NERVANA SYSTEMS, TERADEEP, Artificial Learning, rigetti

DATA PREP
TRIFACTA, Paxata, tamr, Alation

DATA COLLECTION
diffbot, kimono, CrowdFlower, Connotate, WorkFusion, import.io

Agenda

- ~~Goals for this Session~~
- What is Machine Learning?
  - And how can it possibly work?
- Shallow Dive Into Deep Neural Net Technology
- Google PUE Use Case
- Q&A

Before We Start

What is the SOTA in Machine Learning?

- "Building High-level Features Using Large S… Andrew Ng, et. al, 2012
  - http://arxiv.org/pdf/1112.6209 p…
  - Training a *deep neural ne*…
  - Showed that it is p… entirely *unla*…
  - In par…
  - …out-of-plane
  - …you are interested what this is/how it
  - …ages, 10^3 machines
  - …R^{40000}
  - …ays to train
  - …% accuracy categorizing 22K object cla…
    - 70% improvement over current results
    - Random guess achieves less than 0.005% a…

Andrew Ng and his crew at Baidu have recently beat this record with their (GPU based) Deep Speech system. See http://arxiv.org/abs/1412.5567

1 http://www.image-net.org/

What is Machine Learning?

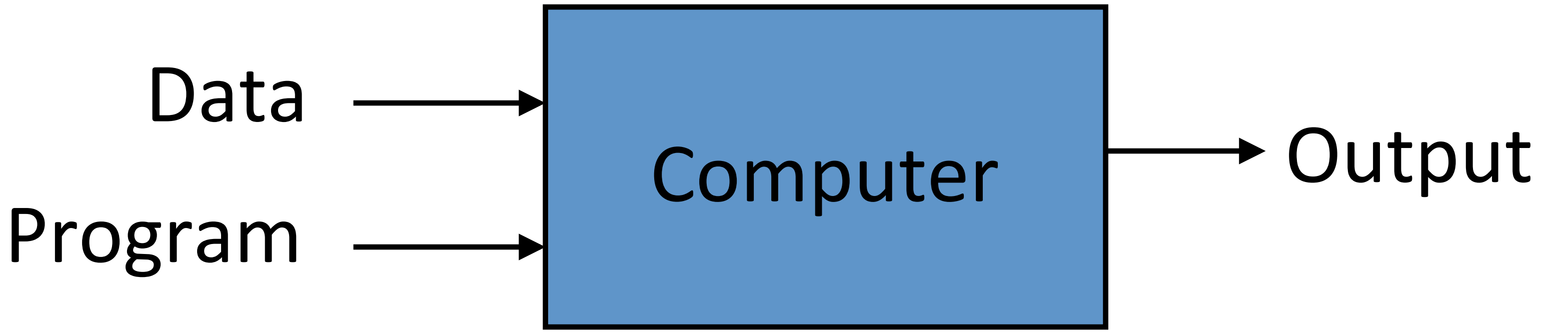
The complexity in traditional computer programming is in the code (programs that people write). In machine learning, algorithms (programs) are in principle simple and the complexity (structure) is in the data. Is there a way that we can automatically learn that structure? That is what is at the heart of machine learning.

-- Andrew Ng

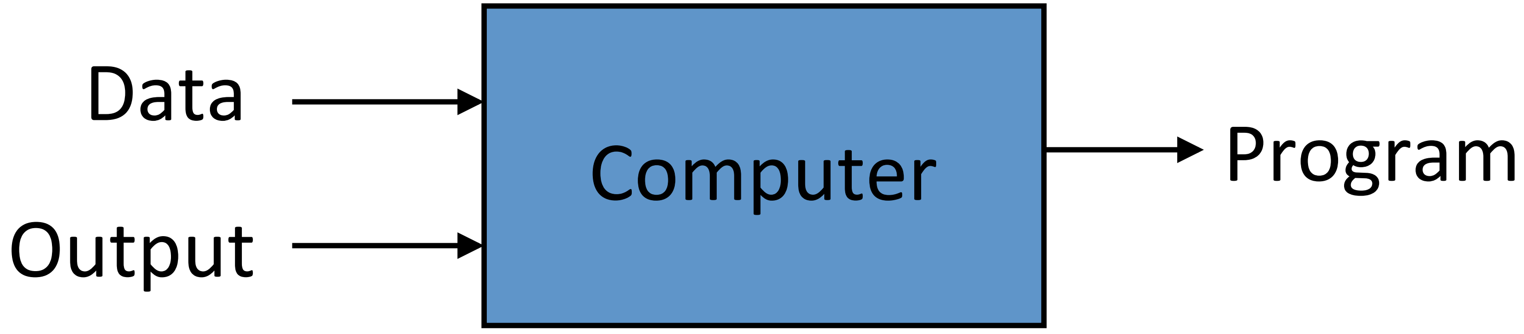
That is, machine learning is the about the construction and study of systems that can learn from data. This is very different than traditional computer programming.

The Same Thing Said in Cartoon Form

Traditional Programming



Machine Learning



When Would We Use Machine Learning?

- When patterns exists in our data
  - Even if we don't know what they are
    - Or perhaps especially when we don't know what they are
- We can not pin down the functional relationships mathematically
  - Else we would just code up the algorithm
- When we have lots of (unlabeled) data
  - Labeled training sets harder to come by
  - Data is of high-dimension
    - High dimension "features"
    - For example, sensor data
  - Want to "discover" lower-dimension representations
    - Dimension reduction
- Aside: Machine Learning is heavily focused on implementability
  - Frequently using well know numerical optimization techniques
  - Lots of open source code available
    - See e.g., libsvm (Support Vector Machines): http://www.csie.ntu.edu.tw/~cjlin/libsvm/
    - Most of my code in python: http://scikit-learn.org/stable/ (many others)
    - Languages (e.g., octave: https://www.gnu.org/software/octave/)

Aside: NVIDA

NVIDIA CUDA ZONE | Getting Started | Downloads | Training | Ecosystem | Register Now | Login

PARALLEL FORALL | Features | CUDACasts | CUDA Pro Tips | CUDA Spotlights

← Previous Next →

Understanding Natural Language with Deep Neural Networks Using Torch

Share:

Posted on March 3, 2015 by Soumith Chintala | 2 Comments
Tagged cuDNN, Deep Learning, Machine Learning, Natural Language Processing

This post was co-written by Soumith Chintala and Wojciech Zaremba of Facebook AI Research.

Language is the medium of human communication. Giving machines the ability to learn and understand language enables products and possibilities that are not imaginable today.

One can understand language at varying granularities. When you learn a new language, you start with words: understanding their meaning, identifying similar and dissimilar words, and developing a sense of contextual appropriateness of a word. You start with a small dictionary of words, building up your dictionary over time, mentally mapping each newly learned word close to similar words in your dictionary. Once you get familiar with your dictionary of words, you put them together into small sentences, learning grammar and structure. You eventually combine sentences in a sensible way, to write paragraphs and pages. Once you get to this stage, you are comfortable with expressing complicated thoughts in language, letting others understand your thoughts and expression.

SUBSCRIBE: RSS EMAIL

CONNECT: Follow @gpucomputing

RESOURCES

- About Parallel Forall
- NVIDIA Developer Forums
- CUDA Newsletter

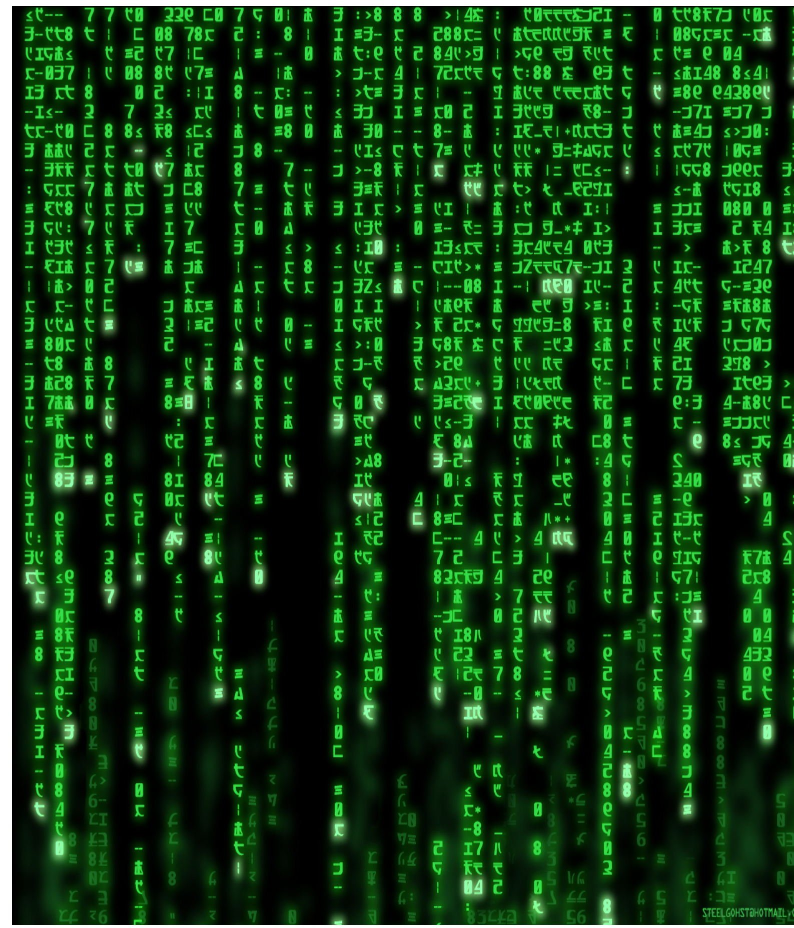
RECENT POSTS

- GPU-Accelerated Graph Analytics in Python with Numba

Why Machine Learning is Hard?

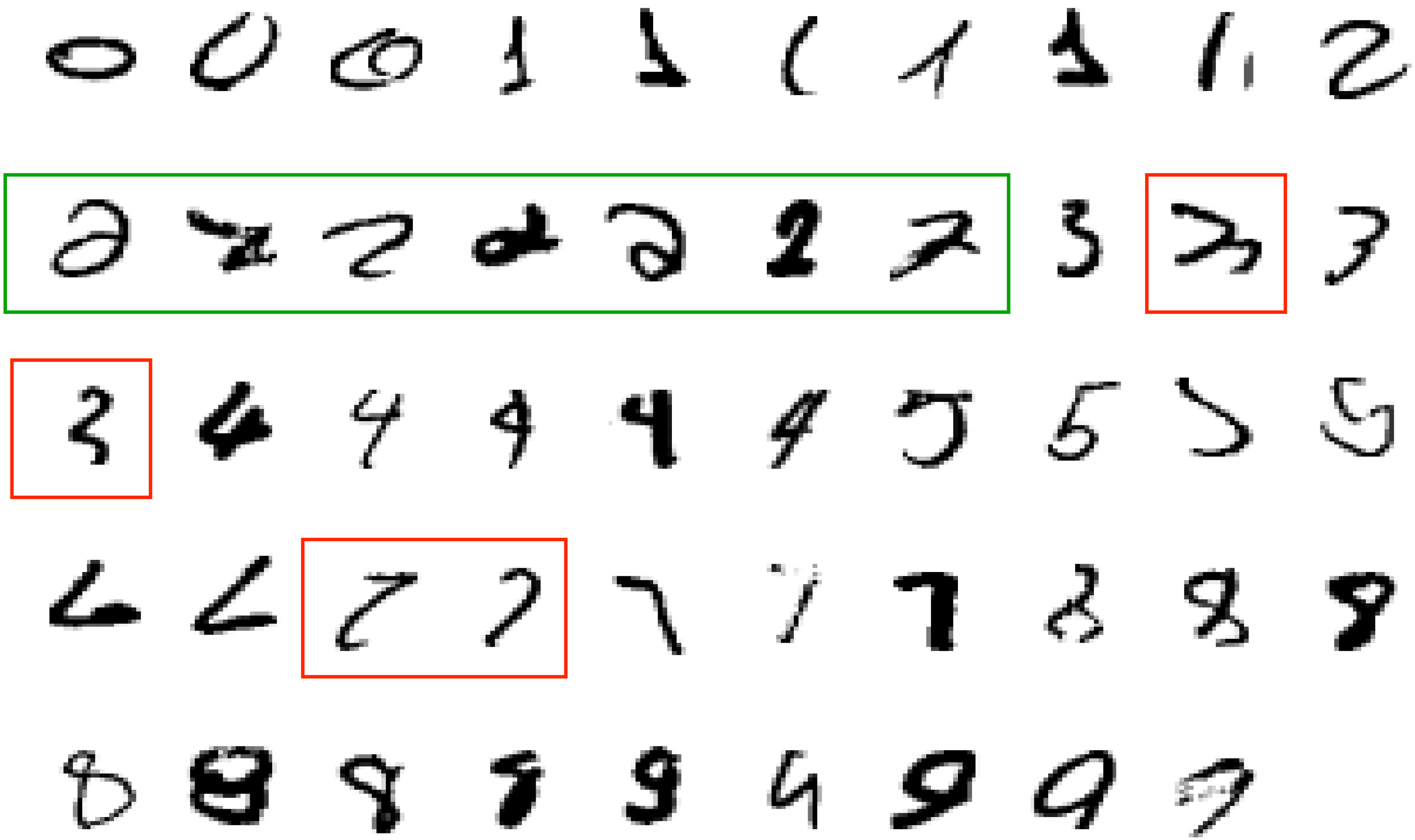
You See

Your ML Algorithm Sees
A Bunch of Bits



Why Machine Learning Is Hard, Redux

What is a "2"?



Examples of Machine Learning Problems

- Pattern Recognition
  - Facial identities or facial expressions
  - Handwritten or spoken words (e.g., Siri)
  - Medical images
  - Sensor Data/IoT
- Optimization
  - Many parameters have "hidden" relationships that can be the basis of optimization
- Pattern Generation
  - Generating images or motion sequences
- Anomaly Detection
  - Unusual patterns in the telemetry from physical and/or virtual plants (e.g., data centers)
  - Unusual sequences of credit card transactions
  - Unusual patterns of sensor data from a nuclear power plant
    - or unusual sound in your car engine or …
- Prediction
  - Future stock prices or currency exchange rates
  - Network events
  - …

Finally, note that ML is a form of Induction

- Given examples of a function *(x, f(x))*
  - *Don't explicitly know f*
    - Rather, trying to learn *f* from the data
  - *Labeled* training data set (i.e., the *f(x)*'s)
  - Training set will be noisy, e.g., $(x, (f(x) + \varepsilon))$
  - Notation: $(x_i, f(x_i))$ denoted $(x^{(i)}, y^{(i)})$
  - $y^{(i)}$ sometimes called t_i (t for "target")

- Predict function ***f(x)*** for new examples *x*
  - Discrimination/Prediction (Regression): *f(x)* continuous
  - Classification: *f(x)* discrete
  - Estimation: *f(x)* = P(Y = c|*x*) for some class c

Agenda

- ~~Goals for this Session~~
- ~~What is Machine Learning?~~
  - And how can it possibly work?
- Shallow Dive Into Deep Neural Net Technology
- PCE
- Q&A

How Can Machine Learning Possibly Work?

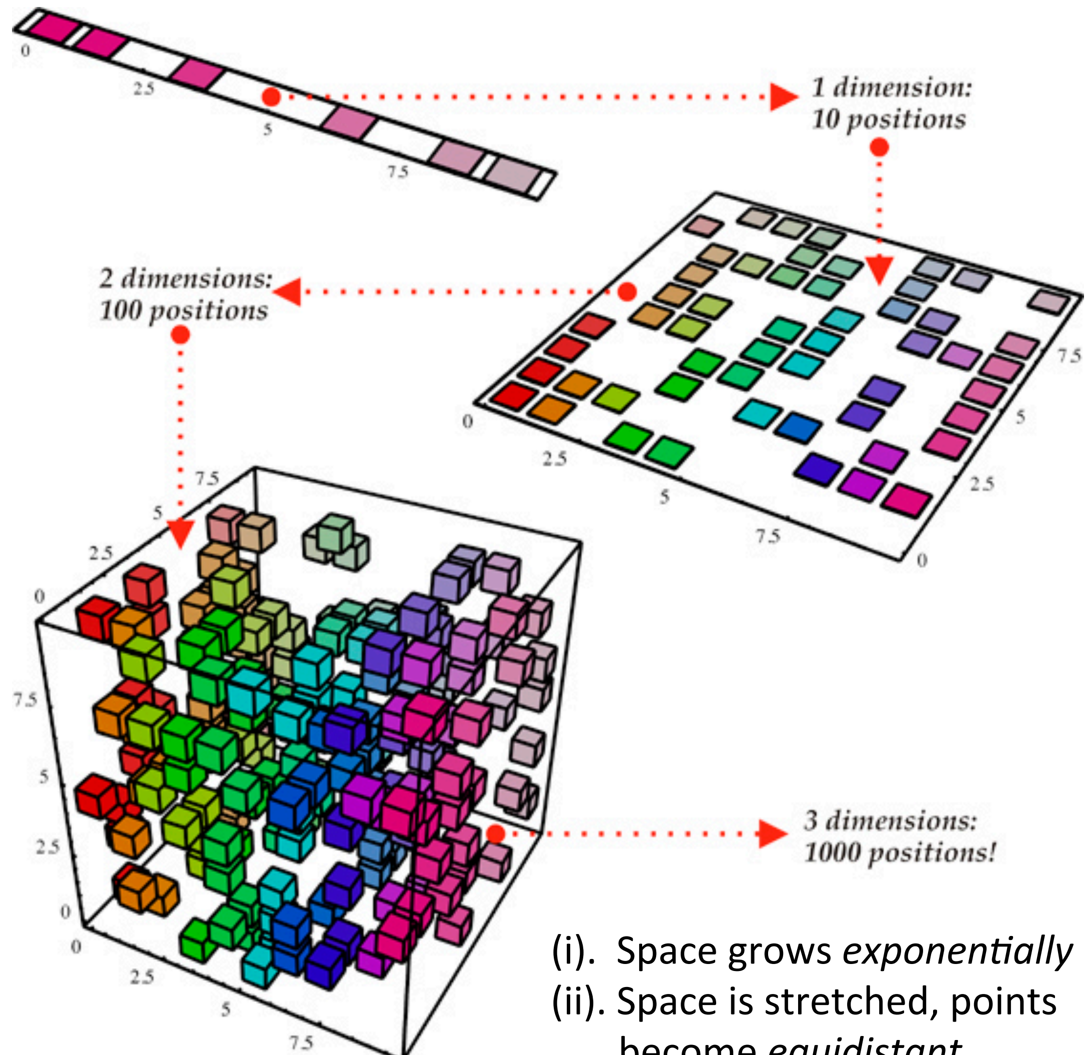
- We want to build statistical models that **generalize to unseen cases**
- What assumptions do we need to do this (essentially predict the future)?
- 4 main "prior" assumptions are (at least) required
  - Smoothness
  - Manifold Hypothesis
  - Distributed Representation/Compositionality
    - Compositionality is useful to describe the world around us efficiently → distributed representations (features) are meaningful by themselves.
    - Non-distributed → # of distinguishable regions linear in # of parameters
    - Distributed → # of distinguishable regions grows almost exponentially in # of parameters
      - Each parameter influences many regions, not just local neighbors
    - Want to generalize non-locally to never-seen regions → essentially ***exponential gain***
  - Shared Underlying Explanatory Factors
    - The assumption here is that there are shared underlying explanatory factors, in particular between p(x) (prior distribution) and p(Y|x) (posterior distribution). ***Disentangling*** these factors is in part what machine learning is about.
- Before this, however: What is the problem in the first place?

Why ML Is Hard
The Curse Of Dimensionality

- To generalize locally, you need representative examples from all relevant variations (and there are an *exponential* number of them)!

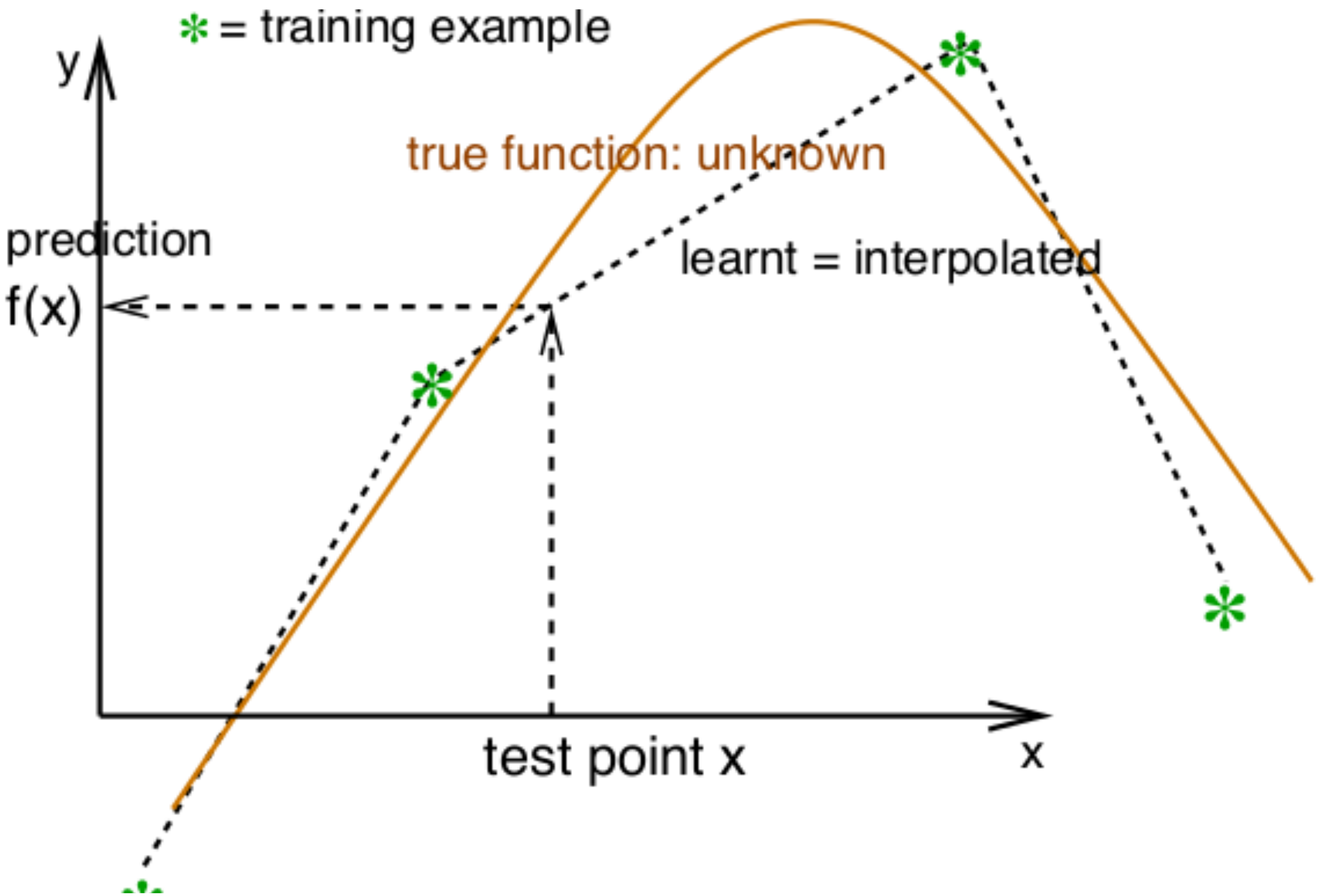
- Classical Solution: Hope for a smooth enough target function, or make it smooth by handcrafting good features or kernels

- *Smooth?*



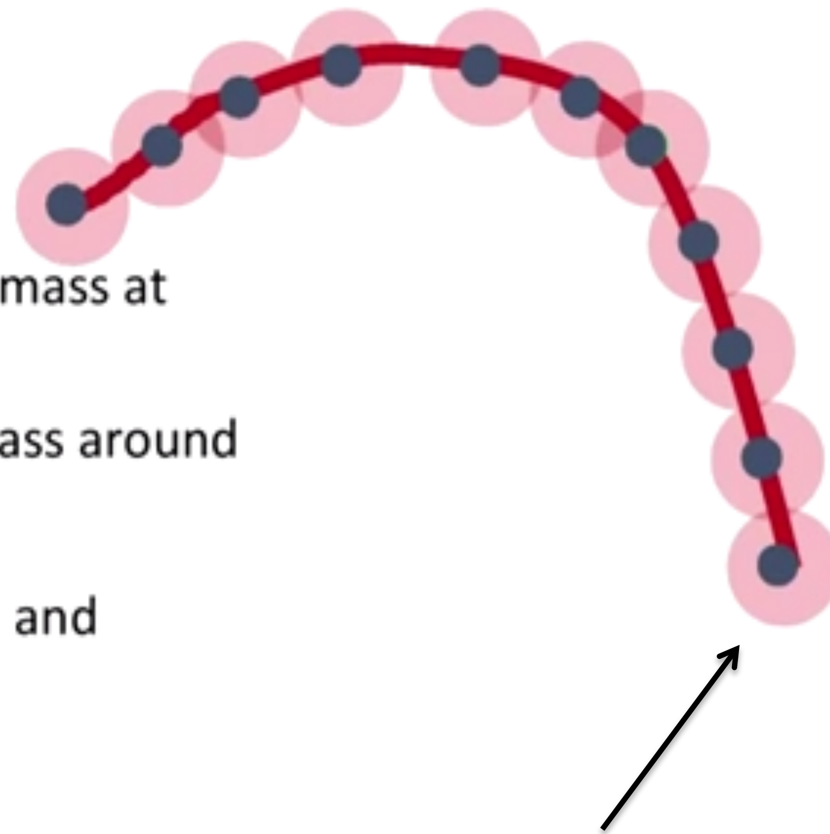
(i). Space grows *exponentially*
(ii). Space is stretched, points become *equidistant*

So What Is Smoothness?



Smoothness → If **x** is geometrically close to **x'** then **f(x) ≈ f(x')**

Smoothness, basically…

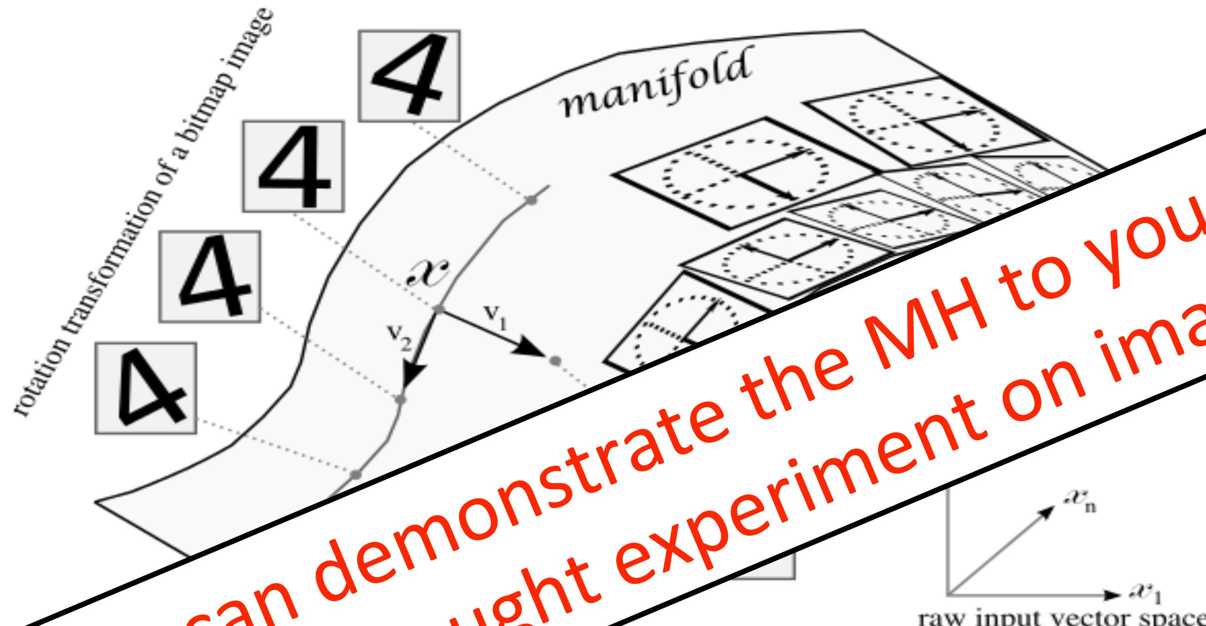


- Empirical distribution: mass at training examples
- Smoothness: spread mass around
- Insufficient
- Guess some ‘structure’ and generalize accordingly

Probability mass $\mathbf{P(Y=c|X;\theta)}$

This is where the *Manifold Hypothesis* comes in…

Manifold Hypothesis

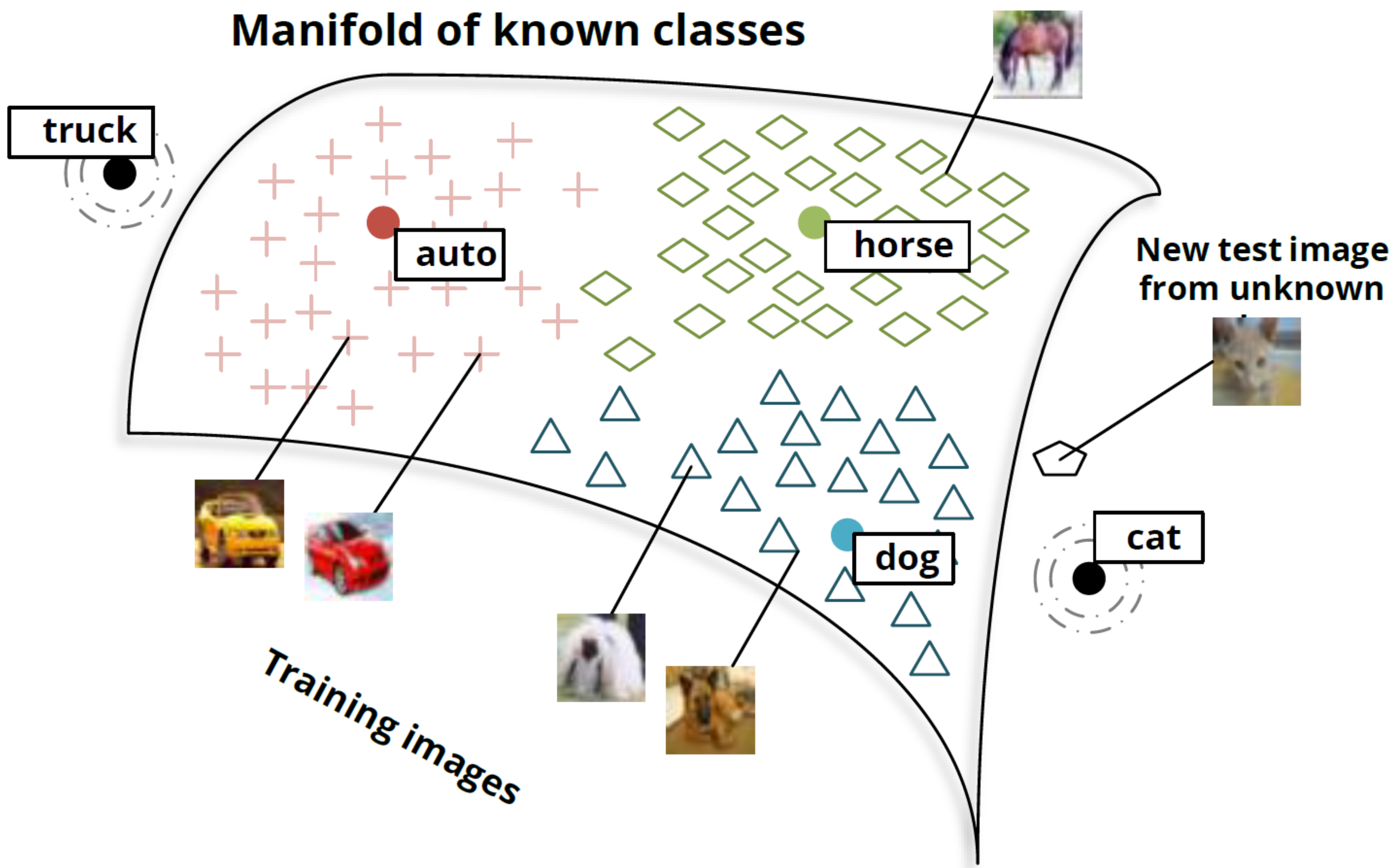


BTW, you can demonstrate the MH to yourself with a simple thought experiment on image data...

...pothesis states that ***natural data*** forms lower dimensional manifolds ...ding space. Why should this be? Well, it seems that there are both th...oretical and experimental reasons to suspect that the Manifold Hypothesis is true.

So if you believe that the MH is true, then the task of a machine learning classification algorithm is fundamentally to separate a bunch of tangled up manifolds.

Another View: Manifolds and Classes



Ok, Great. What Then Is Learning?

- Learning is a procedure that consists of estimating the model parameters so that the learned model (algorithm) can perform a specific task
  - In Artificial Neural Networks, these parameters are the *weight matrix* ($w_{i,j}$'s)

- 2 types of learning considered here
  - Supervised
  - Unsupervised
  - Semi-supervised learning
  - Reinforcement learning

- Supervised learning
  - Present the algorithm with a set of inputs and their corresponding outputs
  - See how closely the actual outputs match the desired ones
    - Note generalization error (bias, variance)
  - Iteratively modify the parameters to better approximate the desired outputs (gradient descent)

- Unsupervised
  - Algorithm learns internal representations and important features

- So let's take a closer look at these learning types

Supervised learning

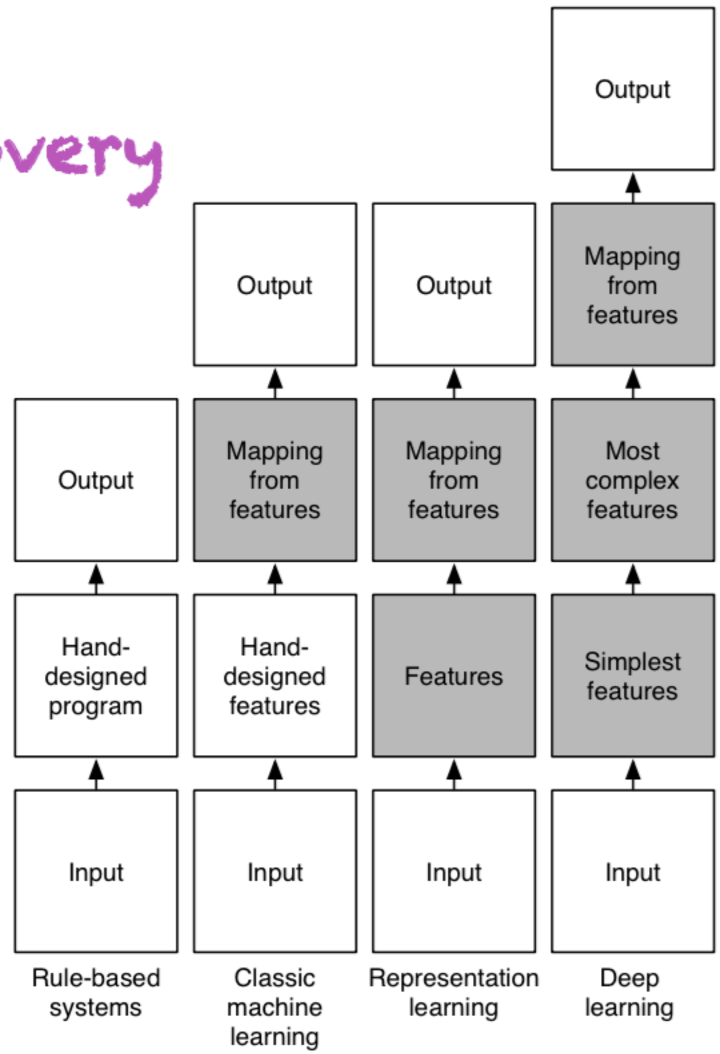
- The desired response (function) of given inputs is well known
  - You are given the “answer” (label) in the training set
  - Training data is set of $(x^{(i)},y^{(i)})$ pairs, $x^{(i)}$ is the input example, $y^{(i)}$ is the label
- There are many 10s (if not 100s or 1000s) of supervised learning learning algorithms
  - These include: Artificial Neural Networks, Decision Trees, Ensembles (Bagging, Boosting, Random Forests, …), k-NN, Linear Regression, Naive Bayes, Logistic Regression (and other CRFs), Support Vector Machines (and other Large Margin Classifiers), …
  - Focus on Artificial Neural Networks (ANNs) here
- The Google Data Center PUE example we will look at later uses supervised learning on a deep neural network

Unsupervised learning

- Basic idea: Discover unknown structure in input data
- Data clustering and dimension reduction
  - More generally: find the relationships/structure in the data set
- No need for labeled data
  - The network itself finds the correlations in the data
- Learning algorithms include (again, many algorithms)
  - K-Means Clustering
  - Auto-encoders/deep neural networks
  - Restricted Boltzmann Machines
    - Hopfield Networks
  - Sparse Encoders
  - …

Taxonomy of Learning Techniques

Automating
Feature Discovery



Slide courtesy Yoshua Bengio

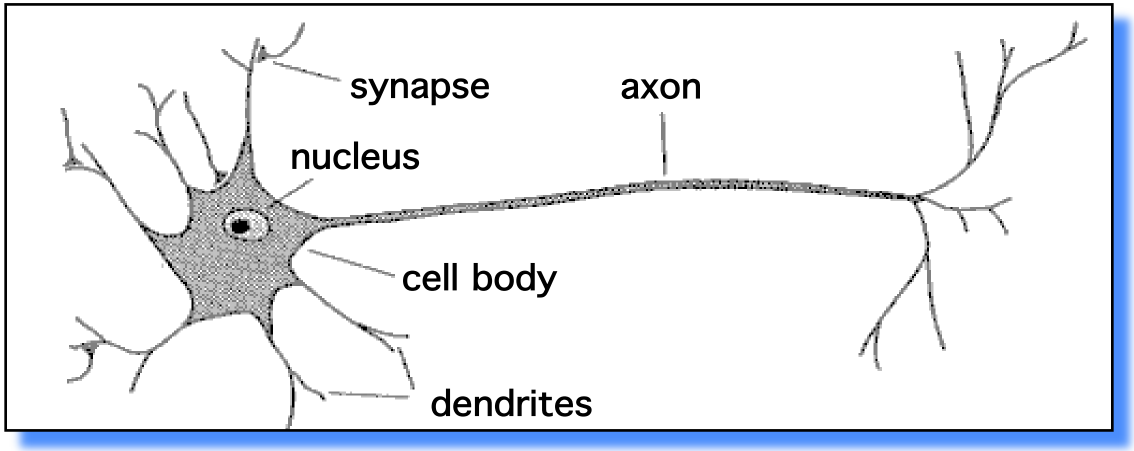
Artificial Neural Networks

- A Bit of History
- Biological Inspiration
- Artificial Neurons (AN)
- Artificial Neural Networks (ANN)
- ~~Computational Power of Single AN~~
- ~~Computational Power of an ANN~~
- Training an ANN -- Learning

Brief History of Neural Networks

- **1943:** McCulloch & Pitts show that neurons can be combined to construct a Turing machine (using ANDs, ORs, & NOTs)
- **1958:** Rosenblatt shows that perceptrons will converge if what they are trying to learn can be represented
- **1969:** Minsky & Papert showed the limitations of perceptrons, killing research for a decade
- **1985:** The backpropagation algorithm revitalizes the field
  - Geoff Hinton et al
- **2006:** The Hinton lab solves the training problem for DNNs

Biological Inspiration: Neurons



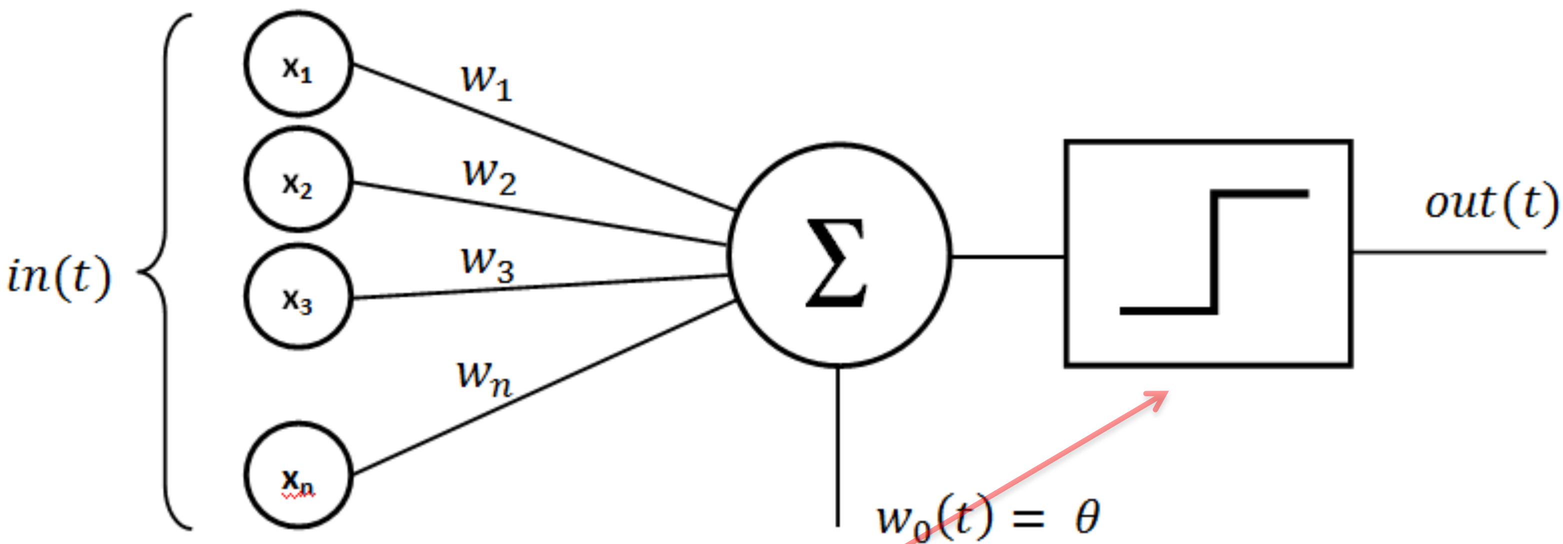
- A neuron has
  - Branching input (dendrites)
  - Branching output (the axon)

- Information moves from the dendrites to the axon via the cell body

- Axon connects to dendrites via synapses
  - Synapses vary in strength
  - Synapses may be excitatory or inhibitory

Basic Perceptron

(Rosenblatt, 1950s and early 60s)

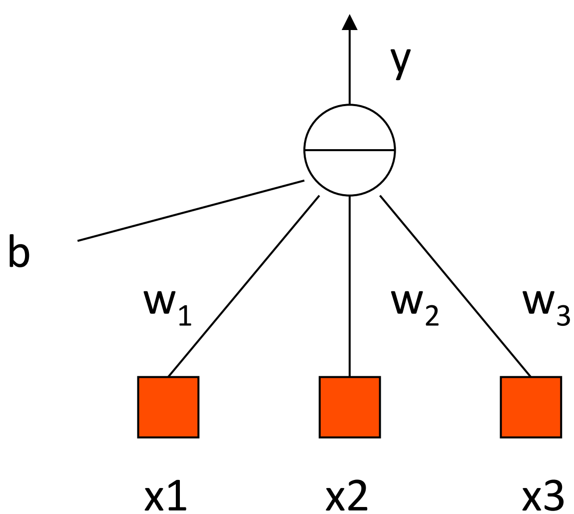


$$O = \begin{cases} 1 : \left(\sum_i w_i x_i \right) + b > 0 \\ 0 : otherwise \end{cases}$$

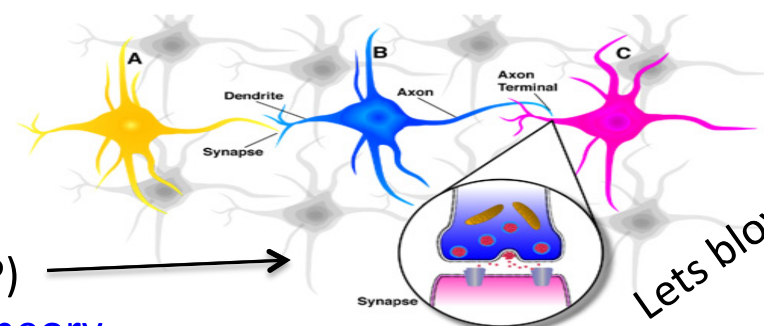
What was the problem here?

What is an Artificial Neuron?

- An Artificial Neuron (AN) is a non-linear parameterized function with restricted output range



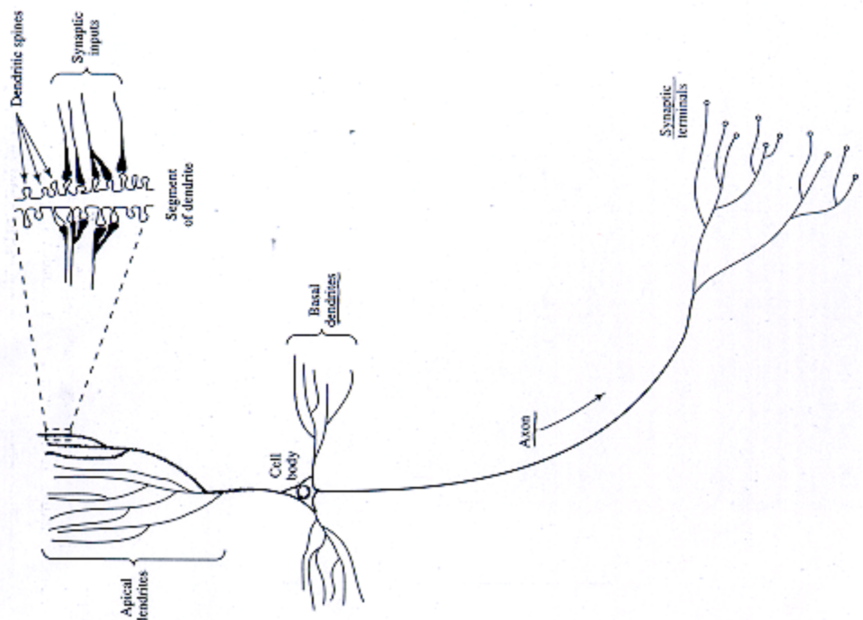
$$y = \boxed{f}\left(b + \sum_{i=1}^{n-1} w_i x_i\right)$$

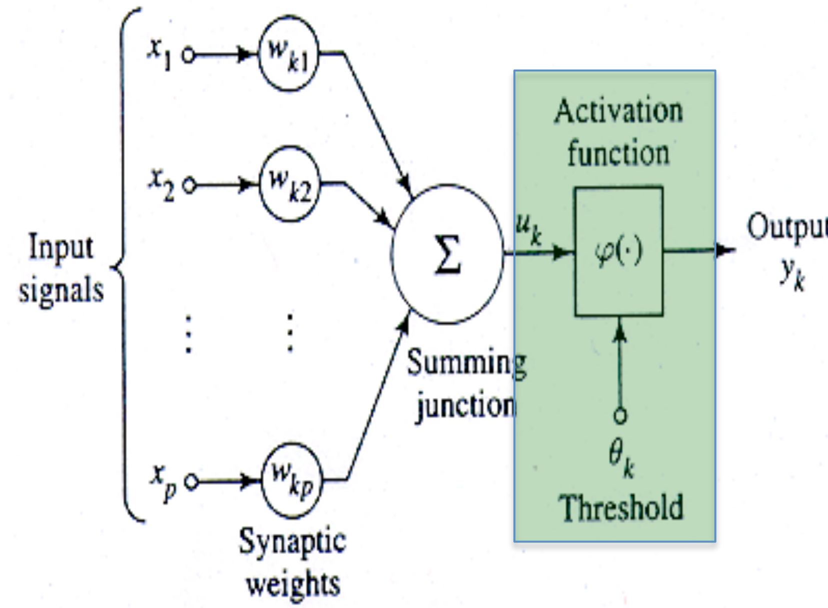


Lets blow this up a bit

Spike-Timing Dependent Plasticity (STDP)
http://en.wikipedia.org/wiki/Hebbian_theory

Mapping to Biological Neurons





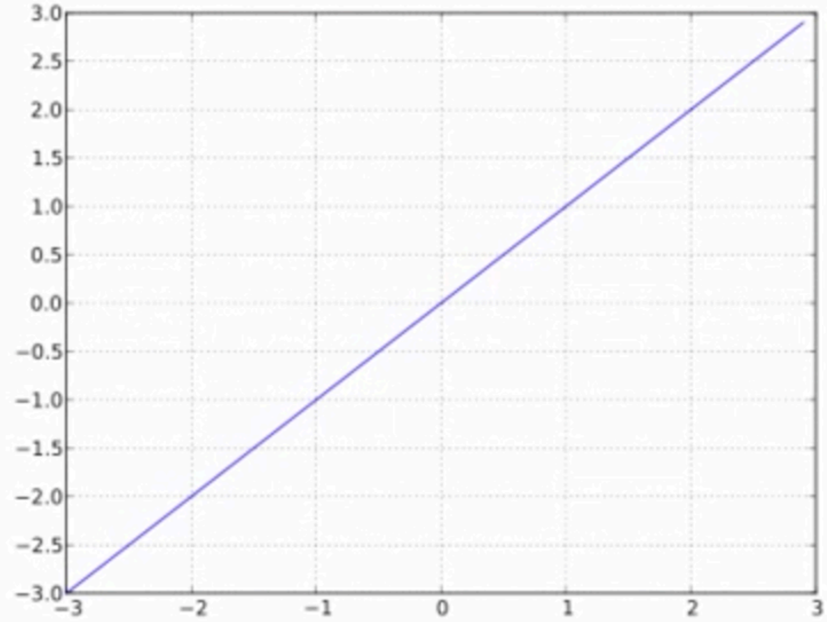
Dendrite

Cell Body

Axon

Activation Functions – Linear Function

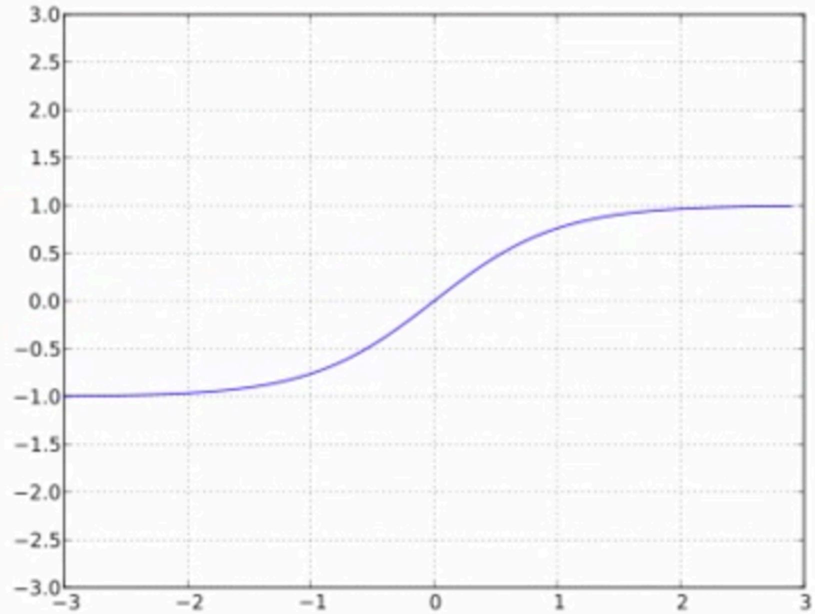
- Performs no input squashing
- Not very interesting...



$$g(a) = a$$

Activation Functions – Hyperbolic Tangent

- Squashes the neuron's input between -1 and 1
- Can be positive or negative
- Bounded
- Strictly increasing

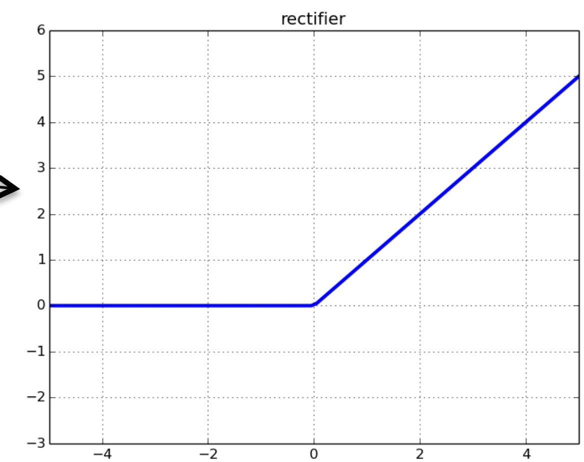


$$g(a) = \tanh(a) = \frac{\exp(a) - \exp(-a)}{\exp(a) + \exp(-a)} = \frac{\exp(2a) - 1}{\exp(2a) + 1}$$

Activation Functions – Sigmoid Function

Recent successes (e.g., Baidu Deep Speech) use (clipped) Rectifier Linear Units:

$h_\theta(x) = g(z)$



f(x) = max(0,x) -- rectifier

f(x) = min(max(0,x), clip)

Smooth approximation (softplus):

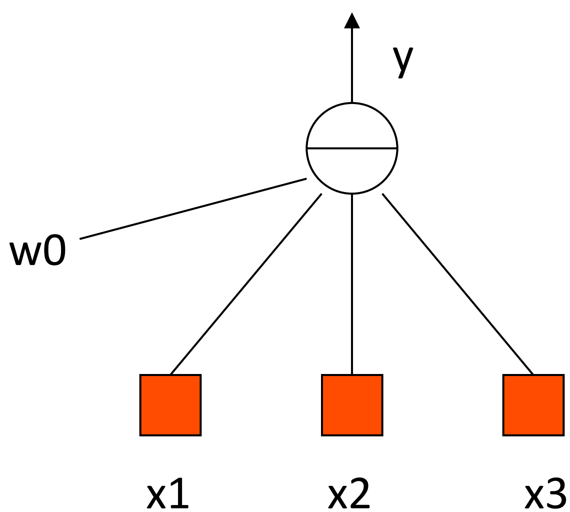
$$f(x) = \log(1 + e^x)$$

Squashes the input (x) onto the open interval [0,1]

$$f'(x) = e^x/(e^x + 1) = 1/(1 + e^{-x})$$

Summary: Artificial neurons

- An *Artificial Neuron* is a non-linear parameterized function with restricted output range



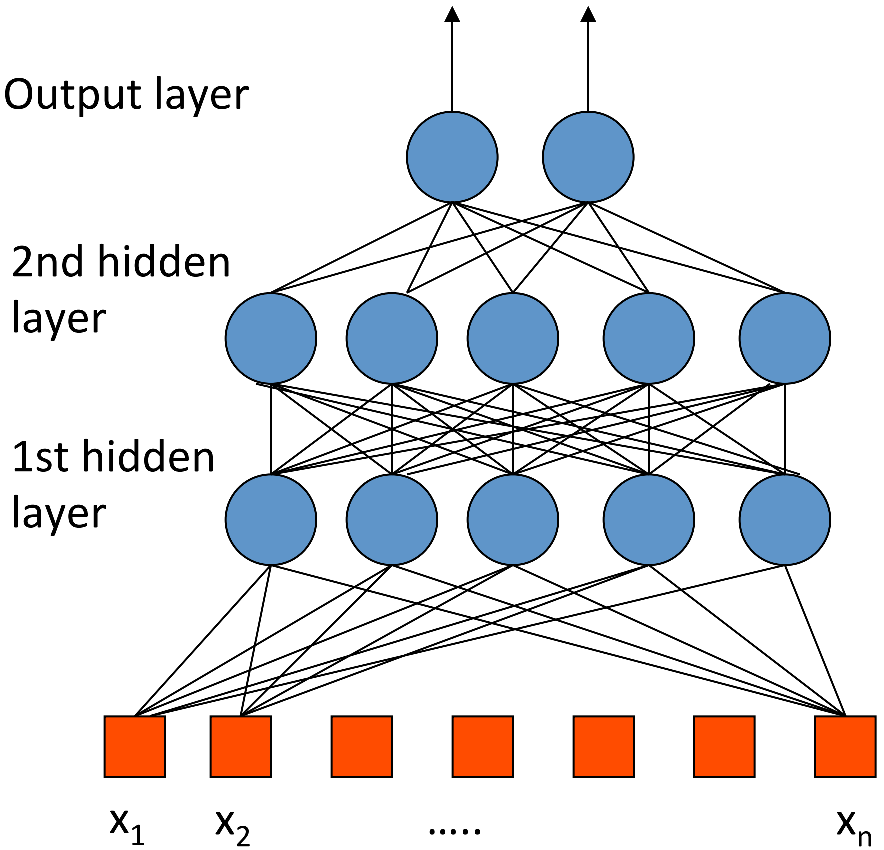
$$y = f\left(w_0 + \sum_{i=1}^{n-1} w_i x_i \right)$$

w_0 also called *a bias term (b_i)*

Ok, Then What is an Artificial Neural Network (ANN)?

- An ANN is mathematical model designed to solve engineering problems
  - Group of highly connected artificial neurons to realize compositions of non-linear functions (usually one of the ones we just looked at)
- Tasks
  - Classification
  - Discrimination
  - Estimation
- 2 main types of networks
  - Feed forward Neural Networks
  - Recurrent Neural Networks

Feed Forward Neural Networks

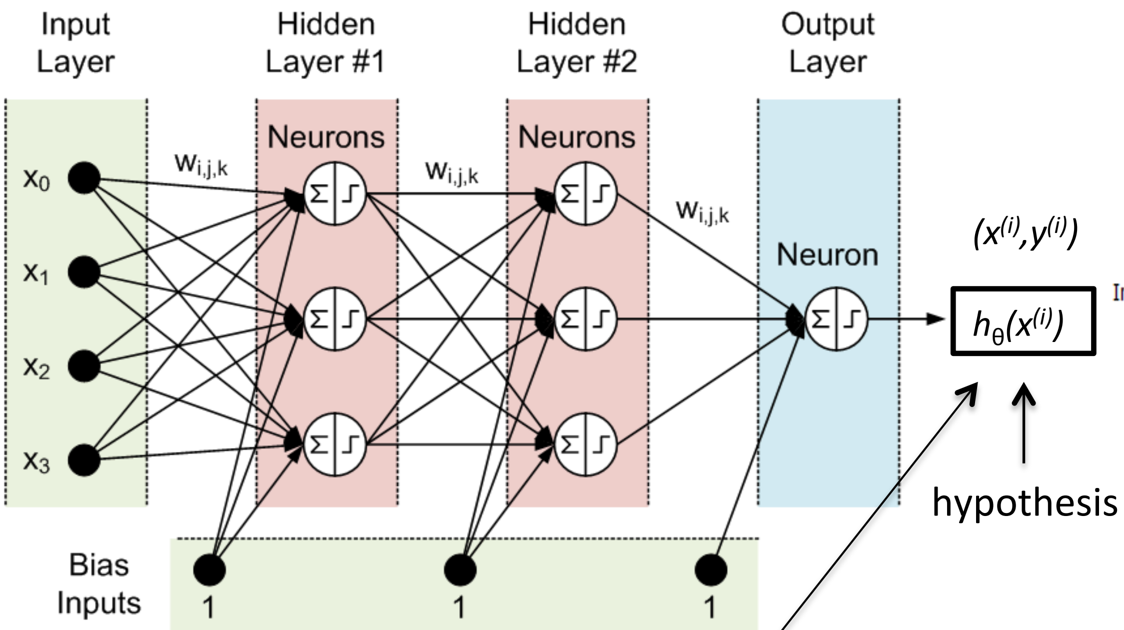


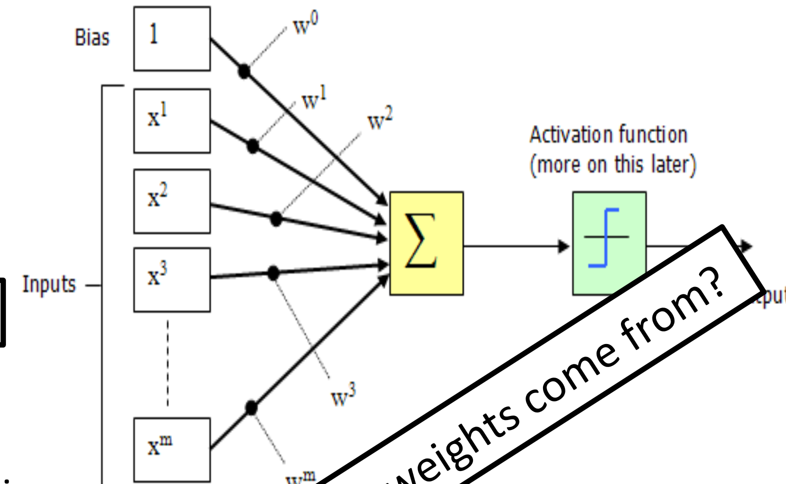
- The information is propagated from the inputs to the outputs
  - Directed Acyclic Graph (DAG)
- Computes one or more non-linear functions
  - Computation is carried out by composition of some number of algebraic functions implemented by the connections, weights and biases of the hidden and output layers
- Hidden layers compute intermediate representations
  - Dimension reduction
- Time has no role -- no cycles between outputs and inputs

We say that the input data, or features, are *n* dimensional

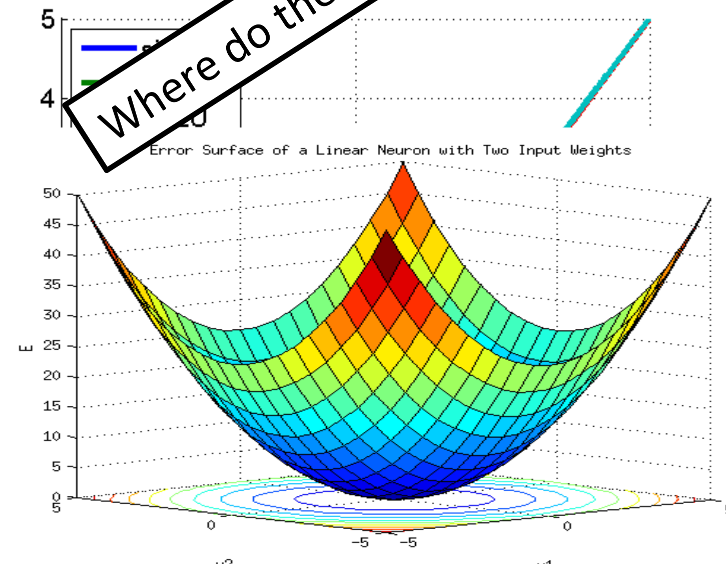
Deep Feed Forward Neural Nets

(in 1 Slide (☺))





Where do the weights come from?



Forward Propagation →

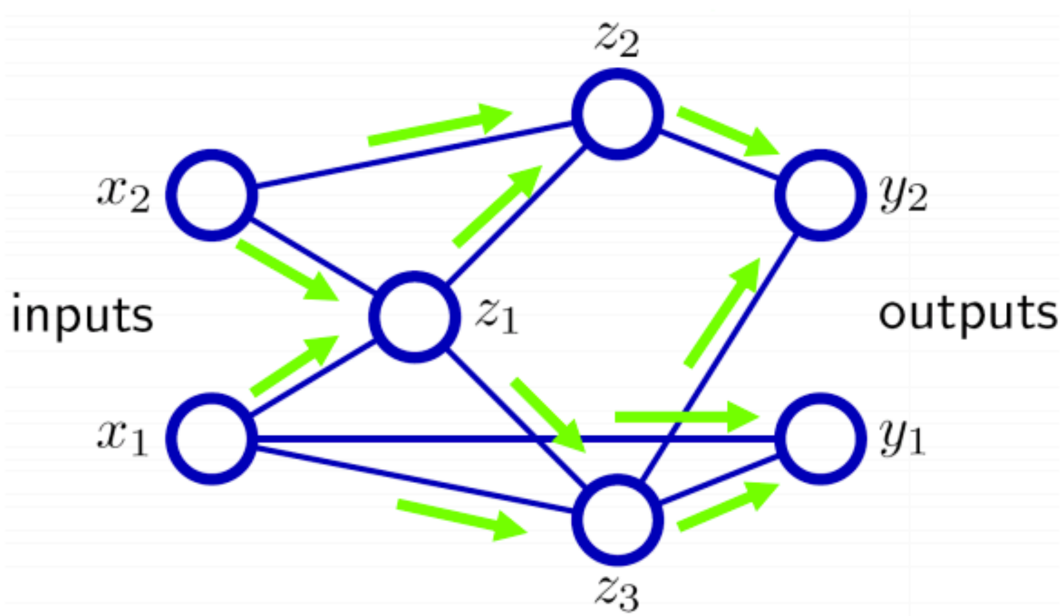
$$J(\theta_0, \theta_1, \ldots, \theta_n) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

Learning is the adjusting of the weights $w_{i,j}$ such that the cost function *J*(**θ**) is minimized (a form of Hebbian learning).

Simple learning procedure: *Back Propagation* (of the error signal)

Forward Propagation Cartoon

- Forward Propagation :
  - Sum inputs, produce activation, feed-forward



Back propagation Cartoon

http://chronicle.com/article/The-Believers/190147

$$J(\theta) = \sum_{i=1}^{n} y^{(i)} \, log\big(h_\theta(x^{(i)})\big) + (1 - y^{(i)}) log(1 - h_\theta(x^{(i)}))$$

More Formally
Empirical Risk Minimization

- Empirical risk minimization
  - framework to design learning algorithms

$$\arg\min_{\boldsymbol{\theta}} \frac{1}{T} \sum_t l(f(\mathbf{x}^{(t)}; \boldsymbol{\theta}), y^{(t)}) + \lambda \Omega(\boldsymbol{\theta})$$

  - $l(f(\mathbf{x}^{(t)}; \boldsymbol{\theta}), y^{(t)})$ is a loss function (loss function also called "cost function" denoted $J(\theta)$)
  - $\Omega(\boldsymbol{\theta})$ is a regularizer (penalizes certain values of $\boldsymbol{\theta}$)
- Learning is cast as optimization
  - ideally, we'd optimize classification error, but it's not smooth
  - loss function is a surrogate for what we truly should optimize (e.g. upper bound)

Any interesting cost function is complicated and non-convex

Solving the Risk (Cost) Minimization Problem
Gradient Descent – Basic Idea

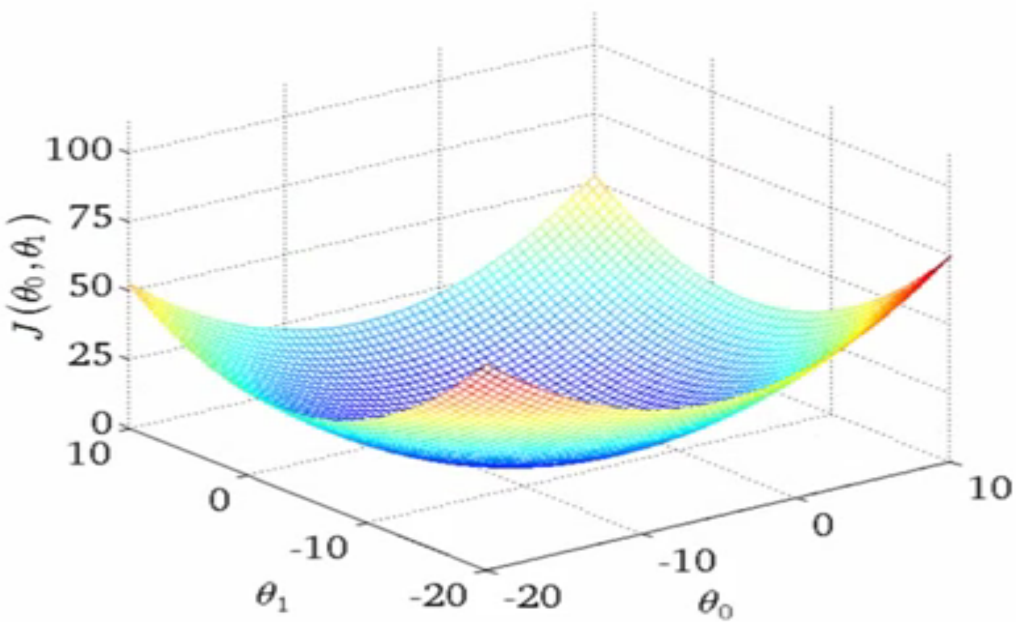
Have some function $J(\theta_0, \theta_1)$

Want $\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$

Outline:

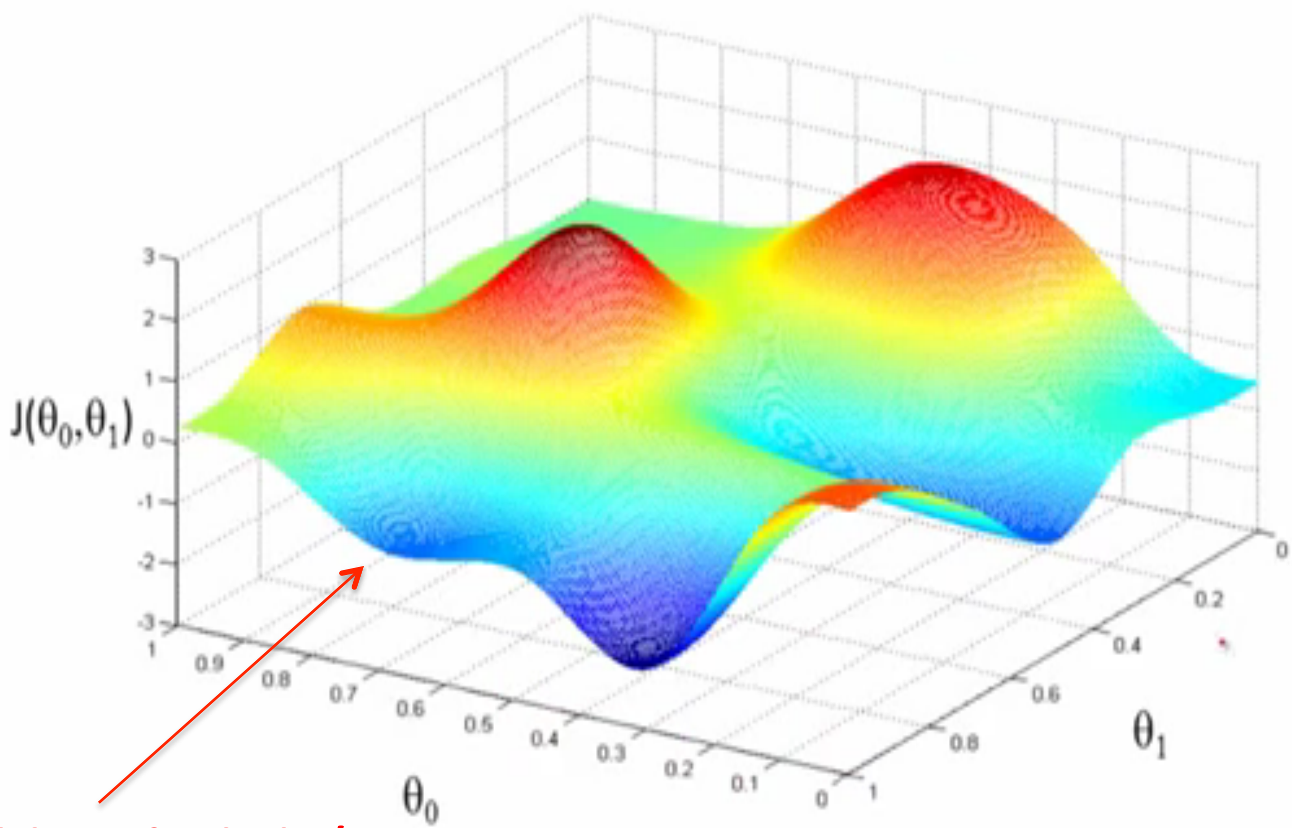
- Start with some θ_0, θ_1
- Keep changing θ_0, θ_1 to reduce $J(\theta_0, \theta_1)$
  until we hopefully end up at a minimum

Gradient Descent Intuition 1
Convex Cost Function



One of the many nice properties of convexity is that any local minimum is also a global minimum

Gradient Decent Intuition 2



Can get stuck here if unlucky/start at the wrong place

Unfortunately, any interesting cost function is likely non-convex

Solving the Optimization Problem
Gradient Descent for Linear Regression

Gradient descent algorithm

repeat until convergence {

$$\theta_j := \theta_j - \boxed{\alpha} \boxed{\frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)}$$

(for $j = 1$ and $j = 0$)

}

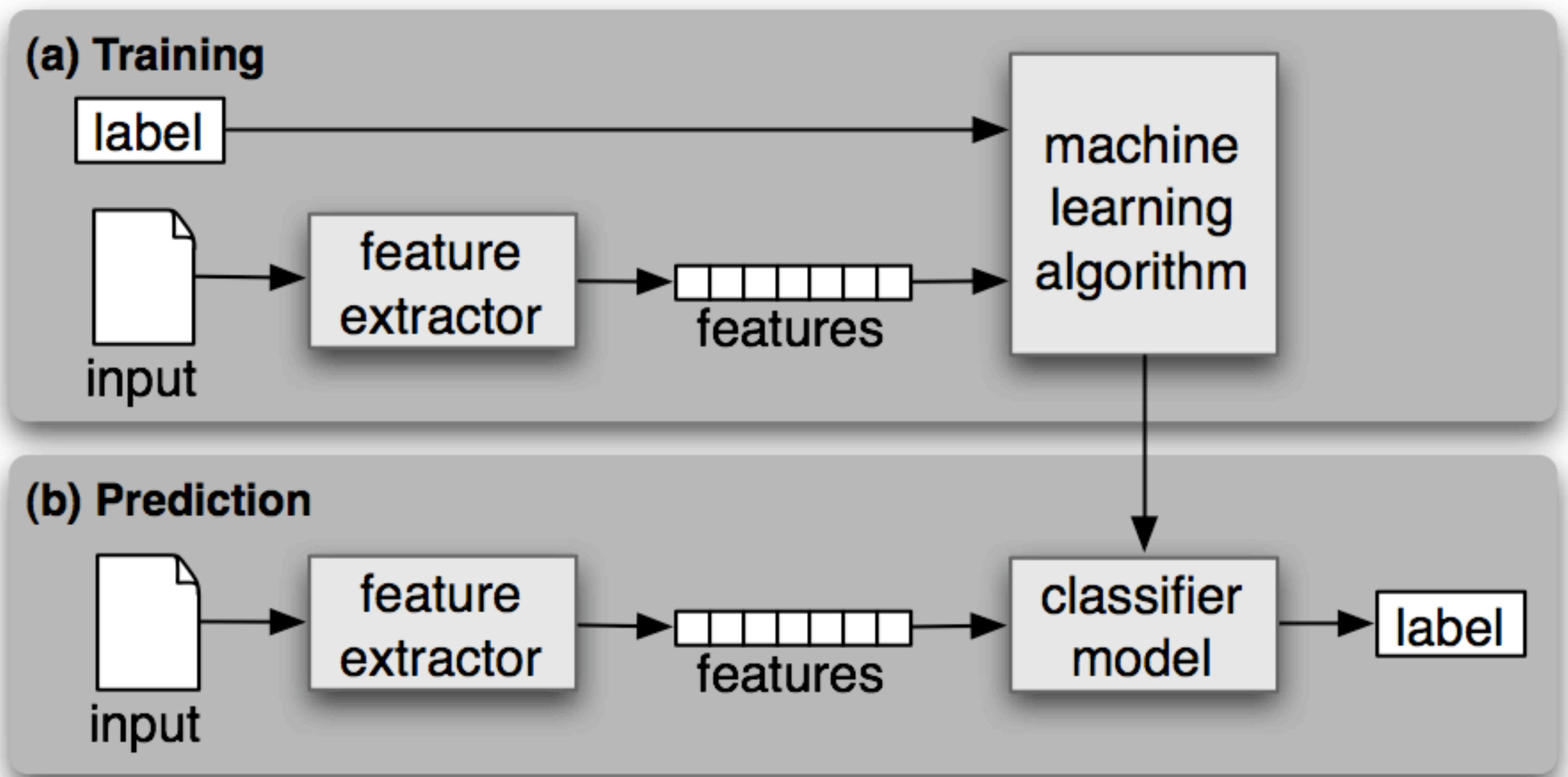
Linear Regression Model

$$h_\theta(x) = \theta_0 + \theta_1 x$$

$$\boxed{J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2}$$

The big breakthrough came from the Hinton lab at UToronto in the mid 80's where the back propagation algorithm was discovered (or perhaps re-discovered). "Backprop" is a simple way of computing the gradient of the loss function with respect to the model parameters θ

Summary: Supervised Learning Process



Agenda

- ~~Goals for this Session~~
- ~~What is Machine Learning?~~
  - ~~And how can it possibly work?~~
- ~~Shallow Dive Into Deep Neural Net Technology~~
- PCE
- Q&A

PCE as a Canonical Application

- PCE ideally suited to SDN and Machine Learning
- Can we infer properties of paths we can't directly see?
  - Likely living in high-dimensional space(es)
  - i.e., those in other domains
- Other inference tasks?
  - Aggregate bandwidth consumption
  - Most loaded links/congestion
  - Cumulative cost of path set
  - Uncover unseen correlations that allow for new optimizations
- How to get there from here
  - The PCE was always a form of "SDN"
  - Applying Machine Learning to the PCE requires understanding the problem you want to solve and what data sets you have

PCE Data Sets

- Assume we have labeled data set
  - $\{(X^{(1)},Y^{(1)}),\ldots,(X^{(n)},Y^{(n)})\}$
    - Where $X^{(i)}$ is an m-dimensional vector, and
    - $Y^{(i)}$ is usually a k dimensional vector, $k < m$

$$\mathbf{X} \in \mathbb{R}^m$$
$$\mathbf{Y} \in \mathbb{Z}^k$$

- Strawman X (information from the TED plus others)

- $X^{(i)}$ = (Path end points,
  Desired path constraints,
  Computed path,
  Aggregate path constraints (e.g. path cost),
  Minimum cost path,
  Minimum load path,
  Maximum residual bandwidth path,
  Aggregate bandwidth consumption,
  Load of the most loaded link,
  Cumulative cost of a set of paths,
  (some measure of buffer occupancy),
  …,
  Other (possibly exogenous) data)

- If we have $Y^{(i)}$'s are a set of classes we want to predict, e.g., congestion, latency, …

What Might the Labels Look Like?

$$\mathbf{Y} = \begin{bmatrix} Congestion \\ Latency \\ Class2 \\ Class3 \\ Class4 \\ \ldots \end{bmatrix} \qquad \rightarrow \qquad Y^{(i)} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ \ldots \end{bmatrix}$$

(instance)

Making this Real
(what do we have to do?)

- Choose the labels of interest
  - What are the classes of interest, what might we want to predict?

- Get the (labeled) data set (this is always the “trick”)
  - Split into training, test, cross-validation
    - Avoid generalization error (bias, variance)
  - Avoid data leakage

- Choose a model
  - I would try supervised DNN
    - We want to find “non-obvious” features, which likely live in high-dimensional space

- Write code
  - Then write more code

- Test on (previously) unseen examples

- Iterate

Agenda

- ~~Goals for this Session~~

- ~~What is Machine Learning?~~
  - ~~And how can it possibly work?~~

- ~~Shallow Dive Into Deep Neural Net Technology~~

- ~~PCE~~

- Q&A

Q & A

Thanks!