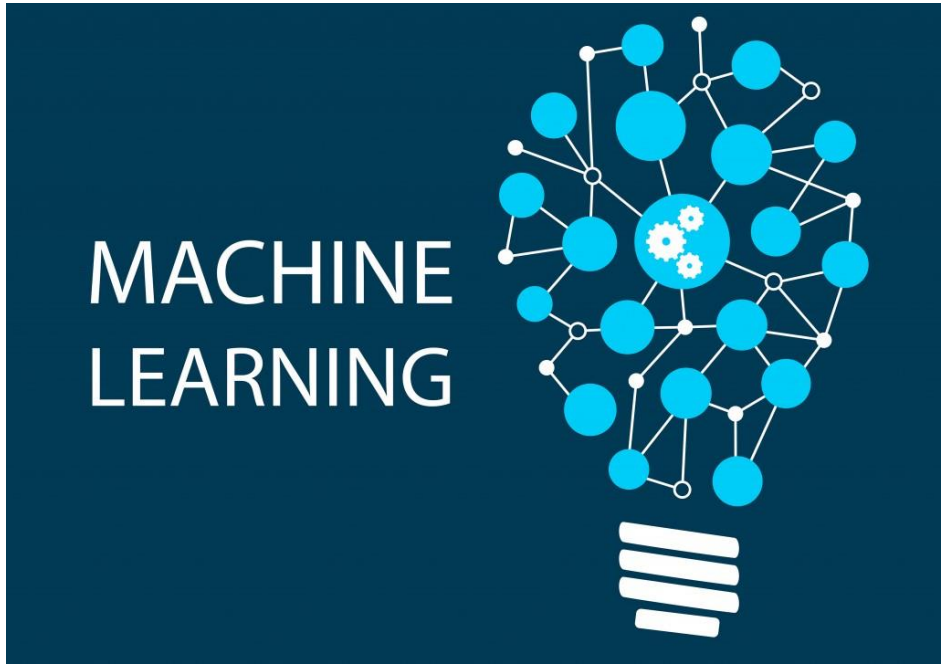
Basic Concepts of Machine Learning

Machine Learning Indonesia Meet Up

Risman Adnan
Samsung R&D Institute Indonesia
University of Indonesia

Outline

- The Learning Problem
- Machine Learning Landscape
- Simple Perceptron Model
- Deep Learning vs Machine Learning
- How to Get Start
- Q&A



The Learning Problem

The Learning Problem

The essence of ML:

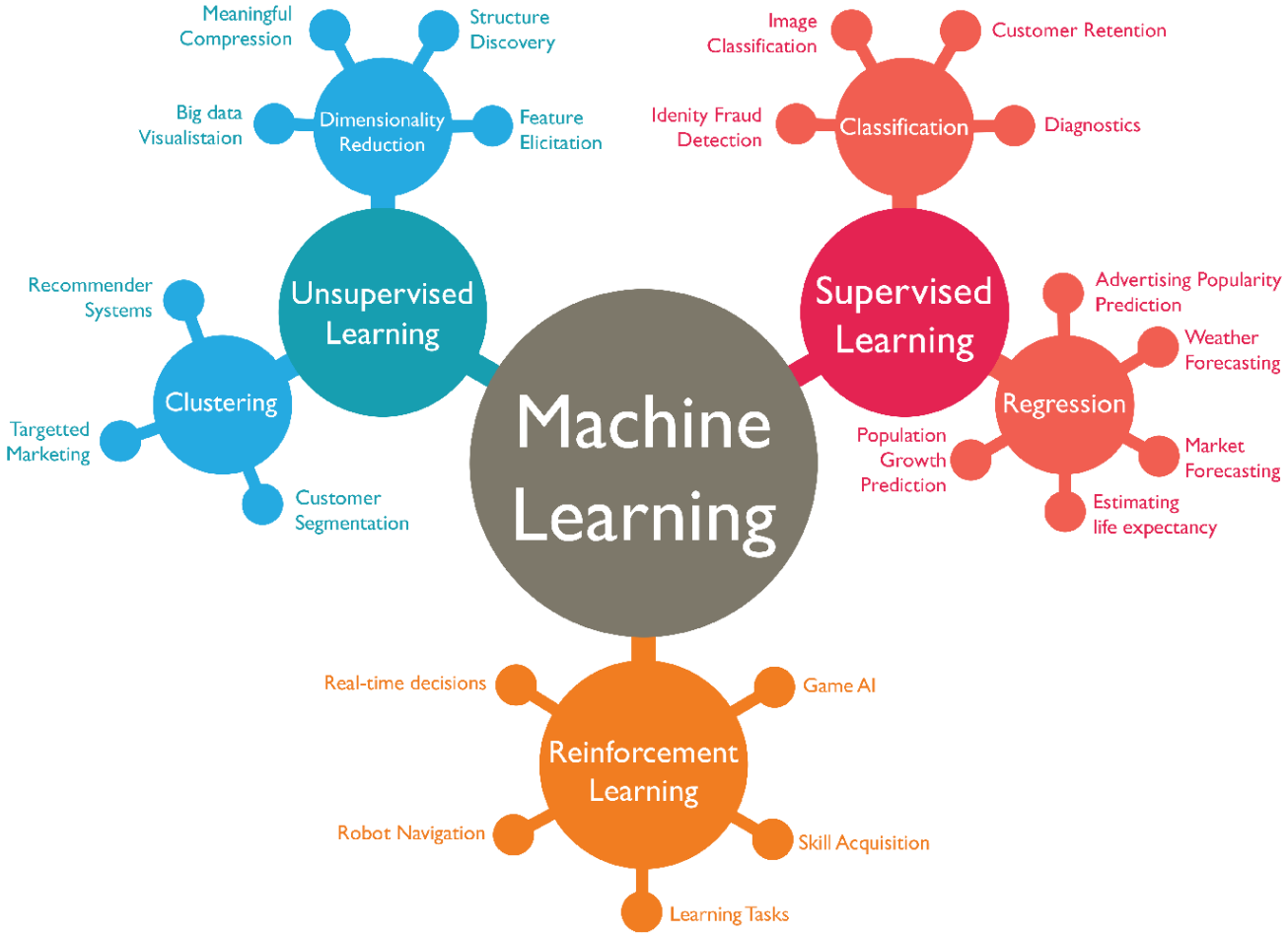
1. We have data
2. Patterns exist in data
3. We cant do math formula

Examples:

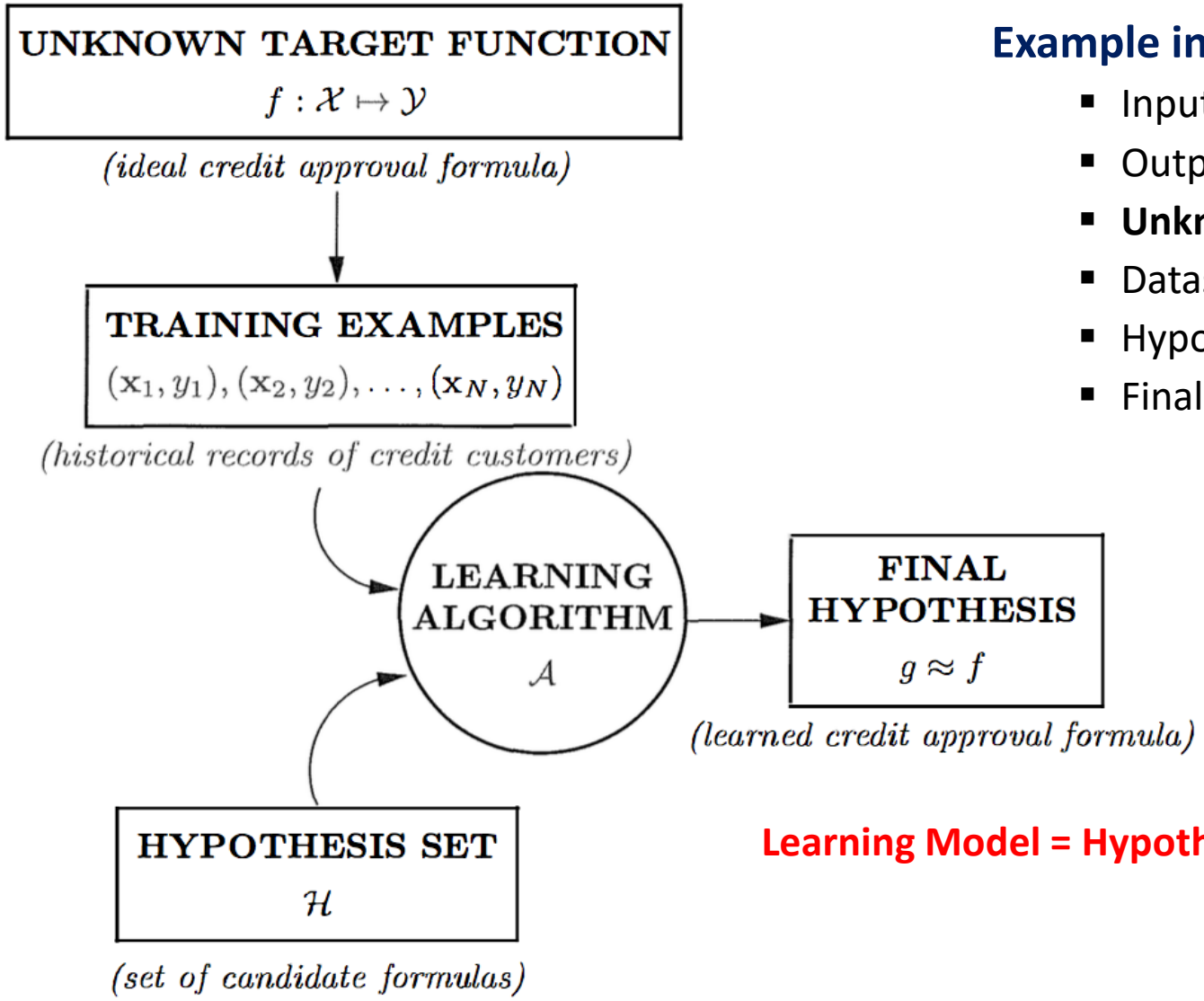
- Movie Rating
- Credit Approval
- Hand Written Recognition

Domain Areas

- Computer Vision
- Natural Language Processing
- Business Intelligence



Components of Learning

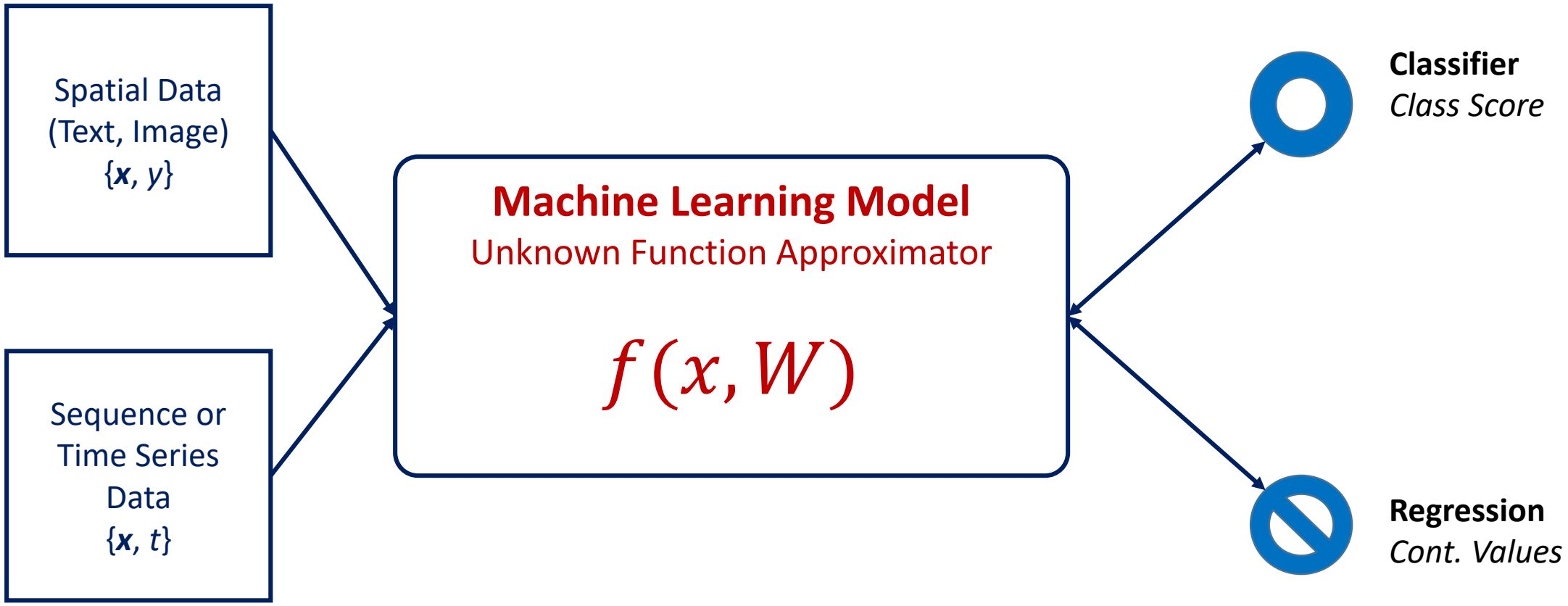


Example in Banking: Credit Card Approval

- Input : **x** (*customer application*)
- Output : *y* (*good/bad customer*)
- **Unknown** Target Function $f : X \rightarrow Y$
- Dataset {*x*, *y*} (*customers record database*)
- Hypothesis Set: $H : X \rightarrow Y$
- Final Hypothesis *g*

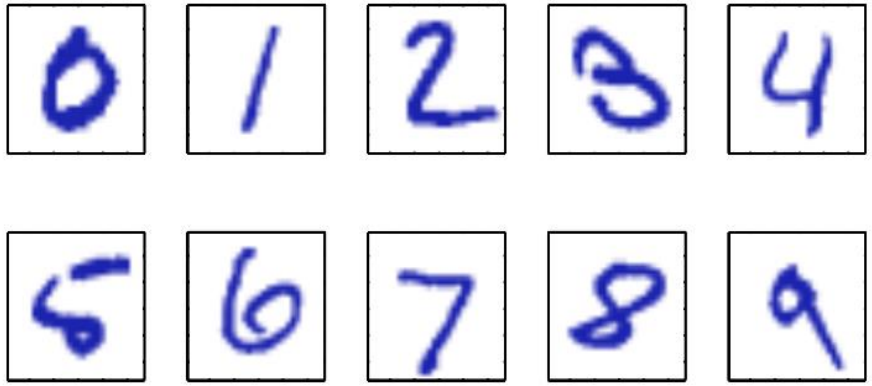
Learning Model = Hypothesis Set + Learning Algorithm

Machine Learning Model

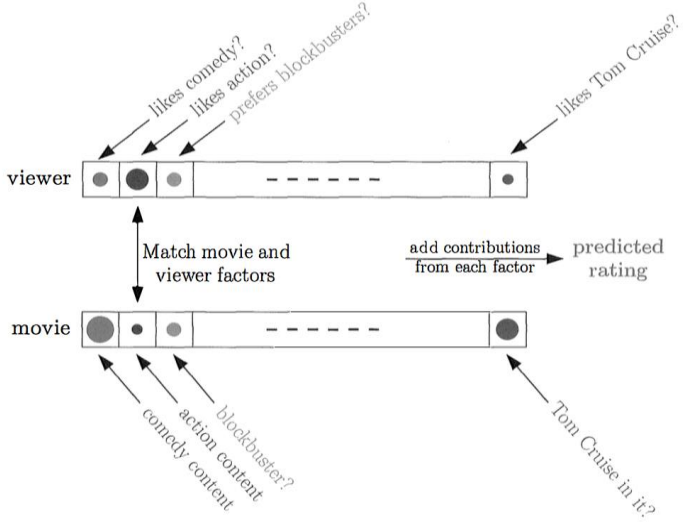


Problem Examples

- **Problem**: Recognize Handwritten Digits



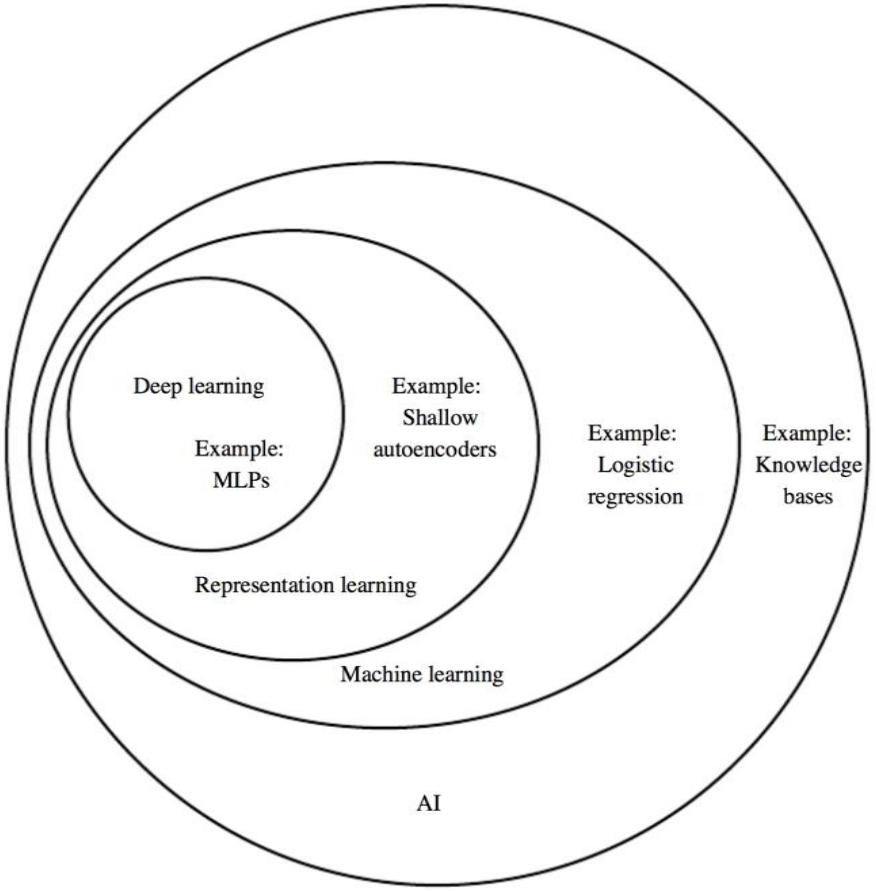
- **Problem**: Predict Viewer Rate Movie



Dataset → Rule Based → Classic Machine Learning → Representation Learning

Machine Learning Landscape

Machine Learning Map



THEORY	TECHNIQUES		PARADIGMS
	MODELS	METHODS	
VC	Linear	**Regularization**	**Supervised**
Bias-Variance	**Neural Networks**	**Validation**	Unsupervised
Complexity	SVM	**Aggregation**	Reinforcement
Bayesian	Nearest Neighbors	**Input Processing**	Active
	RBF		Online
	Gaussian Processes		
	SVD		

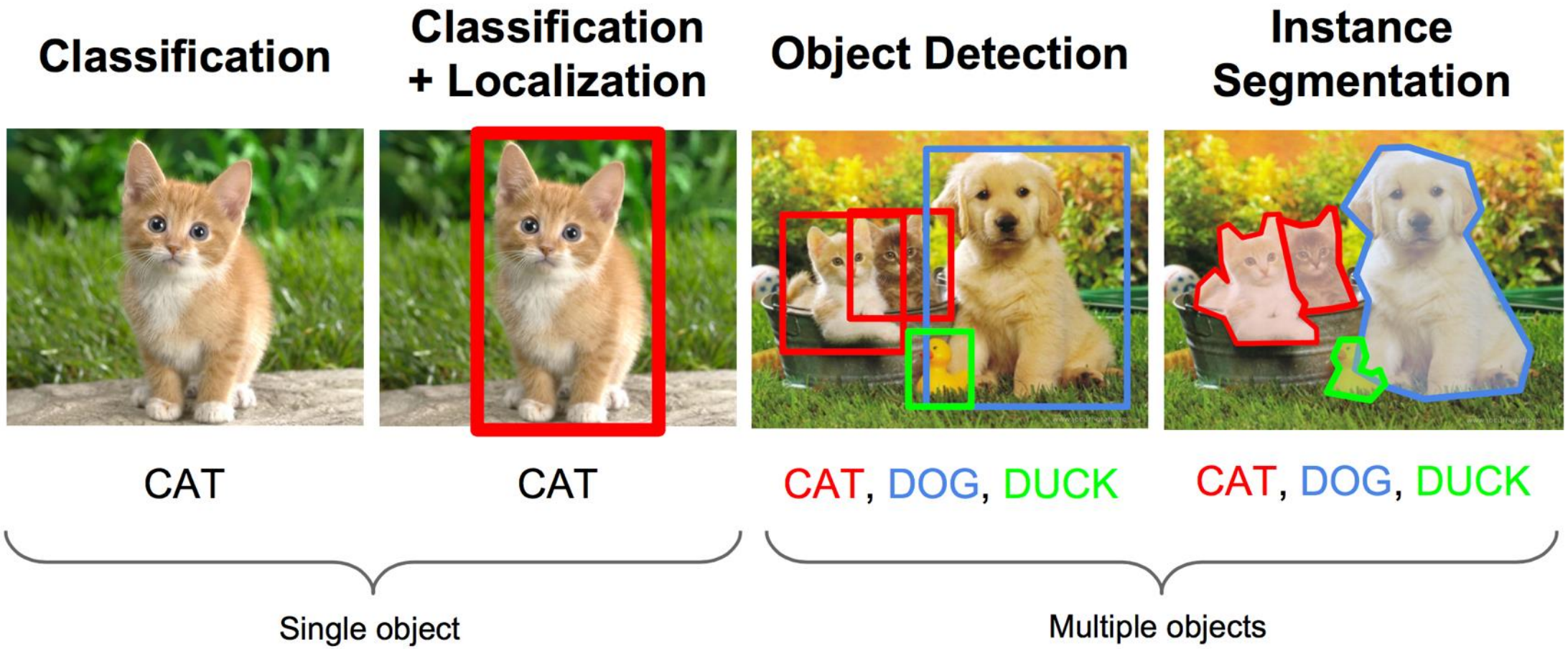
Machine learning become a jungle of models, methods!

Main Paradigms

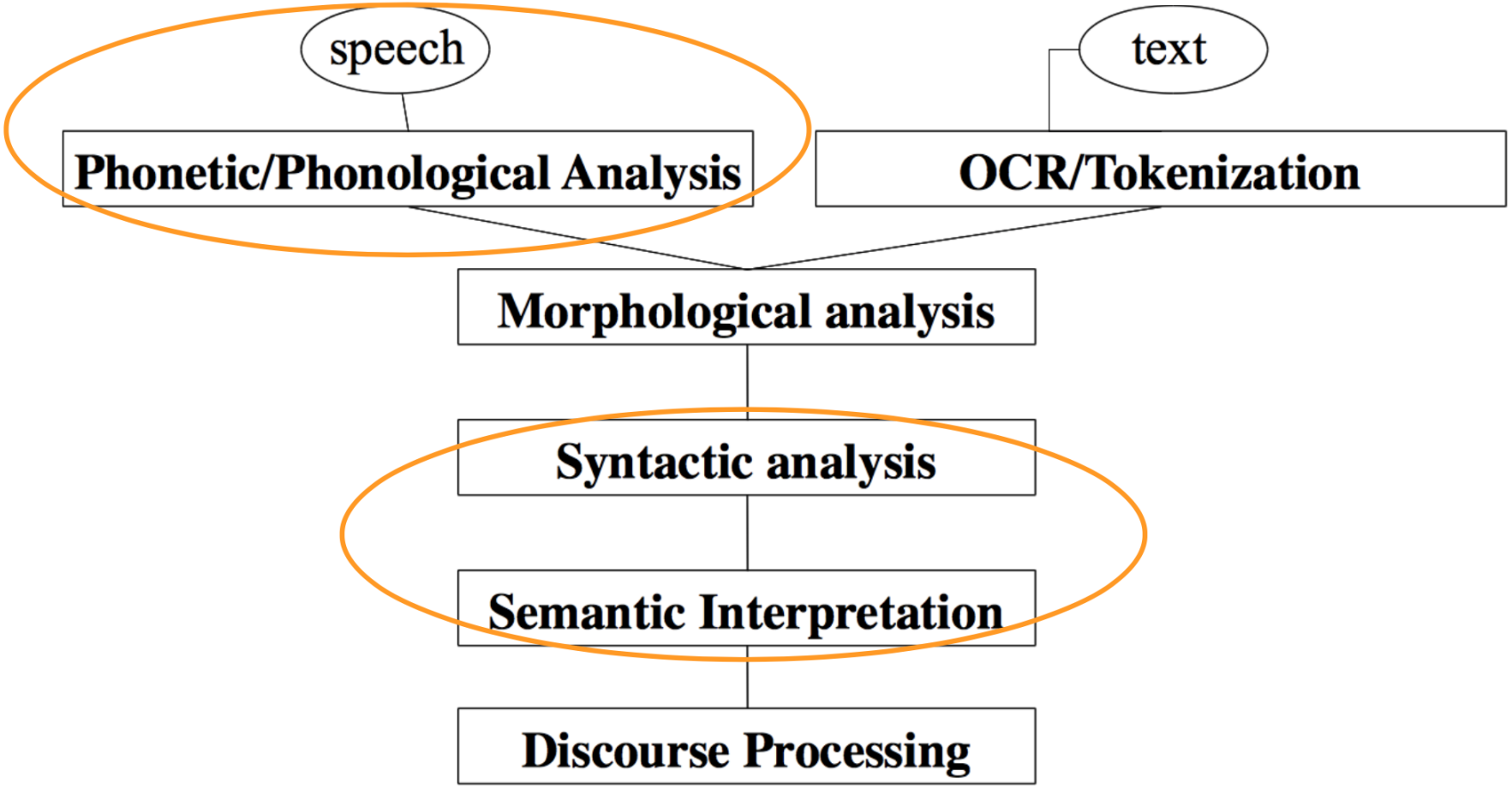
Automatic discovery of patterns in data through computer algorithms and the use of those patterns to take actions such as ***classifying*** or ***clustering*** the data into categories.

- **Supervised Learning**: Learning by labeled example
  - E.g. An email spam detector
  - We have (**input**, **correct output**), and we can predict (**new input**, **predicted output**)
  - Amazingly effective if you have lots of data
- **Unsupervised Learning**: Discovering Patterns
  - E.g. Data clustering
  - Instead of (**input**, **correct output**), we get (**input**, **?**)
  - Difficult in practices but useful if we lack labeled data
- **Reinforcement Learning**: Feedback & Error
  - E.g. Learning to play chess
  - Instead of (**input**, **correct output**), we get (**input**, **only some output**, **grade of this output**)
  - Works well in some domains, becoming more important

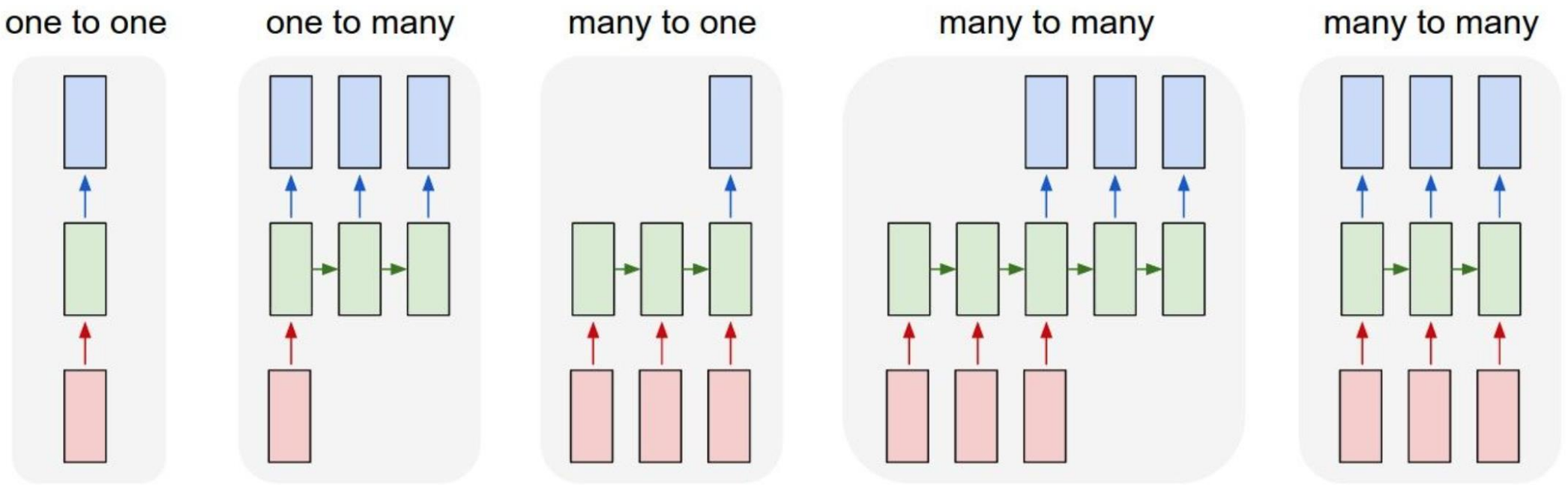
Computer Vision Domain



Natural Language Processing Domain



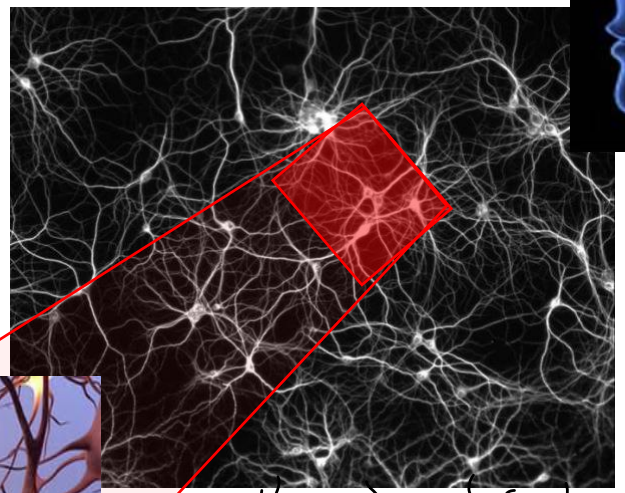
Sequence Labeling in NLP Domain

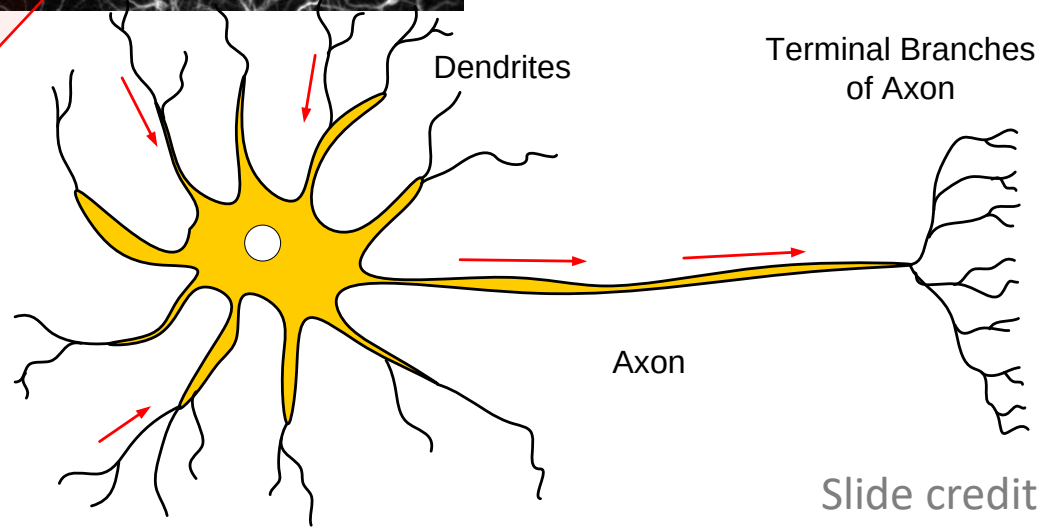


Basic | Image Captioning | Sentiment Classification | Machine Translation | Video Classification

Simple Perceptron Model

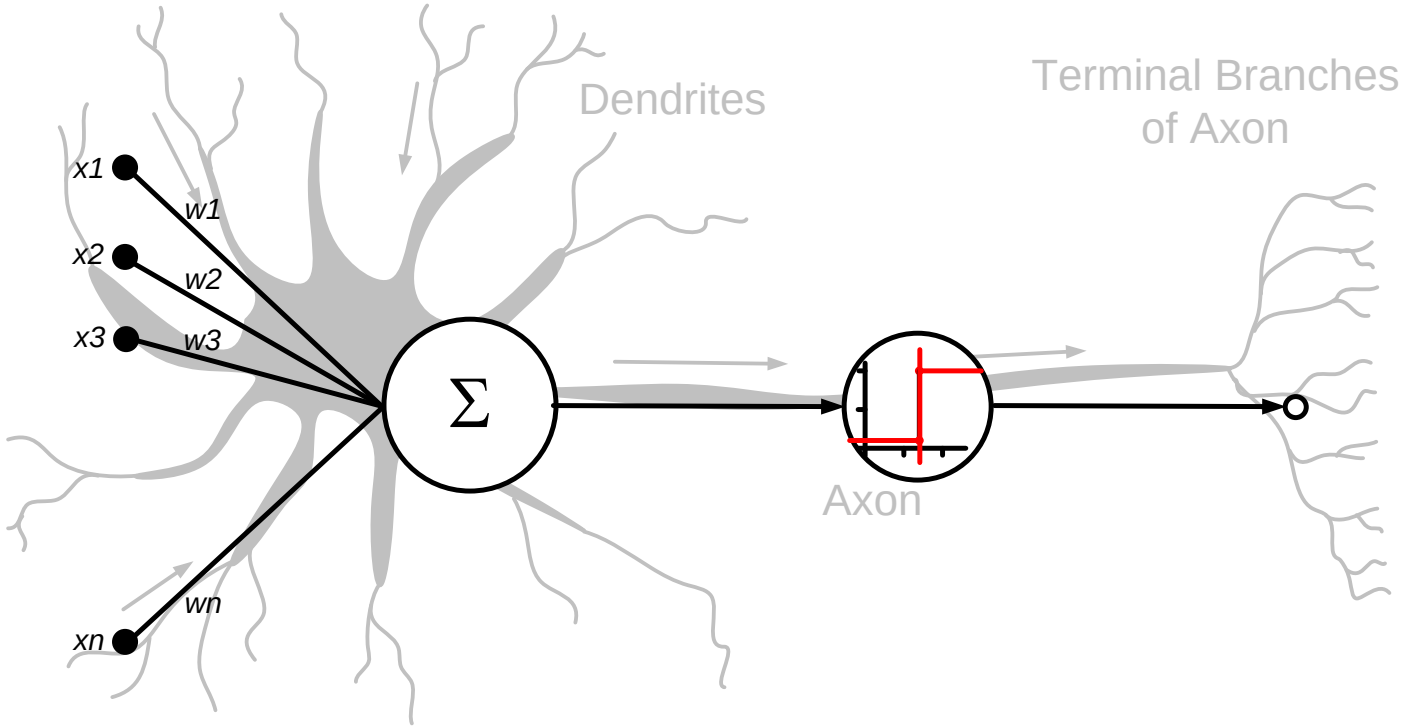
Perceptron Inspired by Biological Neurons





Slide credit : Geoffrey Hinton

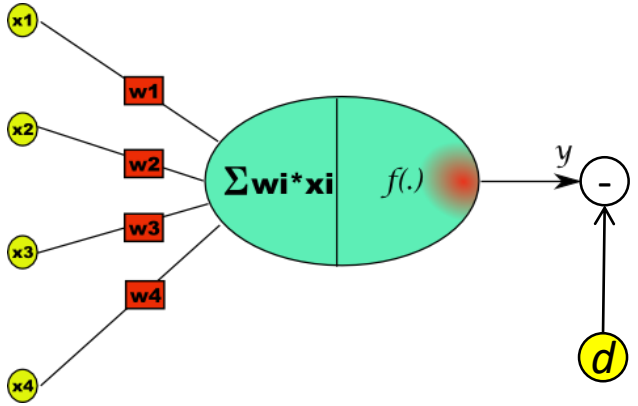
Perceptron in Artificial Neural Networks



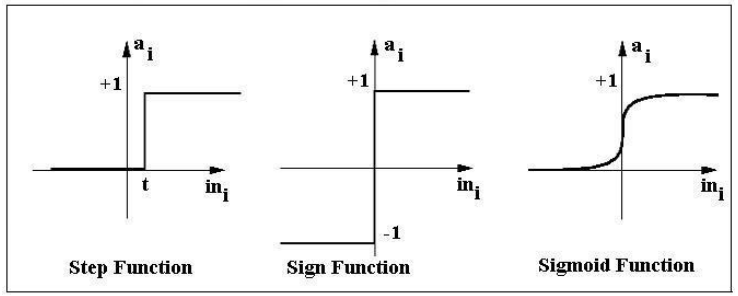
Slide credit : Andrew L. Nelson

A Simple Model - Perceptron

Perceptron:



Activation functions:



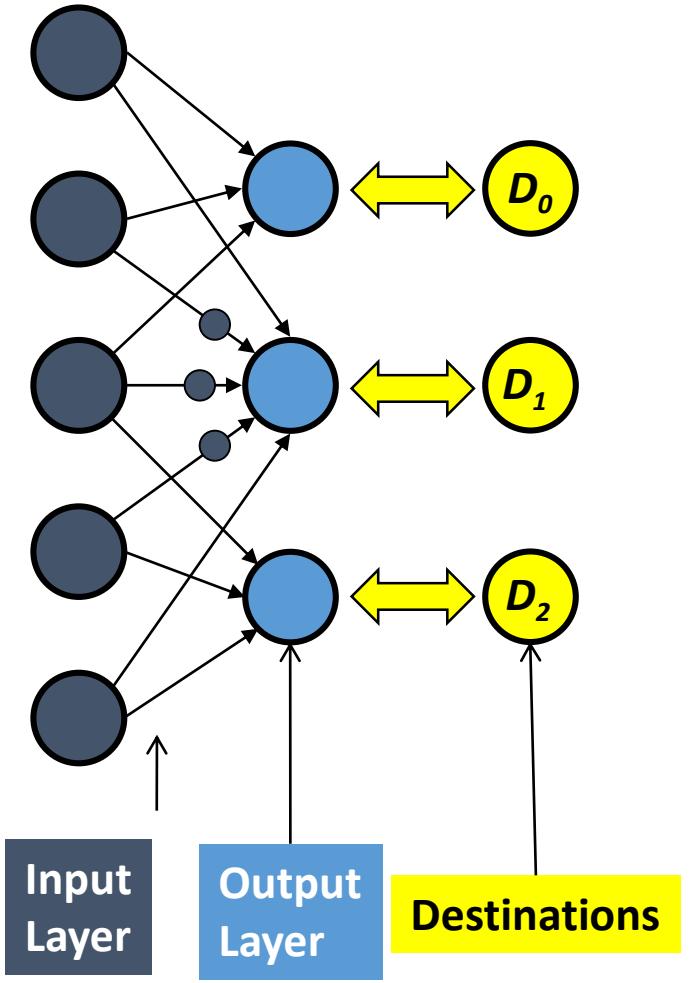
Learning:

$$y^{(t)} = f\left\{\sum_i w_i^{(t)} x_i^{(t)}\right\}$$

Update

$$\Delta w_i^{(t)} = \varepsilon(d^{(t)} - y^{(t)}) x_i^{(t)}$$

$$w_i^{(t+1)} = w_i^{(t)} + \Delta w_i^{(t)}$$



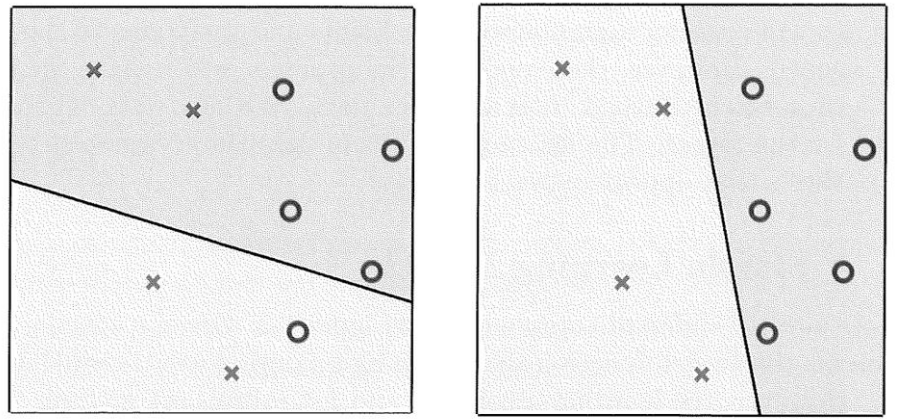
Slide credit : Geoffrey Hinton

A Simple Hypothesis Set – The Perceptron

- For input **x** = (x_1, …., x_d) ‘ attributes for a customer’

$$\text{Approve credit if} \quad \sum_{i=1}^{d} w_i x_i > \text{threshold},$$

$$\text{Deny credit if} \quad \sum_{i=1}^{d} w_i x_i < \text{threshold}.$$



(a) Misclassified data (b) Perfectly classified data

- This formula can be written more compact as

$$h(\mathbf{x}) = \text{sign}\left(\left(\sum_{i=1}^{d} w_i x_i\right) + b\right),$$

$$h(\mathbf{x}) = \text{sign}(\mathbf{w}^{\mathrm{T}}\mathbf{x}).$$

Perceptron Learning Algorithm

- The perceptron implement:

$$h(\mathbf{x}) = \text{sign}(\mathbf{w}^{\text{T}}\mathbf{x}).$$

- Given the training set:

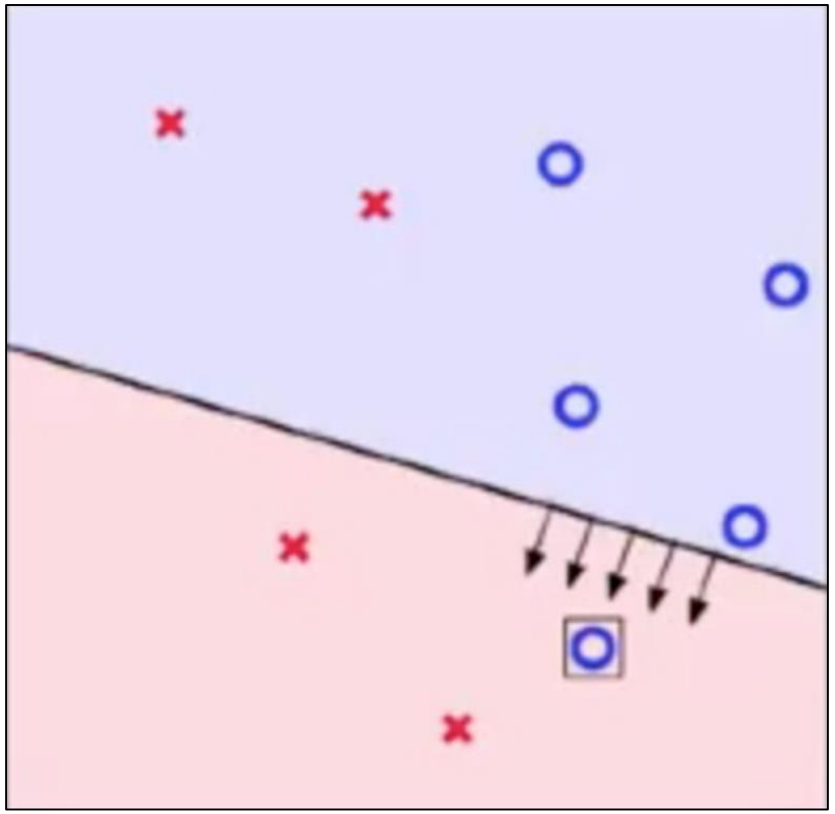
$$(\mathbf{x}_1, y_1) \cdots (\mathbf{x}_N, y_N)$$

- Pick a misclassified point:

$$(\mathbf{x}(t), y(t))$$

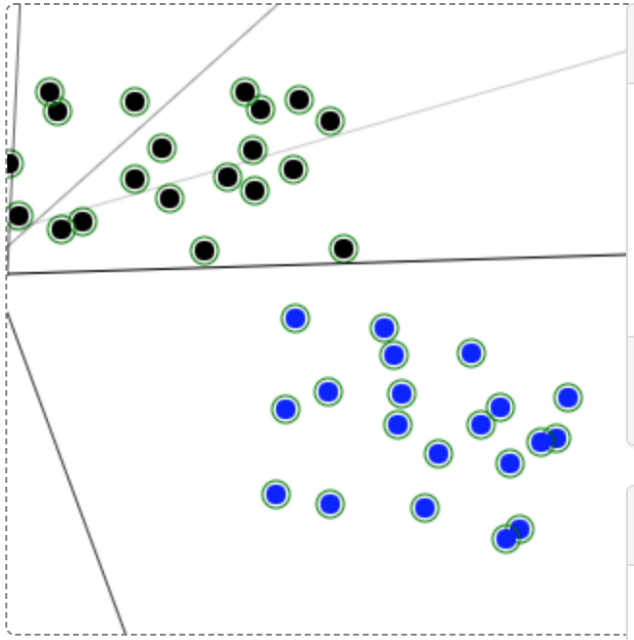
- And update the weight vector:

$$\mathbf{w}(t+1) = \mathbf{w}(t) + y(t)\mathbf{x}(t).$$



Simple Perceptron Demo

http://ditam.github.io/demos/perceptron/perceptronDemo.html



Parameters

random seed:	10	**base width:**	100
separator trail length:	5	**width variance:**	100
run delay (ms):	200	**base height:**	100
cloud size:	20	**height variance:**	50

apply params | step | run

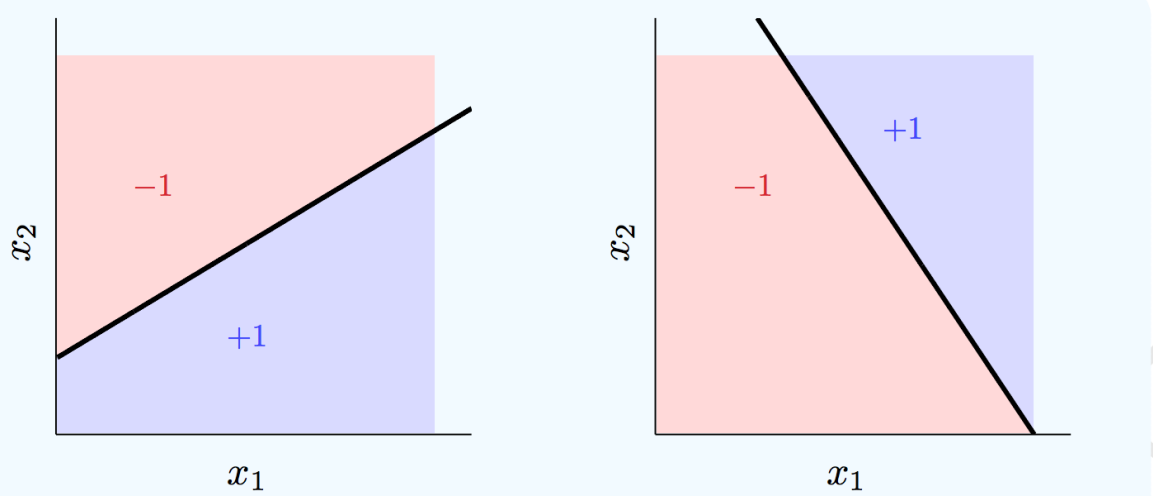
Current Classification

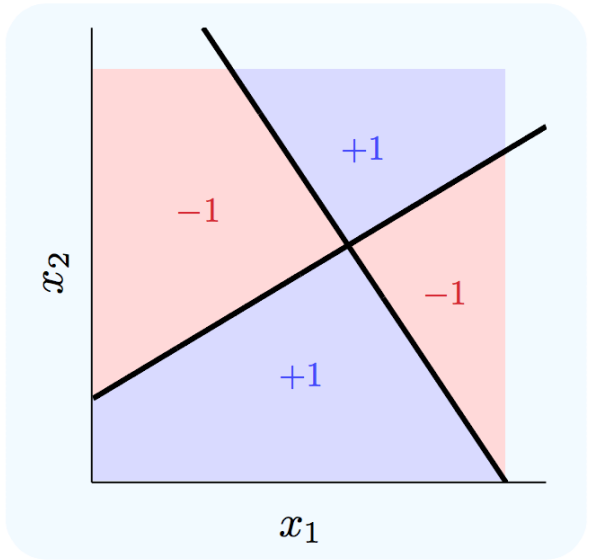
TP	FP	TN	FN	Precision	Recall	F1
20	0	20	0	1.00	1.00	0.50

w(x) = -0.03110135382363703 x + 139

Combining Perceptron

- Can perceptron combined to perform more complex target function?
- Yes if we can write *f* using simple AND and OR operations:





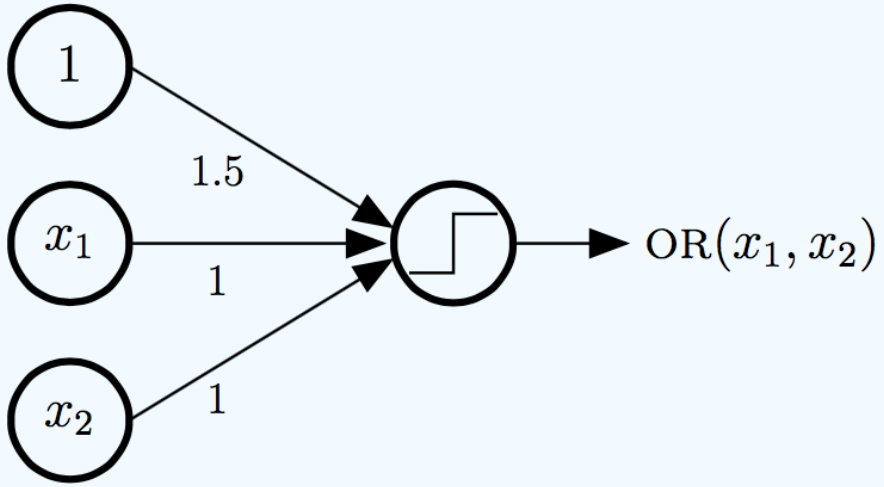
$$f = h_1\overline{h_2} + \overline{h_1}h_2.$$

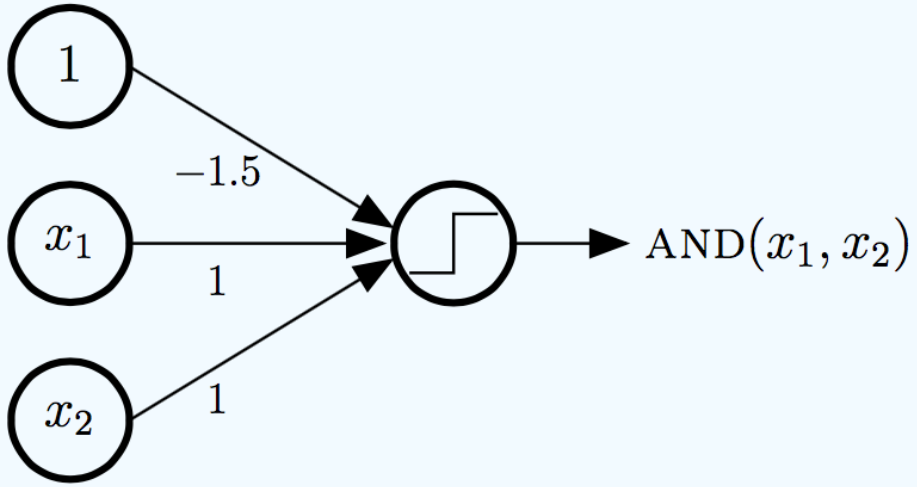
Note: Using standard Boolean notation (multiplication for AND, addition for OR, and overbar for negation)

Combining Perceptron

- Can perceptron implement AND and OR operation?

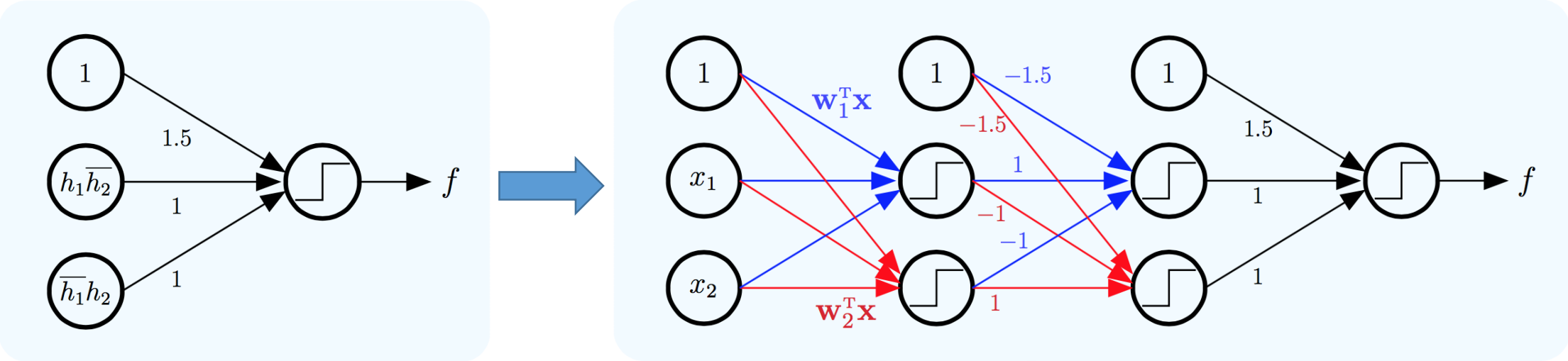
$$\begin{aligned} \text{OR}(x_1, x_2) &= \text{sign}(x_1 + x_2 + 1.5); \\ \text{AND}(x_1, x_2) &= \text{sign}(x_1 + x_2 - 1.5). \end{aligned}$$





Combining Perceptron

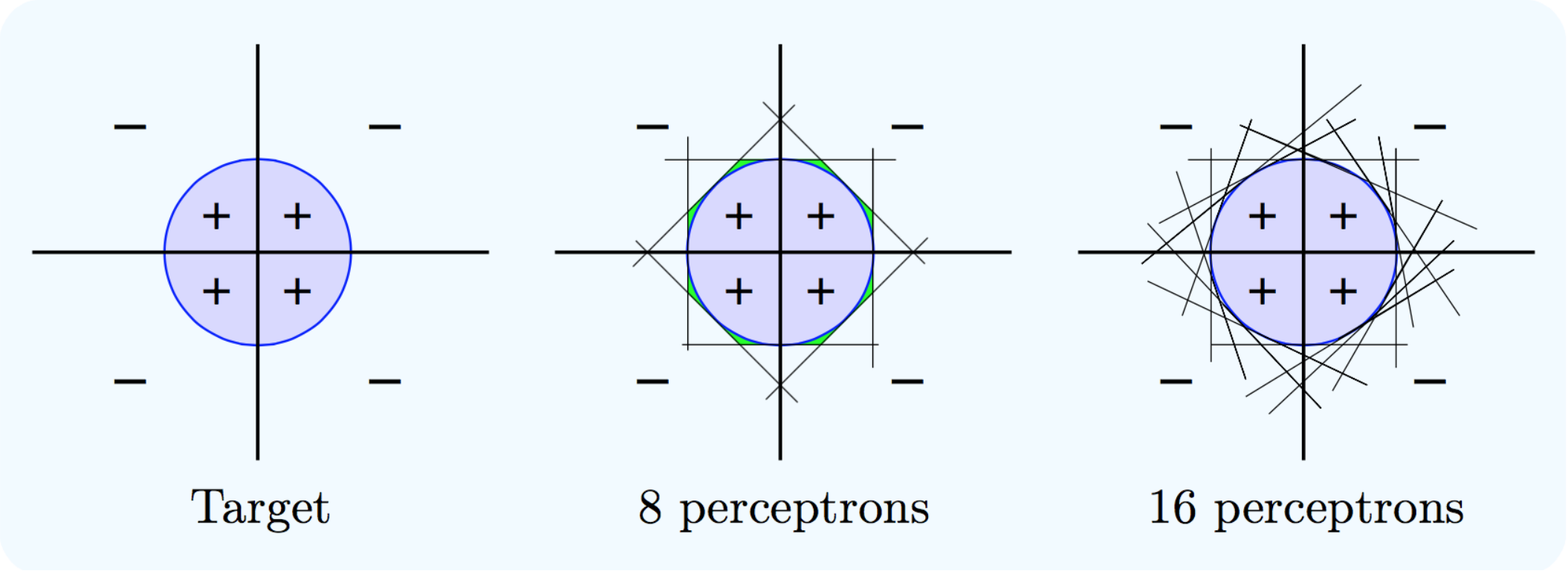
- We can visualize $f = h_1\overline{h_2} + \overline{h_1}h_2$ as the following:



- **Key Insight**: More layers of nodes are used between input and output to implement more complex target function *f*. We called Multi-Layer Perceptron. Additional layers are called *hidden layers*.

A Powerful Model

- If *f* can be decomposed into perceptrons using an OR of ANDs, then it can be implemented by a 3-layer perceptron. If *f* is not strictly decomposable into perceptrons, but the decision boundary is smooth, then a 3-layer perceptron can come arbitrarily close to implementing *f*. **Visual Proof**:

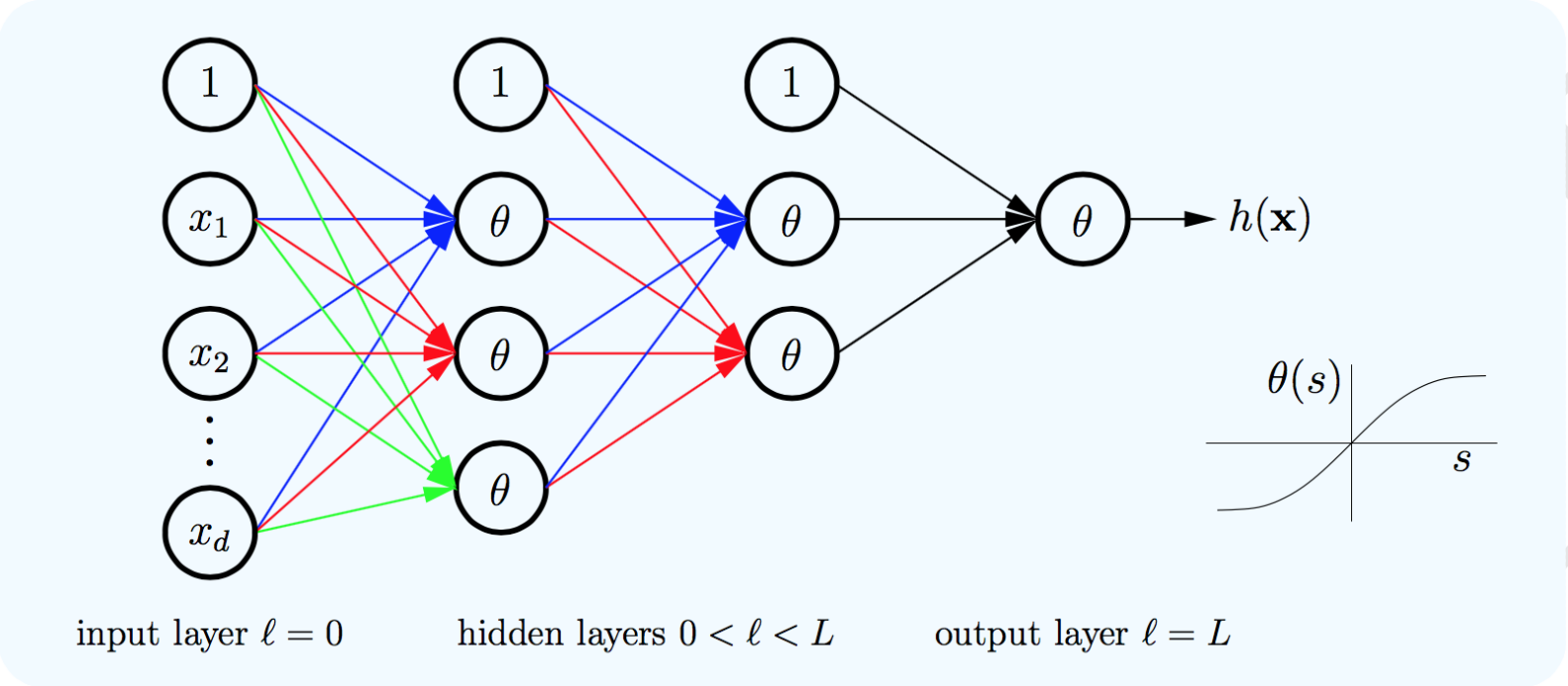


Deep Learning vs Machine Learning

Multi Layer Perceptron

Architecture:

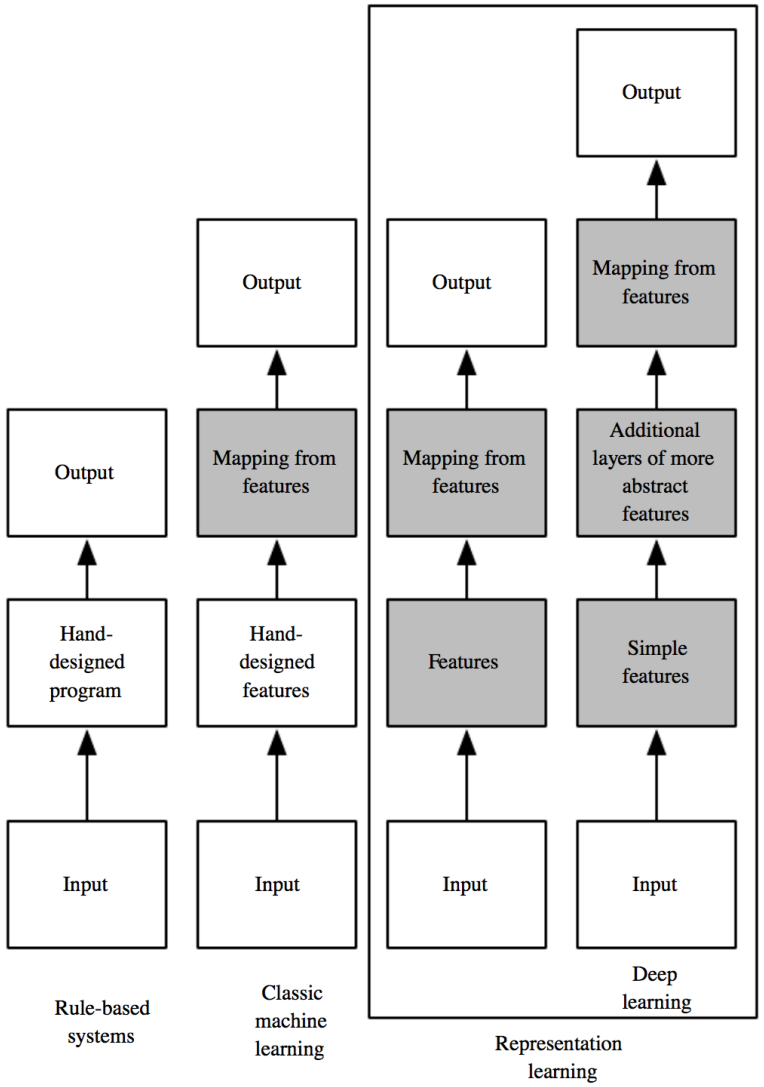
- **Input** Layer
- **Hidden** Layers
- **Output** Layer
- One Direction Flow

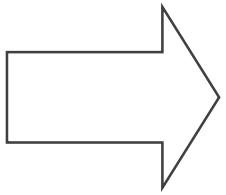


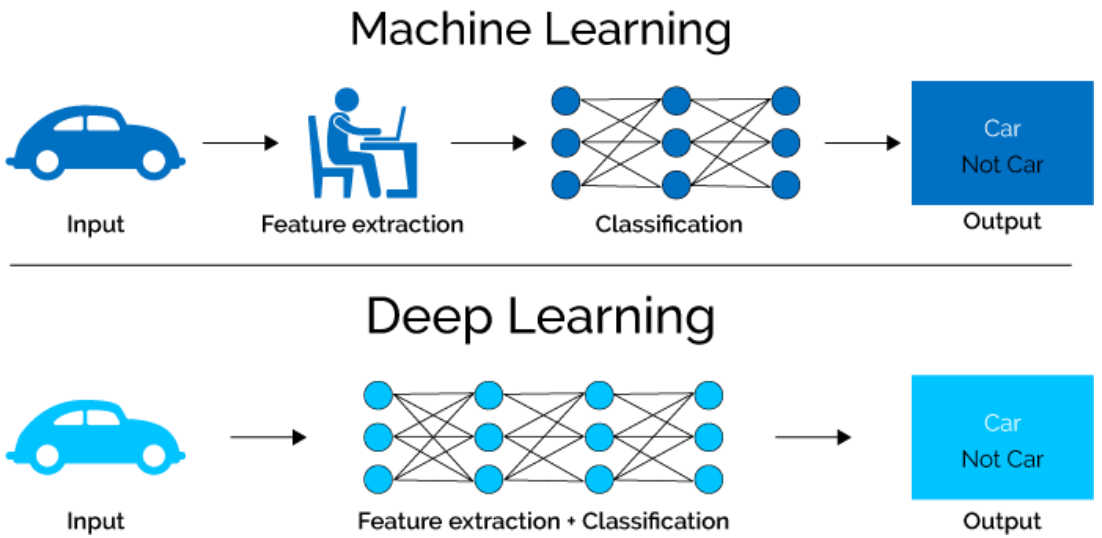
Deep Neural Networks = Many Hidden Layers MLP

Hidden Layers Represent Features …… Calm Down !

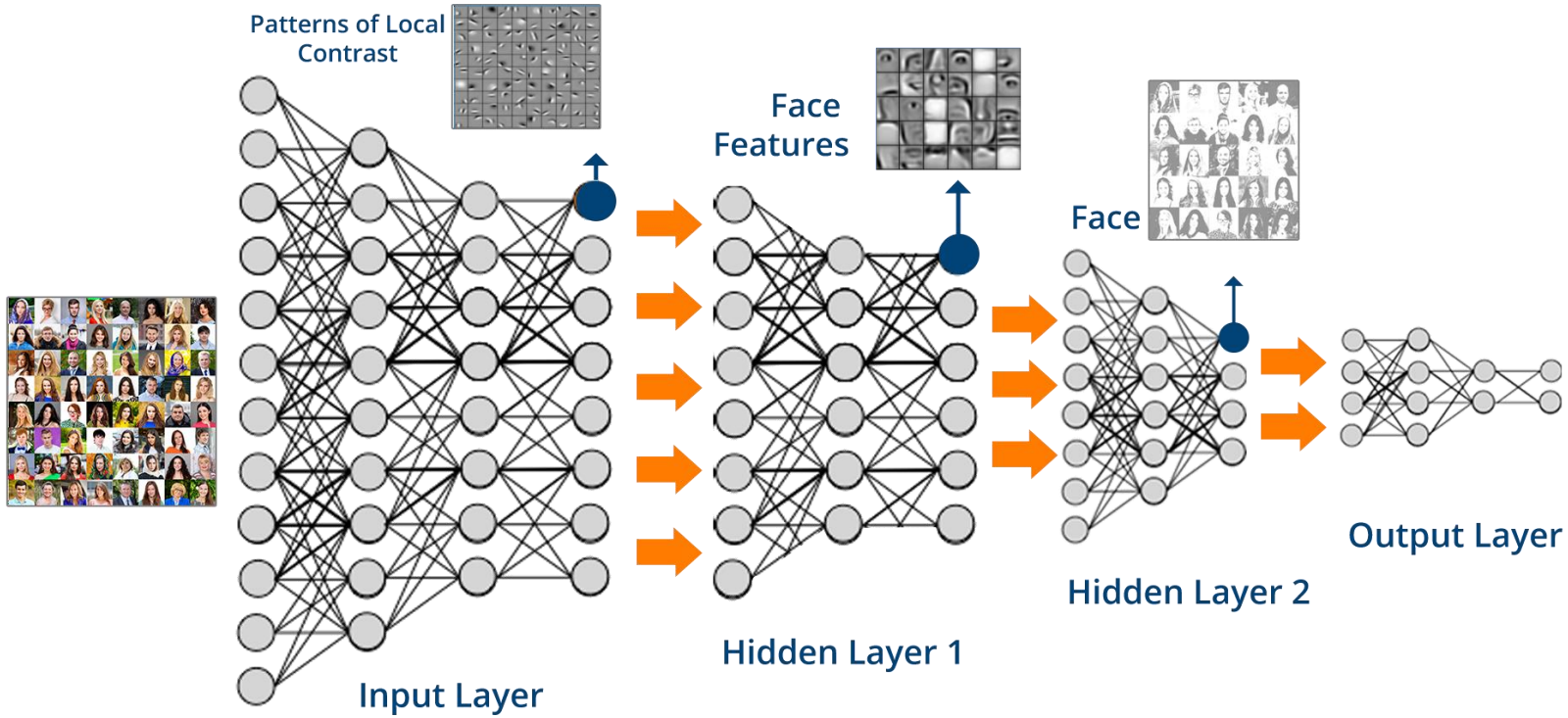
Deep Representation Learning

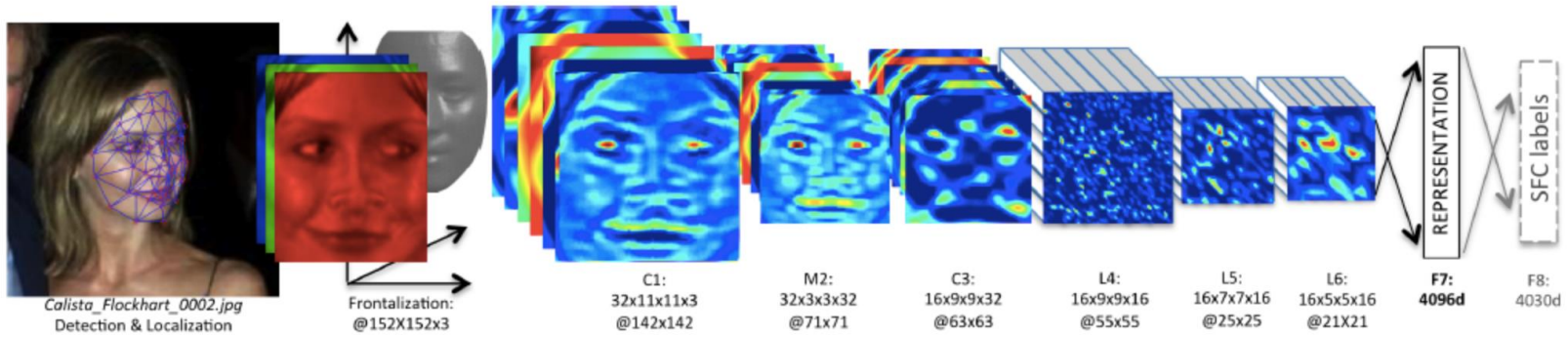




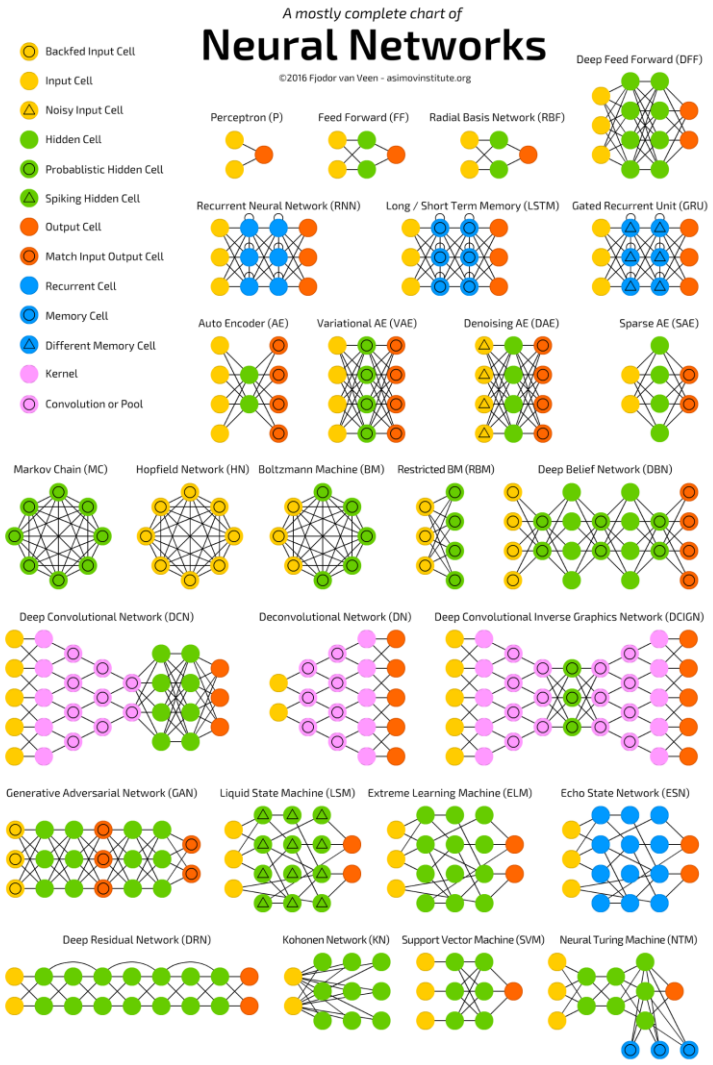


Example: Deep Learning

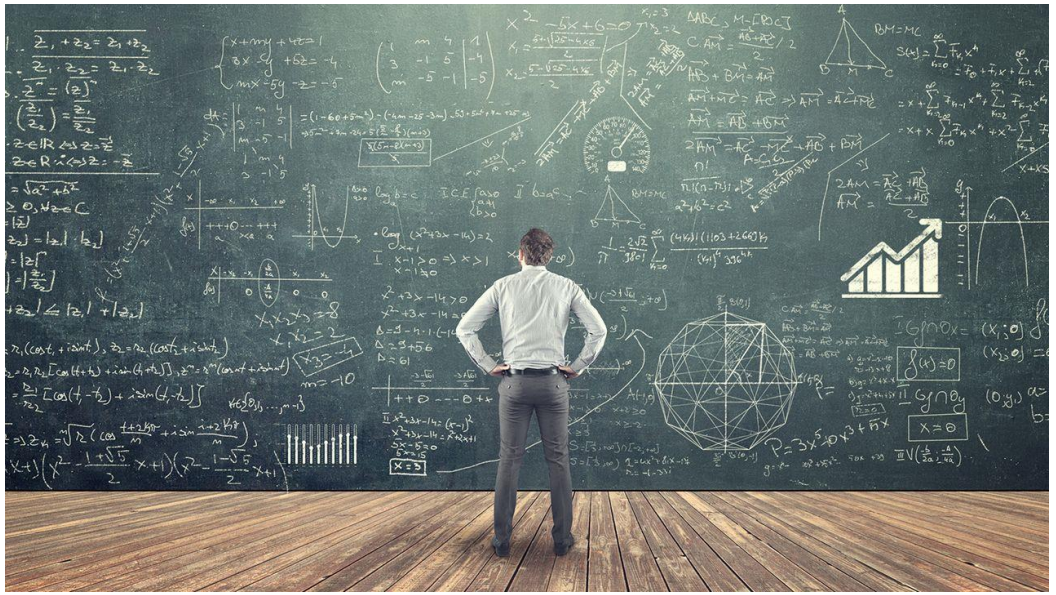




Deep Learning Landscape



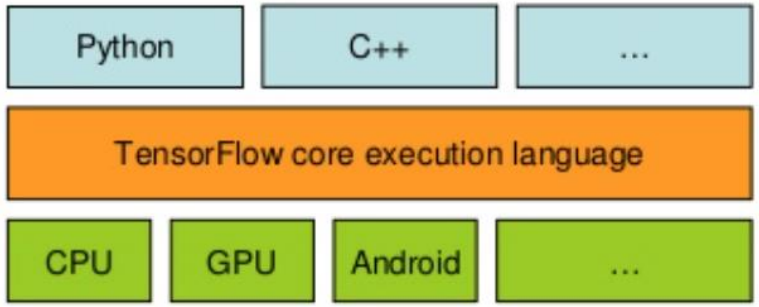
But….
Don’t Skip Math!



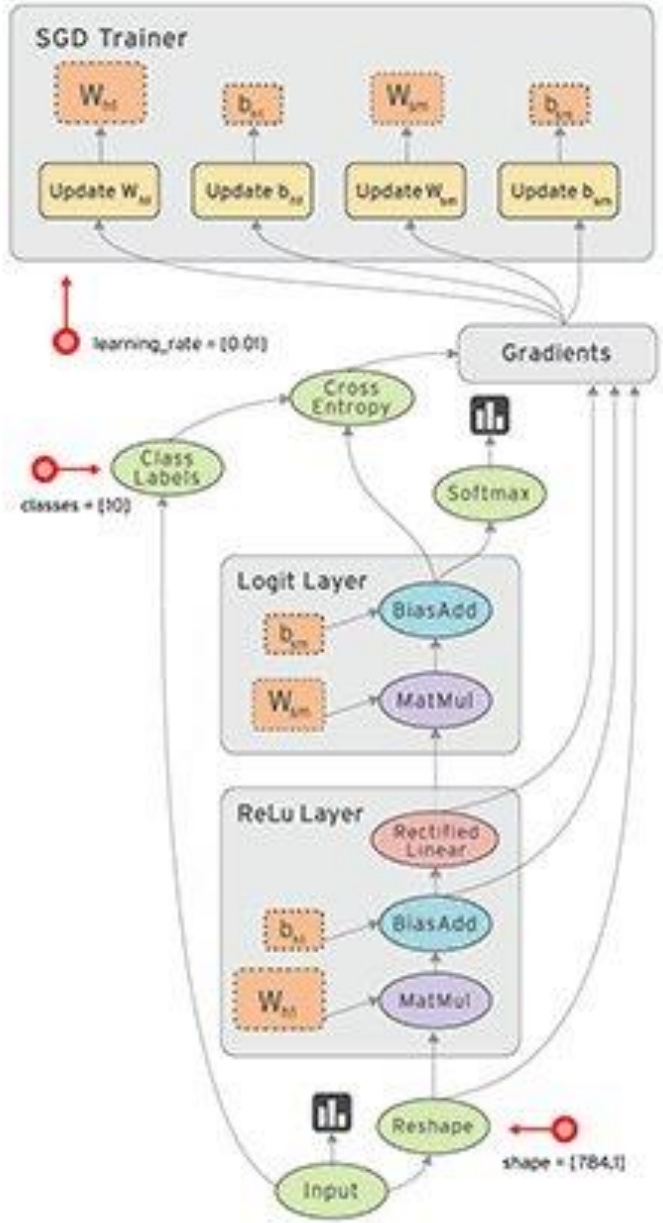
How to Get Started

I Love Tensorflow .....

- **Tensor**
  - **Definition**: An array with more than two axes
  - Arbitrary dimensionality array
- **Directed Graph**
  - Used describes computation
  - **Node**: Instantiation of an operation
- **Operation**
  - An abstract computation
  - Have attributes
- **Kernel**
  - Particular implementation of an operation
  - Run on a type of device (e.q. CPU, GPU)
- **Variable**
  - Special operation to persistent mutable tensor
- **Session**
  - Created to interact with TensorFlow system



Download at www.tensorflow.org



But Many Frameworks

Additional: Keras

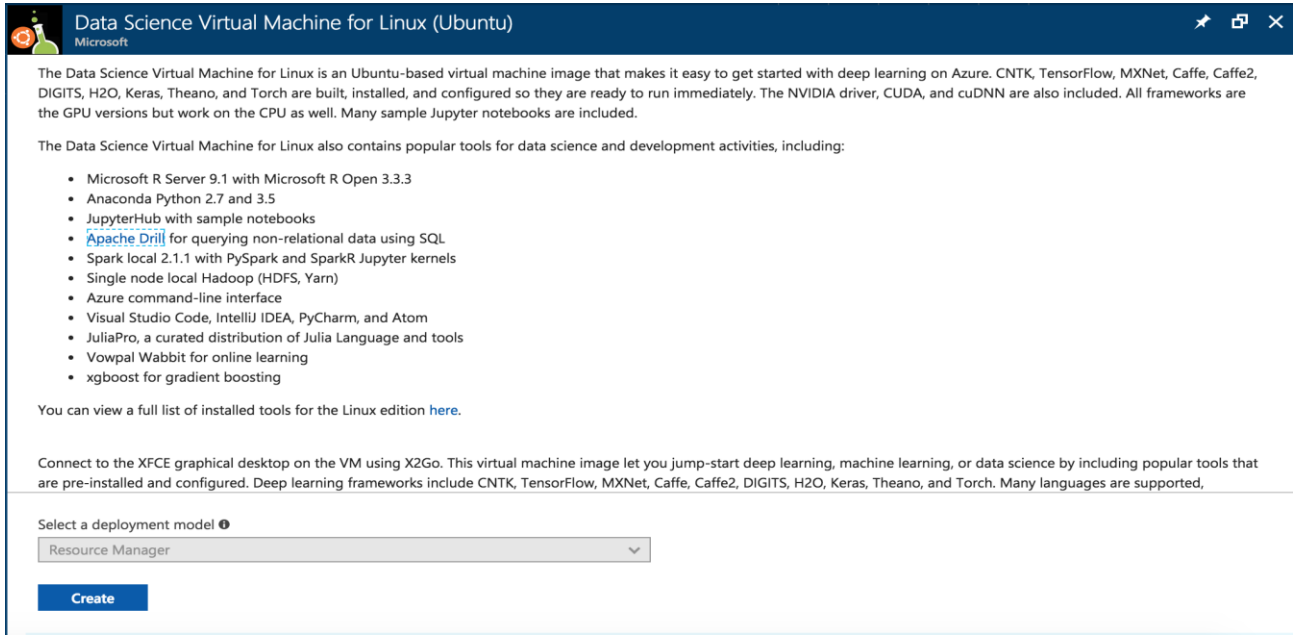
	Caffe	**Torch**	**Theano**	**TensorFlow**
Language	C++, Python	Lua	Python	Python
Pretrained	Yes ++	Yes ++	Yes (Lasagne)	Inception
Multi-GPU: Data parallel	Yes	Yes cunn. DataParallelTable	Yes platoon	Yes
Multi-GPU: Model parallel	No	Yes fbcunn.ModelParallel	Experimental	Yes (best)
Readable source code	Yes (C++)	Yes (Lua)	No	No
Good at RNN	No	Mediocre	Yes	Yes (best)

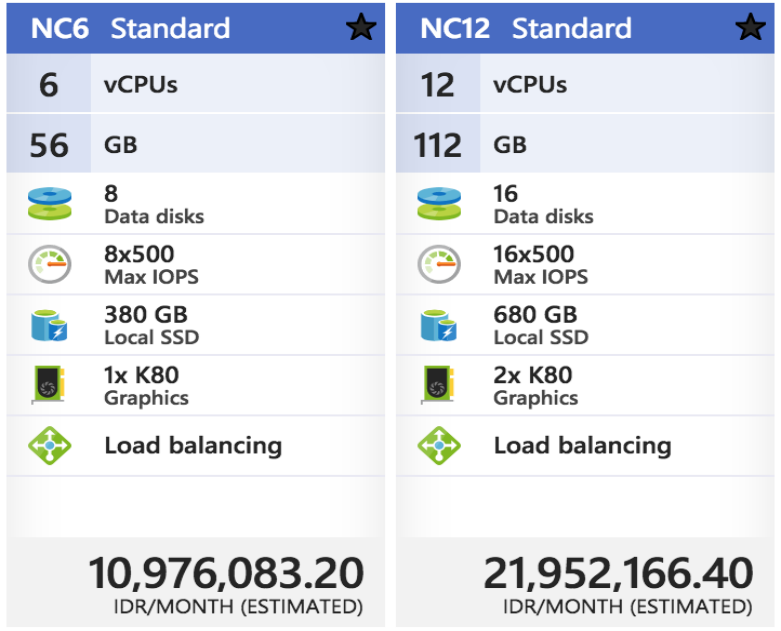
TensorFlow – Pros & Cons

- (+) Python + Numpy (Anaconda)
- (+) Computational graph abstraction, like Theano; great for RNNs
- (+) Much faster compile times than Theano
- (+) Slightly more convenient than raw Theano?
- (+) TensorBoard for visualization and Tensorflow Lite for Mobile
- (+) Data AND model parallelism; best of all frameworks
- (+/-) Distributed models, but not open-source yet
- (-) Slower than other frameworks right now
- (-) Much “fatter” than Torch; more magic
- (-) Not many pre-trained models

Use GPU Cloud Please ….

- **CPU is too slow! Please use GPU instance.**
- **You can use both Azure and Amazon EC2**
- **Azure GPU instance is available in South Central US Region.**

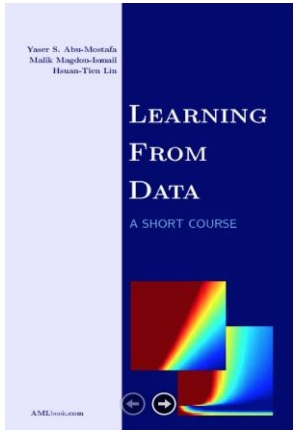


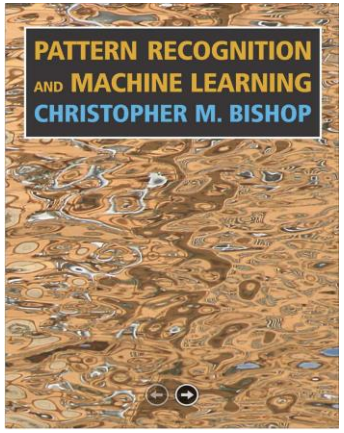


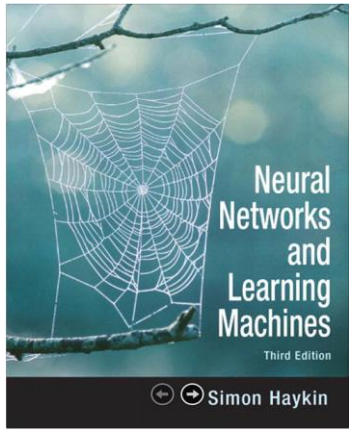
For Amazon EC2 GPU Guide, please look here: https://aws.amazon.com/ec2/elastic-gpus/

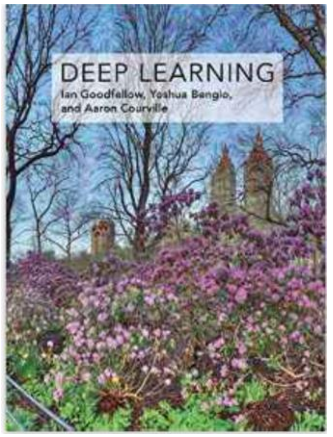
Additional Reading Materials

- Neural Networks and Deep Learning, **Michael Nielsen**
- Learning from Data, **Yaser S. Abu-Mostafa**
- Pattern Recognition and Machine Learning, **Christopher Bishop**
- Neural Networks and Learning Machines, **Simon Haykin**
- Neural Networks, Coursera, **Geoff Hinton**
- Convolutional Neural Networks, Stanford CS231s, **Fei-Fei Li**
- Deep Learning**, Ian Goodfellow, Yoshua Bengio, Aaron Courville**









Q&A

See you in next meet up