Week 9: Introduction to Machine Learning

ASTR 324

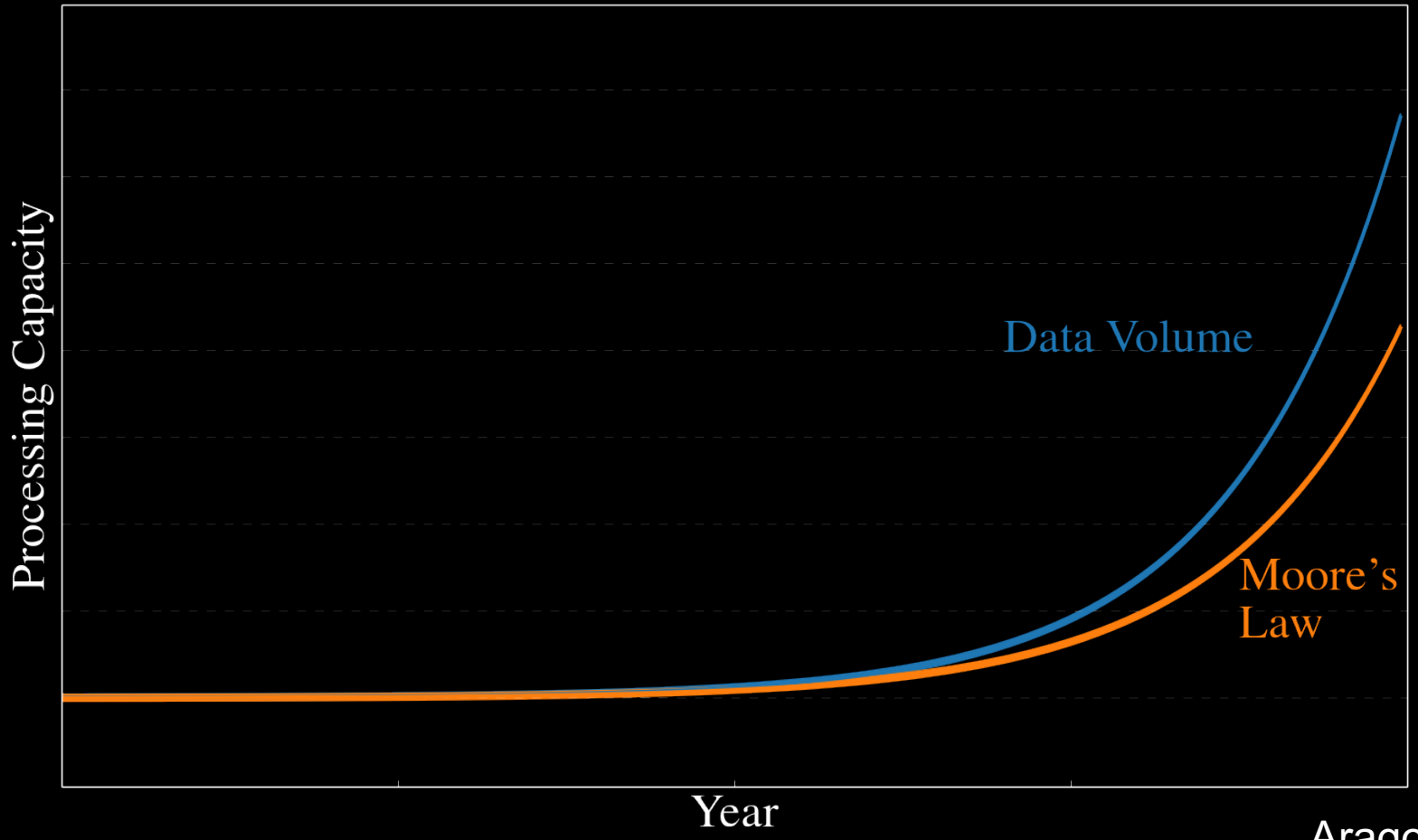
University of Washington, DIRAC Institute

Data Intensive Research in Astrophysics and Cosmology

Motivation: Astronomy in the 2020ies

- LSST, WFIRST, Euclid, …
- Hundreds of PB of imaging data
- Tens of billions of observed objects
- Trillions of observations

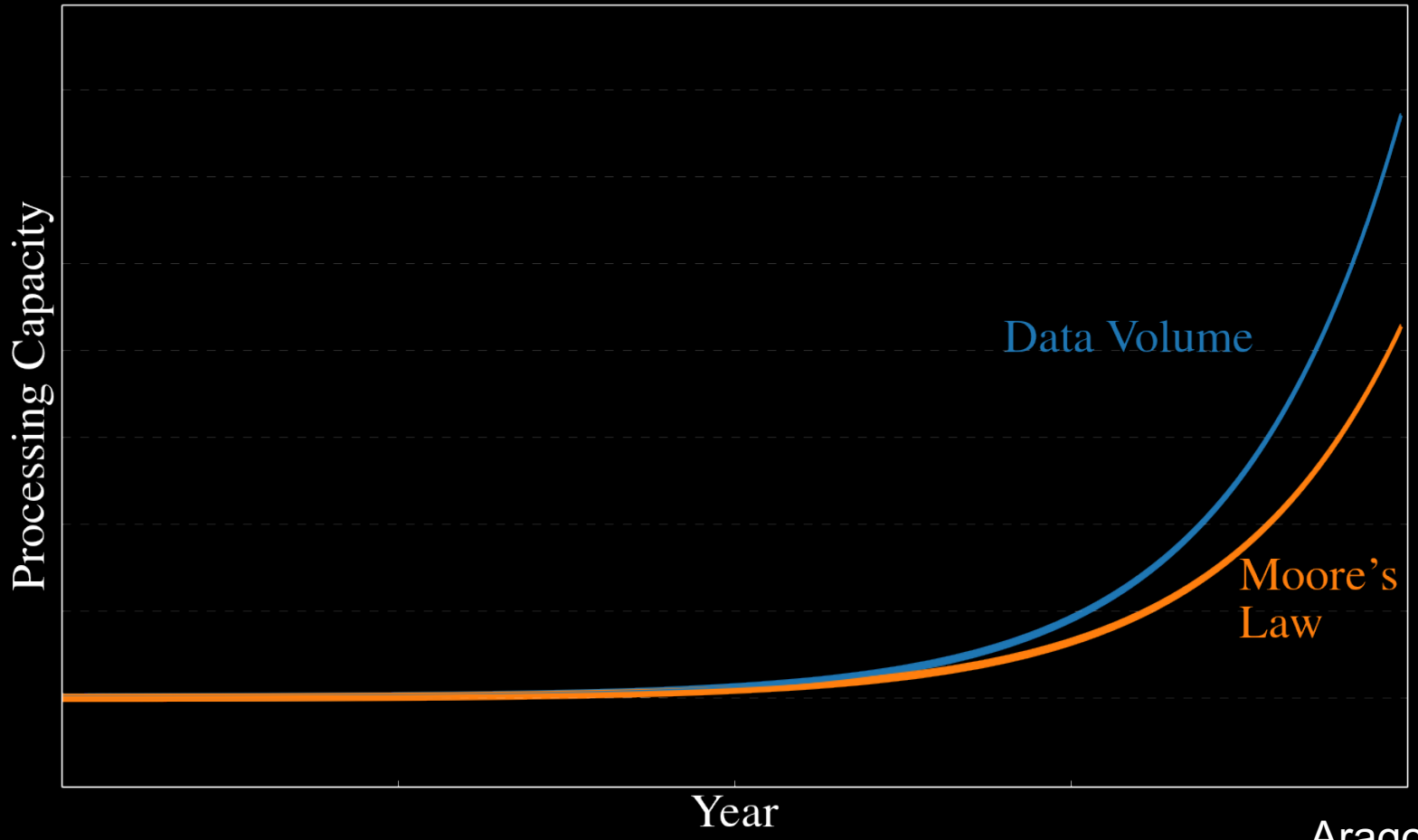
Can we reduce all these data?



Aragon 2008

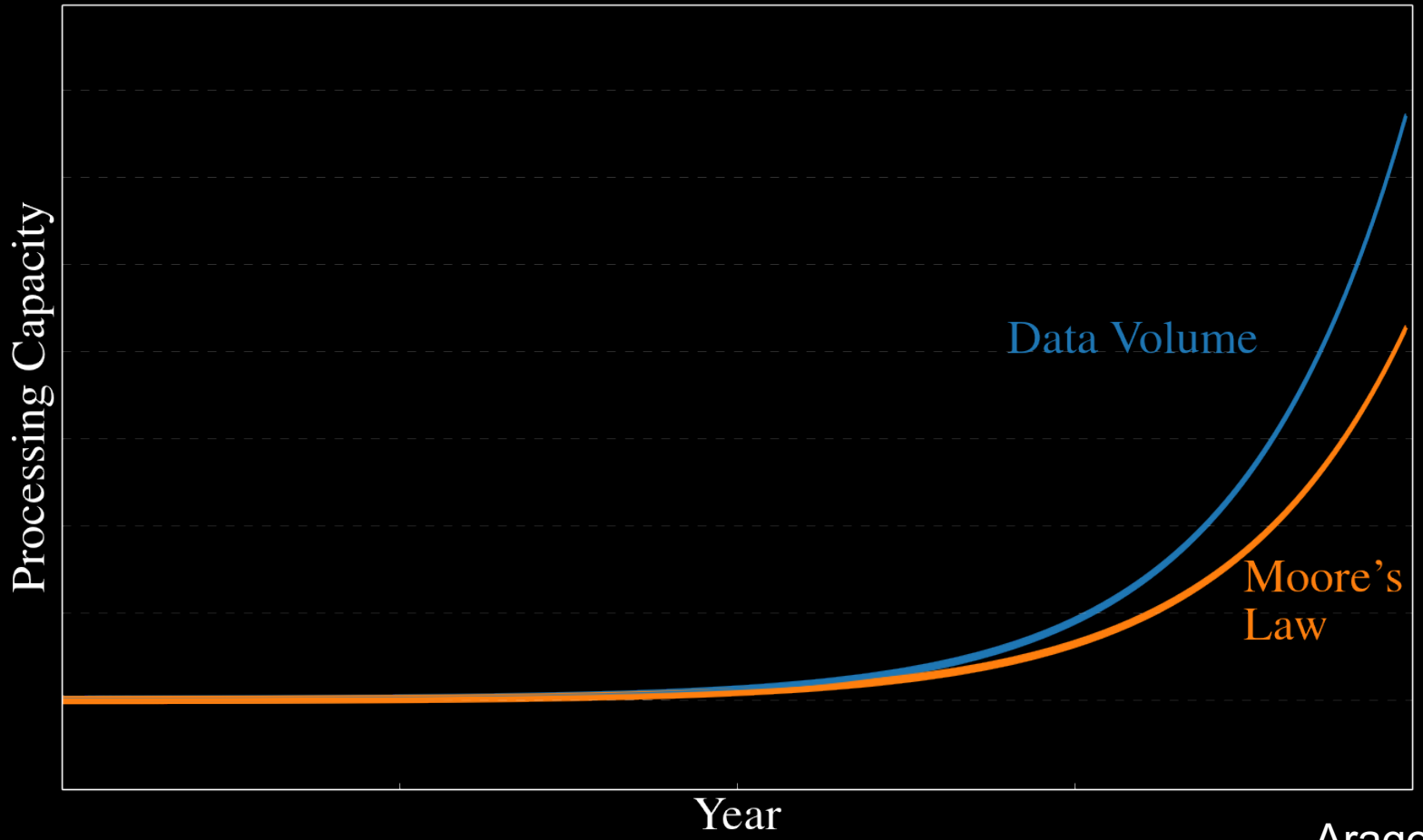
A small supercomputer facility (1.8 PFlops)

Can we reduce all these data?



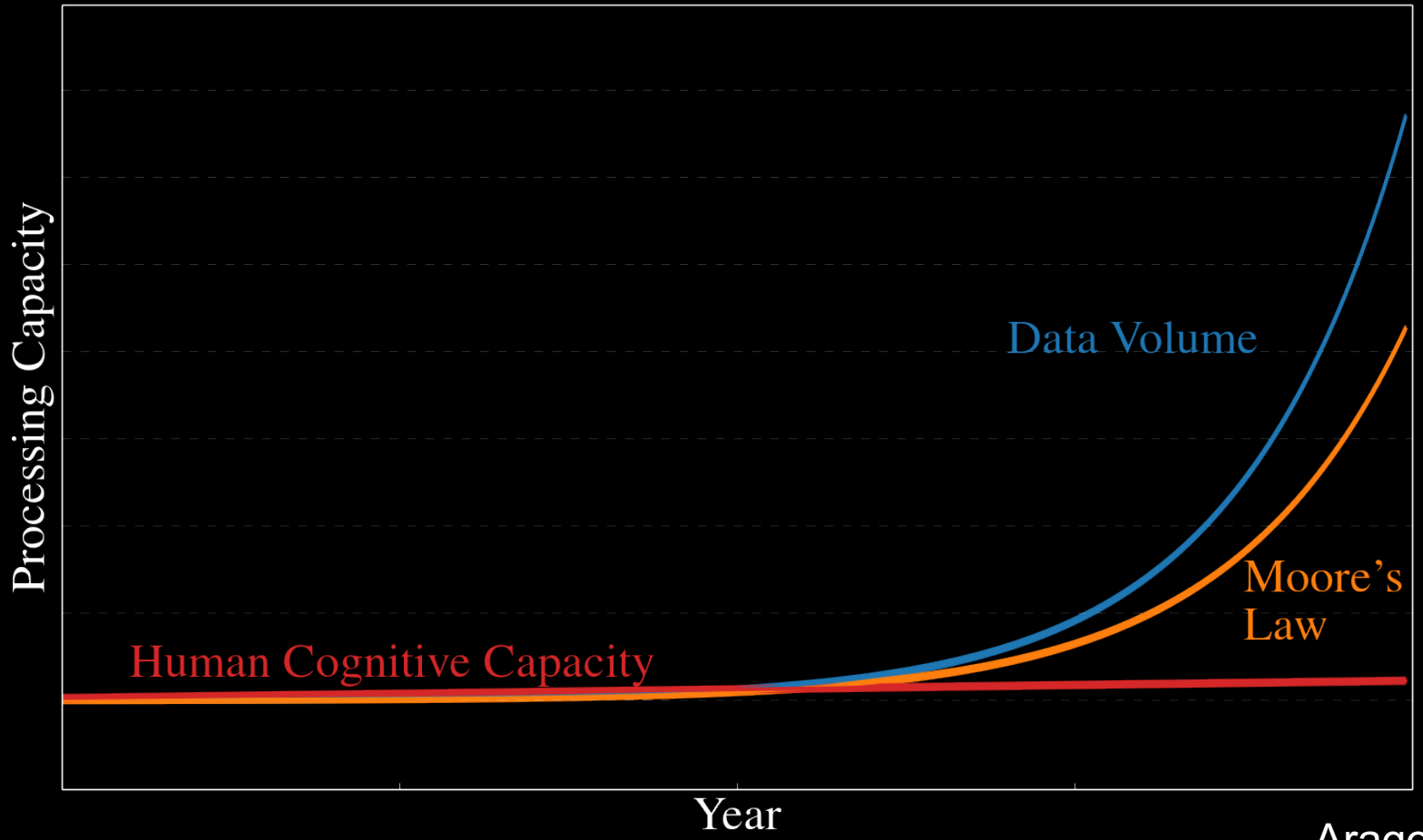
Aragon 2008

Can we *comprehend* all that it's telling us?



Aragon 2008

Can we *comprehend* all that it's telling us?



Aragon 2008

The story so far

Computers were our mental beasts of burden.

We used them to perform menial tasks and carry loads that we as humans are too slow (or too distracted) to do.

Examples: Detect stars on all images. Measure their positions. Measure their brightness.

The “that’s funny” moments were largely reserved for us.

Looking ahead

- Future datasets will be incredibly large (~trillions of rows), complex (~thousands of features), with potentially interesting signals buried deep within the noise.

- To even notice some of those signals will be beyond our cognitive capacity (try visualizing a 100-dimensional space!). We need the computer to find these, and point them out.

- To deduce the causal relationships, to “connect the dots”, may be out of our reach as well. We may need to understand how to teach the computer to do this for us as well.

Machine Learning

What is Machine Learning?

Machine learning (ML) is the scientific
study of algorithms and statistical models that computer systems use to progressively improve their performance on a specific task.

-- Wikipedia

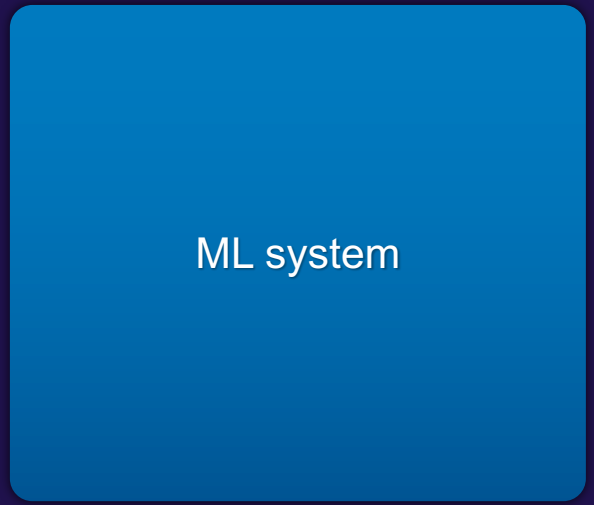
What is Machine Learning?

A system that learns from examples, rather than being explicitly programmed what to do.

-- colloquially

A Toy Example (Training Phase)

	ftr_Nz2	emissivity
0	30.571119	48.971987
1	1.271871	19.341034
2	18.887398	36.707097
3	14.590665	34.120698
4	17.721658	36.171351
5	2.089411	20.348163
6	42.661662	60.551494
7	0.591402	19.066962
8	2.353133	21.515238
9	29.955232	48.188376
10	23.045119	42.420823



Note: The “ML box” was not pre-programmed with these data in mind(*).

A Toy Example (Inference Phase)

ML system

Note: The “ML box” was not pre-programmed with these data in mind(*).

	ftr_Nz2	emissivity_pred	emissivity_true	error
0	12.017282	30.573918	30.582191	-0.000271
1	25.702186	44.233306	43.717787	0.011792
2	44.786225	63.281763	64.199395	-0.014293
3	38.051029	56.559124	56.392191	0.002960
4	42.186956	60.687340	59.935506	0.012544

Inference

What can these systems do?

An approximate taxonomy of machine learning

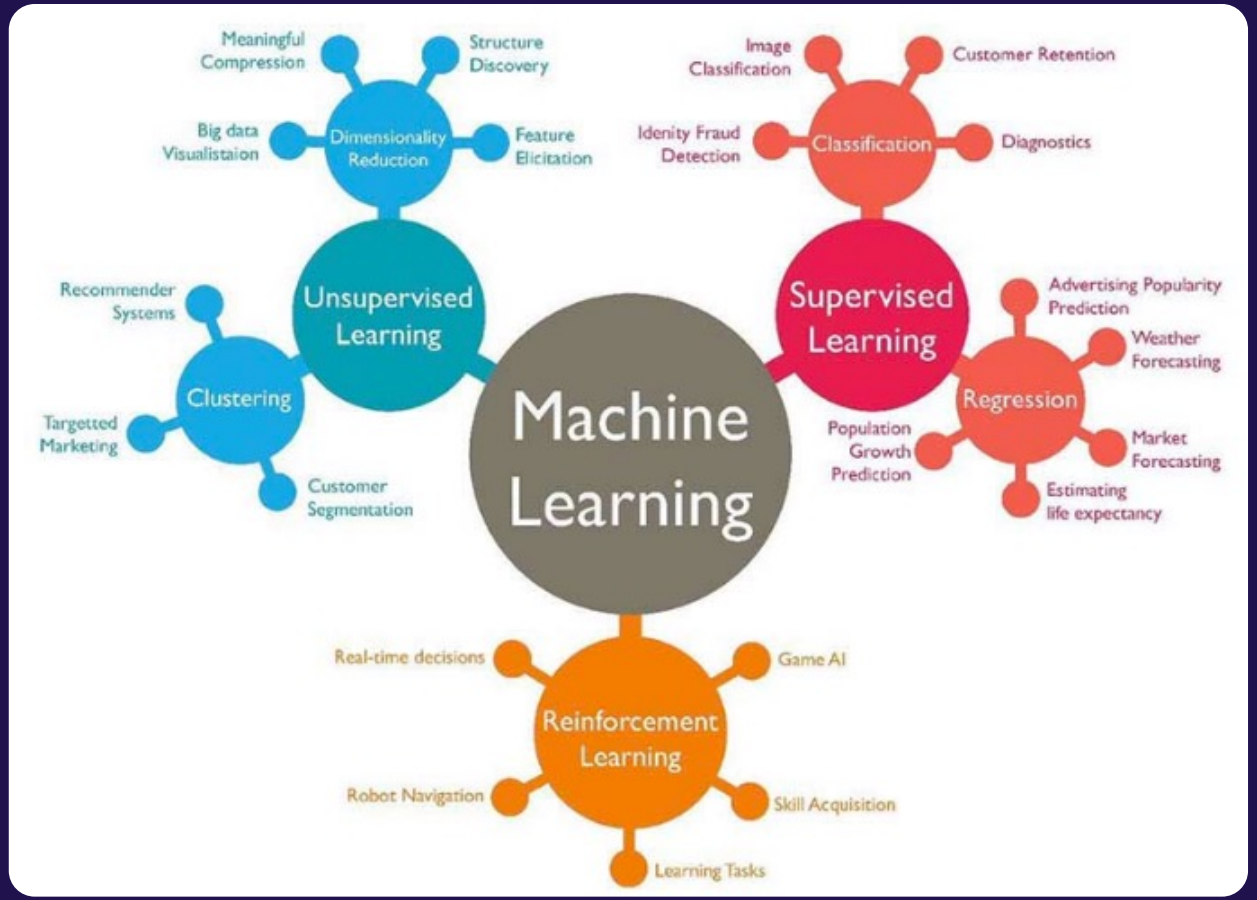


Figure: https://medium.com/@machadogj/ml-basics-supervised-unsupervised-and-reinforcement-learning-b18108487c5a

An approximate taxonomy of machine learning

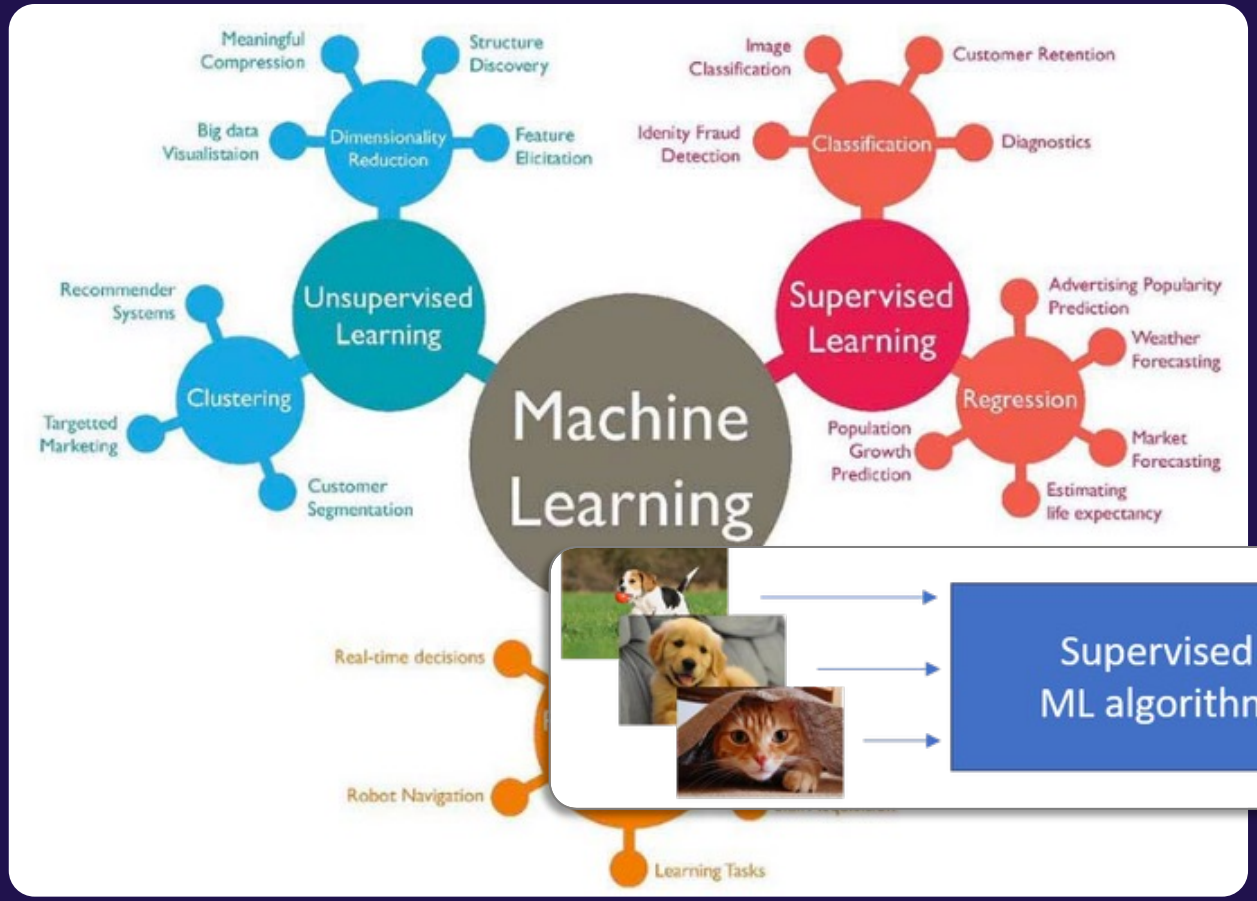


Figure: https://medium.com/@machadogj/ml-basics-supervised-unsupervised-and-reinforcement-learning-b18108487c5a

An approximate taxonomy of machine learning

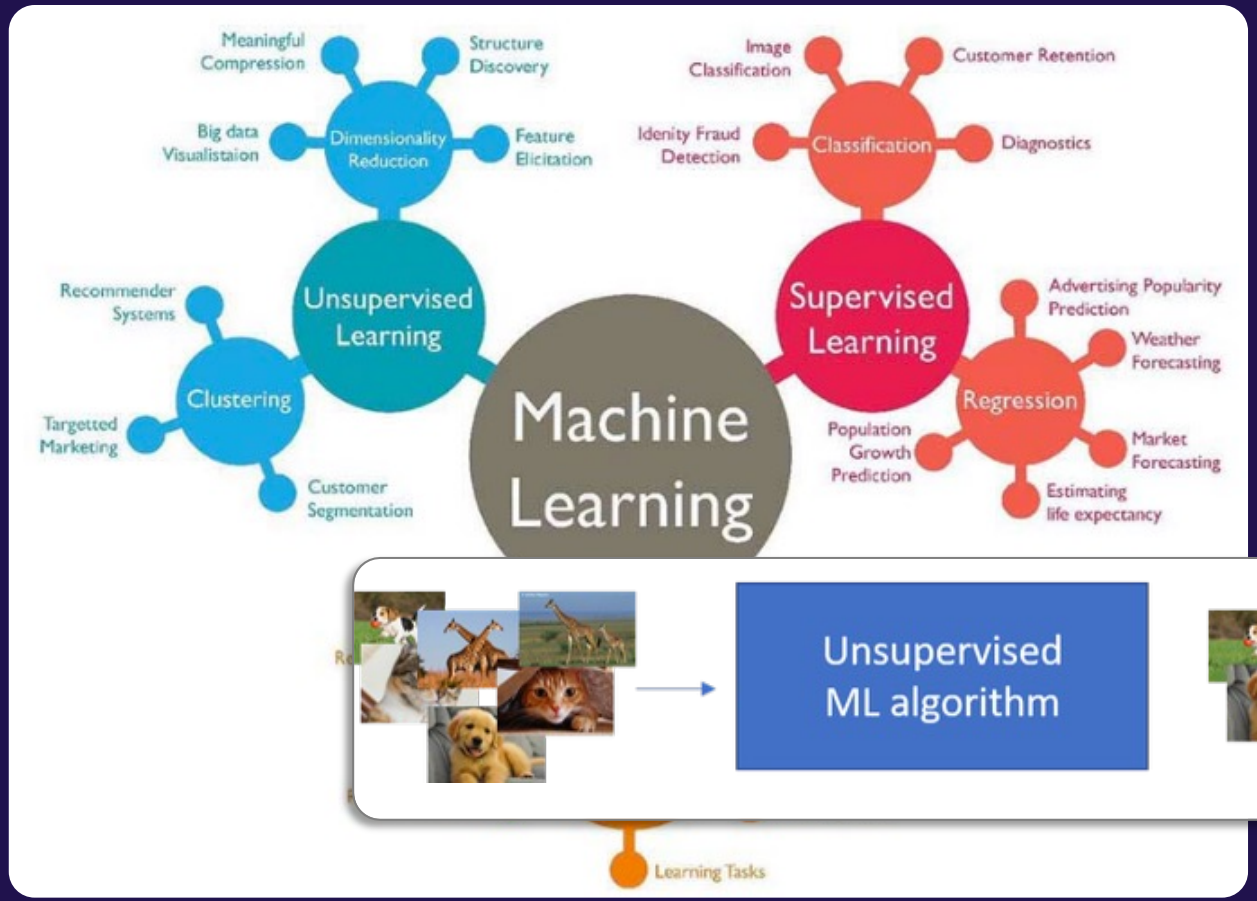
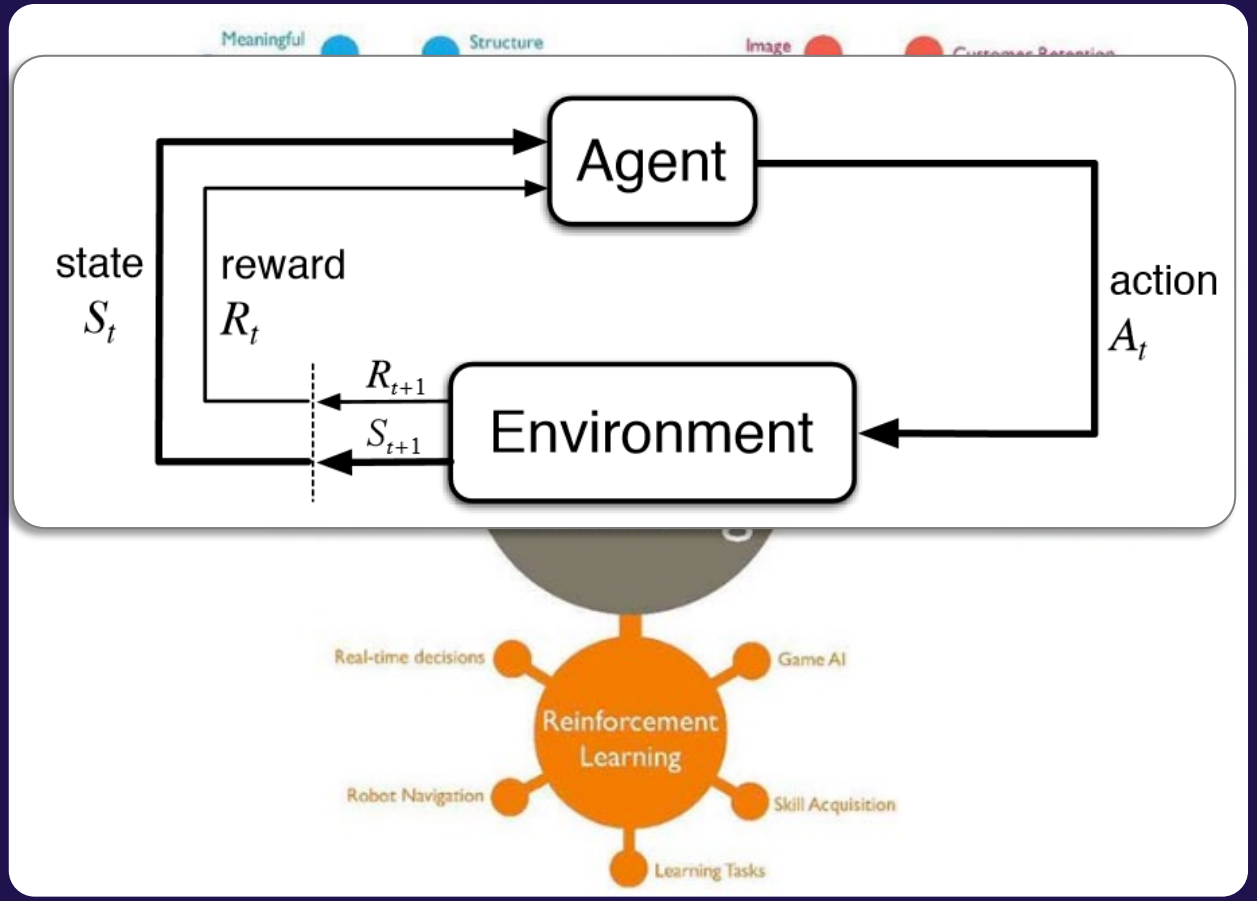


Figure: https://medium.com/@machadogj/ml-basics-supervised-unsupervised-and-reinforcement-learning-b18108487c5a

An approximate taxonomy of machine learning



Figures: https://www.geeksforgeeks.org/what-is-reinforcement-learning

ARTICLE

doi:10.1038/nature16961

Mastering the game of Go with deep neural networks and tree search

David Silver[1]*, Aja Huang[1]*, Chris J. Maddison[1], Arthur Guez[1], Laurent Sifre[1], George van den Driessche[1], Julian Schrittwieser[1], Ioannis Antonoglou[1], Veda Panneershelvam[1], Marc Lanctot[1], Sander Dieleman[1], Dominik Grewe[1], John Nham[2], Nal Kalchbrenner[1], Ilya Sutskever[2], Timothy Lillicrap[1], Madeleine Leach[1], Koray Kavukcuoglu[1], Thore Graepel[1] & Demis Hassabis[1]

The game of Go has long been viewed as the most challenging of classic games for artificial intelligence owing to its enormous search space and the difficulty of evaluating board positions and moves. Here we introduce a new approach to computer Go that uses 'value networks' to evaluate board positions and 'policy networks' to select moves. These deep neural networks are trained by a novel combination of supervised learning from human expert games, and reinforcement learning from games of self-play. Without any lookahead search, the neural networks play Go at the level of state-of-the-art Monte Carlo tree search programs that simulate thousands of random games of self-play. We also introduce a new search algorithm that combines Monte Carlo simulation with value and policy networks. Using this search algorithm, our program AlphaGo achieved a 99.8% winning rate against other Go programs, and defeated the human European Go champion by 5 games to 0. This is the first time that a computer program has defeated a human professional player in the full-sized game of Go, a feat previously thought to be at least a decade away.

Mastering the game of Go without human knowledge

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel & Demis Hassabis

Nature **550**, 354–359 (19 October 2017) | Download Citation

Abstract

A long-standing goal of artificial intelligence is an algorithm that learns, *tabula rasa*, superhuman proficiency in challenging domains. Recently, AlphaGo became the first program to defeat a world champion in the game of Go. The tree search in AlphaGo evaluated positions and selected moves using deep neural networks. These neural networks were trained by supervised learning from human expert moves, and by reinforcement learning from self-play. Here we introduce an algorithm based solely on reinforcement learning, without human data, guidance or domain knowledge beyond game rules. AlphaGo becomes its own teacher: a neural network is trained to predict AlphaGo's own move selections and also the winner of AlphaGo's games. This neural network improves the strength of the tree search, resulting in higher quality move selection and stronger self-play in the next iteration. Starting ***tabula rasa***, our new program AlphaGo Zero achieved superhuman performance, winning 100–0 against the previously published, champion-defeating AlphaGo.

What’s in the ML Box?

A Toy Example

	ftr_Nz2	emissivity
0	30.571119	48.971987
1	1.271871	19.341034
2	18.887398	36.707097
3	14.590665	34.120698
4	17.721658	36.171351
5	2.089411	20.348163
6	42.661662	60.551494
7	0.591402	19.066962
8	2.353133	21.515238
9	29.955232	48.188376
10	23.045119	42.420823

ML system

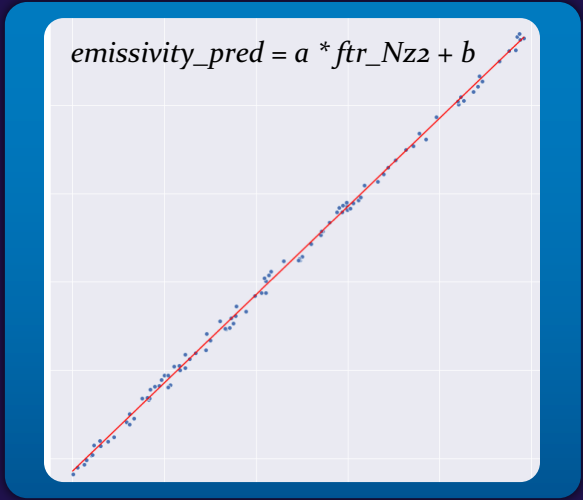
Training

Inference

	ftr_Nz2	emissivity_pred	emissivity_true	error
0	12.017282	30.573918	30.582191	-0.000271
1	25.702186	44.233306	43.717787	0.011792
2	44.786225	63.281763	64.199395	-0.014293
3	38.051029	56.559124	56.392191	0.002960
4	42.186956	60.687340	59.935506	0.012544

A Toy Example

	ftr_Nz2	emissivity
0	30.571119	48.971987
1	1.271871	19.341034
2	18.887398	36.707097
3	14.590665	34.120698
4	17.721658	36.171351
5	2.089411	20.348163
6	42.661662	60.551494
7	0.591402	19.066962
8	2.353133	21.515238
9	29.955232	48.188376
10	23.045119	42.420823



Training: Fitting the coefficients a, b
Inference: Evaluating the model

	ftr_Nz2	emissivity_pred	emissivity_true	error
0	12.017282	30.573918	30.582191	-0.000271
1	25.702186	44.233306	43.717787	0.011792
2	44.786225	63.281763	64.199395	-0.014293
3	38.051029	56.559124	56.392191	0.002960
4	42.186956	60.687340	59.935506	0.012544

Training

Inference

Machine learning	Statistics
network, graphs	model
weights	parameters
learning	fitting
generalization	test set performance
supervised learning	regression/classification
unsupervised learning	density estimation, clustering
large grant = $1,000,000	large grant = $50,000
nice place to have a meeting: Snowbird, Utah, French Alps	nice place to have a meeting: Las Vegas in August

-- Rob Tibshirani, Stanford

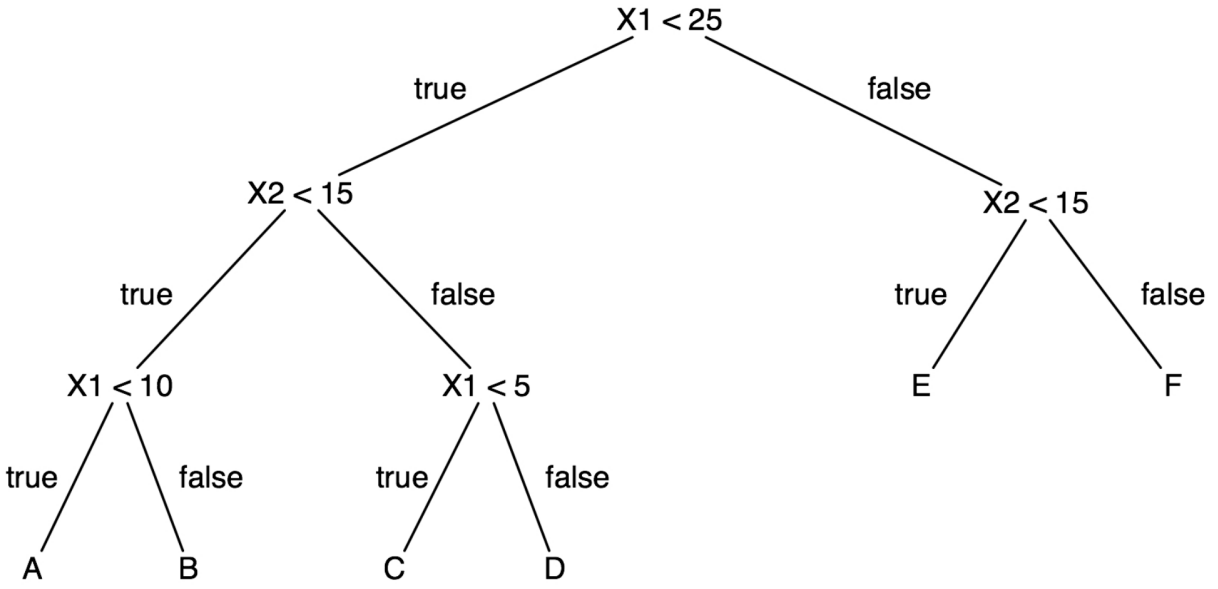
What's in the Box

Techniques to enable machine learning behavior are numerous. Some examples:

- Support Vector Machines
- Decision Trees
- Random Forests
- Artificial Neural Networks
- …

Some work better in certain domains (or are faster, or are easier to apply).
We'll discuss trees and ANNs.

Decision Trees



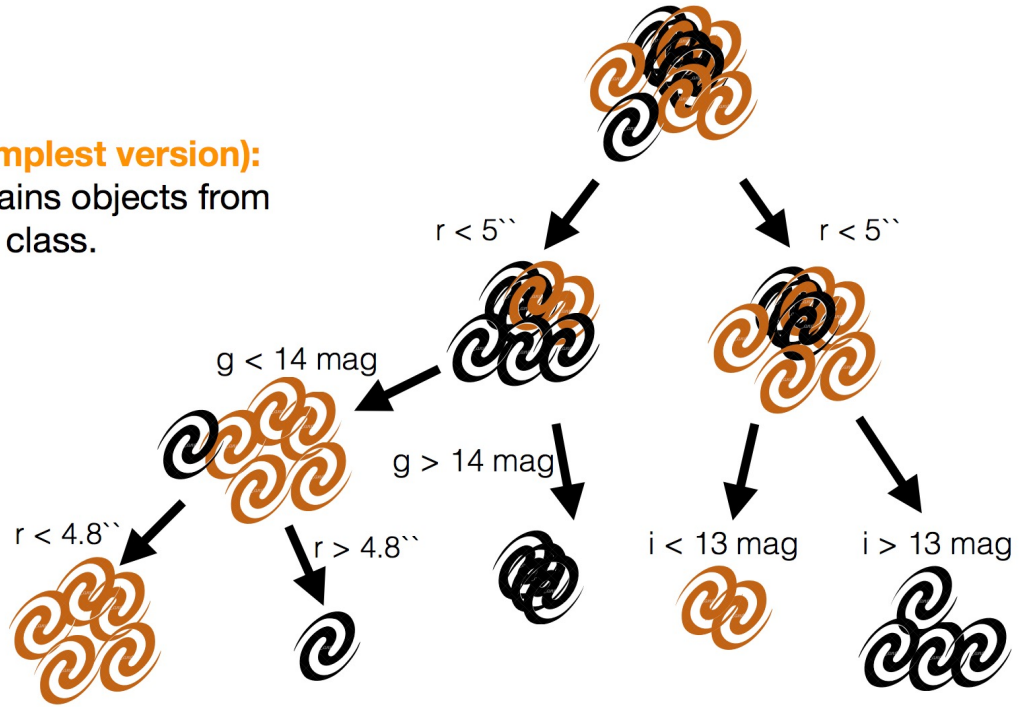
Decision tree: a non-parametric model, constructed during training, which is described by a tree-like graph. It can be used for classification or regression.

From lectures on Machine Learning
by Dalya Baron

Decision Tree Construction

Input training set: a list of objects with measured features and known labels.
Classes: “black” and “brown” galaxies.
Measured features: r (arcsec), g (mag), i(mag).

Stop criterion (simplest version):
each terminal contains objects from a single class.



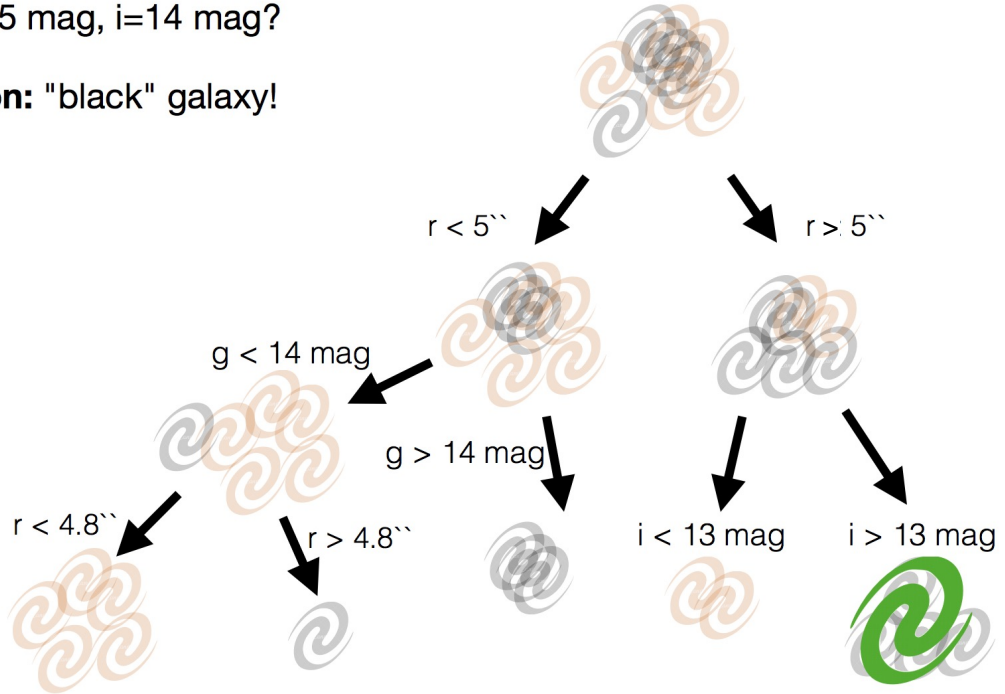
From lectures on Machine Learning by Dalya Baron

Decision Tree Prediction

Input set: a list of objects with measured features and unknown labels.
Objects are propagated through the tree according to their measured features.

Example: what is the predicted label for a galaxy with the measured features:
r=8``, g=15 mag, i=14 mag?

Prediction: "black" galaxy!



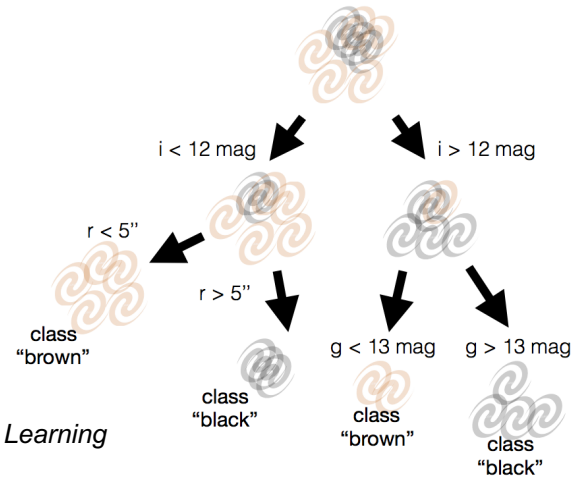
From lectures on Machine Learning by Dalya Baron

Random Forests

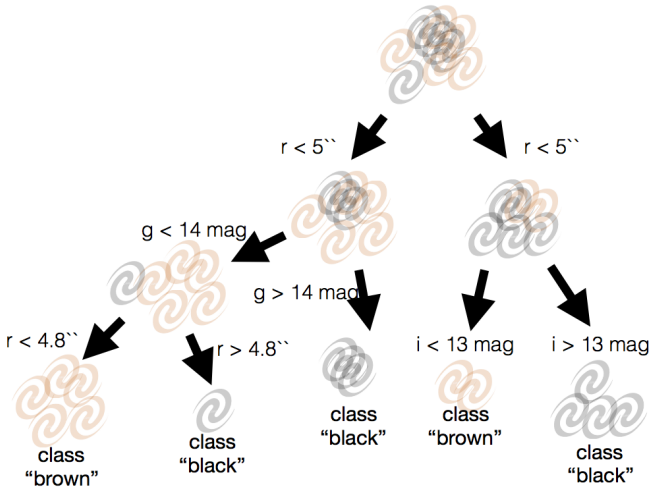
Random Forest is an ensemble of decision trees, where **randomness** is injected into the training process of each individual tree with a **bagging** approach.

Bagging: -The training set is split into randomly-selected subsets, and each decision tree is trained on a subset of the data.
-In each node in the decision tree, only a randomly-selected subset of the feature is considered.

decision tree #2

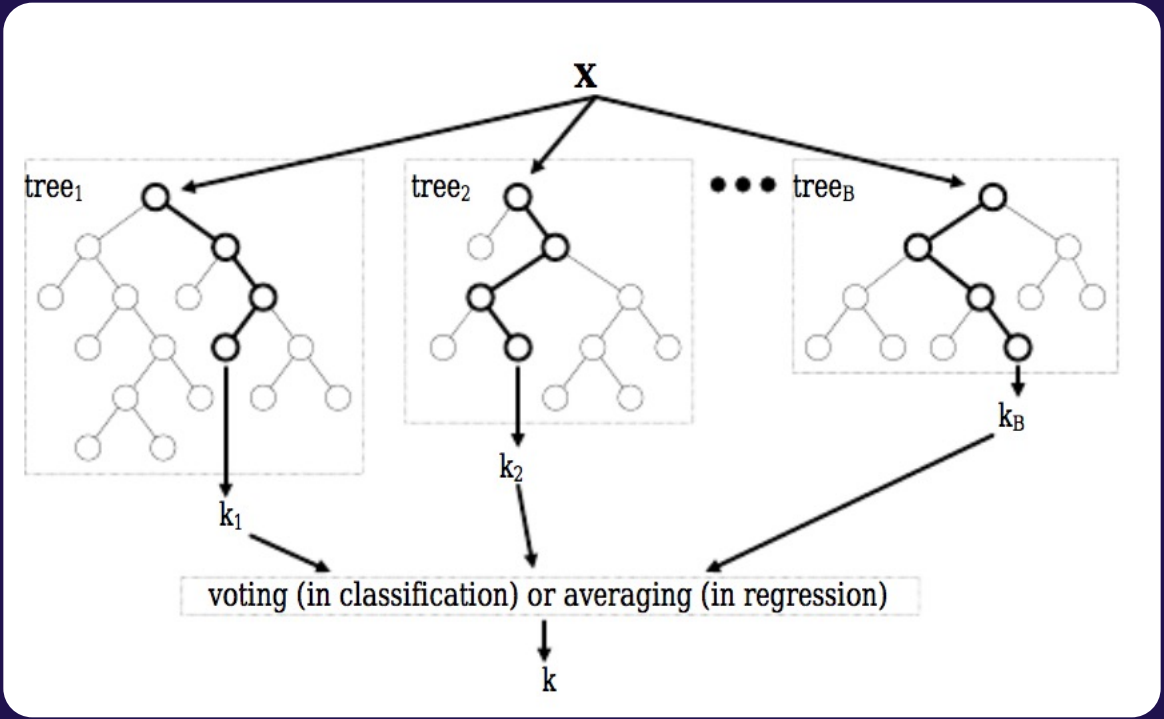


decision tree #1



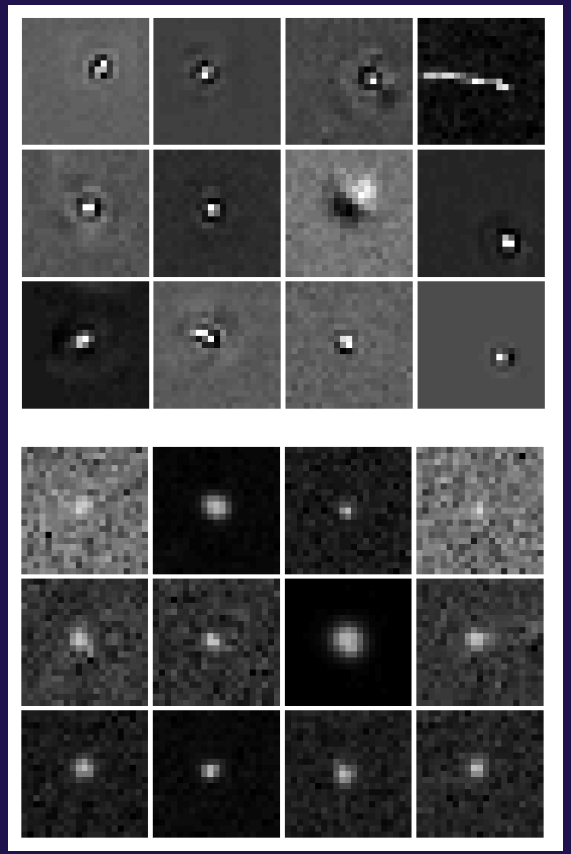
From lectures on Machine Learning by Dalya Baron

Inference with Random Forests



Verikas et al. (2016)

Applications in Astronomy: “Real-Bogus Classifiers”



Set	Selected	Feature	Description
RB1		mag	USNO-B1.0 derived magnitude of the candidate on the difference image
		mag_err	estimated uncertainty on mag
		a_image	semi-major axis of the candidate
	✓	b_image	semi-minor axis of the candidate
		fwhm	full-width at half maximum (FWHM) of the candidate
	✓	flag	numerical representation of the SExtractor extraction flags
	✓	mag_ref	magnitude of the nearest object in the reference image if less than 5 arcsec from the candidate
	✓	mag_ref_err	estimated uncertainty on mag_ref
	✓	a_ref	semi-major axis of the reference source
	✓	b_ref	semi-minor axis of the reference source
		n2sig3	number of at least negative 2 σ pixels in a 5×5 box centered on the candidate
		n3sig3	number of at least negative 3 σ pixels in a 5×5 box centered on the candidate
		n2sig5	number of at least negative 2 σ pixels in a 7×7 box centered on the candidate
	✓	n3sig5	number of at least negative 3 σ pixels in a 7×7 box centered on the candidate
	✓	flux_ratio	ratio of the aperture flux of the candidate relative to the aperture flux of the reference source
		ellipticity	ellipticity of the candidate using a_image and b_image
	✓	ellipticity_ref	ellipticity of the reference source using a_ref and b_ref
	✓	nn_dist_renorm	distance in arcseconds from the candidate to reference source
		magdiff	when a reference source is found nearby, the difference between the candidate magnitude and the reference source. Else, the difference between the candidate magnitude and the limiting magnitude of the image
	✓	maglim	True if there is no nearby reference source, False otherwise.
		sigflux	significance of the detection, the PSF flux divided by the estimated uncertainty in the PSF flux
		seeing_ratio	ratio of the FWHM of the seeing on the new image to the FWHM of the seeing on the reference image
	✓	mag_from_limit	limiting magnitude minus the candidate magnitude
		normalized_fwhm	ratio of the FWHM of the candidate to the seeing in the new image
	✓	normalized_fwhm_ref	ratio of the FWHM of the reference source to the seeing in the reference image
	✓	good_cand_density	ratio of the number of candidates in that subtraction to the total usable area on that array
	✓	min_distance_to_edge_in_new	distance in pixels to the nearest edge of the array on the new image
New	✓	ccdid	numerical ID of the specific camera detector (1 – 12)
		sym	Measure of symmetry, based on dividing the object into quadrants
	✓	seeingnew	FWHM of the seeing on the new image
	✓	extracted	number of candidates on that exposure found by Sextractor
	✓	obsaved	number of candidates on that exposure saved to the database (a subset of extracted)
		pos	True for a positive (i.e., brighter) residual, False for a negative (fading) one
	✓	gauss	gaussian best fit squared difference value
		corr	gaussian best fit correlation value
		scale	gaussian scale value
	✓	amp	gaussian amplitude value
	✓	l1	sum of absolute pixel values
		smooth1	filter 1 output
		smooth2	filter 2 output
		pca1	1st principal component
		pca2	2nd principal component
Test		empty	zero for all candidates (i.e., no information)
		random	a random number generated for every candidate (i.e., pure noise)

Table 1. List of all of the features used in our analysis. The first set of features, labeled 'RB1', were first introduced by Bloom et al. (2011) and we repeat here their Table 1. The second, labeled 'New' is introduced here. The last set of features, called 'Test' serves as a benchmark for feature selection in §3.1, where we expect good features to perform better than these. The check-marked as 'selected' represent the optimal subset found by our incremental feature selection algorithm in §3.1.

Problem: In transient searches, image differencing generates many artefacts (”false positive detections”). These overwhelm real candidates (by ~100:1).

Solution: RF-based classifiers.

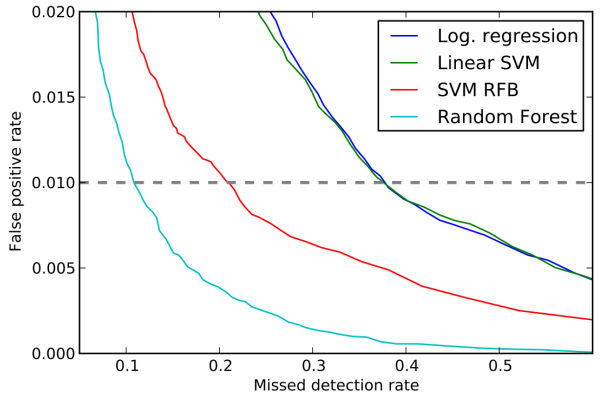


Figure 3. Comparison of a few well known classification algorithms applied to the full dataset. ROC curves enable a trade-off between false positives and missed detections, but the best classifier pushes closer towards the origin. Linear models (Logistic Regression or Linear SVMs) perform poorly as expected, while non-linear models (SVMs with radial basis function kernels or random forests) are much more suited for this problem. Random forests perform well with minimal tuning and efficient training, so we will use those in the remainder of this paper.

Pioneering work by Bloom et al (2011)
Figures from Brink et al (2012)

Real Bogus, Dark Energy Survey

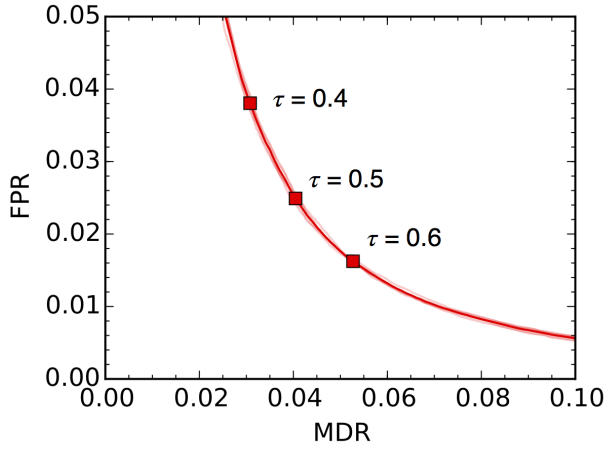


Fig. 7.— 5-fold cross-validated receiver operating characteristics of the best-performing classifier from §3.5. Six visually indistinguishable curves are plotted: one translucent curve for each round of cross-validation, and one opaque curve representing the mean. Points on the mean ROC corresponding to different class discrimination boundaries τ are labeled. $\tau = 0.5$ was adopted in DES-SN.

TABLE 4

CAN ON REPROCESSED DES Y1 TRANSIENT CANDIDATE S

	No ML	ML ($\tau = 0.5$)	ML / No ML
N_c[a]	100,450	7,489	0.075
$\langle N_A/N_{NA}\rangle$[b]	13	0.34	0.027
ϵ_F[c]	1.0	0.990	0.990

[a]Total number of science candidates discovered.

[b]Average ratio of artifact to non-artifact detections in human scanning pool.

[c]`autoScan` candidate-level efficiency for fake SNe Ia.

- **Raw false detection rates of 13:1**
- **Post-filtering rates of 1:3 (!!)**

Stellar Variability

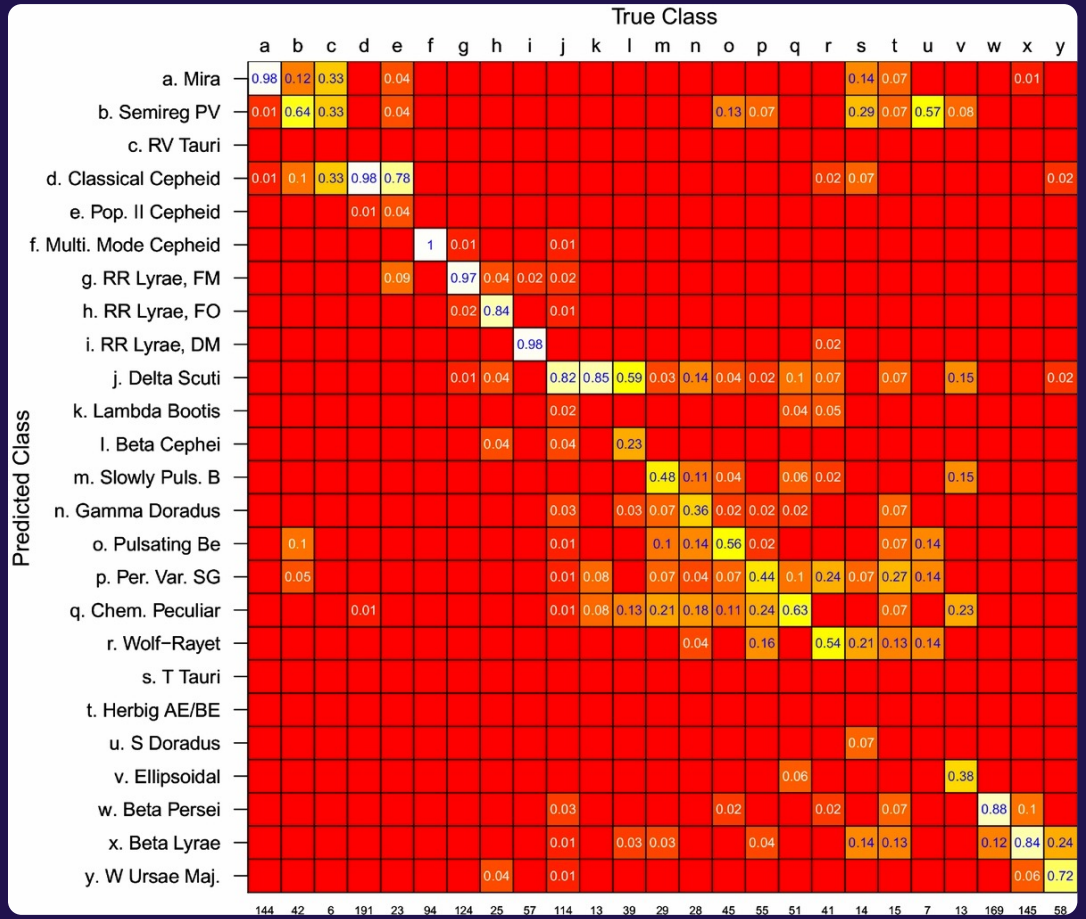
Classifying variable sources in (noisy) survey datasets.

RF-based classifiers trained on features computed from time-series of well-known variables.

Outperformed all other classifiers (by ~25%).

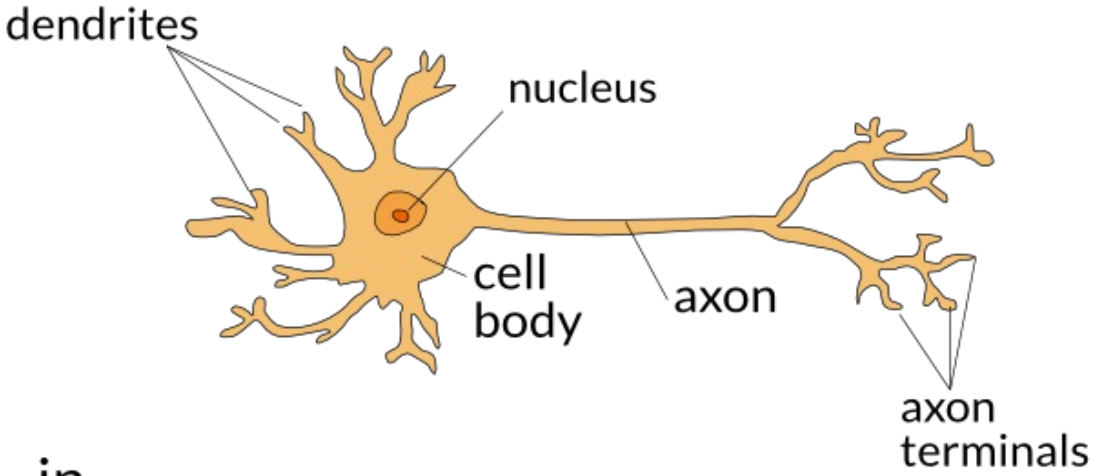
Extremely efficient discovery tool (e.g., >95% for pulsational variables).

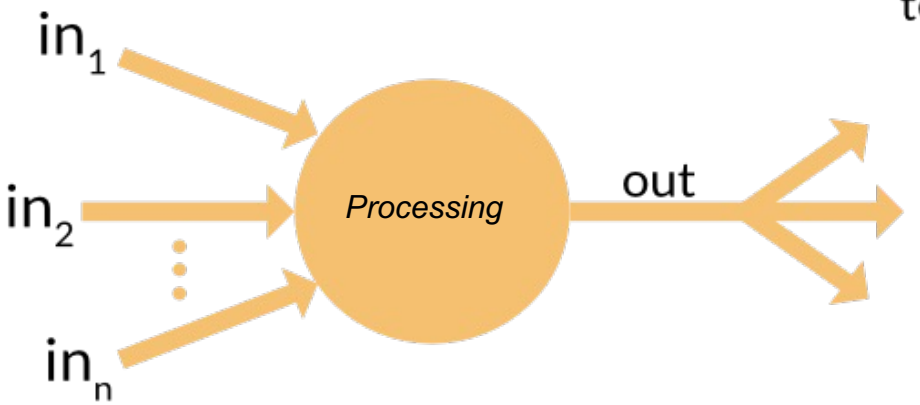
Richards et al. (2011)



Artificial Neural Networks

Inspired by Nature

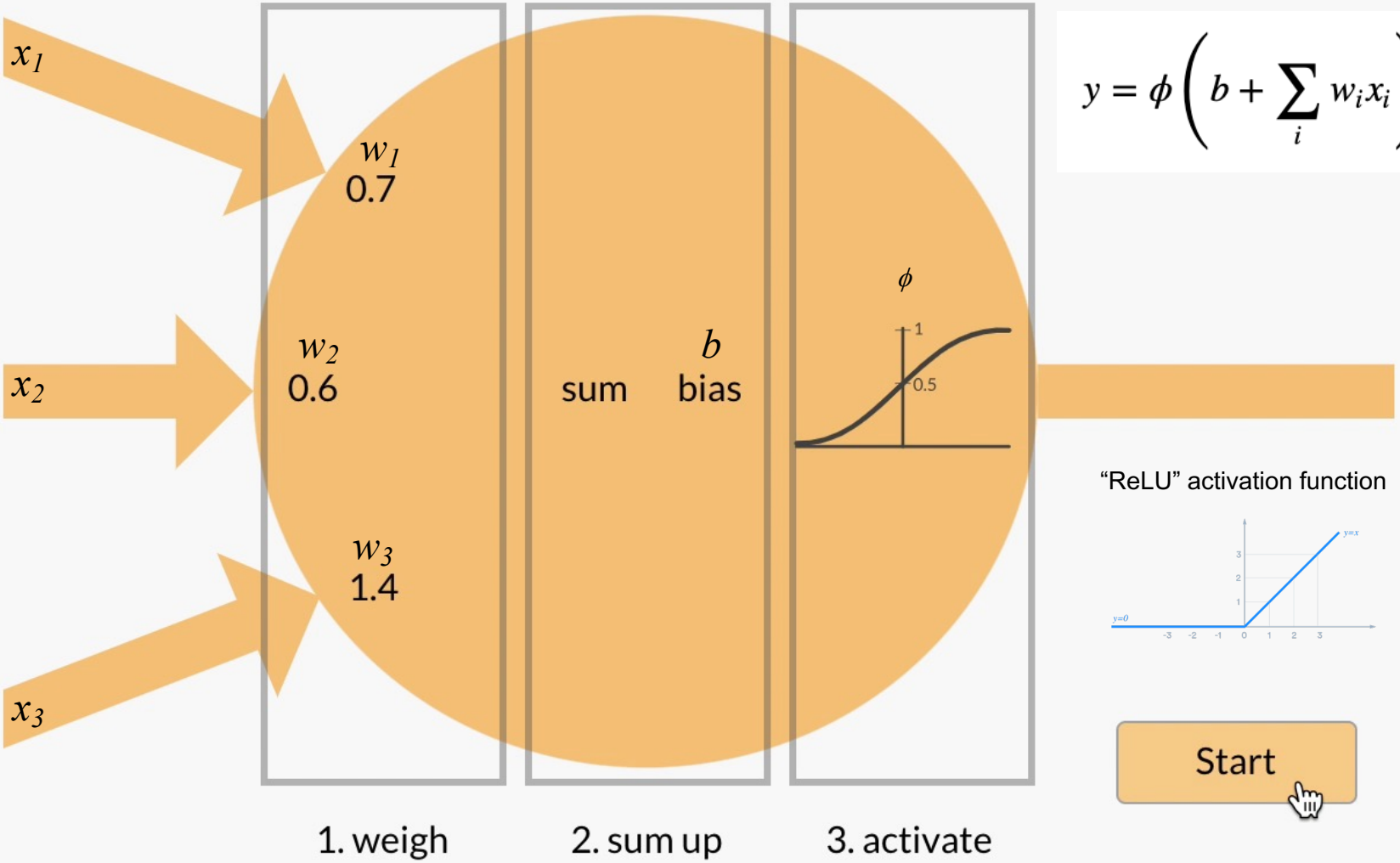




Input signals (e.g., measurements)

Output signal

https://appliedgo.net/perceptron/



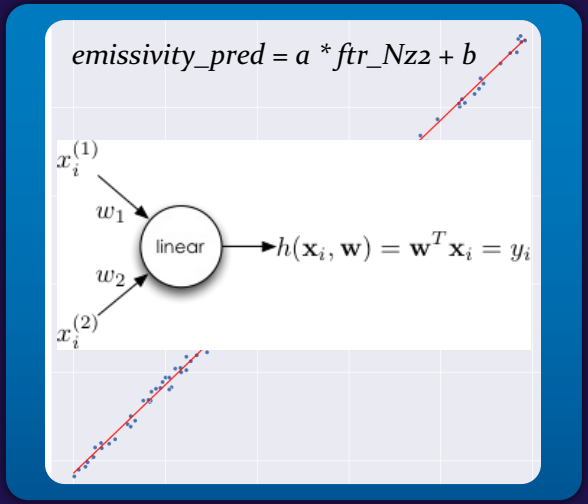
$$y = \phi\left(b + \sum_i w_i x_i\right)$$

"ReLU" activation function

Start

Fitting a Straight Line as a ML problem

	ftr_Nz2	emissivity
0	30.571119	48.971987
1	1.271871	19.341034
2	18.887398	36.707097
3	14.590665	34.120698
4	17.721658	36.171351
5	2.089411	20.348163
6	42.661662	60.551494
7	0.591402	19.066962
8	2.353133	21.515238
9	29.955232	48.188376
10	23.045119	42.420823



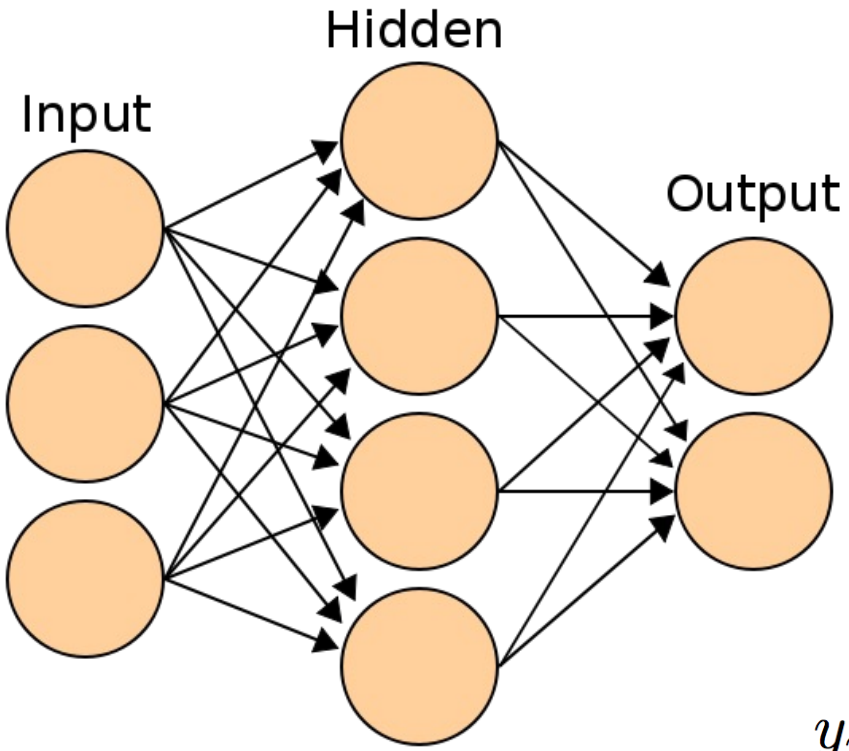
Model: Single neuron with a linear activation function
Training: Fitting the coefficients a, b
Inference: Evaluating the model

	ftr_Nz2	emissivity_pred	emissivity_true	error
0	12.017282	30.573918	30.582191	-0.000271
1	25.702186	44.233306	43.717787	0.011792
2	44.786225	63.281763	64.199395	-0.014293
3	38.051029	56.559124	56.392191	0.002960
4	42.186956	60.687340	59.935506	0.012544

Training

Inference

An Artificial Neural *Network*



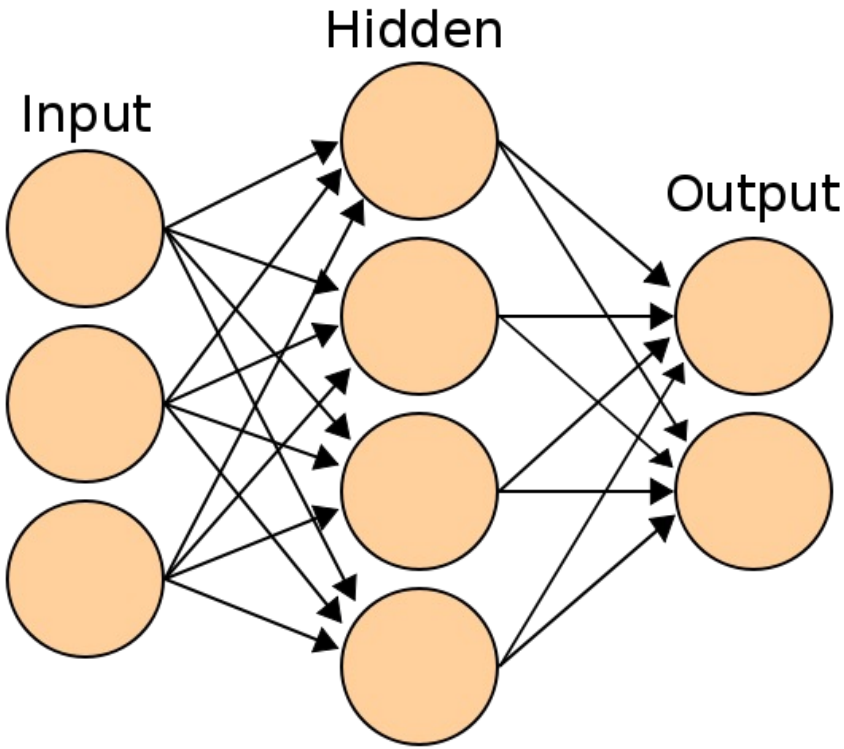
An Artificial Neural Network connects outputs of neurons to inputs of other neurons.

It's typically organized in *layers*, having an input, output, and zero or more hidden layers.

Mathematically, just lots of linear algebra (plus the non-linear activation functions).

$$y_n = \sigma\left(...\sigma\left(...\sigma\left(\sum w_0 x\right)\right)\right)$$

Why ANNs? Universal Approximators.



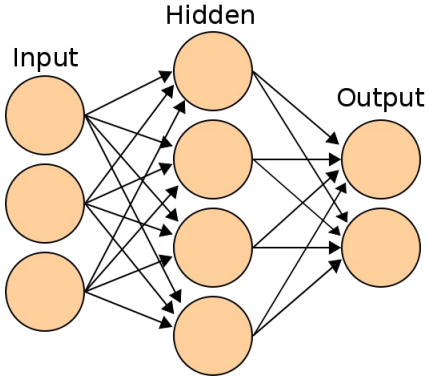
The **universal approximation theorem** states that a feed-forward network with a single hidden layer containing a finite number of neurons can approximate almost any continuous function.

Therefore, even simple neural networks can represent a wide variety of interesting functions when given appropriate parameters.

However, the theorem says nothing about algorithmic *learnability* of those parameters. An object of much research today.

See https://en.wikipedia.org/wiki/Universal_approximation_theorem for a quick introduction.

Learning: Gradient Descent, Back-Propagation, +Tricks



Given a set of known inputs and known outputs, adjust the weights of the network so that inputs approximate the outputs (as measured by some *loss function*).

Do so for networks with many nodes (millions!), in finite computational time, and w/o overfitting.

This is where a lot of the research is today.

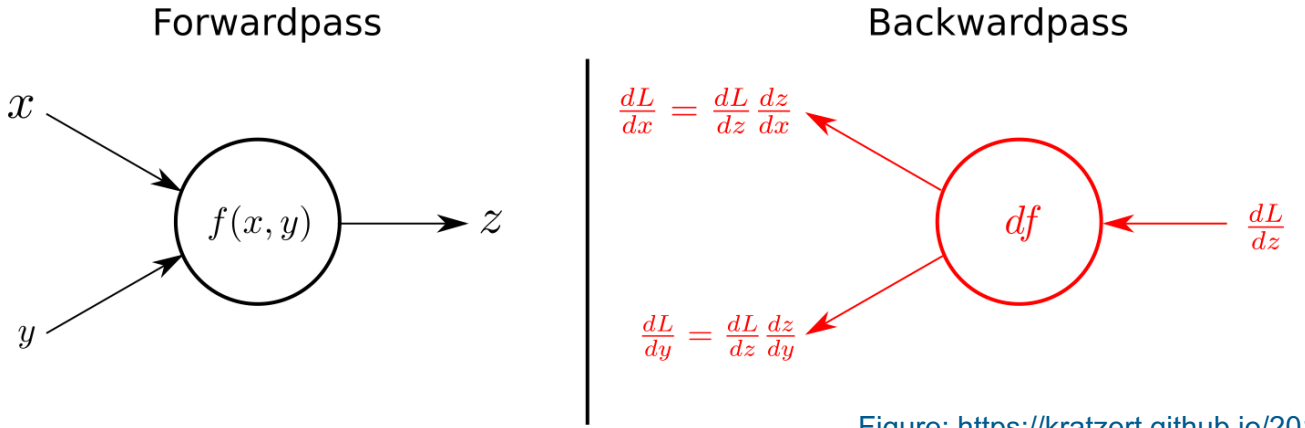
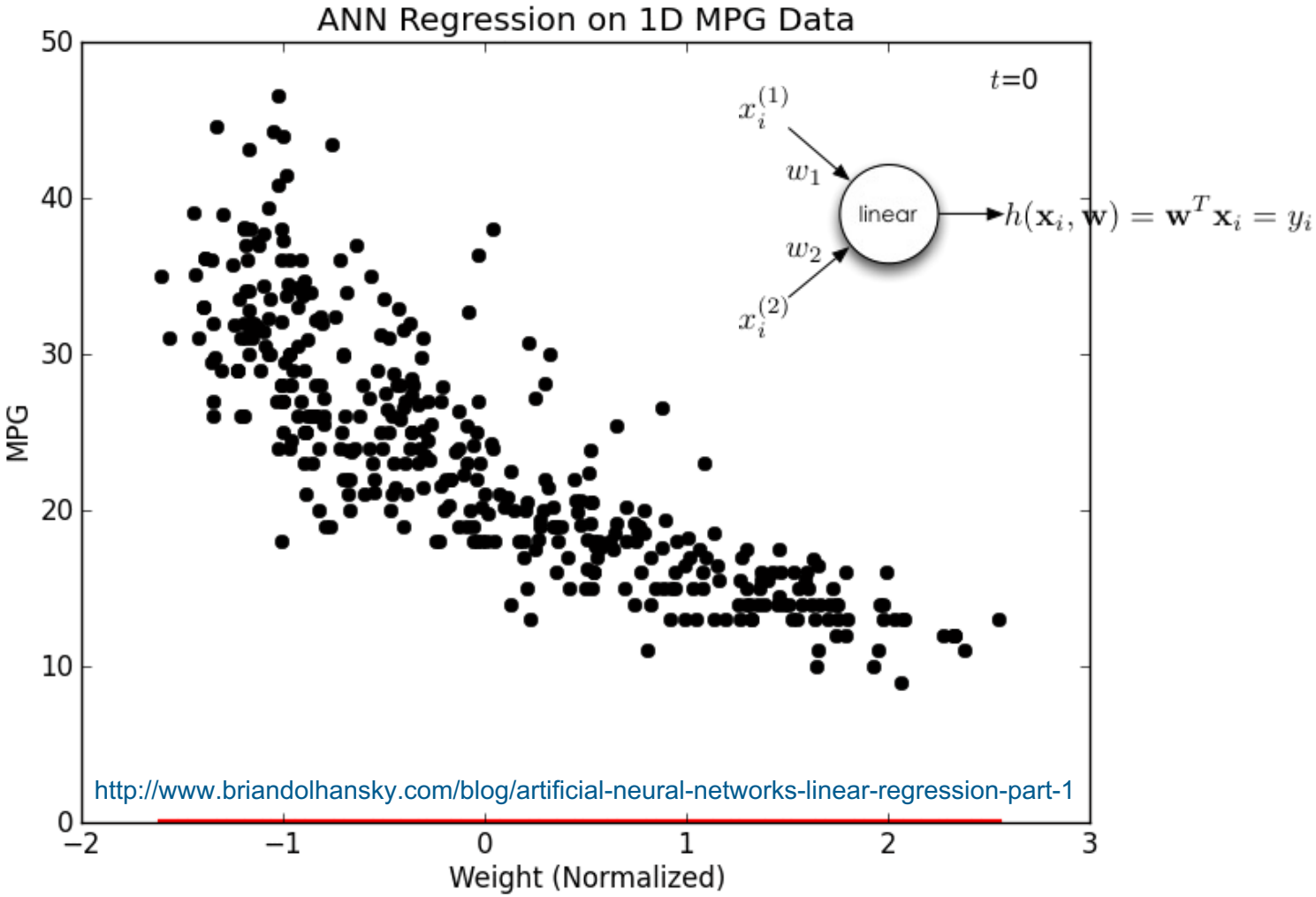


Figure: https://kratzert.github.io/2016/02/12/understanding-the-gradient-flow-through-the-batch-normalization-layer.html (Frederik Kratzert)


ANN Regression on 1D MPG Data
t=0
$x_i^{(1)}$
w_1
linear
w_2
$x_i^{(2)}$
$h(\mathbf{x}_i, \mathbf{w}) = \mathbf{w}^T \mathbf{x}_i = y_i$
MPG
Weight (Normalized)
50
40
30
20
10
0
−2
−1
0
1
2
3
http://www.briandolhansky.com/blog/artificial-neural-networks-linear-regression-part-1

ANNs Excel in Signal Processing

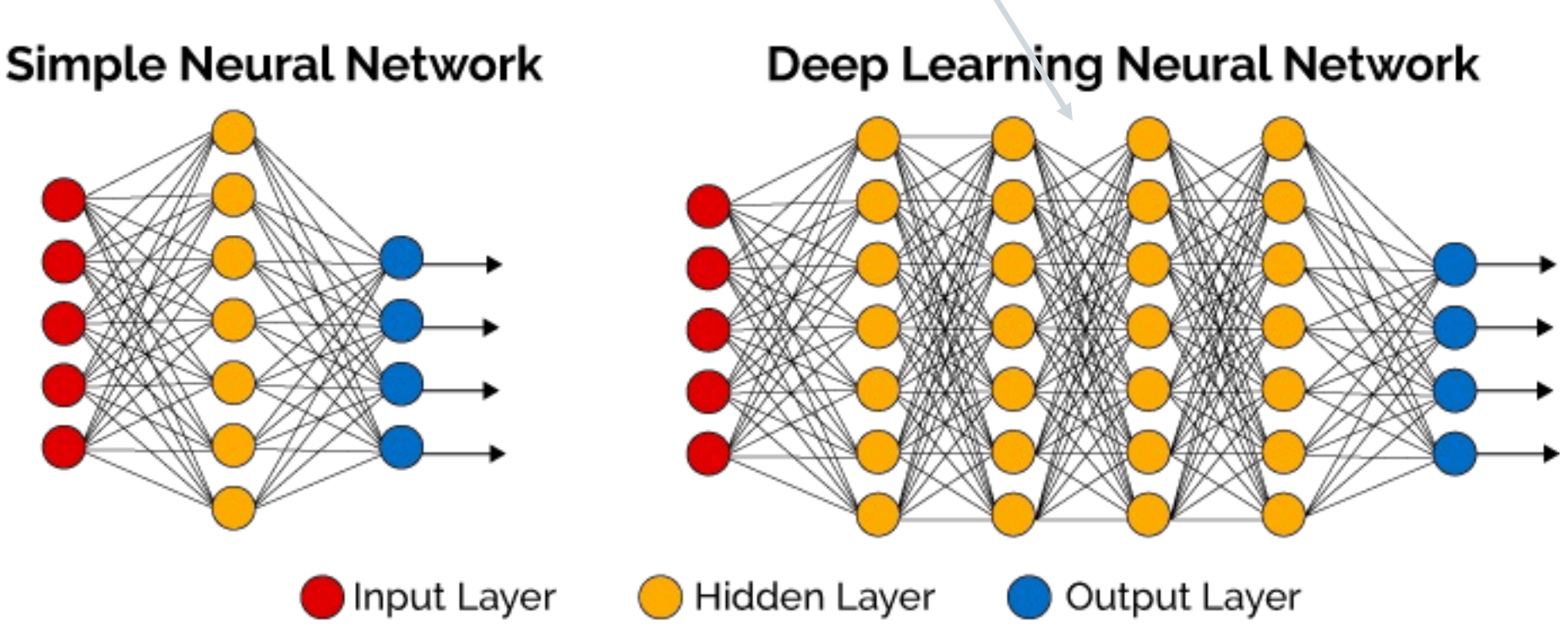
- Speech recognition
- Image recognition
- Image segmentation
- Recognition in video

- Speech synthesis
- Image synthesis
- Video synthesis
- …

Mazzini, Buzzelli, Pauy and Schettini (2018)
https://www.youtube.com/watch?v=L81GJsTd1yY

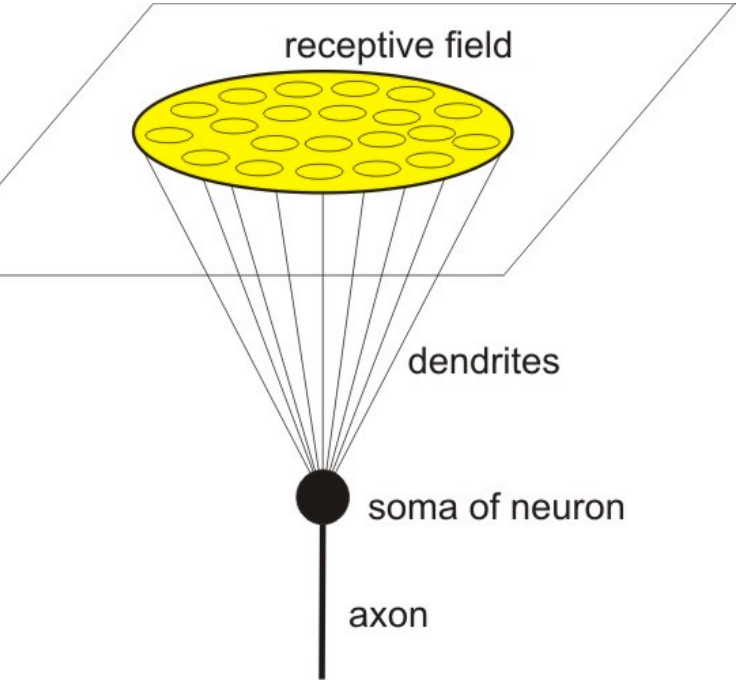
Deep Neural Networks

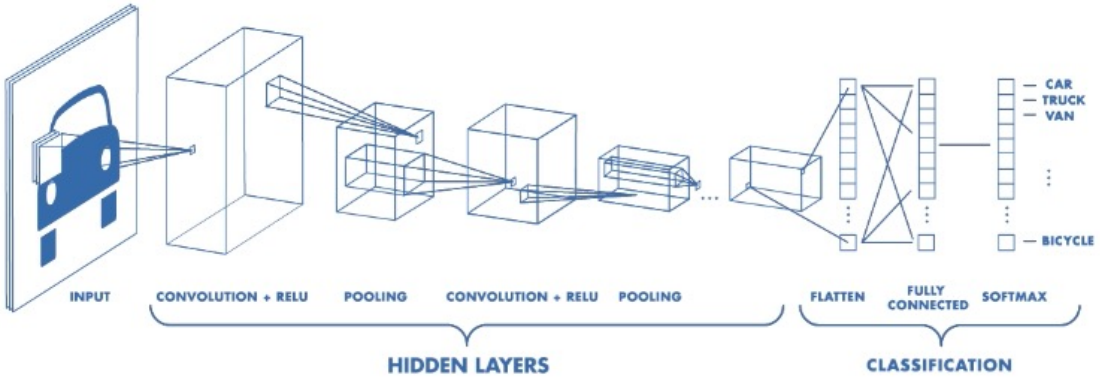
Often millions of parameters!



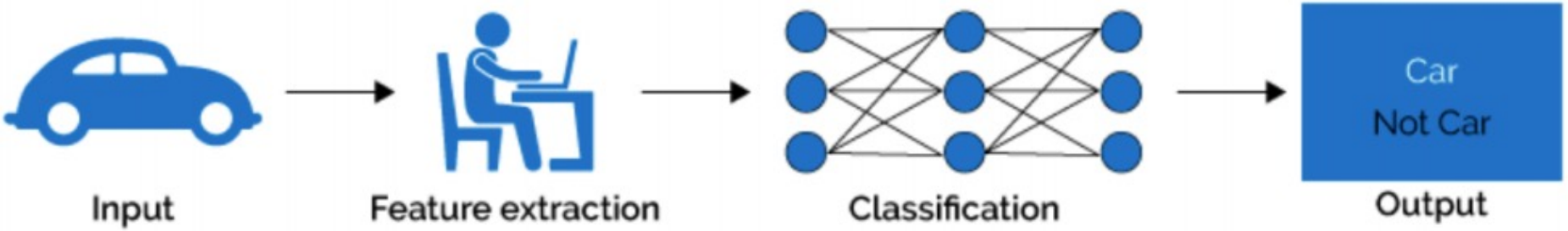
https://becominghuman.ai/deep-learning-made-easy-with-deep-cognition-403fbe445351

Convolutional Neural Networks (CNNs)



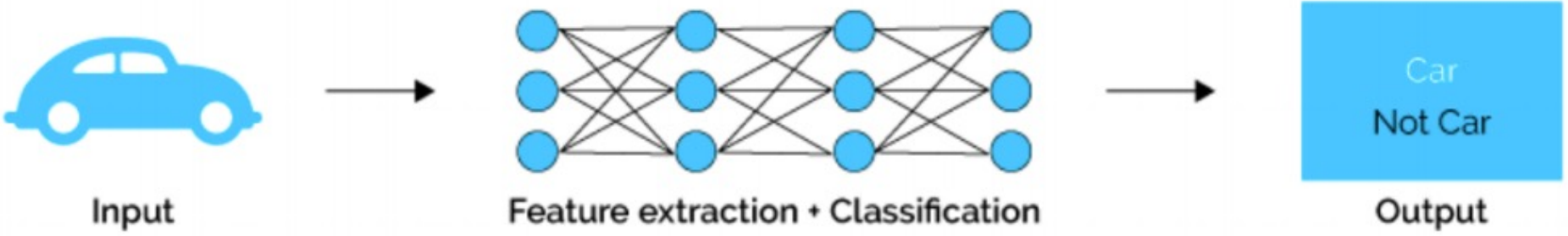


Machine Learning



Input → Feature extraction → Classification → Output

Deep Learning



Input → Feature extraction + Classification → Output

Applications in Astronomical Data Analysis

Rule of Thumb:

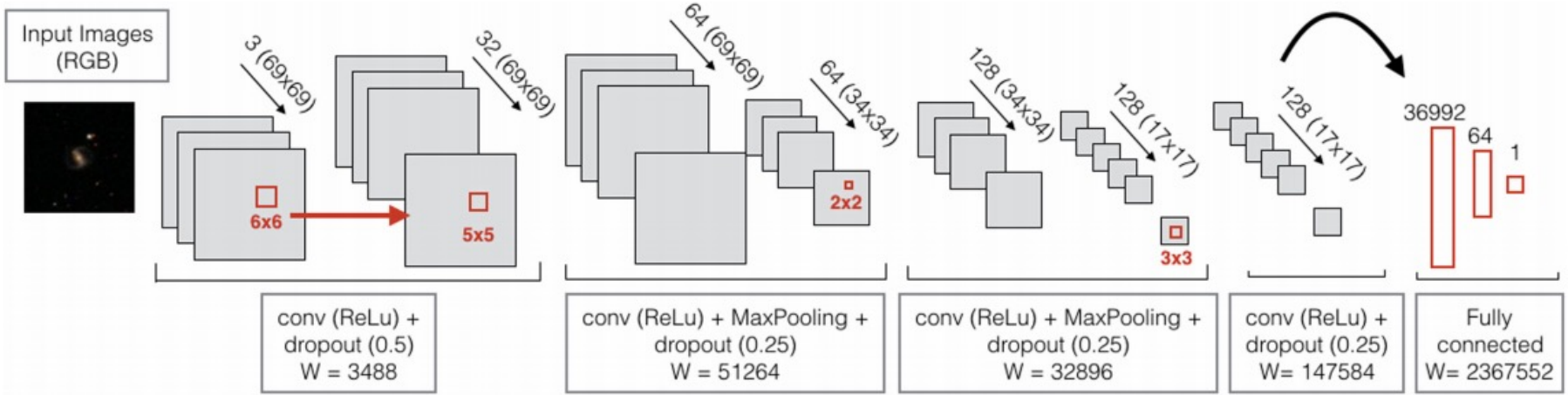
Anywhere a human does well (almost 100% certain)

“Superhuman”:

Where the problem involves complex, multi-D, correlations

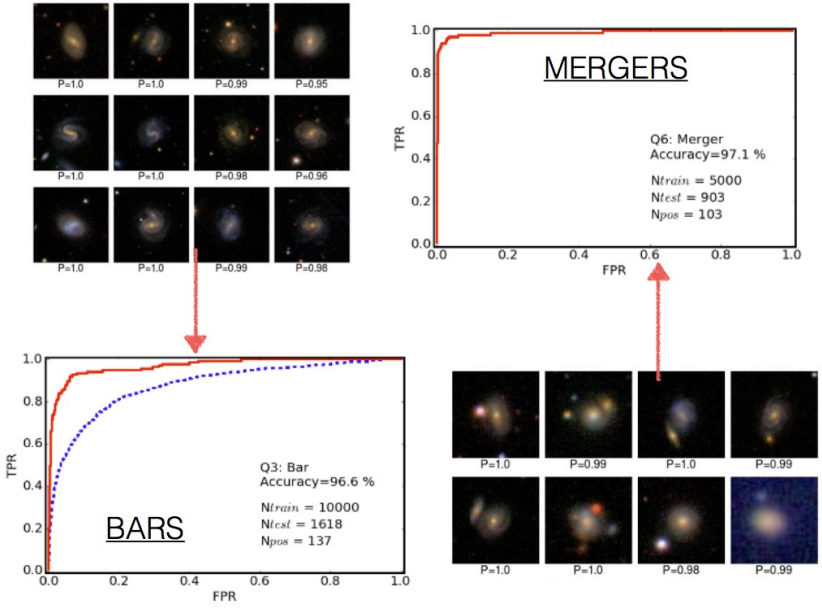
Example: Galaxy Classification

A (simple) CNN trained on ~15k galaxies, used to classify 670k galaxies in SDSS.
Four convolutional layers, and a fully connected layer (2 million parameters).



Dominguez Sanchez et al. (2018)

How well does it do?



Outperforms SVM-based models.

Large accuracy (> 97\%) for distinguishing between disk features/bars/edge or face on galaxies/etc..

Note: this is using a fairly simple CNN – improvements are likely.

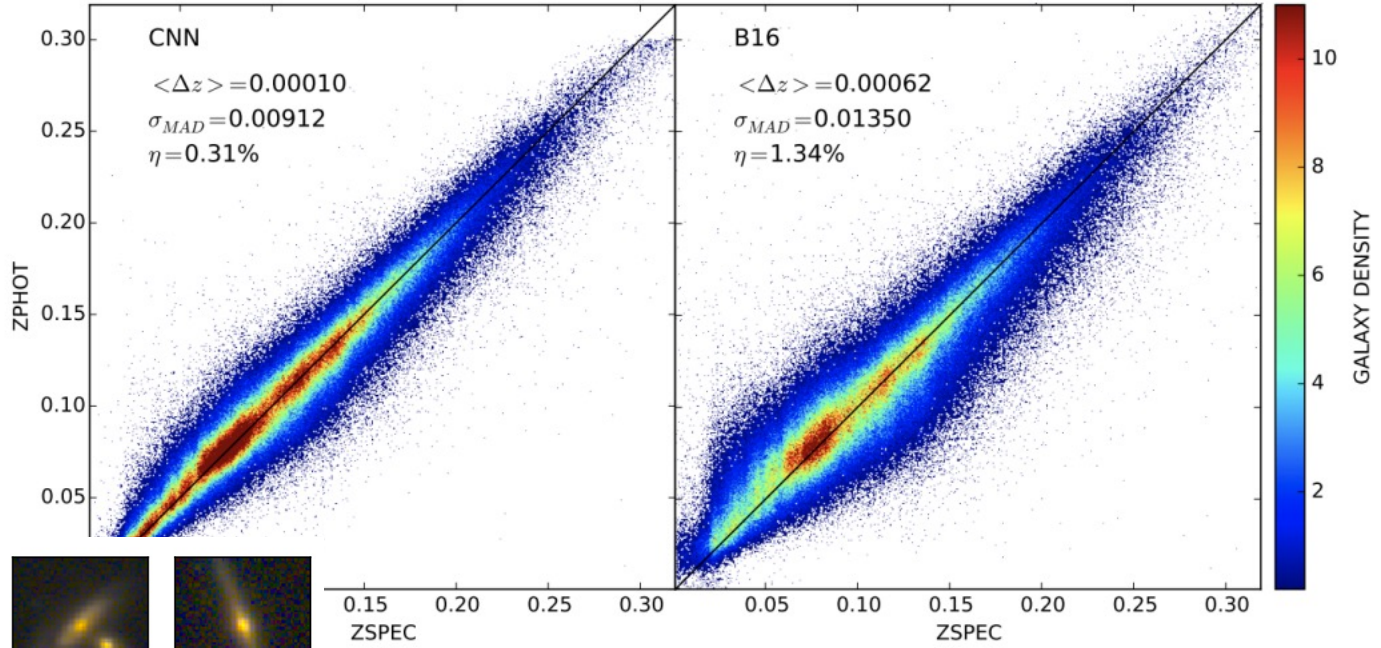
```
#========= Model definition=======

#Convolutional Layers

model = Sequential()
model.add(Convolution2D(32, 6,6, border_mode='same',
                        input_shape=(img_channels, img_rows, img_cols)))
model.add(Activation('relu'))
model.add(Dropout(0.5))

model.add(Convolution2D(64, 5, 5, border_mode='same'))
model.add(Activation('relu'))

model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Convolution2D(128, 2, 2, border_mode='same'))
model.add(Activation('relu'))

model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Convolution2D(128, 3, 3, border_mode='same'))
model.add(Activation('relu'))

model.add(Dropout(0.25))

#Fully Connected start here
#-----------------------------#

model.add(Flatten())
model.add(Dense(64, activation='relu'))
model.add(Dropout(.5))
model.add(Dense(1, init='uniform', activation='sigmoid'))

print("Compilation...")

model.compile(loss='binary_crossentropy',optimizer='adam',metrics=['accuracy'])
```

Implementation in Keras
(courtesy of M. Huertas-Company)

Example: Photo-Z Estimation

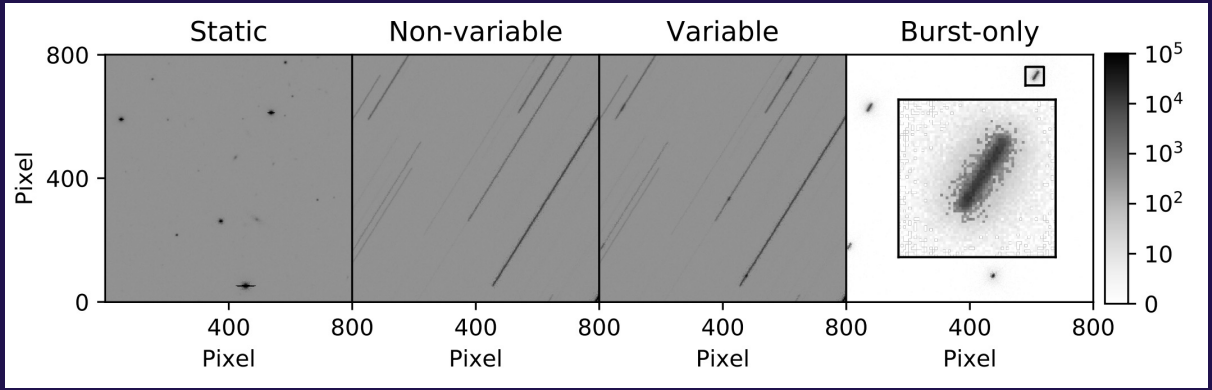
Photo Z model trained on 400k images of galaxies from SDSS, for which spectroscopic redshifts are available.

Outperforms best known photo-Z estimators.



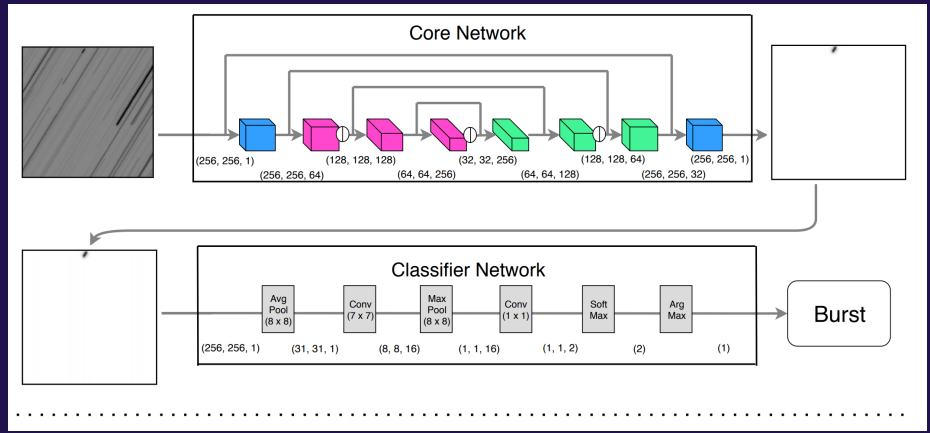
Pasquet et al. 2018

Example: Detecting Short Bursts in Trailed Images



Idea: take trailed images of the sky; have the neural net identify trails where the intensity of the source changed during the exposure.

Result: a network that detects variability at ~10ms timescales (bright sources, high variability) in trailed LSST images.



Thomas and Kahn, submitted

… and many, many,
more.

Summary

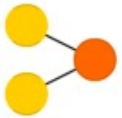
- **Machine learning may be a major source of ”unknown unknowns” in the big-data era.**
- Why does it work? Incredible capacity to soak up complex correlations (number of degrees of freedom).
- These ideas are generally not new! E.g., neural networks go back to the 1950-ies…
  - New: Large Training Datasets | New: Compute Power
- … but the understanding of their theory still in its infancy. We’re in the “Edison Era” of Machine Learning (esp. ANNs, Deep Learning, etc.): experiment and see what works, practice is ahead of theory.

A mostly complete chart of

Neural Networks

- Backfed Input Cell
- Input Cell
- Noisy Input Cell
- Hidden Cell
- Probablistic Hidden Cell
- Spiking Hidden Cell
- Output Cell
- Match Input Output Cell
- Recurrent Cell
- Memory Cell
- Different Memory Cell
- Kernel
- Convolution or Pool

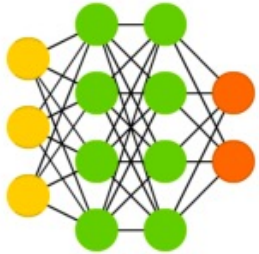
Perceptron (P)



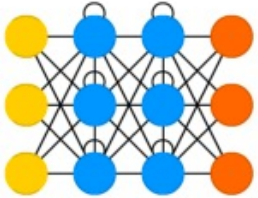
Feed Forward (FF)

Radial Basis Network (RBF)

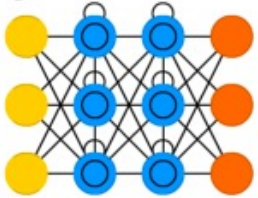
Deep Feed Forward (DFF)



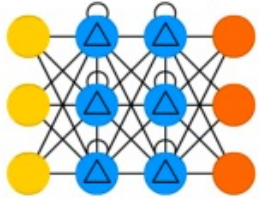
Recurrent Neural Network (RNN)



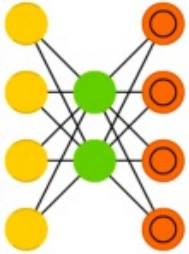
Long / Short Term Memory (LSTM)



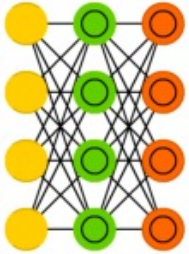
Gated Recurrent Unit (GRU)



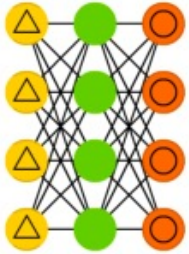
Auto Encoder (AE)



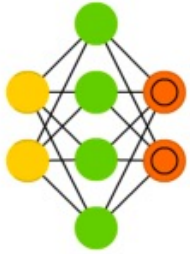
Variational AE (VAE)



Denoising AE (DAE)



Sparse AE (SAE)



A mostly complete chart of

Neural Networks

- Backfed Input Cell
- Input Cell
- Noisy Input Cell
- Hidden Cell
- Probablistic Hidden Cell
- Spiking Hidden Cell
- Output Cell
- Match Input Output Cell
- Recurrent Cell
- Memory Cell
- Different Memory Cell
- Kernel
- Convolution or Pool