Chapter 3

Introduction to Machine Learning

Introduction to Machine Learning

Artificial Intelligence: AI is the simulation of human intelligence by software coded heuristics

Machine Learning (ML) is a branch that uses statistical mathematics and related algorithms to achieve the same goal.

Deep learning deals with creating ML models that mimic the functionality of our brain, that is, the way neurons are connected and function together to draw inferences from the presented information.

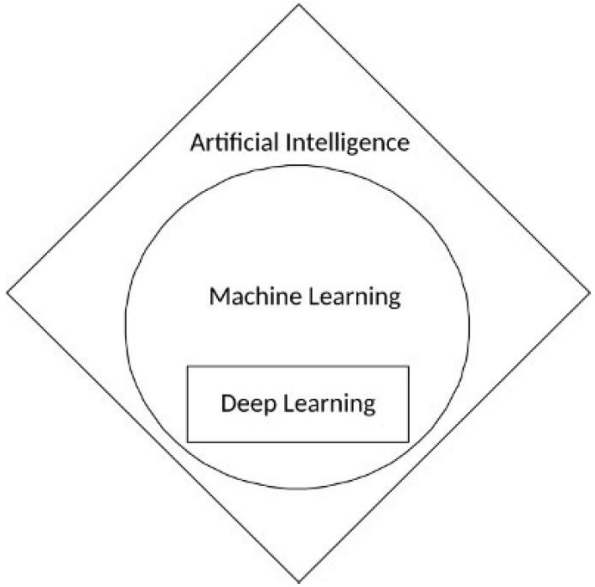


Figure 3.1: *Differentiating between AI and ML*

What is Machine Learning?

"Learning is any process by which a system improves performance from experience."

- Herbert Simon

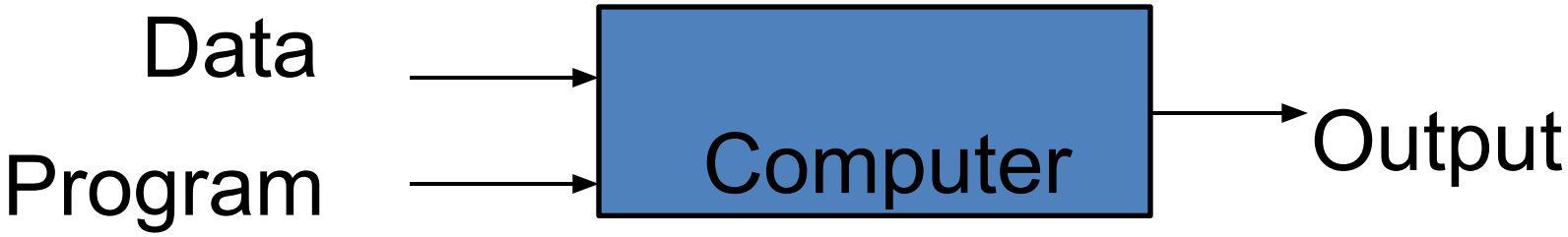
Definition by Tom Mitchell (1998):

Machine Learning is the study of algorithms that

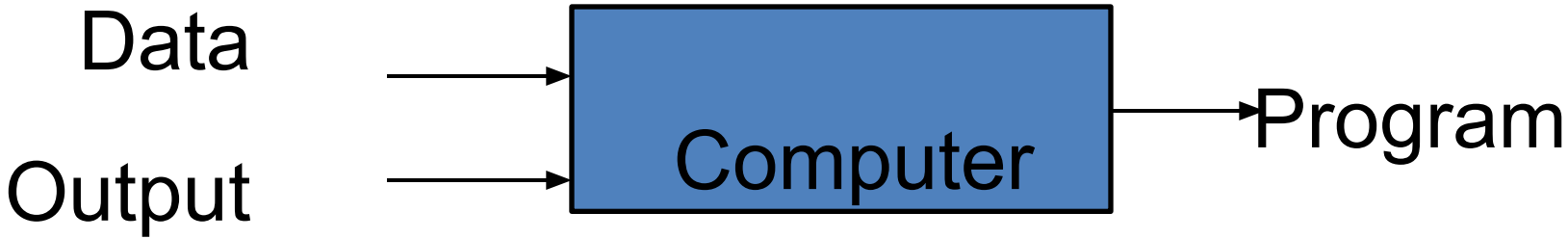
- improve their performance P
- at some task T
- with experience E.

A well-defined learning task is given by <P, T, E>.

Traditional Programming



Machine Learning



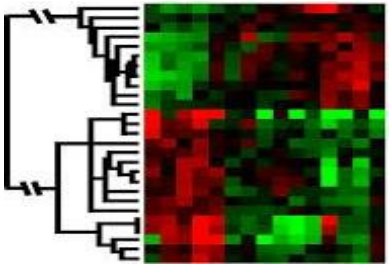
Slide credit: Pedro

When Do We Use Machine Learning?

ML is used when:

- Human expertise does not exist (navigating on Mars)
- Humans can't explain their expertise (speech recognition)
- Models must be customized (personalized medicine)
- Models are based on huge amounts of data (genomics)



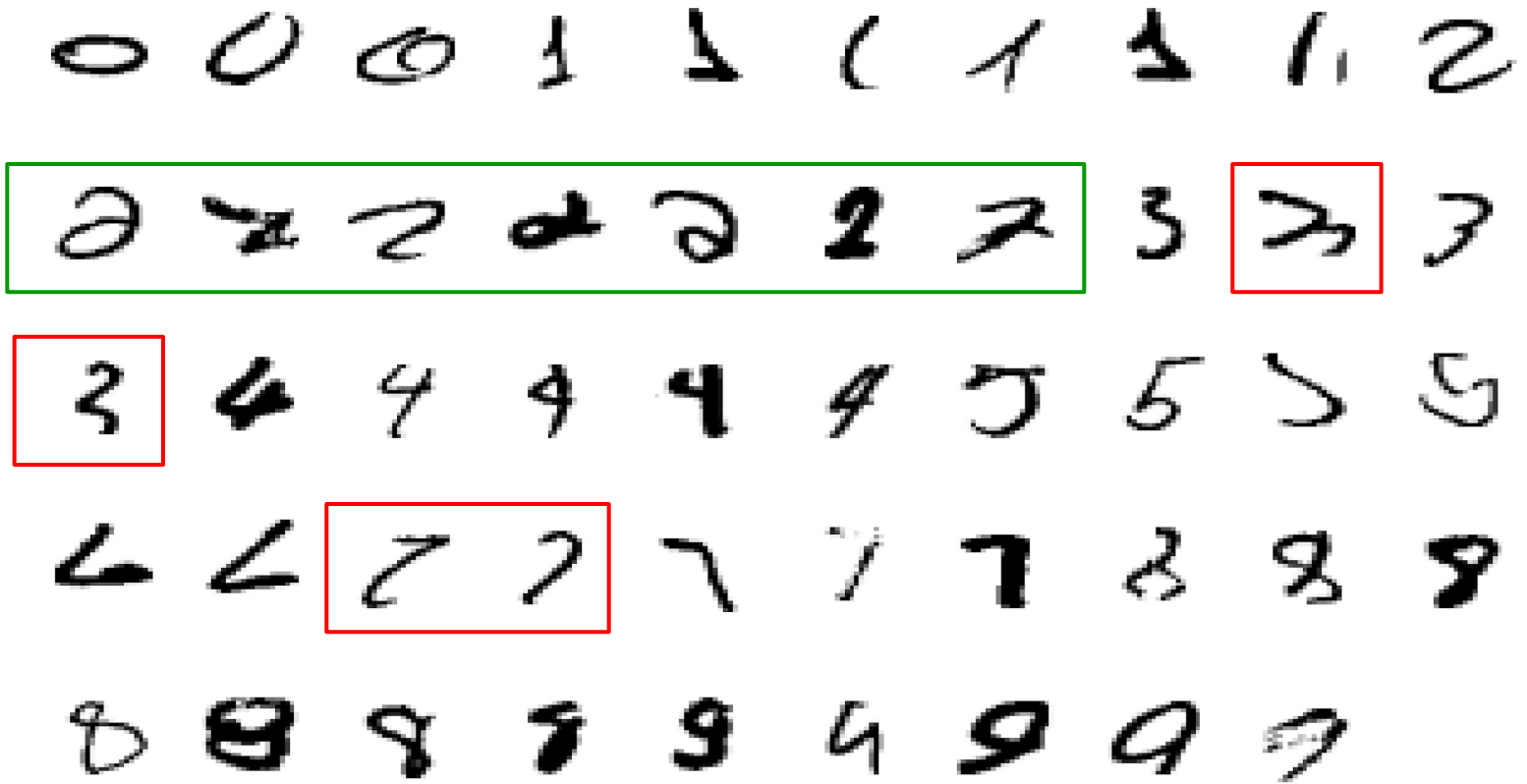



Learning isn't always useful:

- There is no need to "learn" to calculate payroll

Based on slide by E. Alpaydin

A classic example : It is very hard to say what makes 2



Slide credit: Geoffrey

Some more examples of Task

- Recognizing patterns:
  - Facial identities or facial expressions
  - Handwritten or spoken words
  - Medical images
- Generating patterns:
  - Generating images or motion sequences
- Recognizing anomalies:
  - Unusual credit card transactions
  - Unusual patterns of sensor readings in a nuclear power plant
- Prediction:
  - Future stock prices or currency exchange rates

Slide credit: Geoffrey

Types of Data

1. Customer data: Information from users who use your product or platform.

2. Feedback data: Reviews and opinions from customers about your product.

3. Performance data: Shows how well a product or employee is doing over time.

4. Open-source data: Publicly available datasets or information found online.

5. Internal company data: Reports and data from within the company, like customer data or feedback.

Glimpse into the dataset

1. Row 1: Column names indicating data types.

2. Index column: Represents row indexes, can be added using pandas.

3. Rows: Contain data observations.

4. Columns: Features or attributes providing insights and relationships.

Index	No. of Bedrooms	Area(sq. ft.)	Location
1	3	1800.23	Downtown
2	2	1000.10	Midtown
3	3	1600.87	Downtown

Variables in a dataset:

In a dataset, variables can be classified as either quantitative or categorical. Understanding this classification is important for data analysis and processing.

Quantitative variables: Numeric

Discrete: Finite, countable values (e.g., number of customer complaints).

Continuous: Infinite values, not countable (e.g., the house prices of houses in a district).

Categorical variables:

Non-numeric, fall into categories (e.g., pass/fail in an exam, location).

Branches of ML

Regression and classification are methods used in data analysis.

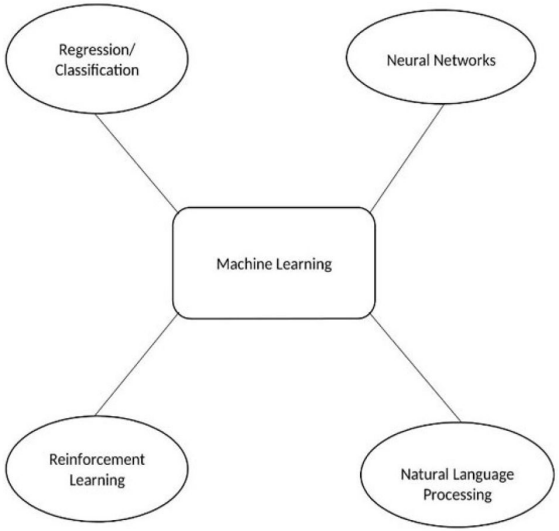
Classification involves categorizing data into classes.Examples of classification include predicting customer purchases (binary) or product choices (multiclass).

Regression predicts continuous variables. Examples of regression include predicting house prices or sales numbers.

Neural networks are a type of machine learning model inspired by the human brain.They are used for tasks like image processing.

Reinforcement learning aims to maximize rewards in a given situation.Example: finding the best strategy for winning at casino slot machines.

Natural language processing involves teaching machines to understand and communicate in human language. Examples include chatbots used on websites for customer support.

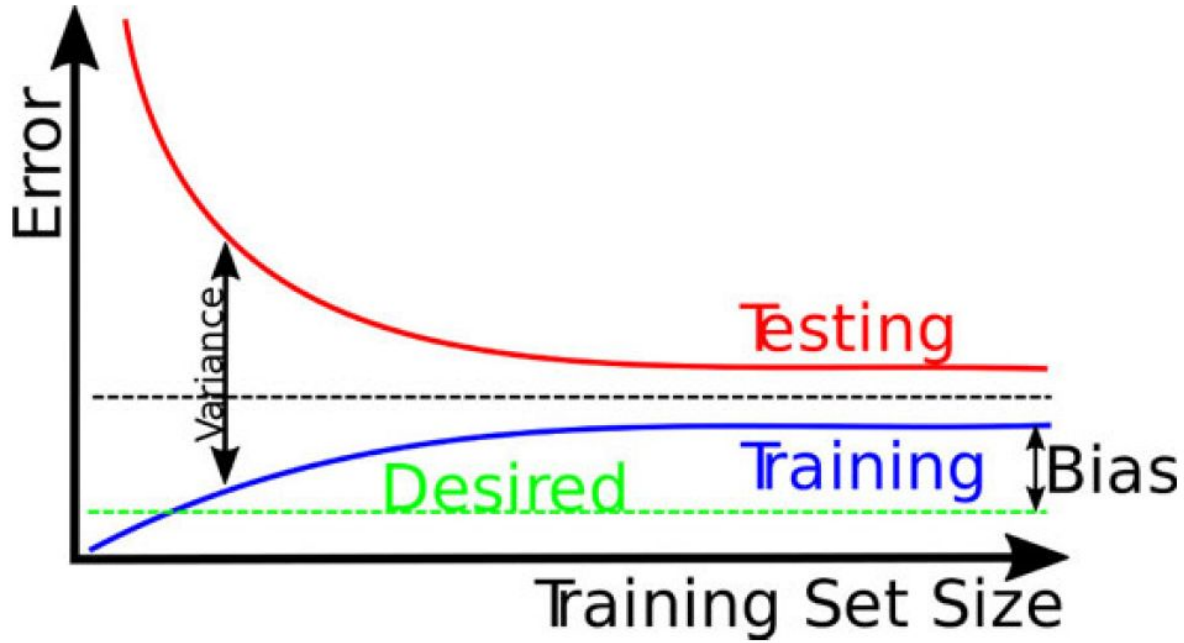


What is Bias?

Bias is the difference between the average prediction of our model and the correct value which we are trying to predict. A model with high bias pays very little attention to the training data and oversimplifies the model.

What is a Variance?

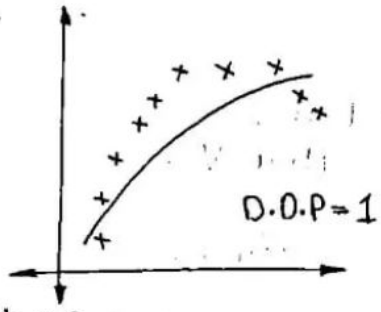
Variance is the variability of model prediction for a given data point or a value that tells us the spread of our data. A model with high variance pays a lot of attention to training data and does not generalize on the data which it hasn’t seen before.


Error
Variance
Testing
Training
Bias
Desired
Training Set Size

Bais and Variance

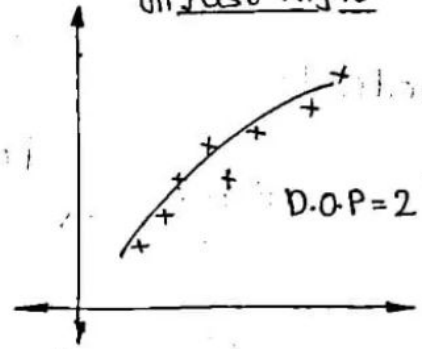
☐ Regression :

(i) Underfitting



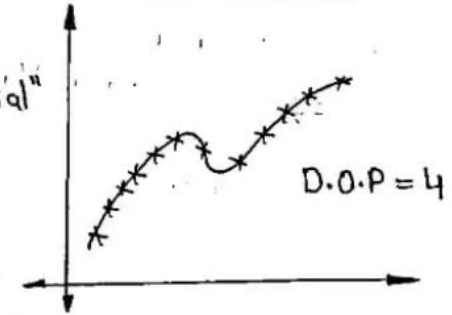
"High Bais, High Variance"

(ii) Just Right



"Low Bais, Low Variance"

(iii) Over-fitting



"Low Bais, High Variance"

Note:
D.O.P =
"Degree of polynomial"

(i) Model 1 (Underfitting) : Train Accuracy ↓
Test Accuracy ↓

(ii) Model 3 (Overfitting) : Train Accuracy ↑
Test Accuracy ↓

- Step 1: Collecting data

  - Define problem statement and gather sufficient data.

  - Data can be from various sources and may require cleaning.

  - Data can be labeled or unlabeled, and relevant features need to be selected.

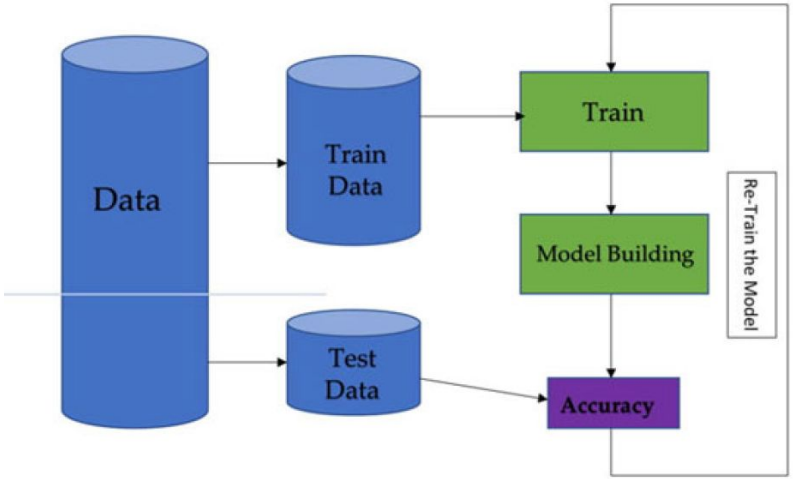


Figure 3.8: *Predictive modeling cycle*

- Step 2: Splitting the data

  - Split data into training, cross-validation, and test datasets.

  - Training data (70% or more) is used to build models.

  - Cross-validation data (10%) validates model performance.

  - Test data (30%) assesses model robustness.

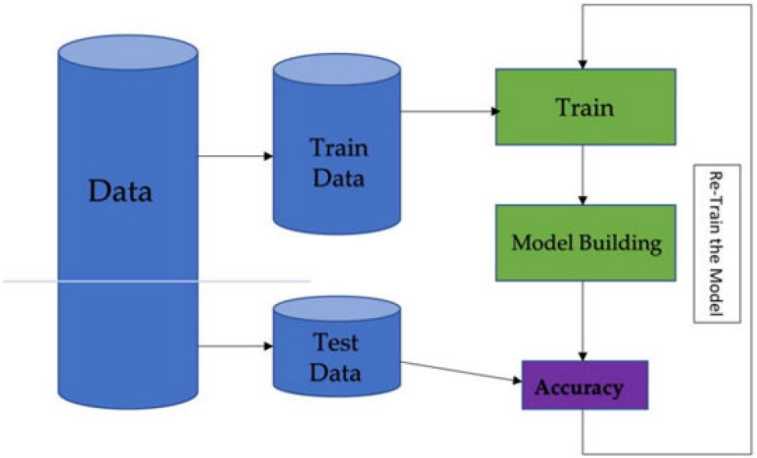


Figure 3.8: Predictive modeling cycle

Step 3: Creating a machine learning model

- Choose appropriate algorithms based on data type and labeling.

- Train models using training data and validate using test data.

- Evaluate model accuracy and adjust parameters if necessary.

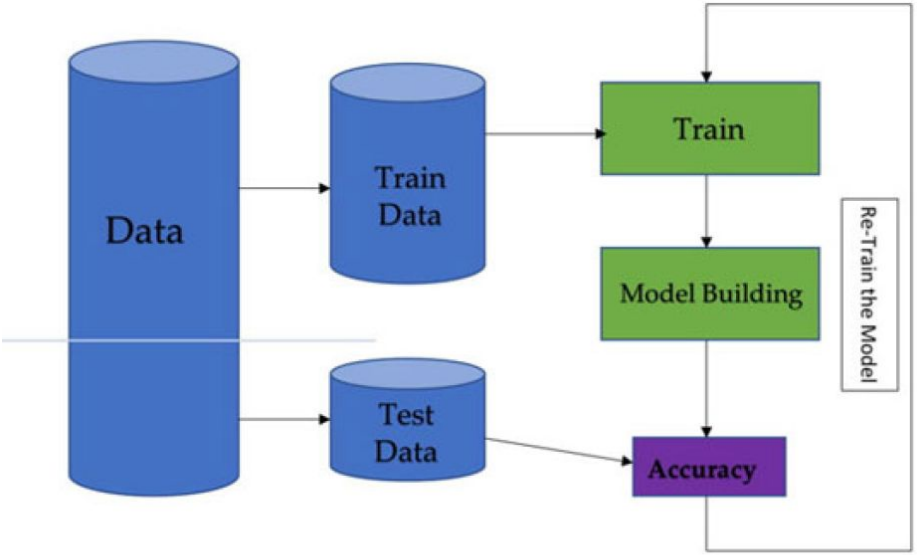


Figure 3.8: *Predictive modeling cycle*

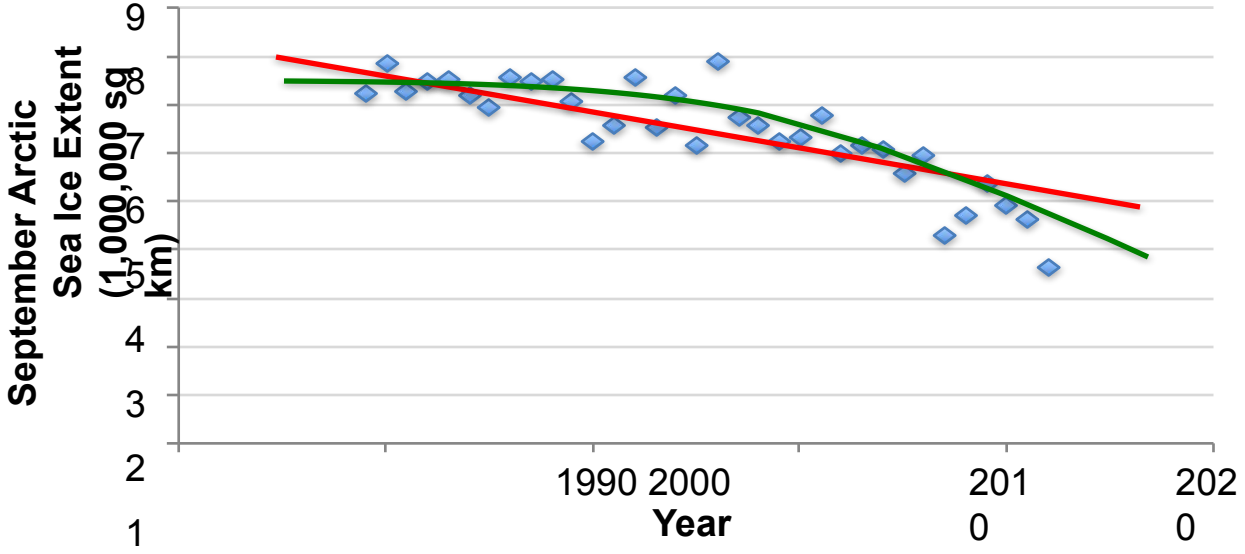
Chapter 4: Supervised and Unsupervised Learning in Python

Introduction

- **Supervised (inductive) learning**
  - Given: training data + desired outputs (labels)
  - Example : Classification, Regression
- **Unsupervised learning**
  - Given: training data (without desired outputs)
  - Example : Clustering, Autoencoders
-

Supervised Learning: Regression

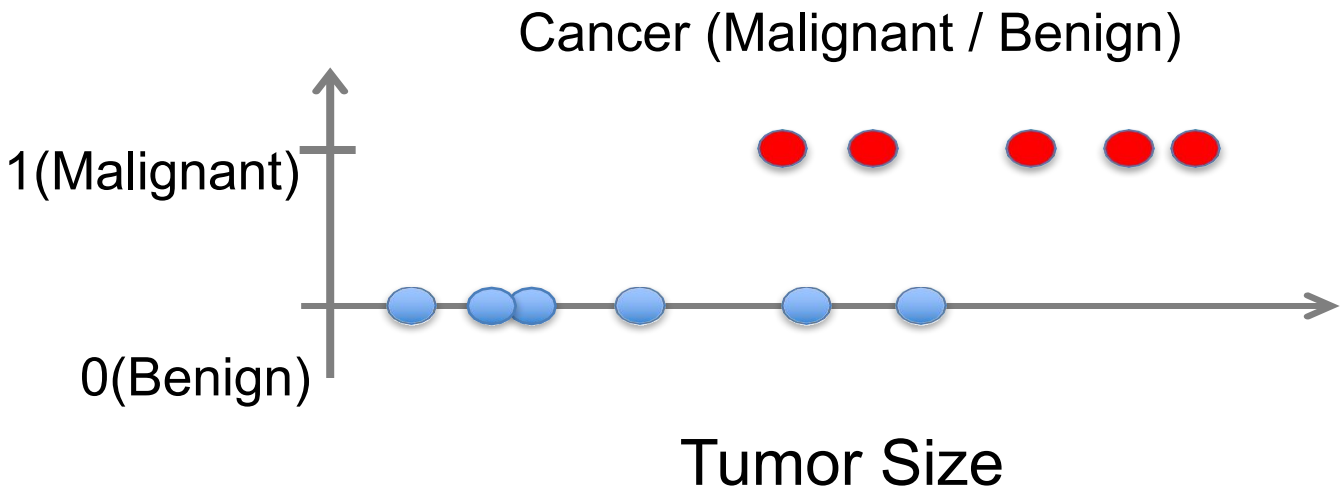
- Given (x_1, y_1), (x_2, y_2), ..., (x_n, y_n)
- Learn a function $f(x)$ to predict y given x
  - y is real-valued == regression



Data from G. Witt, Journal of Statistics Education, Volume 21, Number 1

Supervised Learning: Classificatio

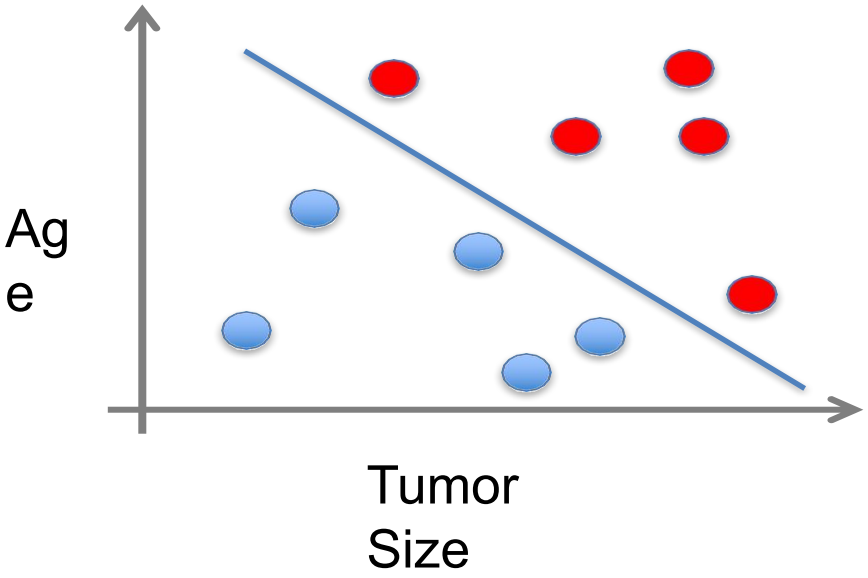
- Given (x_1, y_1), (x_2, y_2), ..., (x_n, y_n)
- Learn a function $f(x)$ to predict y given x
  - y is categorical == classification



Based on example by Andrew

Supervised Learning

- x can be multi-dimensional
  - Each dimension corresponds to an attribute



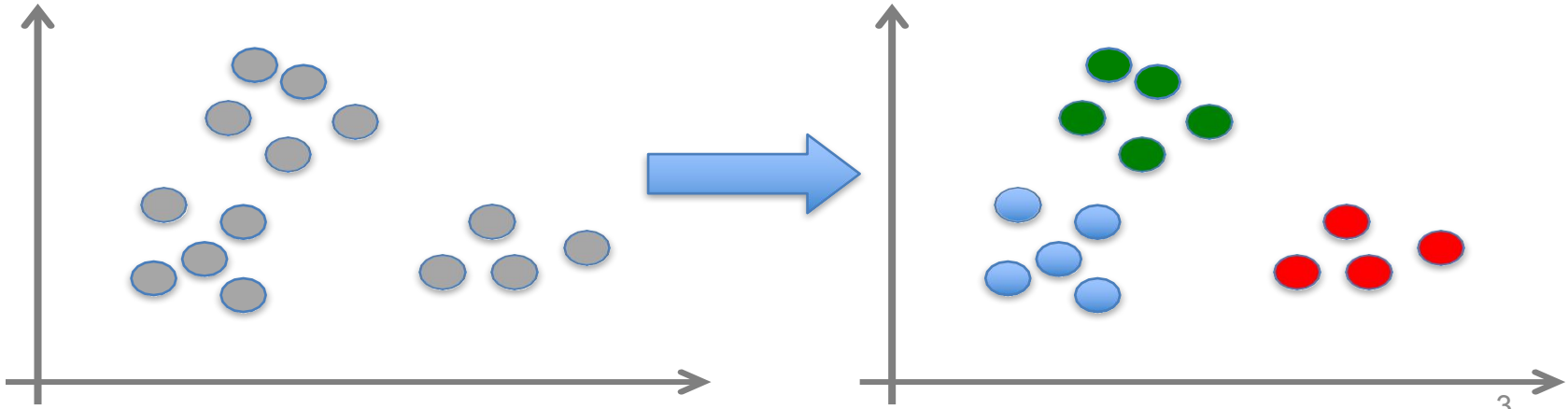
- Clump Thickness
- Uniformity of Cell Size
- Uniformity of Cell Shape

…

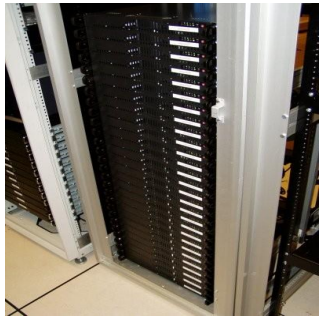
Based on example by Andrew

Unsupervised Learning

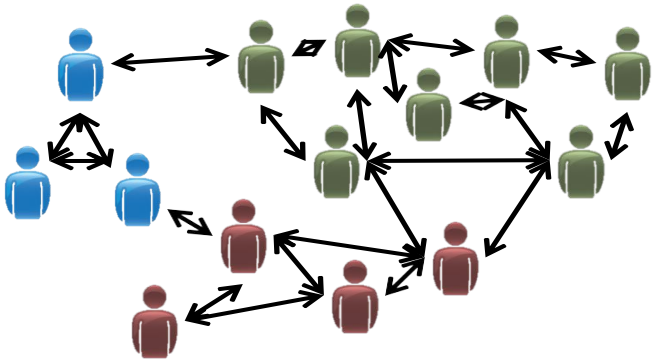
- Given x_1, x_2, ..., x_n (without labels)
- Output hidden structure behind the x's
  - E.g., clustering



Unsupervised Learning



Organize computing clusters



Social network analysis

Market segmentation

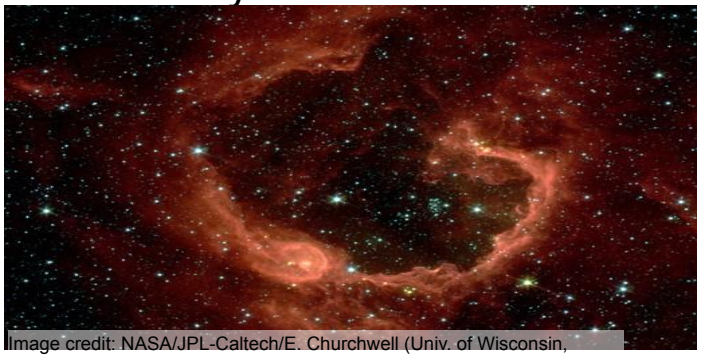


Image credit: NASA/JPL-Caltech/E. Churchwell (Univ. of Wisconsin, Madison)

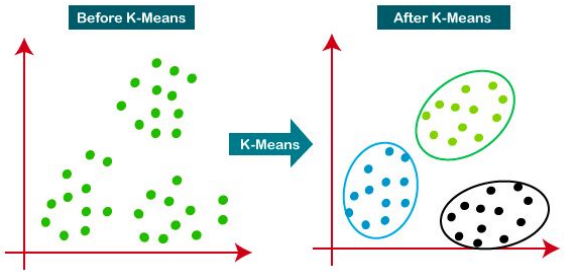
Astronomical data

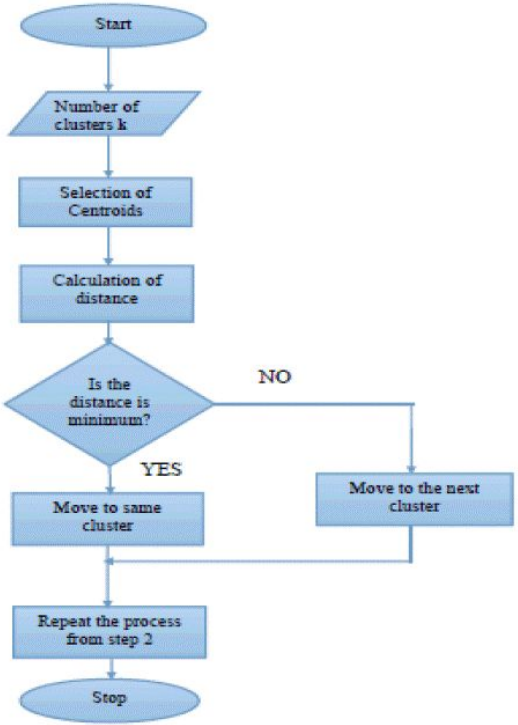
k-means clustering

K-Means Clustering is an unsupervised learning, which groups the

unlabeled dataset into different clusters.

Here K defines the number of pre-defined clusters that need to be created in the process, as if K=2, there will be two clusters.



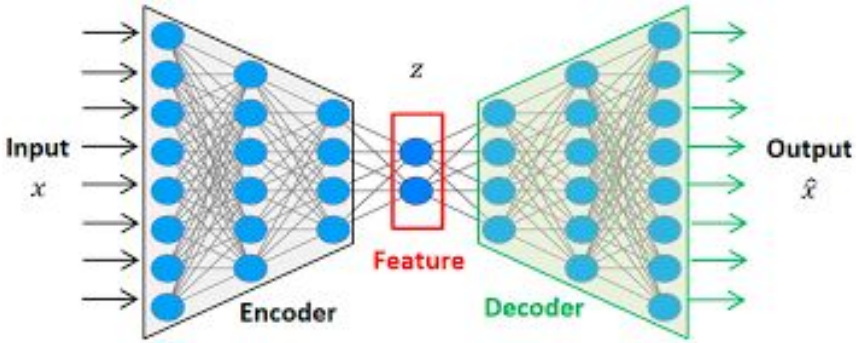


Given a set of observations ($\mathbf{x}_1$, $\mathbf{x}_2$, ..., $\mathbf{x}_n$), where each observation is a d-dimensional real vector, k-means clustering aims to partition the n observations into k ($\leq n$) sets $\mathbf{S} = \{S_1, S_2, ..., S_k\}$ so as to minimize the within-cluster sum of squares (WCSS) (i.e. variance). Formally, the objective is to find:

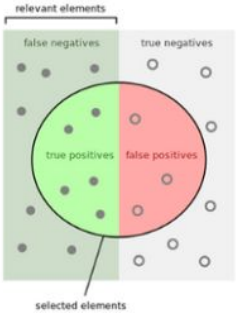
$$\arg\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2$$

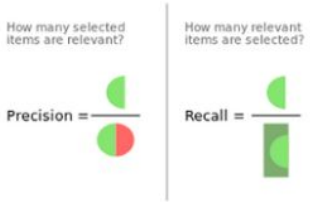
Auto Encoder

An autoencoder is a type of artificial neural network used to learn efficient codings of unlabeled data (unsupervised learning). An autoencoder learns two functions: an encoding function that transforms the input data, and a decoding function that recreates the input data from the encoded representation.



True Positive and True Negative





$$\textbf{Precision} = \frac{\text{True Positive}}{\text{Actual Results}} \quad \text{or} \quad \frac{\text{True Positive}}{\text{True Positive + False Positive}}$$

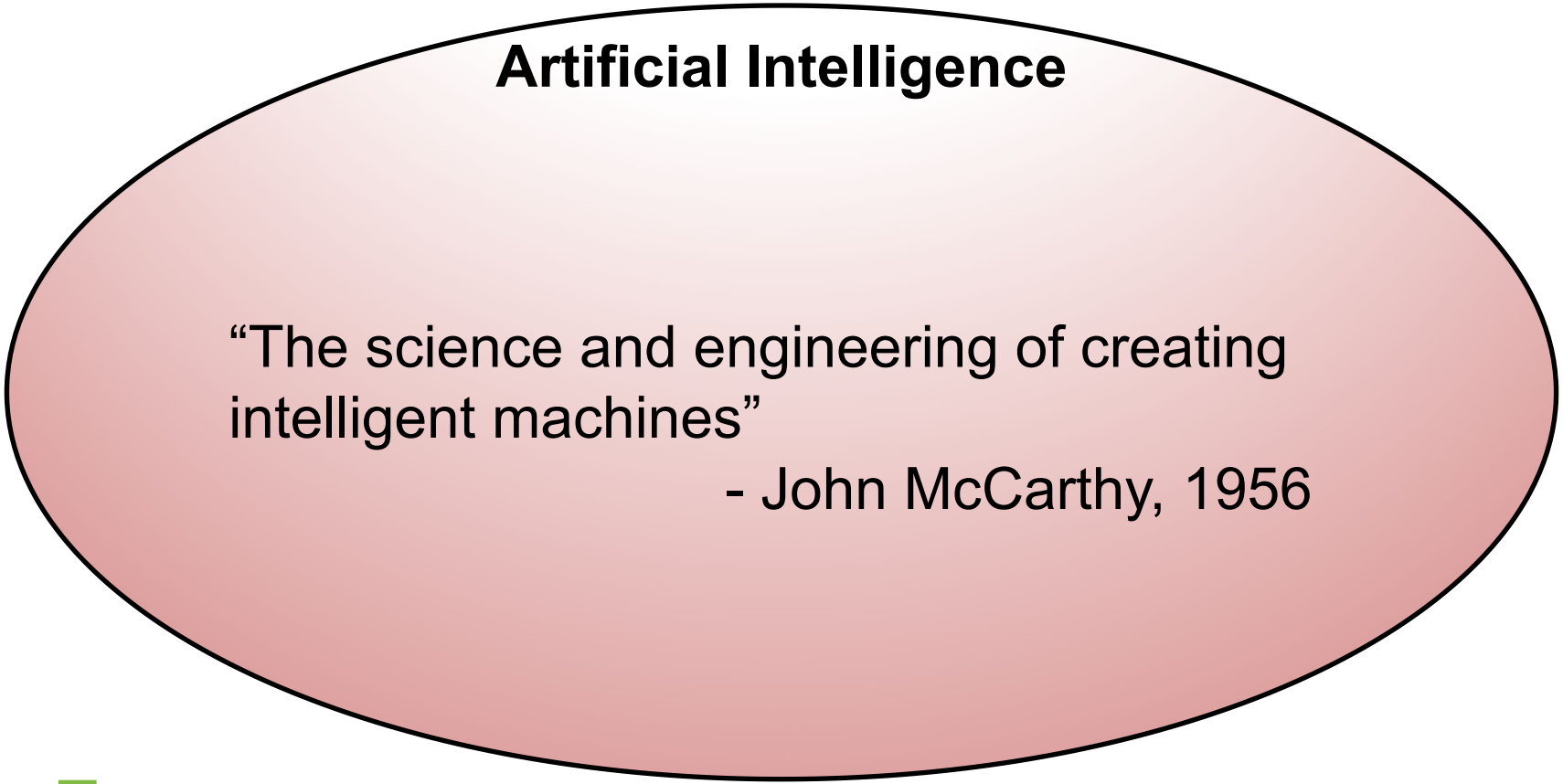
$$\textbf{Recall} = \frac{\text{True Positive}}{\text{Predicted Results}} \quad \text{or} \quad \frac{\text{True Positive}}{\text{True Positive + False Negative}}$$

$$\textbf{Accuracy} = \frac{\text{True Positive + True Negative}}{\text{Total}}$$

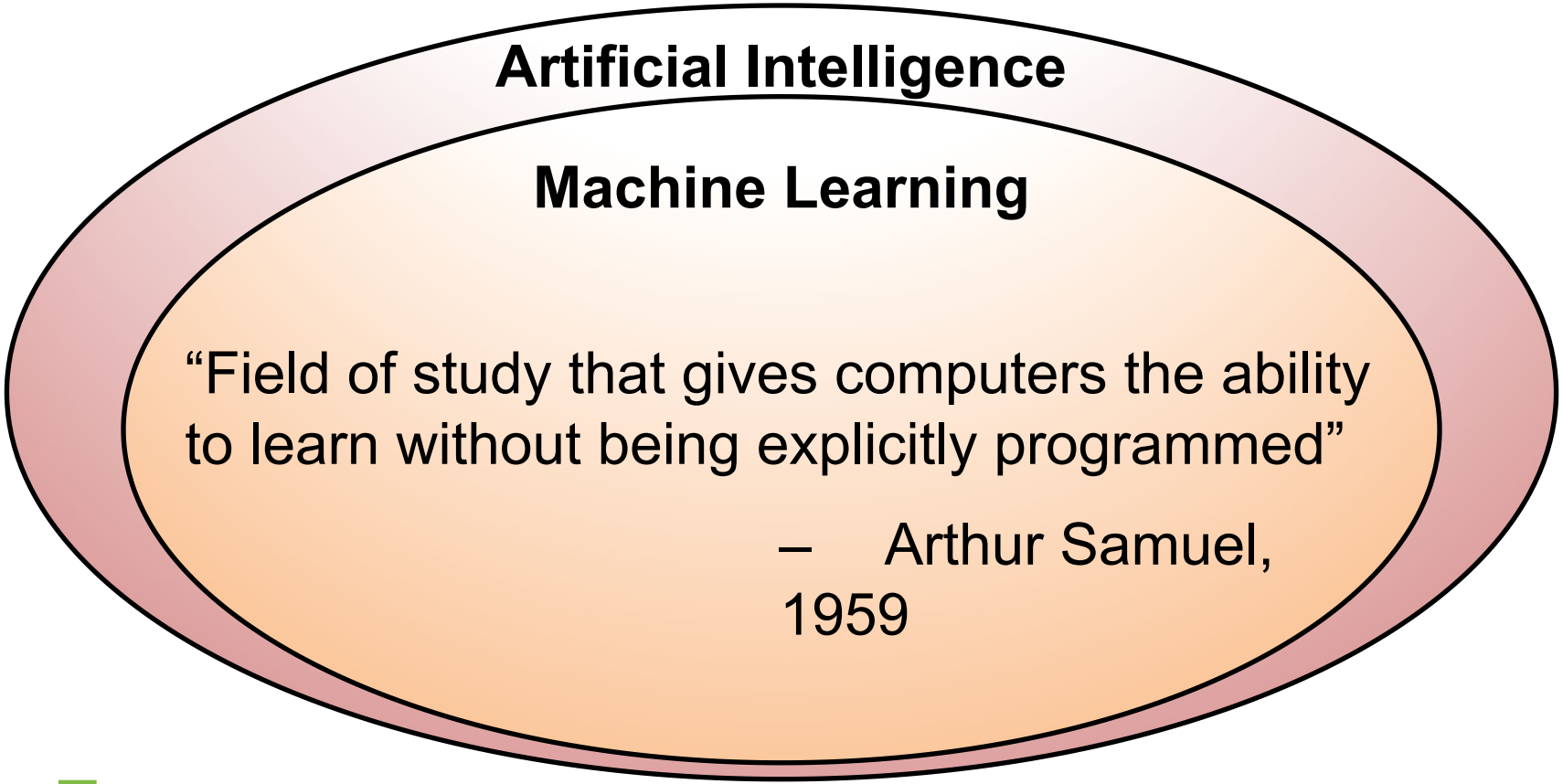
For Chapter 2

Tensorflow and Keras

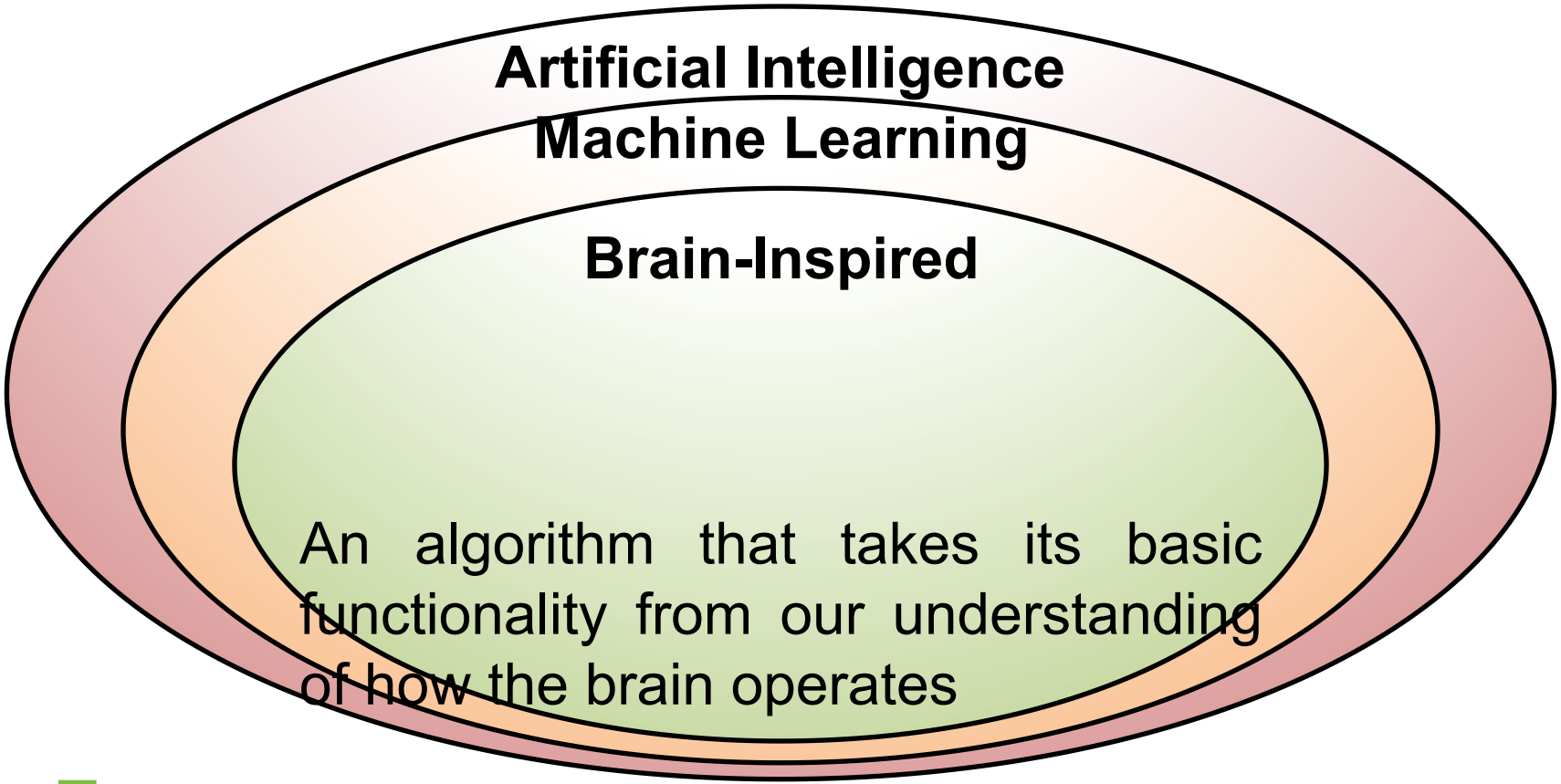
Artificial Intelligence



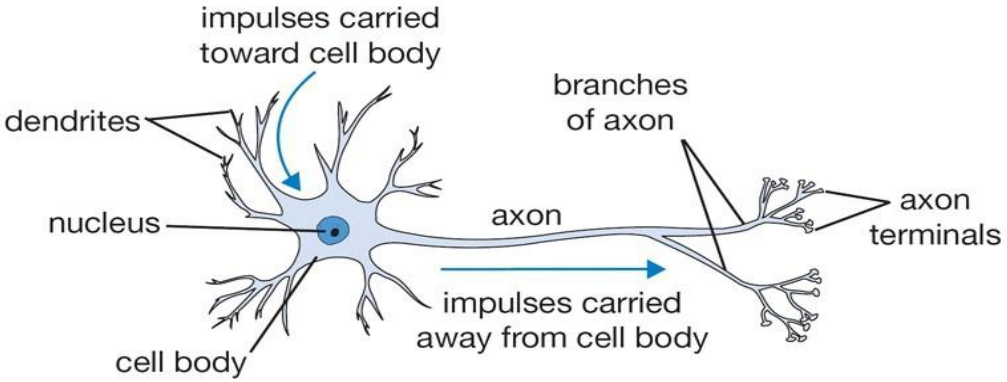
AI and Machine Learning

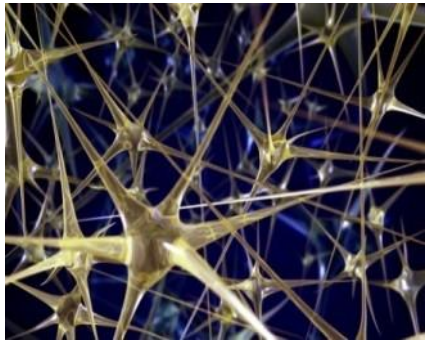


Brain-Inspired Machine Learning



How Does the Brain Work?

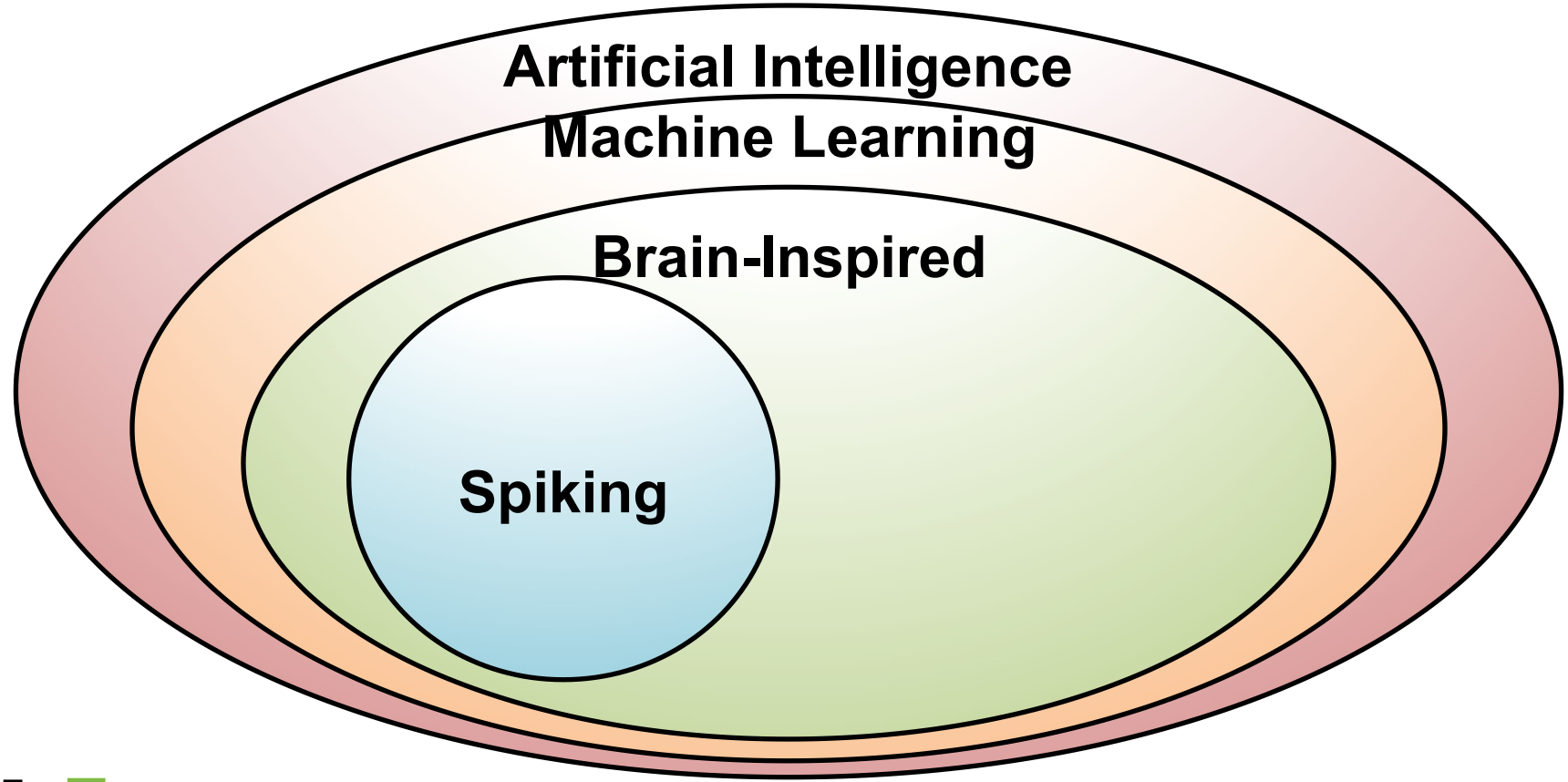




- The basic computational unit of the brain is a **neuron**
  - 86B neurons in the brain
- Neurons are connected with nearly **10^{14} – 10^{15} synapses**
- Neurons receive input signal from **dendrites** and produce output signal along **axon**, which interact with the dendrites of other neurons via **synaptic weights**

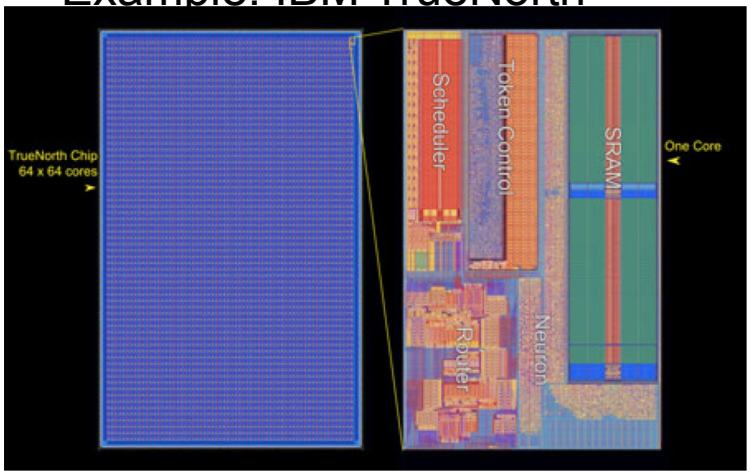
Image Source: Stanford

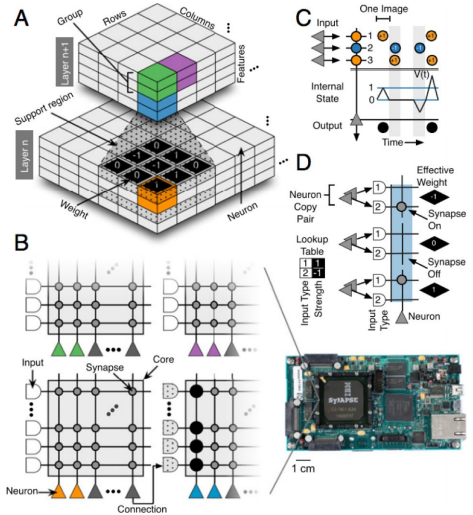
Spiking-based Machine Learning



Spiking Architecture

- Brain-inspired
- Integrate and fire: Once a neuron reaches a certain potential, it spikes and resets its potential
- Example: IBM TrueNorth

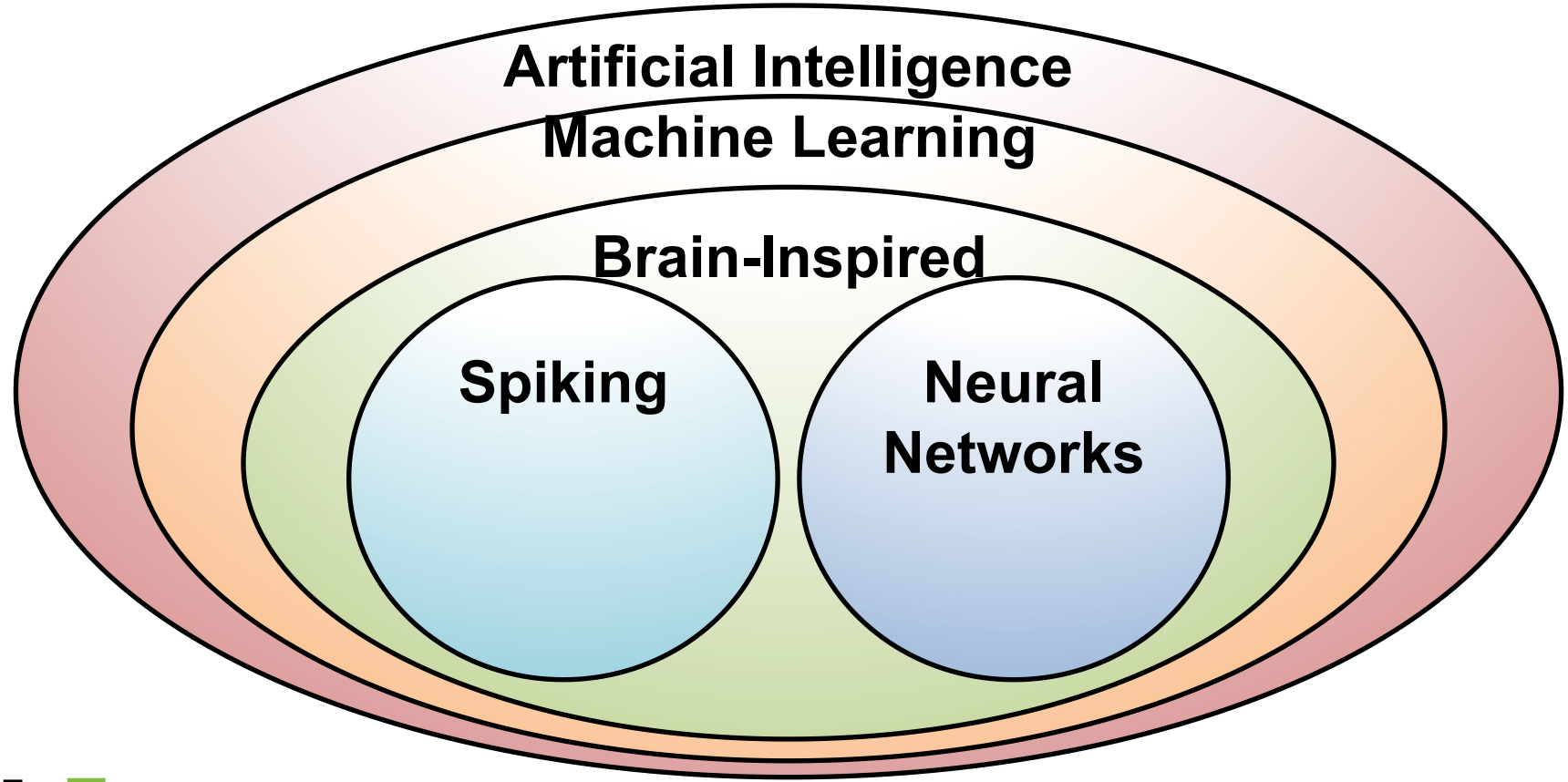




[Merolla et al., Science 2014; Esser et al., PNAS 2016]

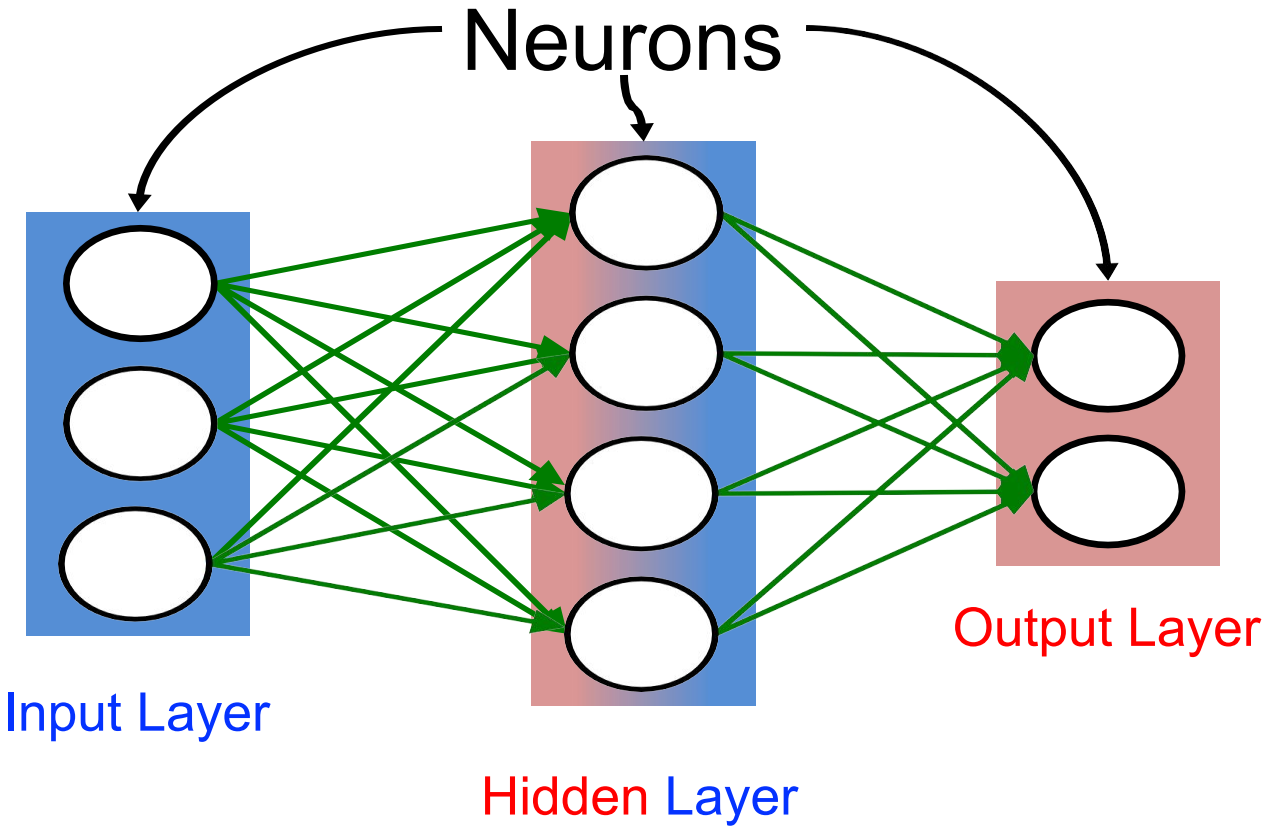
http://www.research.ibm.com/articles/brain-chip.shtml

Machine Learning with Neural Networks

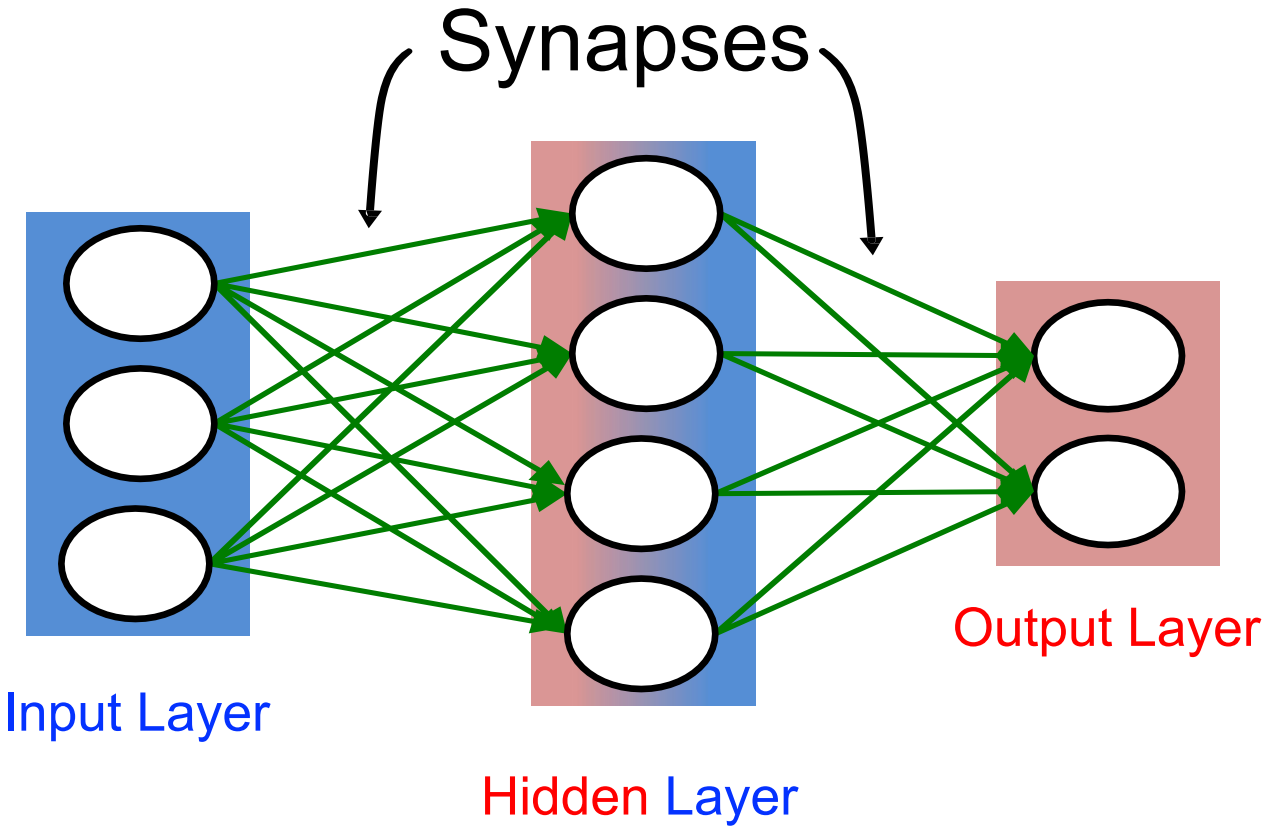


Overview of Deep Neural Networks

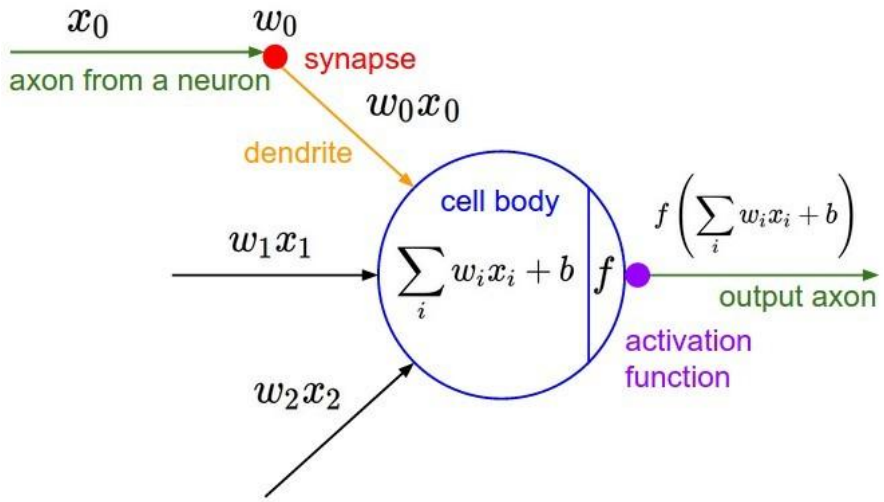
DNN Terminology 101



DNN Terminology 101



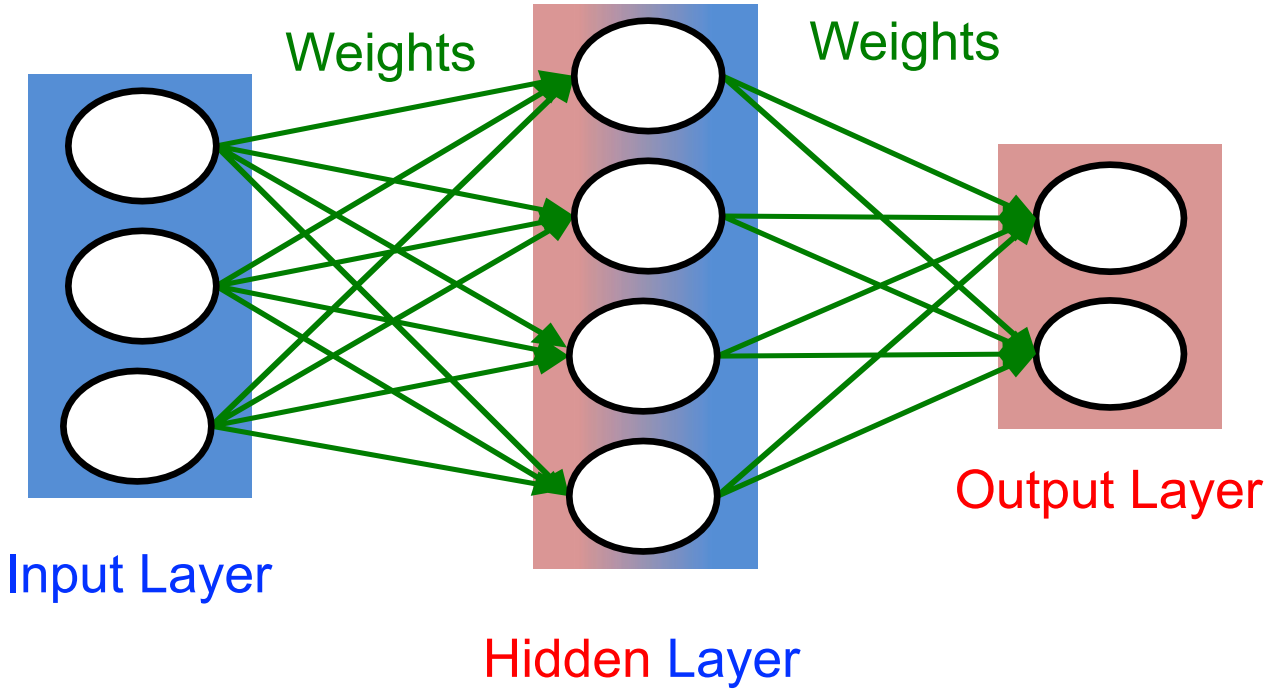
Neural Networks: Weighted Sum



- A node, also called a neuron or Perceptron, is a computational unit that has one or more weighted input connections, a transfer function that combines the inputs in some way, and an output connection

Image Source: Stanford

Many Weighted Sums



What is Deep Learning?

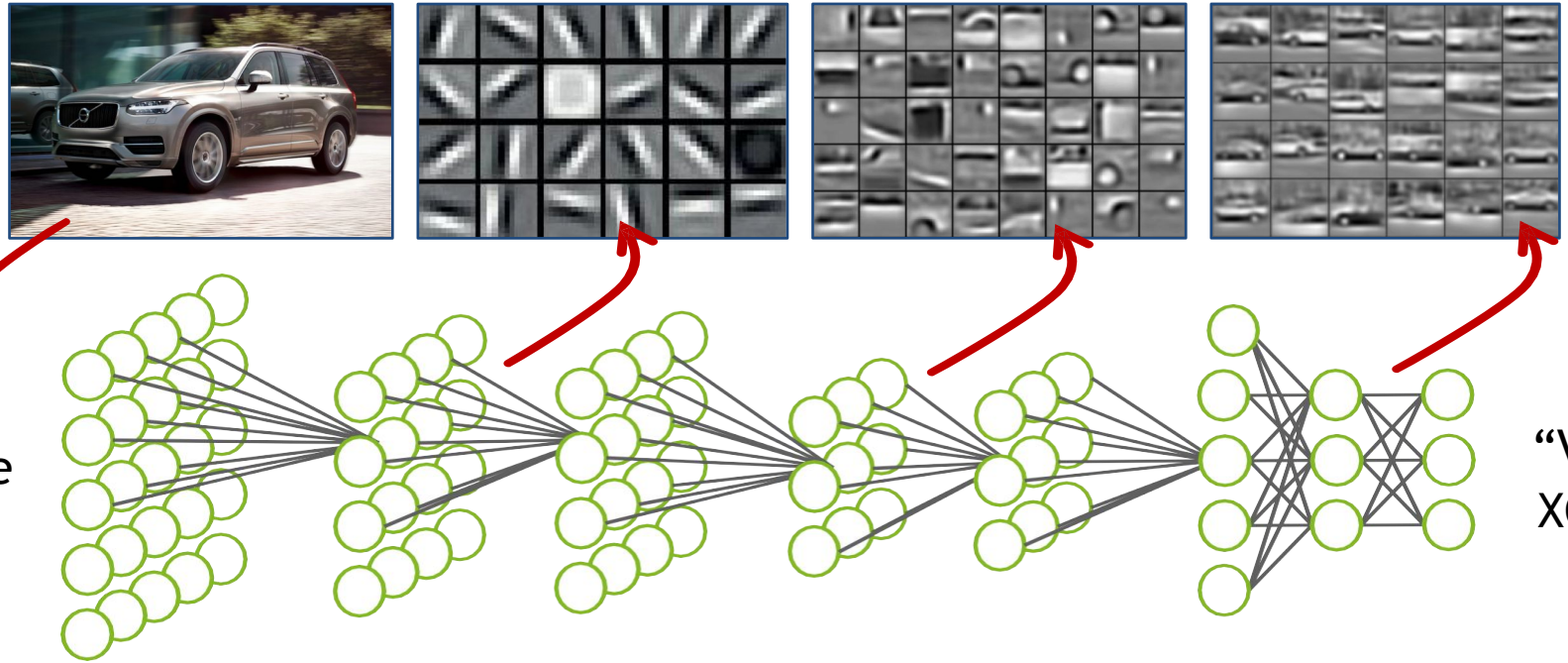
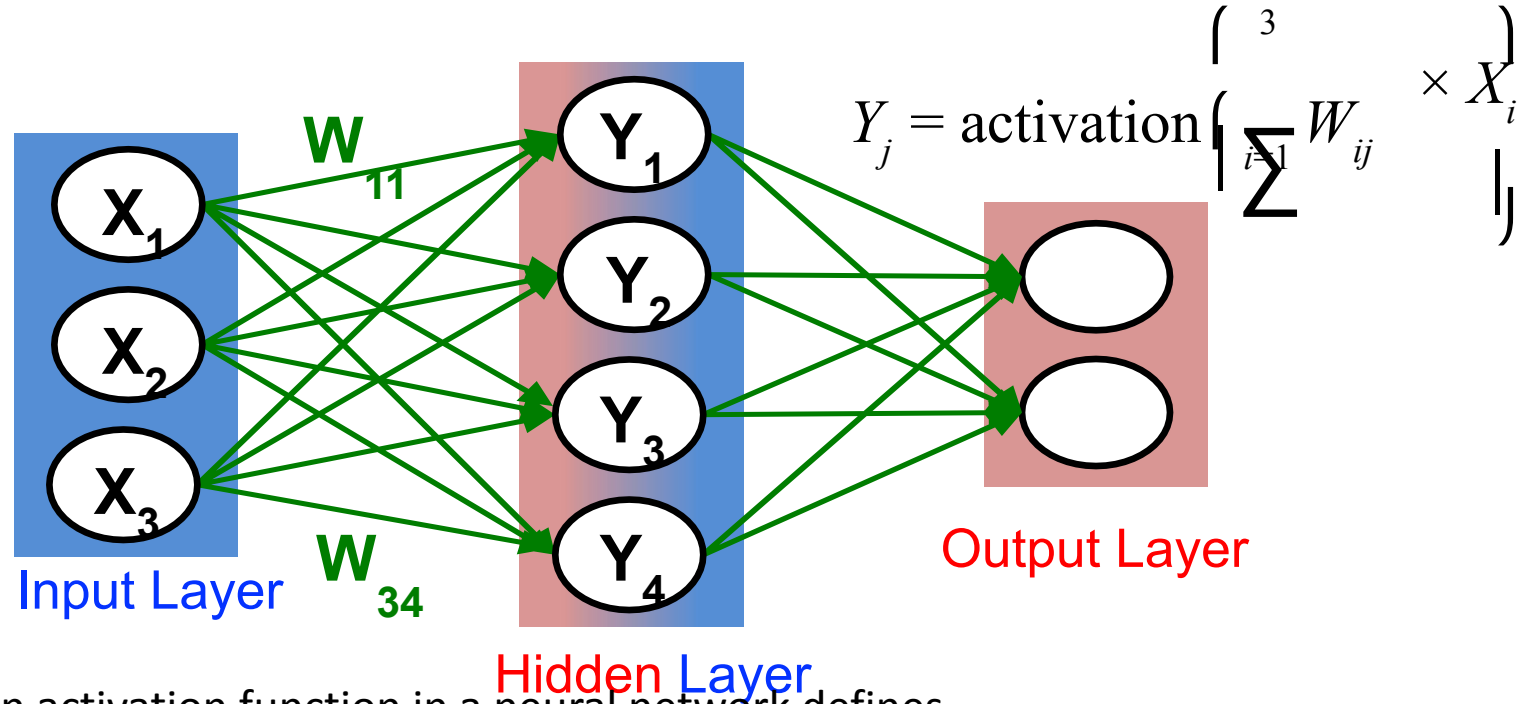


Image Source: [Lee et al., Comm. ACM 2011]

DNN Terminology 101

Each synapse has a **weight** for neuron **activation**

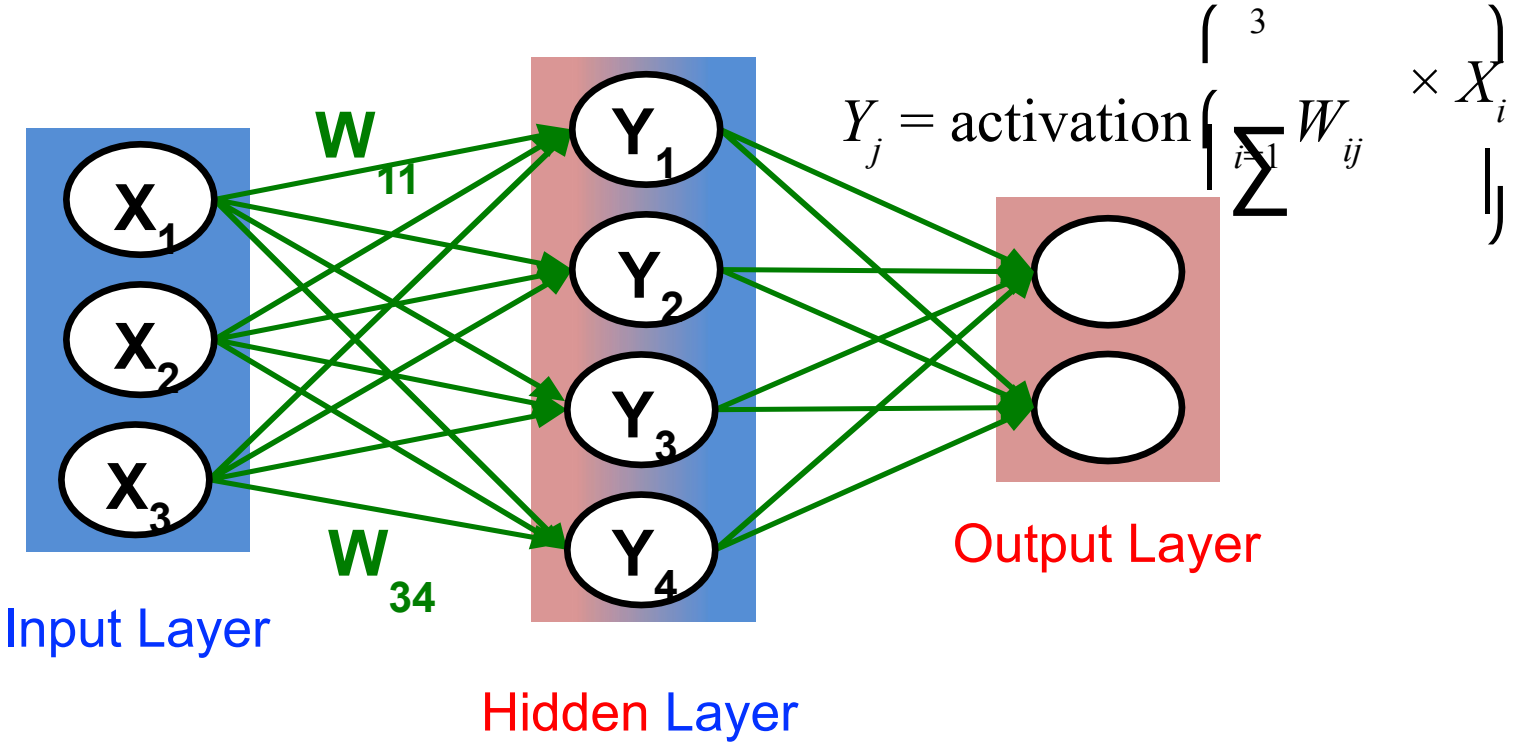


- An activation function in a neural network defines

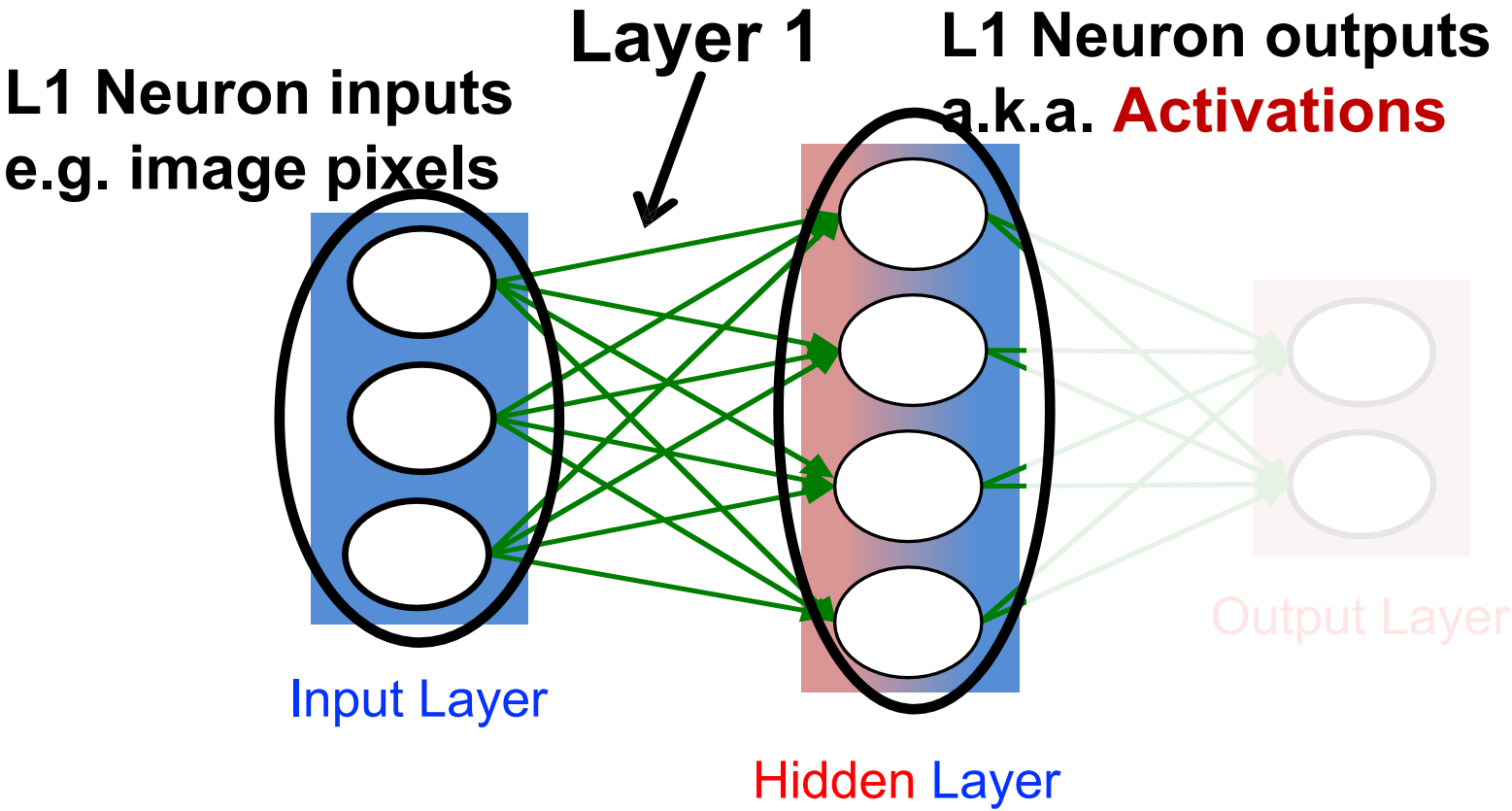
how the weighted sum of the input is transformed into an output from a node

DNN Terminology 101

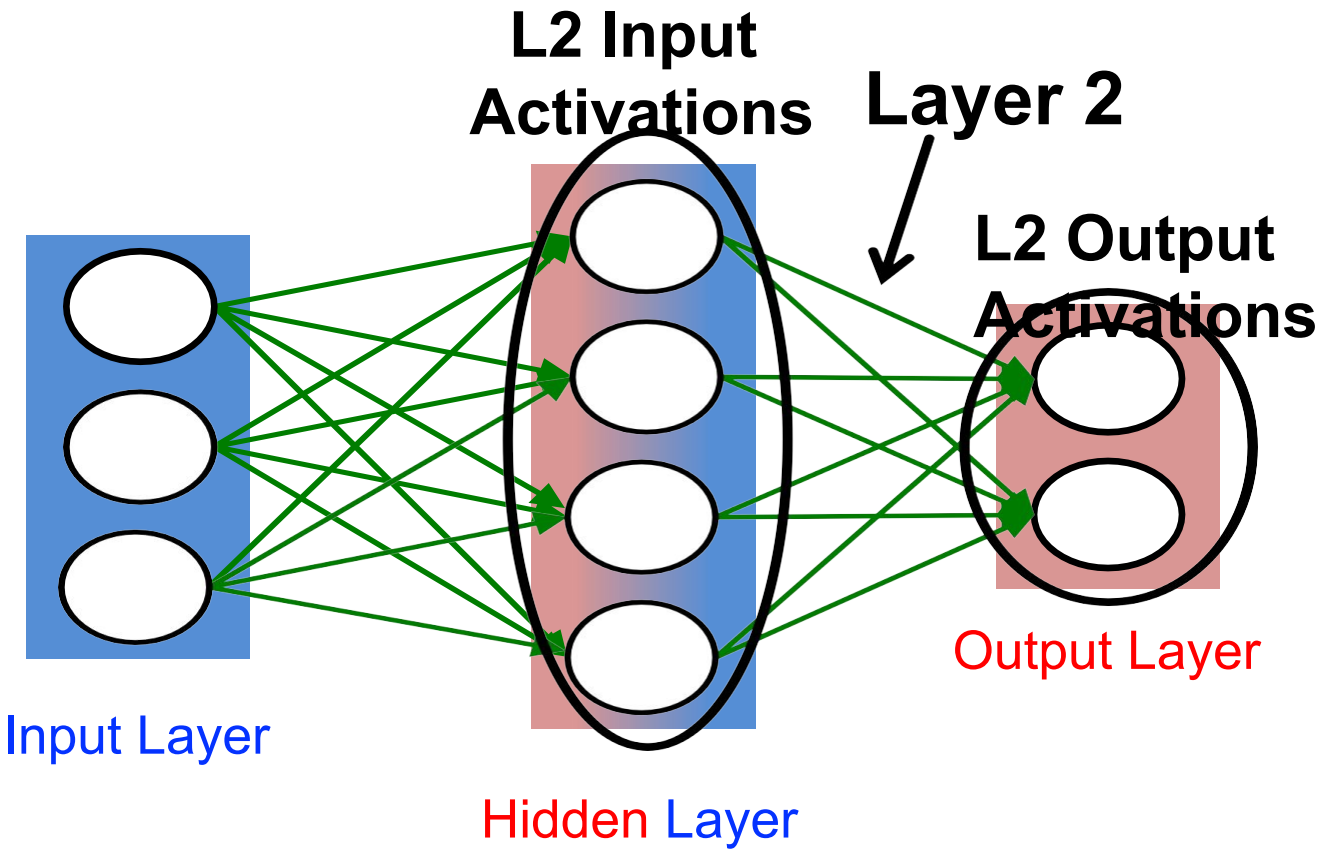
Weight Sharing: multiple synapses use the **same weight value**



DNN Terminology 101

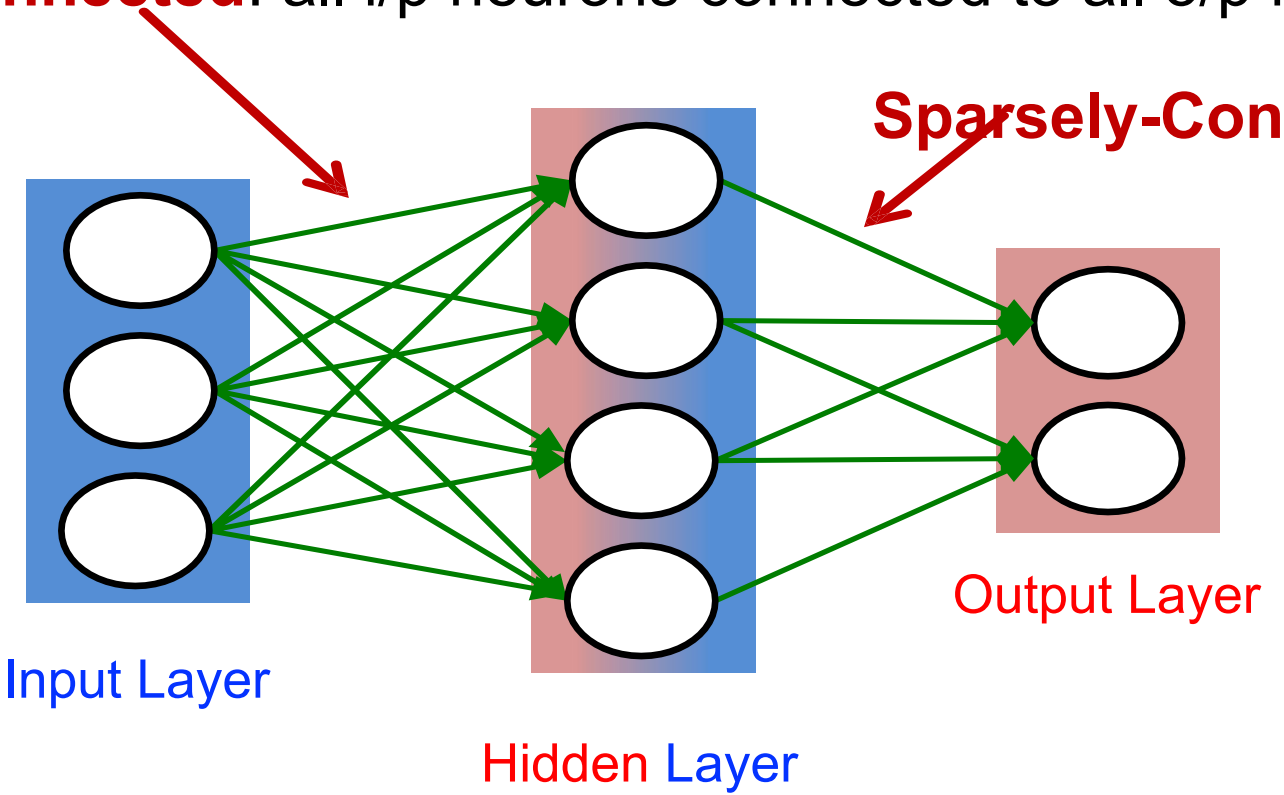


DNN Terminology 101

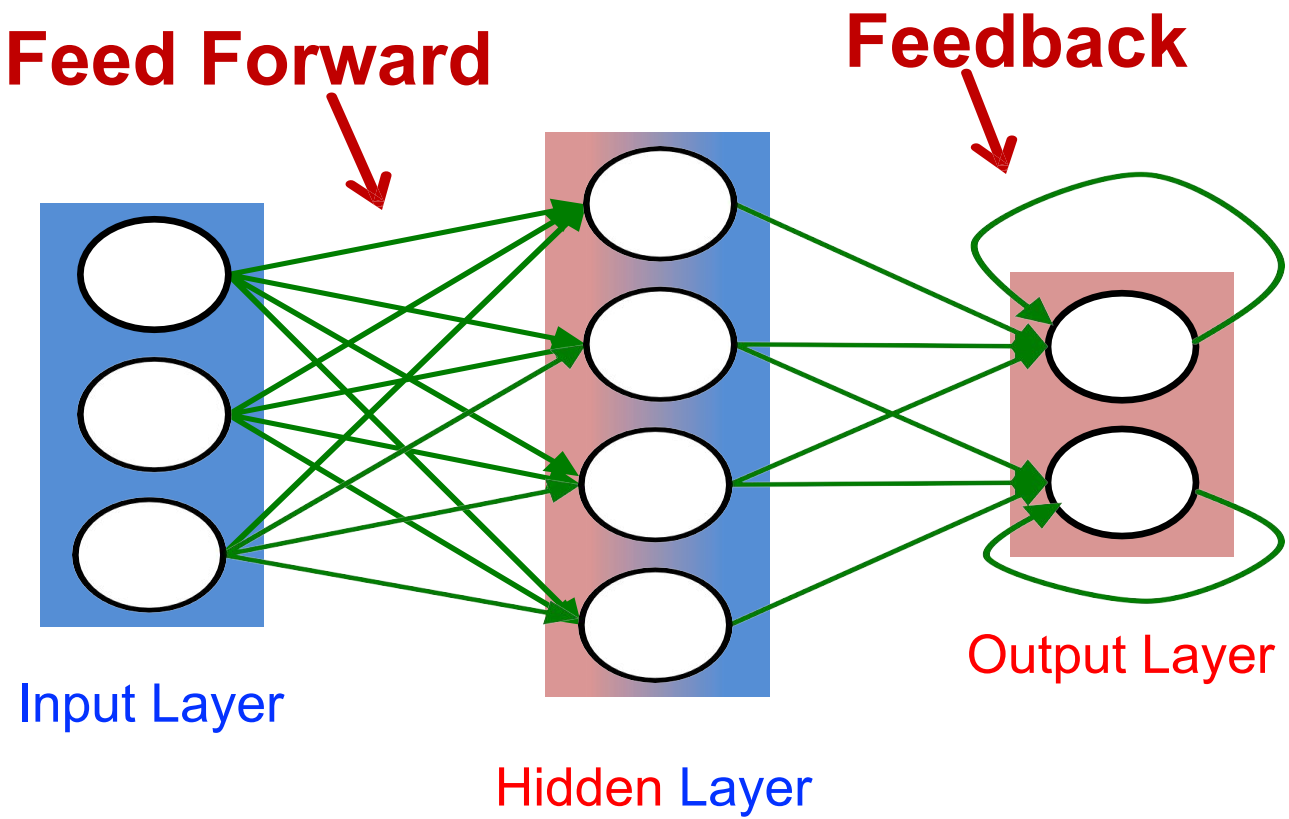


DNN Terminology 101

Fully-Connected: all i/p neurons connected to all o/p neurons



DNN Terminology 101

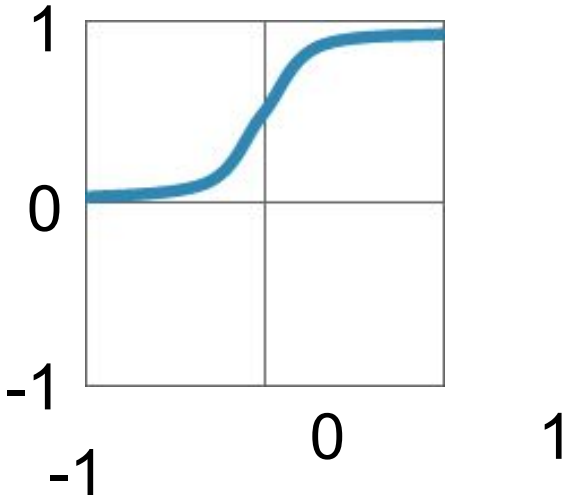


Popular Types of DNNs

- **Fully-Connected NN**
  - feed forward, a.k.a. multilayer perceptron (MLP)
- **Convolutional NN (CNN)**
  - feed forward, sparsely-connected w/ weight sharing
- **Recurrent NN (RNN)**
  - feedback
- **Long Short-Term Memory (LSTM)**
  - feedback + storage

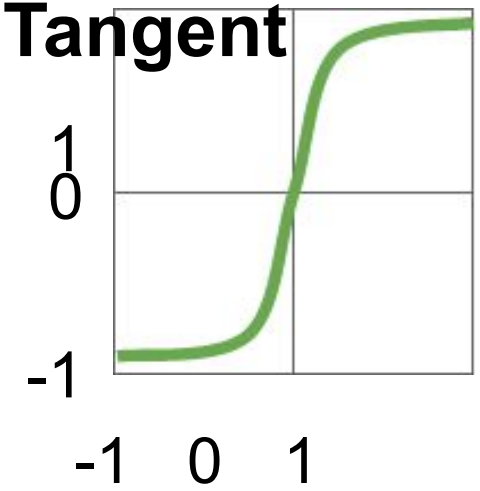
Traditional Activation Functions

Sigmoid



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

Hyperbolic Tangent

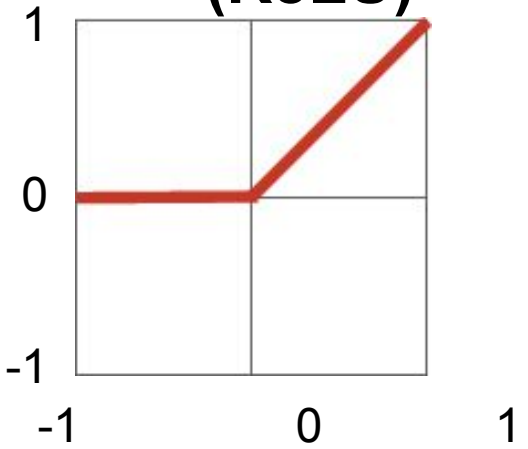


$y=(e^{x}-e^{-x})/(e^{x}+e^{-x})$

Image Source: Caffe Tutorial

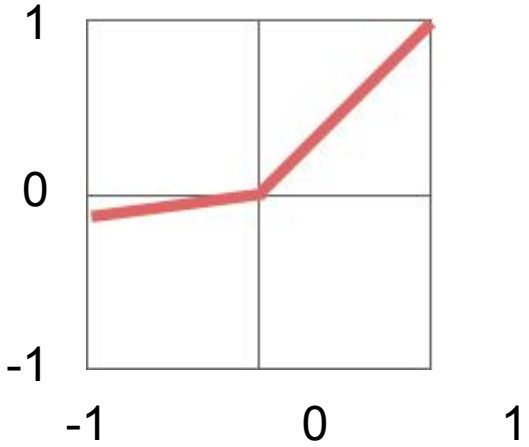
Modern Activation Functions

Rectified Linear Unit (ReLU)



$y=\max(0,x)$

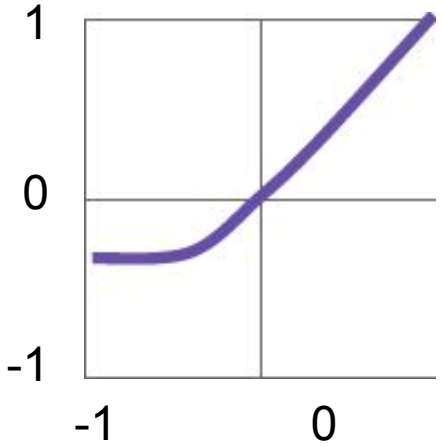
Leaky ReLU



$y=\max(\alpha x,x)$

α = small const. (e.g. 0.1)

Exponential LU



$$y=\begin{cases} x, & x\geq 0 \\ \alpha(e^{x}-1), & x<0 \end{cases}$$

Image Source: Caffe Tutorial

What is the purpose of Leaky ReLU?

Activation Functions

- The choice of activation function in the hidden layer will control how well the network model learns the training dataset. The choice of activation function in the output layer will define the type of predictions the model can make.

- Sometimes the activation function is called a "*transfer function*."
- If the output range of the activation function is limited, then it may be called a "*squashing function*."
- Many activation functions are nonlinear and may be referred to as the "*nonlinearity*" in the layer or the network design.

- The activation function is used within or after the internal processing of each node in the network, although networks are designed to use the same activation function for all nodes in a layer.

Activation Functions

- A network may have three types of layers: input layers that take raw input from the domain, **hidden layers** that take input from another layer and pass output to another layer, and **output layers** that make a prediction.

- All hidden layers typically use the same activation function.

- The output layer will typically use a different activation function from the hidden layers and is dependent upon the type of prediction required by the model.

- Activation functions are also typically differentiable, meaning the first-order derivative can be calculated for a given input value. This is required given that neural networks are typically trained using the backpropagation of error algorithm that requires the derivative of prediction error in order to update the weights of the model.

Activation for Hidden Layers

A hidden layer in a neural network is a layer that receives input from another layer (such as another hidden layer or an input layer) and provides output to another layer (such as another hidden layer or an output layer).
A neural network may have zero or more hidden layers.

Typically, a differentiable nonlinear activation function is used in the hidden layers of a neural network. This allows the model to learn more complex functions than a network trained using a linear activation function.:

- **ReLU Hidden Layer Activation Function**
- It is common because it is both simple to implement and effective at overcoming the limitations of other previously popular activation functions, such as Sigmoid and Tanh.
- it is less susceptible to vanishing gradients that prevent deep models from being trained,

Why ReLU is less susceptible to vanishing gradients?

Output Activation Functions

Linear Output Activation Function

The linear activation function is also called "*identity*" (multiplied by 1.0) or "*no activation*." This is because the linear activation function does not change the weighted sum of the input in any way and instead returns the value directly.

Sigmoid Output Activation Function

- Target labels used to train a model with a sigmoid activation function in the output layer will have the values 0 or 1.

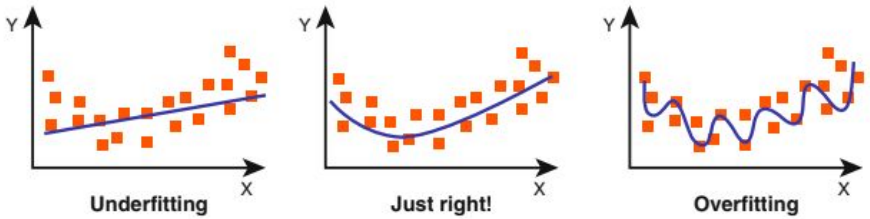
Softmax Output Activation Function

The softmax function outputs a vector of values that sum to 1.0 that can be interpreted as probabilities of class membership.

As such, the input to the function is a vector of real values and the output is a vector of

Regularization

A network is overfitting, when it performs well on its training data, but not on new unseen inputs.



Solutions:

- Data augmentation : fake data is generated and added to the training set.
  - not for many other tasks such as natural language processing.
- Parameter norm penalties : a penalty is added to the loss function to optimally limit the capacity of a model.
- Dropout : some connections are randomly put to zero every iteration.
- k-Fold Cross-Validation: Cross-validation is a resampling procedure used to evaluate machine learning models on a limited data sample.

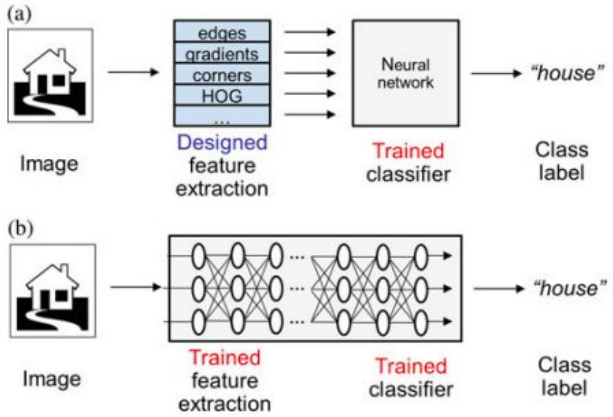


Fig. 1.2 Comparing the (**a**) classical approach to machine learning using hand-designed features to the (**b**) multi-layered representational approach of deep learning operating on raw inputs

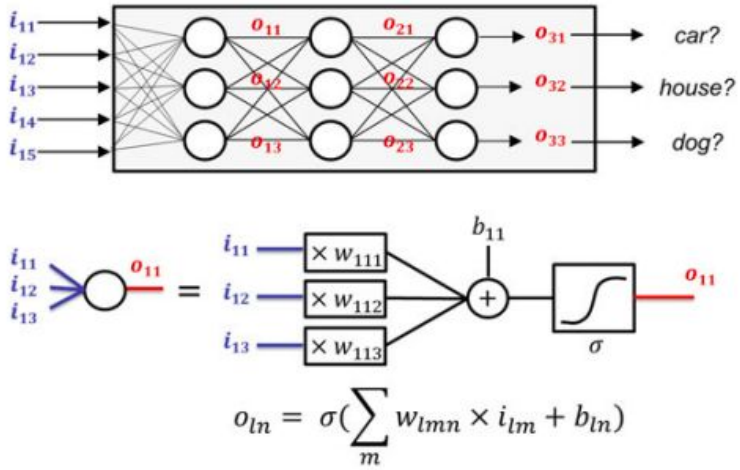
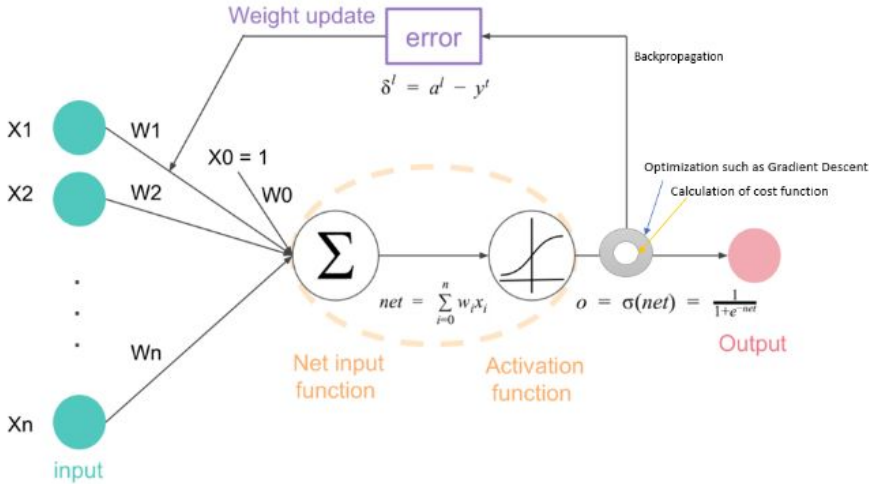


Fig. 1.4 Graphical representation of a simple deep feed-forward neural network

$$O = a\left(\sum_{m=0}^{M} W[m] \times x[m] + B\right)$$

Vanishing Gradient Problem



Deep networks are hard to train because of the notorious vanishing gradient problem — as the gradient is back-propagated to earlier layers, repeated multiplication may make the gradient infinitively small.

As a result, as the network goes deeper, its performance gets saturated or even starts degrading rapidly.

Vanishing Gradient Problem