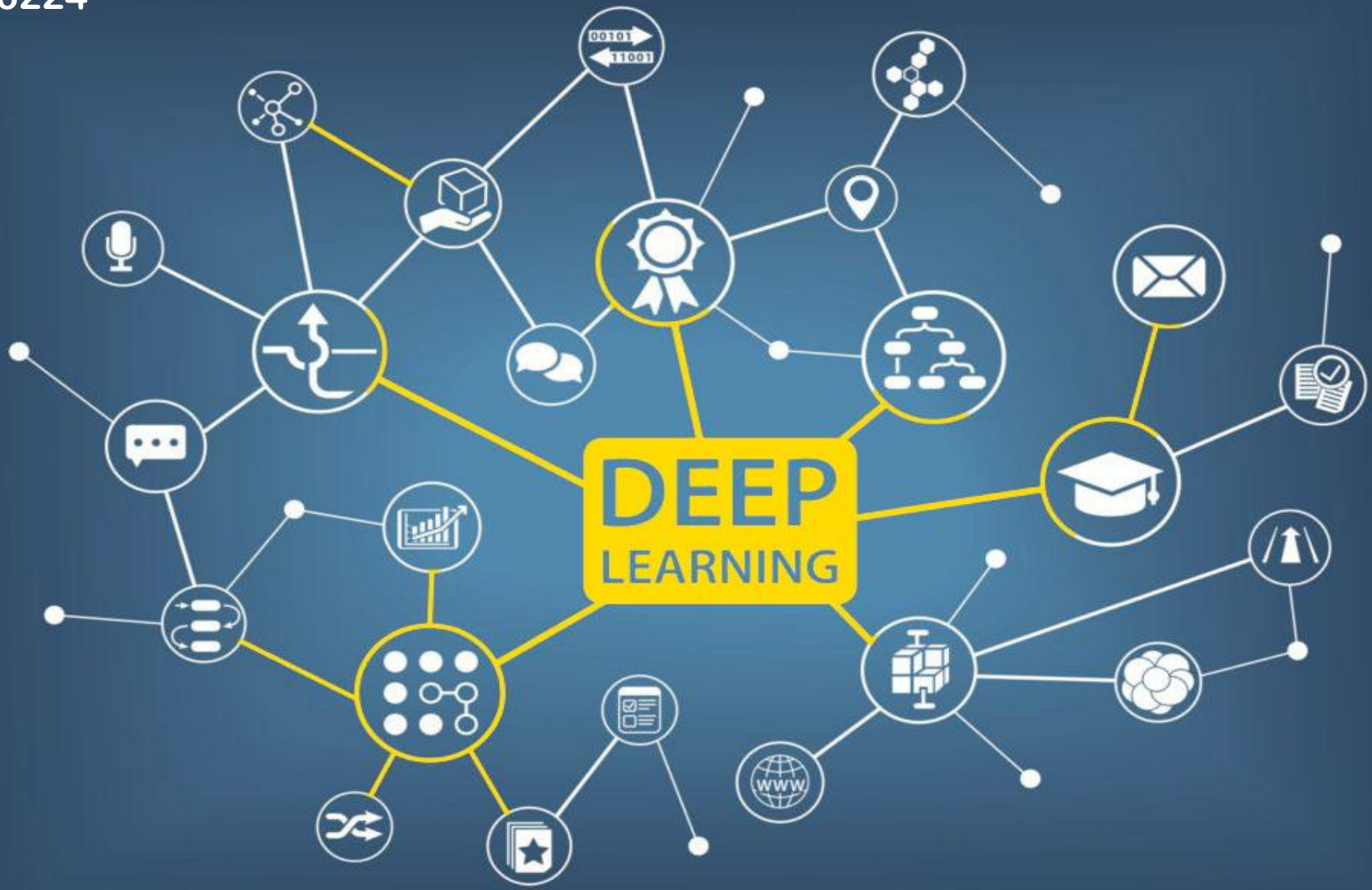
MDS6224



Deep Learning and Their Applications

Dr. Chen Chen SME, CUHK Shenzhen

Lecture 1.5
CONTENTS

Introduction to ML

Supervised vs Unsupervised

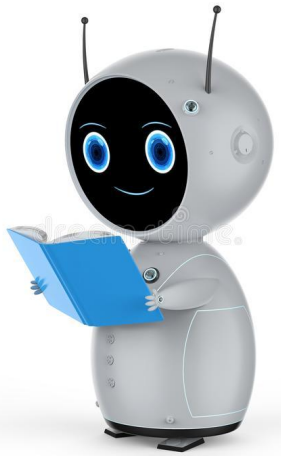
Learning is about trials and errors

- Human are not born with omnipotence. We become capable by learning.
- We start out usually very badly, but gradually improve ourselves through multiple rounds of failures, with the some helpful instructions.
- Sometime you need to set a goal for them. Let them know when they did right or wrong. Reward when they do right.
- Examples: babies learning to walk, you learning to code. Some are better learners: accurate, fast and adaptive.

Machine learns from trials and errors too. Just like human being.

Definition of ML:

Give a task **T** and a performances measure **P**, a machine learning program is to find a way **A** (typically an algorithm) to learn from past experience **E** (i.e. training on some existing data), so that the performance on T as measured by P is improved with E (by Tom Mitchell).



Key components of machine learning

- The task T (of course)
- The experience, typically a dataset E
- The performance measure P
- A way to improve P, usually an algorithm

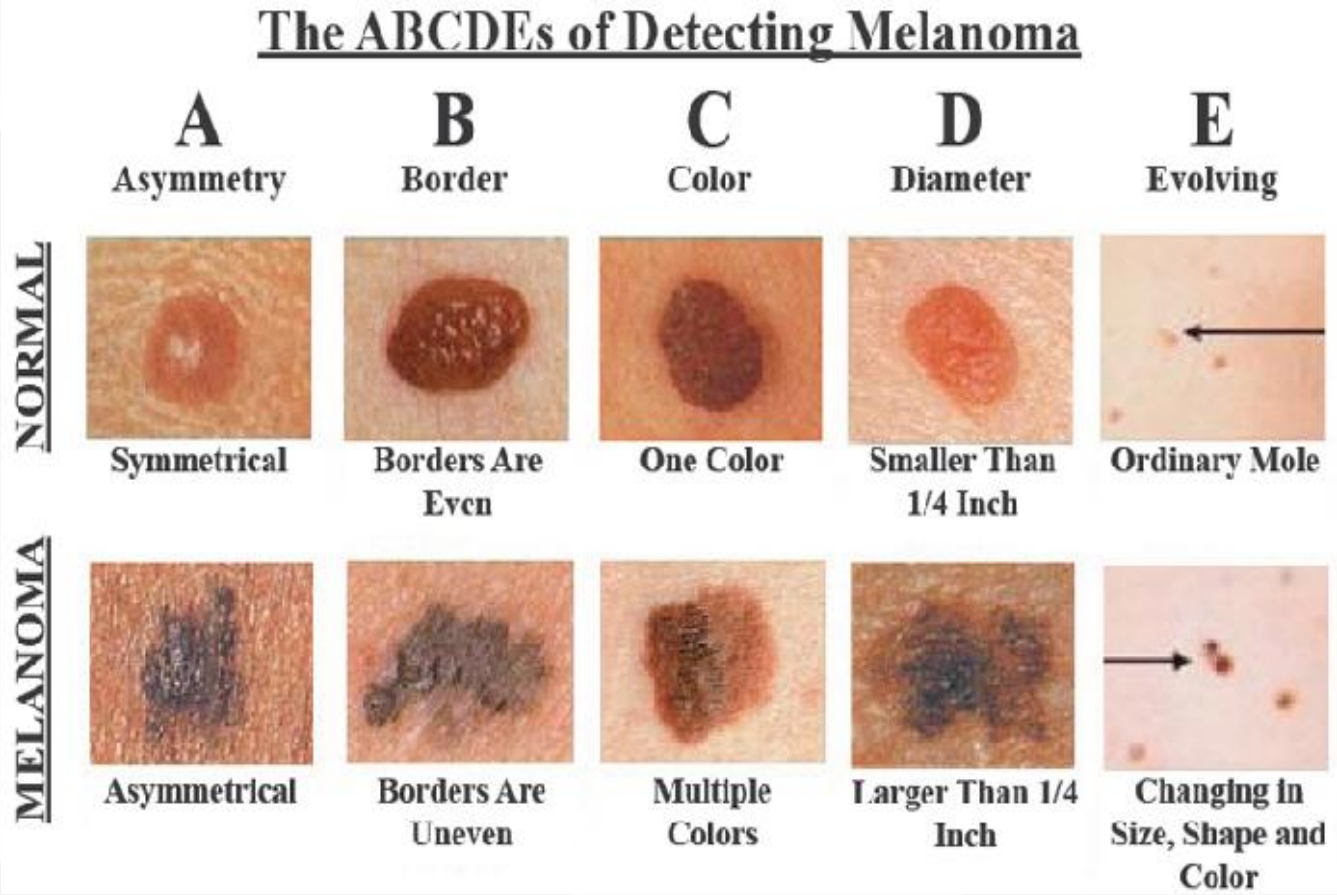
- A loss/cost function, usually in a form that is easily differentiable, and consistent with the performance measure
- A machine learning algorithm that typically involves multiple runs on data

A Example: Spam Detection

- The task T: Finding out the spams
- The dataset: a collection of *labeled* emails (1 if a spam, 0 if not)
- The performance measure P: accuracy, precision, recall, etc.
- A way to improve P, usually an algorithm

- This is a typical binary classification, where you classify samples from your dateset into one of two classes.
- The simpliest solution is logitistic regression, which we will talk about later.

Detecting Melanoma, a skin cancer



ImageNet

More than 14 million images have been hand-annotated by the project to indicate what objects are pictured. It contains more than 20,000 categories

Computer vision and picture caption

A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.

A little girl sitting on a bed with a teddy bear.

A group of people sitting on a boat in the water.

A giraffe standing in a forest with trees in the background.

What in common so far?

- The common scheme of these problems are that we provided the machine with human supervision.
  - E.g. before training, Chen asked his friends to help label 50K emails as spam or ham.

What in common so far?

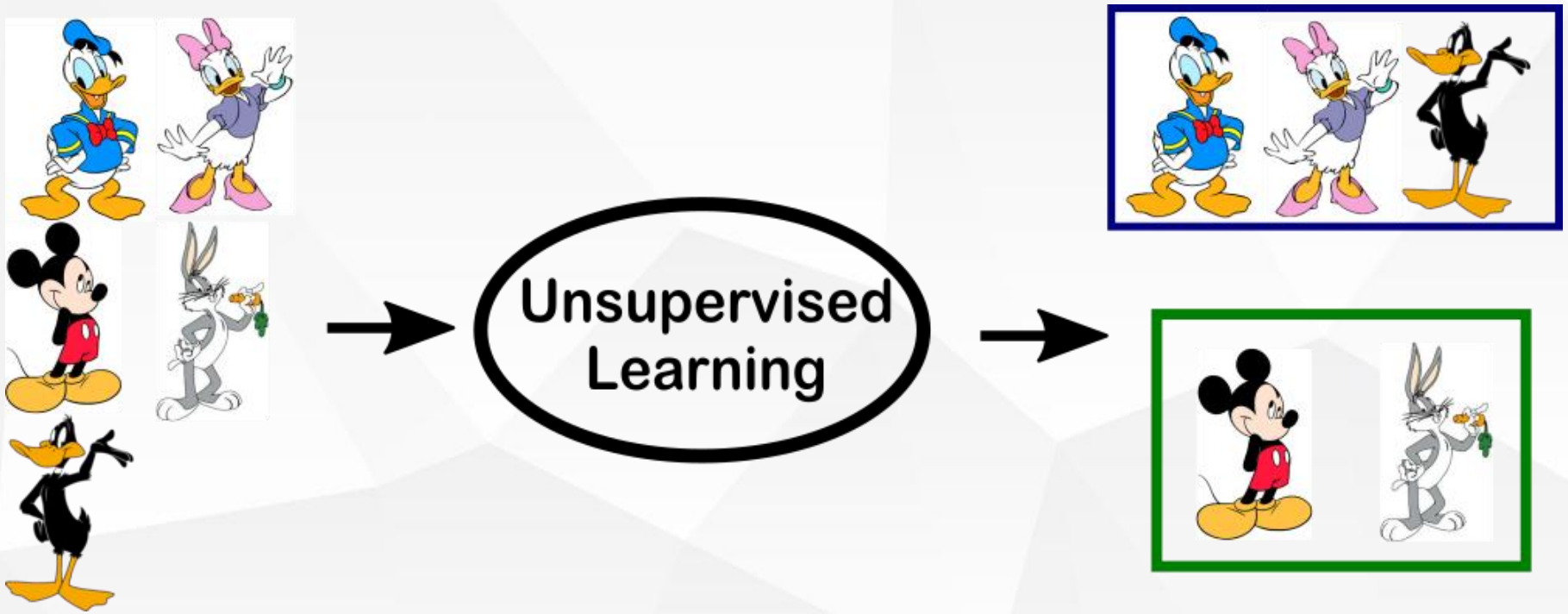
- The common scheme of these problems are that we provided the machine with human supervision.
  - E.g. before training, Chen asked his friends to help label 50K emails as spam or ham.
- Human puts their hands on the machine during the training process.
  - Equivalent to teaching baby walk: hands-on ==》 hands-off

Supervised learning

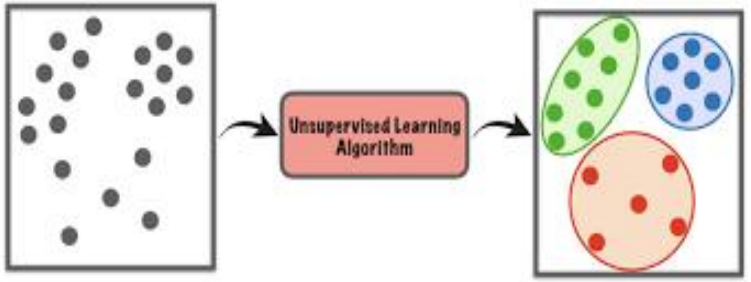
- What is the common theme of previous ML examples?
  - We are asked to build something, typically called a ***classifier,*** to assign labels to unlabeled data, based on labeled data from a set of instances, known as the ***training set***.
- This type of machine learning problems are called ***supervised learning*** problems (as we have true labels to *teach* our classifier), also known as ***classification*** (as it involves *classifying* data with learned classifiers).
- A classifier is typically a model or an ensemble of models (i.e. a collection of models), but not always so.
- The process of building a classifier from the training data is called induction, while the process of applying the model to predict on unseen test instances is called deduction. For prediction, we often need to specify a *decision rule*.

But what if the training data do not have labels

- There are some patterns in data which make some more similar to others.

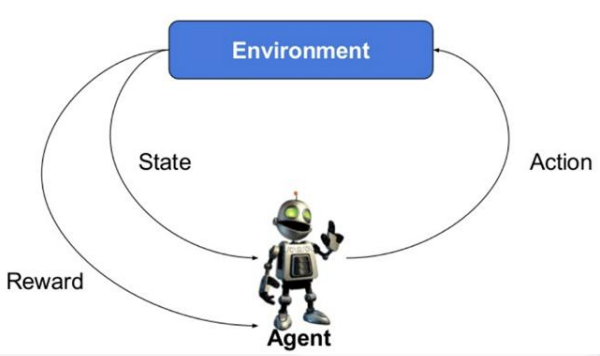


Unsupervised learning



- ***Unsupervised learning***, or ***clustering***, is another major type of machine learning problems that involves a data set with no pre-existing labels (i.e. minimum human supervision). The goal of unsupervised learning is to group objects into clusters so that objects within the same cluster will be as similar as possible.
  - Clustering is often used to detect unknown patterns and features.
  - When total human supervision is impossible to achieve, clustering is often used for feature extraction.
  - It often precedes supervised learning: extracting features (i.e. the input) for future labeling task

Reinforcement Learning



- The last major machine learning paradigm is *Reinforcement learning* (RL), which concerned with how intelligent agents ought to take actions in an environment in order to maximize the notion of cumulative reward.
  - In economics and game theory, reinforcement learning may be used to explain how equilibrium may arise under bounded rationality.
  - A *Markov decision process*
  - Widely involved in robot design, AI. A premium example is AlphaGo.
  - Instead of learning a label for a new object, it attempts to find a *policy* that maximizes the *return* under a certain *state*. The reward is measured in *value function.*
  - Not a focus of this class since it is not commonly used for text analysis.
  - However, combining with deep neural network, it does have a lot of applications in NLP.

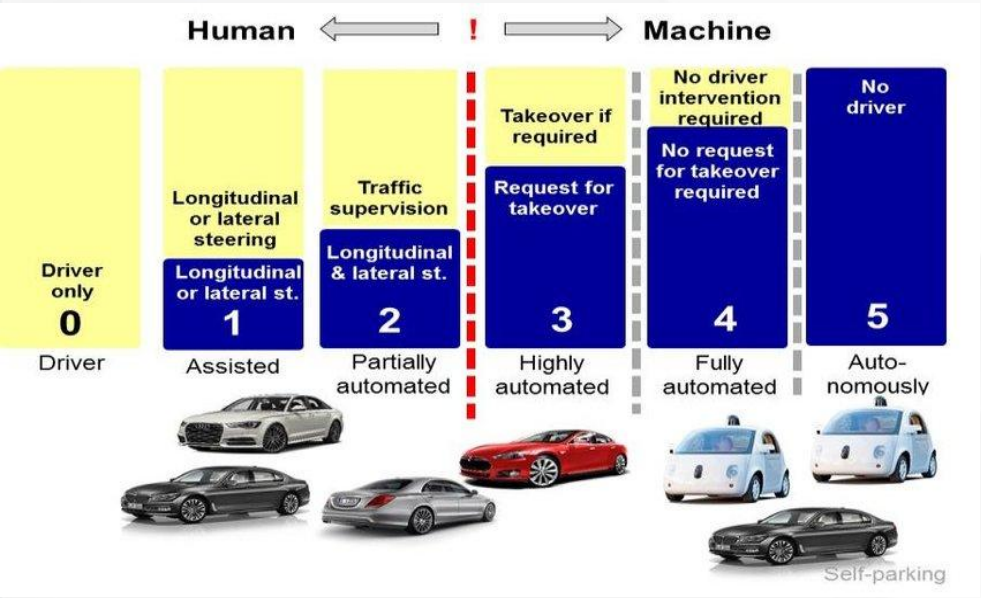
A fundamental assumption of machine learning

Assumption: The distribution of training examples is identical to the distribution of test examples (including future unseen examples).

- In practice, this assumption is **often** violated to certain degree.
- Strong violations will clearly result in poor classification accuracy.
- To achieve good accuracy on the test data, training examples must be sufficiently representative of the test data.
- In cases where the samples you wish to make a prediction on do not follow the same distribution as your training samples, all traditional ML methods would fail. And it might cause a huge problem
  - Example: self-driving algorithm

How far away are we from autonomous cars?

- Can we make the right predictions for objects that are not seen in the training set?



Unlike AI, human brains are able to generalize well

Calculus taught in class

Calculus in homework

Calculus in the Final

The common classification algorithms

- Rule based classifiers
- Logistic regression
- Decision trees
- Nearest neighbour classifiers (kNN)
- Naive Bayes (NB) and Bayesian networks
- Support Vector machines (SVM)
- Neural network (NNet) and deep learning (DL)
- Ensemble methods *
- And so on...

Linear Algebra and Matrix Calculus

Optional!

Matrix Calculus

- Matrix calculus is a specialized notation for doing multivariable calculus, especially over spaces of matrices.
- Two competing notational conventions split the field of matrix calculus into two separate groups.

Numerator-Layout Notation

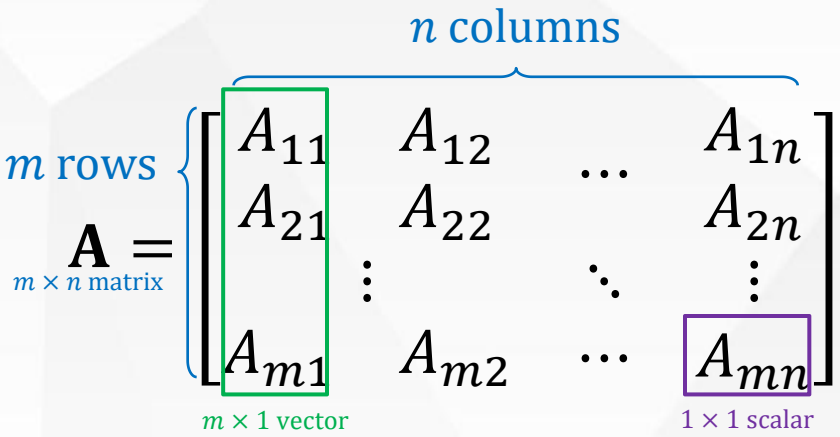
$$\frac{\partial \mathbf{y}}{\partial \mathrm{x}} = \begin{bmatrix} \frac{\partial y_1}{\partial x} \\ \frac{\partial y_2}{\partial x} \\ \vdots \\ \frac{\partial y_m}{\partial x} \end{bmatrix}, \quad \frac{\partial y}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial y}{\partial x_1} & \frac{\partial y}{\partial x_2} & \cdots & \frac{\partial y}{\partial x_n} \end{bmatrix}$$

Denominator-Layout Notation

$$\frac{\partial \mathbf{y}}{\partial \mathrm{x}} = \begin{bmatrix} \frac{\partial y_1}{\partial x} & \frac{\partial y_2}{\partial x} & \cdots & \frac{\partial y_m}{\partial x} \end{bmatrix}, \quad \frac{\partial y}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial y}{\partial x_1} \\ \frac{\partial y}{\partial x_2} \\ \vdots \\ \frac{\partial y}{\partial x_n} \end{bmatrix}$$

Notation

- Matrix: $\mathbf{A}$, $\mathbf{X}$, $\mathbf{Y}$
  - bold capital letter
- Vector: $\mathbf{a}$, $\mathbf{x}$, $\mathbf{y}$ (**column**)
  - boldface lowercase letter
- Scalar: a, x, y
  - lowercase italic typeface
- Transpose: $\mathbf{A}^T$, $\mathbf{a}^T$
- Trace: $\mathrm{tr}(\mathbf{A})$
  - $\mathrm{tr}(\mathbf{A}) = A_{11} + A_{22} + \cdots + A_{nn} = \sum_{i=1}^{n} A_{ii}$
- Determinant: $\det(\mathbf{A})$



Properties of Transpose

- $(\mathbf{A}^T)^T = \mathbf{A}$
- $(\mathbf{A} + \mathbf{B})^T = \mathbf{A}^T + \mathbf{B}^T$
- $(\mathbf{A} + \mathbf{B} + \mathbf{C})^T = \mathbf{A}^T + \mathbf{B}^T + \mathbf{C}^T$
- $(r\mathbf{A})^T = r\mathbf{A}^T$
- $(\mathbf{AB})^T = \mathbf{B}^T\mathbf{A}^T$
- $(\mathbf{ABC})^T = \mathbf{C}^T\mathbf{B}^T\mathbf{A}^T$

Trace & Determinant

- If $\mathbf{A}$ is a square $n-$by$-n$ matrix and if $\lambda_1, \cdots, \lambda_n$ are the eigenvalues of $\mathbf{A}$, then
  - $\mathrm{tr}(\mathbf{A}) = A_{11} + A_{22} + \cdots + A_{nn} = \sum_{i=1}^{n} A_{ii} = \sum_{i=1}^{n} \lambda_i$
  - $\det(\mathbf{A}) = \sum_{i=1}^{n} (-1)^{i+j} a_{i,j} M_{i,j} = \prod_{i=1}^{n} \lambda_i$
    - Minor $M_{i,j}$: the determinant of the $(n-1) \times (n-1)$-matrix that results from $\mathbf{A}$ by removing the ith row and the jth column.
    - Cofactor $C_{i,j}$: $(-1)^{i+j} M_{i,j}$
    - Cofactor matrix: $\mathbf{C} = \begin{bmatrix} C_{11} & C_{12} & \cdots & C_{1n} \\ C_{21} & C_{22} & & C_{2n} \\ \vdots & & \ddots & \vdots \\ C_{n1} & C_{n2} & \cdots & C_{nn} \end{bmatrix}$

$$\mathbf{A}^{-1} = \frac{(\mathbf{C})^T}{\det(\mathbf{A})}$$

Hadamard Product

- For two matrices, $\mathbf{A}$, $\mathbf{B}$, of the same dimension, $m \times n$ the Hadamard product, $\mathbf{A} \circ \mathbf{B}$, is a matrix, of the same dimension as the operands, with elements given by

$$(\mathbf{A} \circ \mathbf{B})_{i,j} = (\mathbf{A})_{i,j} \cdot (\mathbf{B})_{i,j}$$

  - For example the Hadamard product for a 3×3 matrix $\mathbf{A}$ with a 3×3 matrix $\mathbf{B}$ is:

$$\begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \circ \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{21} & B_{22} & B_{23} \\ B_{31} & B_{32} & B_{33} \end{bmatrix} = \begin{bmatrix} A_{11}B_{11} & A_{12}B_{12} & A_{13}B_{13} \\ A_{21}B_{21} & A_{22}B_{22} & A_{23}B_{23} \\ A_{31}B_{31} & A_{32}B_{32} & A_{33}B_{33} \end{bmatrix}$$

Kronecker Product

- If $\mathbf{A}$ is an $m \times n$ matrix and $\mathbf{B}$ is a $p \times q$ matrix, then the Kronecker product $\mathbf{A} \otimes \mathbf{B}$ is the $mp \times nq$ block matrix:

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} A_{11}\mathbf{B} & \cdots & A_{1n}\mathbf{B} \\ \vdots & \ddots & \vdots \\ A_{m1}\mathbf{B} & \cdots & A_{mn}\mathbf{B} \end{bmatrix}$$

  - For example, the Kronecker product for a 2×2 matrix $\mathbf{A}$ with a 2×3 matrix $\mathbf{B}$ is:

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} A_{11}B_{11} & A_{11}B_{12} & A_{11}B_{13} & A_{12}B_{11} & A_{12}B_{12} & A_{12}B_{13} \\ A_{11}B_{21} & A_{11}B_{22} & A_{11}B_{23} & A_{12}B_{21} & A_{12}B_{22} & A_{12}B_{23} \\ A_{21}B_{11} & A_{21}B_{12} & A_{21}B_{13} & A_{22}B_{11} & A_{22}B_{12} & A_{22}B_{13} \\ A_{21}B_{21} & A_{21}B_{22} & A_{21}B_{23} & A_{22}B_{21} & A_{22}B_{22} & A_{22}B_{23} \end{bmatrix}$$

Outline

- Introduction
- Derivatives in Numerator-Layout Notation
  - List of Differentiation
  - Derivative Formulas
  - Chain rule
  - The Matrix Differential
- Derivatives in Denominator-Layout Notation
- Identities

Vector-by-Scalar

- The derivative of $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$, by x is written as:

$$\frac{\partial \mathbf{y}}{\partial \mathrm{x}} \overset{\text{def}}{=} \begin{bmatrix} \frac{\partial y_1}{\partial x} \\ \frac{\partial y_2}{\partial x} \\ \vdots \\ \frac{\partial y_m}{\partial x} \end{bmatrix}$$

Scalar-by-Vector

- The derivative of y by $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$ is written as:

$$\frac{\partial \mathrm{y}}{\partial \mathbf{x}} \stackrel{\text{def}}{=} \begin{bmatrix} \frac{\partial y}{\partial x_1} & \frac{\partial y}{\partial x_2} & \cdots & \frac{\partial y}{\partial x_n} \end{bmatrix}$$

Vector-by-Vector

- The derivative of $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$ with respect to $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$:

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}} \stackrel{\text{def}}{=} \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_1}{\partial x_2} & \cdots & \frac{\partial y_1}{\partial x_n} \\ \frac{\partial y_2}{\partial x_1} & \frac{\partial y_2}{\partial x_2} & & \frac{\partial y_2}{\partial x_n} \\ \vdots & & \ddots & \vdots \\ \frac{\partial y_m}{\partial x_1} & \frac{\partial y_m}{\partial x_2} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix}$$

  - Also known as the Jacobian matrix

Example 1

- Given $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$, $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$, and $y_1 = x_1^2 - 2x_2$, $y_2 = x_3^2 - 4x_2$,

  the Jacobian matrix $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$ is:

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_1}{\partial x_2} & \frac{\partial y_1}{\partial x_3} \\ \frac{\partial y_2}{\partial x_1} & \frac{\partial y_2}{\partial x_2} & \frac{\partial y_2}{\partial x_3} \end{bmatrix} = \begin{bmatrix} 2x_1 & -2 & 0 \\ 0 & -4 & 2x_3 \end{bmatrix}$$

Example 2

- The transformation from spherical to Cartesian coordinates is defined by

$$x = r\sin\theta\cos\phi\,,\ y = r\sin\theta\sin\phi\,,\ z = r\cos\theta$$

➢ Let $\mathbf{y} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$, $\mathbf{x} = \begin{bmatrix} r \\ \theta \\ \phi \end{bmatrix}$, the Jacobian matrix $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$ is:

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial x}{\partial r} & \frac{\partial x}{\partial \theta} & \frac{\partial x}{\partial \phi} \\ \frac{\partial y}{\partial r} & \frac{\partial y}{\partial \theta} & \frac{\partial y}{\partial \phi} \\ \frac{\partial z}{\partial r} & \frac{\partial z}{\partial \theta} & \frac{\partial z}{\partial \phi} \end{bmatrix} = \begin{bmatrix} \sin\theta\cos\phi & r\cos\theta\cos\phi & -r\sin\theta\sin\phi \\ \sin\theta\sin\phi & r\cos\theta\sin\phi & r\sin\theta\cos\phi \\ \cos\theta & -r\sin\theta & 0 \end{bmatrix}$$

Matrix-by-Scalar

- The derivative of a matrix function $\mathbf{Y}$ by a scalar x is known as the tangent matrix and is given by

$$\frac{\partial \mathbf{Y}}{\partial x} \stackrel{\text{def}}{=} \begin{bmatrix} \frac{\partial Y_{11}}{\partial x} & \frac{\partial Y_{12}}{\partial x} & \cdots & \frac{\partial Y_{1n}}{\partial x} \\ \frac{\partial Y_{21}}{\partial x} & \frac{\partial Y_{22}}{\partial x} & & \frac{\partial Y_{2n}}{\partial x} \\ \vdots & & \ddots & \vdots \\ \frac{\partial Y_{m1}}{\partial x} & \frac{\partial Y_{m2}}{\partial x} & \cdots & \frac{\partial Y_{mn}}{\partial x} \end{bmatrix}$$

Scalar-by-Matrix

- The derivative of a scalar y function by a matrix $\mathbf{X}$ is known as the gradient matrix and is given by

$$\frac{\partial y}{\partial \mathbf{X}} \stackrel{\text{def}}{=} \begin{bmatrix} \frac{\partial y}{\partial X_{11}} & \frac{\partial y}{\partial X_{21}} & \cdots & \frac{\partial y}{\partial X_{m1}} \\ \frac{\partial y}{\partial X_{12}} & \frac{\partial y}{\partial X_{22}} & & \frac{\partial y}{\partial X_{m2}} \\ \vdots & & \ddots & \vdots \\ \frac{\partial y}{\partial X_{1n}} & \frac{\partial y}{\partial X_{2n}} & \cdots & \frac{\partial y}{\partial X_{mn}} \end{bmatrix}$$

List of Differentiation

- Result of differentiating various kinds of aggregates with other kinds of aggregates.

	Scalar y		Vector $\mathbf{y}$ (size m)		Matrix $\mathbf{Y}$ (size $m \times n$)	
	Notation	Type	Notation	Type	Notation	Type
Scalar x	$\frac{\partial y}{\partial x}$	scalar	$\frac{\partial \mathbf{y}}{\partial x}$	size-m column vector	$\frac{\partial \mathbf{Y}}{\partial x}$	$m \times n$ matrix
Vector $\mathbf{x}$ (size n)	$\frac{\partial y}{\partial \mathbf{x}}$	size-n row vector	$\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$	$m \times n$ matrix	$\frac{\partial \mathbf{Y}}{\partial \mathbf{x}}$	–
Matrix $\mathbf{X}$ (size $p \times q$)	$\frac{\partial y}{\partial \mathbf{X}}$	$q \times p$ matrix	$\frac{\partial \mathbf{y}}{\partial \mathbf{X}}$	–	$\frac{\partial \mathbf{Y}}{\partial \mathbf{X}}$	–

Derivative Formulas

$\mathbf{y}$	$\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$
$\mathbf{Ax}$	$\mathbf{A}$
$\mathbf{x}^T\mathbf{A}$	$\mathbf{A}^T$
$\mathbf{x}^T\mathbf{x}$	$2\mathbf{x}^T$
$\mathbf{x}^T\mathbf{Ax}$	$\mathbf{x}^T\mathbf{A} + \mathbf{x}^T\mathbf{A}^T$

- Hint: Derive $\mathbf{x}$
  - If you have to differentiate $\mathbf{x}^T$, transpose the rest.
  - If you have two $\mathbf{x}$-terms, differentiate them separately in turn and then sum up the two derivatives.

Chain Rule (1/2)

- Let $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$, $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$, and $\mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_r \end{bmatrix}$, where $\mathbf{z}$ is a function of $\mathbf{y}$, which is in turn a function of $\mathbf{x}$. Then

$$\frac{\partial \mathbf{z}}{\partial \mathbf{x}} \overset{\text{def}}{=} \begin{bmatrix} \frac{\partial z_1}{\partial x_1} & \frac{\partial z_1}{\partial x_2} & \cdots & \frac{\partial z_1}{\partial x_n} \\ \frac{\partial z_2}{\partial x_1} & \frac{\partial z_2}{\partial x_2} & & \frac{\partial z_2}{\partial x_n} \\ \vdots & & \ddots & \vdots \\ \frac{\partial z_r}{\partial x_1} & \frac{\partial z_r}{\partial x_2} & \cdots & \frac{\partial z_r}{\partial x_n} \end{bmatrix}, \text{ where } \frac{\partial z_i}{\partial x_j} = \sum_{k=1}^{m} \frac{\partial z_i}{\partial y_k} \frac{\partial y_k}{\partial x_j} \quad \begin{cases} i = 1,2,\cdots,r \\ j = 1,2,\cdots,n \end{cases}$$

Chain Rule (2/2)

$$\frac{\partial \mathbf{z}}{\partial \mathbf{x}} = \begin{bmatrix} \sum \frac{\partial z_1}{\partial y_k}\frac{\partial y_k}{\partial x_1} & \sum \frac{\partial z_1}{\partial y_k}\frac{\partial y_k}{\partial x_2} & \cdots & \sum \frac{\partial z_1}{\partial y_k}\frac{\partial y_k}{\partial x_n} \\ \sum \frac{\partial z_2}{\partial y_k}\frac{\partial y_k}{\partial x_1} & \sum \frac{\partial z_2}{\partial y_k}\frac{\partial y_k}{\partial x_2} & & \sum \frac{\partial z_2}{\partial y_k}\frac{\partial y_k}{\partial x_n} \\ \vdots & & \ddots & \vdots \\ \sum \frac{\partial z_r}{\partial y_k}\frac{\partial y_k}{\partial x_1} & \sum \frac{\partial z_r}{\partial y_k}\frac{\partial y_k}{\partial x_2} & \cdots & \sum \frac{\partial z_r}{\partial y_k}\frac{\partial y_k}{\partial x_n} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{\partial z_1}{\partial y_1} & \frac{\partial z_1}{\partial y_2} & \cdots & \frac{\partial z_1}{\partial y_m} \\ \frac{\partial z_2}{\partial y_1} & \frac{\partial z_2}{\partial y_2} & & \frac{\partial z_2}{\partial y_m} \\ \vdots & & \ddots & \vdots \\ \frac{\partial z_r}{\partial y_1} & \frac{\partial z_r}{\partial y_2} & \cdots & \frac{\partial z_r}{\partial y_m} \end{bmatrix} \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_1}{\partial x_2} & \cdots & \frac{\partial y_1}{\partial x_n} \\ \frac{\partial y_2}{\partial x_1} & \frac{\partial y_2}{\partial x_2} & & \frac{\partial y_2}{\partial x_n} \\ \vdots & & \ddots & \vdots \\ \frac{\partial y_m}{\partial x_1} & \frac{\partial y_m}{\partial x_2} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix} = \frac{\partial \mathbf{z}}{\partial \mathbf{y}}\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$$

Exercise 1 (Numerator-Layout)

- Find $\mathbf{w}^*$ to minimize $E(\mathbf{w})$, where

$$E(\mathbf{w}) = \frac{1}{N}\|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2 = \frac{1}{N}(\mathbf{X}\mathbf{w} - \mathbf{y})^T(\mathbf{X}\mathbf{w} - \mathbf{y})$$
$$= \frac{1}{N}(\mathbf{w}^T\mathbf{X}^T\mathbf{X}\mathbf{w} - 2\mathbf{w}^T\mathbf{X}^T\mathbf{y} + \mathbf{y}^T\mathbf{y})$$

  - Assume $\mathbf{X}^T\mathbf{X}$ is invertible.

$$\mathbf{w}^* = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

The Matrix Differential (1/2)

- For a scalar function $f(\mathbf{x})$, where $\mathbf{x}$ is an n-vector, the ordinary differential of multivariate calculus is defined as

$$df = \sum_{i=1}^{n} \frac{\partial f}{\partial x_i} dx_i$$

- In harmony with this formula, we define the differential of an $m \times n$ matrix $\mathbf{X} = [X_{ij}]$ to be

$$d\mathbf{X} \stackrel{\text{def}}{=} \begin{bmatrix} dX_{11} & dX_{12} & \cdots & dX_{1n} \\ dX_{21} & dX_{22} & & dX_{2n} \\ \vdots & & \ddots & \vdots \\ dX_{m1} & dX_{m2} & \cdots & dX_{mn} \end{bmatrix}$$

The Matrix Differential (2/2)

- This definition complies with the multiplicative and associative rules

$$d(\alpha\mathbf{X}) = \alpha d\mathbf{X} \qquad d(\mathbf{X}+\mathbf{Y}) = d\mathbf{X} + d\mathbf{Y}$$

- If $\mathbf{X}$ and $\mathbf{Y}$ are product-conforming matrices, it can be verified that the differential of their product is

$$d(\mathbf{XY}) = (d\mathbf{X})\mathbf{Y} + \mathbf{X}(d\mathbf{Y})$$

Summary of Numerator-Layout

- Three straightforward key points:

1. Ice cream
   - Derivatives
2. Derive $\mathbf{x}$
   - Derivative Formulas
3. Chain rule
   - From left to right

Vector-by-Vector

- Identities: vector-by-vector $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$

Condition	Expression	Numerator layout	Denominator layout
$a = a(\mathbf{x}),$ $\mathbf{u} = \mathbf{u}(\mathbf{x})$	$\frac{\partial a\mathbf{u}}{\partial \mathbf{x}} =$	$a\frac{\partial \mathbf{u}}{\partial \mathbf{x}} + \mathbf{u}\frac{\partial a}{\partial \mathbf{x}}$	$a\frac{\partial \mathbf{u}}{\partial \mathbf{x}} + \frac{\partial a}{\partial \mathbf{x}}\mathbf{u}^T$
$\mathbf{u} = \mathbf{u}(\mathbf{x}),$ $\mathbf{v} = \mathbf{v}(\mathbf{x})$	$\frac{\partial(\mathbf{u}+\mathbf{v})}{\partial \mathbf{x}} =$	$\frac{\partial \mathbf{u}}{\partial \mathbf{x}} + \frac{\partial \mathbf{v}}{\partial \mathbf{x}}$	
$\mathbf{u} = \mathbf{u}(\mathbf{x})$	$\frac{\partial \mathbf{f}(\mathbf{g}(\mathbf{u}))}{\partial \mathbf{x}} =$	$\frac{\partial \mathbf{f}(\mathbf{g})}{\partial \mathbf{g}}\frac{\partial \mathbf{g}(\mathbf{u})}{\partial \mathbf{u}}\frac{\partial \mathbf{u}}{\partial \mathbf{x}}$	$\frac{\partial \mathbf{u}}{\partial \mathbf{x}}\frac{\partial \mathbf{g}(\mathbf{u})}{\partial \mathbf{u}}\frac{\partial \mathbf{f}(\mathbf{g})}{\partial \mathbf{g}}$

Scalar-by-Vector

- Identities: scalar-by-vector $\frac{\partial y}{\partial \mathbf{x}}$

Condition	Expression	Numerator layout	Denominator layout
$u = u(\mathbf{x})$	$\frac{\partial f(g(u))}{\partial \mathbf{x}} =$	$\frac{\partial f(g)}{\partial g}\frac{\partial g(u)}{\partial u}\frac{\partial u}{\partial \mathbf{x}}$	
$\boldsymbol{u} = \mathbf{u}(\mathbf{x}),$ $\mathbf{v} = \mathbf{v}(\mathbf{x})$	$\frac{\partial \mathbf{u}^T\mathbf{v}}{\partial \mathbf{x}} =$	$\mathbf{u}^T\frac{\partial \mathbf{v}}{\partial \mathbf{x}} + \mathbf{v}^T\frac{\partial \mathbf{u}}{\partial \mathbf{x}}$	$\frac{\partial \mathbf{u}}{\partial \mathbf{x}}\mathbf{v} + \frac{\partial \mathbf{v}}{\partial \mathbf{x}}\mathbf{u}$
$\mathbf{A} \neq \mathbf{A}(\mathbf{x}),$ $\mathbf{u} = \mathbf{u}(\mathbf{x}),$ $\mathbf{v} = \mathbf{v}(\mathbf{x})$	$\frac{\partial \mathbf{u}^T\mathbf{A}\mathbf{v}}{\partial \mathbf{x}} =$	$\mathbf{u}^T\mathbf{A}\frac{\partial \mathbf{v}}{\partial \mathbf{x}} + \mathbf{v}^T\mathbf{A}^T\frac{\partial \mathbf{u}}{\partial \mathbf{x}}$	$\frac{\partial \mathbf{u}}{\partial \mathbf{x}}\mathbf{A}\mathbf{v} + \frac{\partial \mathbf{v}}{\partial \mathbf{x}}\mathbf{A}^T\mathbf{u}$
$\mathbf{a} \neq \mathbf{a}(\mathbf{x}),$ $\mathbf{u} = \mathbf{u}(\mathbf{x})$	$\frac{\partial \mathbf{a}^T\mathbf{u}}{\partial \mathbf{x}} =$	$\mathbf{a}^T\frac{\partial \mathbf{u}}{\partial \mathbf{x}}$	$\frac{\partial \mathbf{u}}{\partial \mathbf{x}}\mathbf{a}$

Vector-by-Scalar

- Identities: vector-by-scalar $\frac{\partial \mathbf{y}}{\partial x}$

Condition	Expression	Numerator layout	Denominator layout
$\mathbf{u} = \mathbf{u}(x)$, $\mathbf{v} = \mathbf{v}(x)$	$\frac{\partial(\mathbf{u}+\mathbf{v})}{\partial x} =$	$\frac{\partial \mathbf{u}}{\partial x} + \frac{\partial \mathbf{v}}{\partial x}$	
$\mathbf{u} = \mathbf{u}(x)$, $\mathbf{v} = \mathbf{v}(x)$	$\frac{\partial(\mathbf{u}\times\mathbf{v})}{\partial x} =$	$\mathbf{u} \times \frac{\partial \mathbf{v}}{\partial x} + \frac{\partial \mathbf{u}}{\partial x} \times \mathbf{v}$	
$\mathbf{u} = \mathbf{u}(x)$	$\frac{\partial \mathbf{f}(\mathbf{g}(\mathbf{u}))}{\partial x} =$	$\frac{\partial \mathbf{f}(\mathbf{g})}{\partial \mathbf{g}} \frac{\partial \mathbf{g}(\mathbf{u})}{\partial \mathbf{u}} \frac{\partial \mathbf{u}}{\partial x}$	$\frac{\partial \mathbf{u}}{\partial x} \frac{\partial \mathbf{g}(\mathbf{u})}{\partial \mathbf{u}} \frac{\partial \mathbf{f}(\mathbf{g})}{\partial \mathbf{g}}$

Scalar-by-Matrix

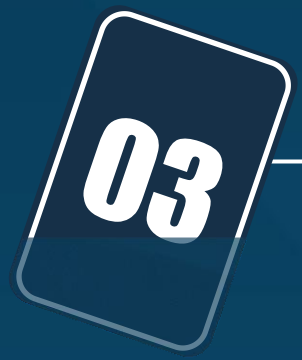
- Identities: scalar-by-matrix $\frac{\partial y}{\partial \mathbf{X}}$

Condition	Expression	Numerator layout	Denominator layout
$u = u(\mathbf{X}),$ $v = v(\mathbf{X})$	$\frac{\partial (u+v)}{\partial \mathbf{X}} =$	$\frac{\partial u}{\partial \mathbf{X}} + \frac{\partial v}{\partial \mathbf{X}}$	
$u = u(\mathbf{X}),$ $v = v(\mathbf{X})$	$\frac{\partial uv}{\partial \mathbf{X}}$	$u\frac{\partial v}{\partial \mathbf{X}} + v\frac{\partial u}{\partial \mathbf{X}}$	
$u = u(\mathbf{X})$	$\frac{\partial f(g(u))}{\partial \mathbf{X}} =$	$\frac{\partial f(g)}{\partial g}\frac{\partial g(u)}{\partial u}\frac{\partial u}{\partial \mathbf{X}}$	

Matrix-by-Scalar

- Identities: scalar-by-matrix $\frac{\partial \mathbf{Y}}{\partial x}$

Condition	Expression	Numerator layout
$\mathbf{U} = \mathbf{U}(x),$ $\mathbf{V} = \mathbf{V}(x)$	$\frac{\partial(\mathbf{U}+\mathbf{V})}{\partial x} =$	$\frac{\partial \mathbf{U}}{\partial x} + \frac{\partial \mathbf{V}}{\partial x}$
$\mathbf{U} = \mathbf{U}(x),$ $\mathbf{V} = \mathbf{V}(x)$	$\frac{\partial(\mathbf{U}\mathbf{V})}{\partial x} =$	$\mathbf{U}\frac{\partial \mathbf{V}}{\partial x} + \frac{\partial \mathbf{U}}{\partial x}\mathbf{V}$
$\mathbf{U} = \mathbf{U}(x),$ $\mathbf{V} = \mathbf{V}(x)$	$\frac{\partial(\mathbf{U}\circ\mathbf{V})}{\partial x} =$	$\mathbf{U}\circ\frac{\partial \mathbf{V}}{\partial x} + \frac{\partial \mathbf{U}}{\partial x}\circ\mathbf{V}$
$\mathbf{U} = \mathbf{U}(x),$ $\mathbf{V} = \mathbf{V}(x)$	$\frac{\partial(\mathbf{U}\otimes\mathbf{V})}{\partial x} =$	$\mathbf{U}\otimes\frac{\partial \mathbf{V}}{\partial x} + \frac{\partial \mathbf{U}}{\partial x}\otimes\mathbf{V}$



Optional: Probability Theory

Bayes Theorem

Sample space and Events

- Ω : Sample Space, result of an experiment
  - If you toss a coin twice $\Omega = \{\mathrm{HH},\mathrm{HT},\mathrm{TH},\mathrm{TT}\}$
- Event: a subset of Ω
  - First toss is head = {HH,HT}
- S: event space, a set of events:
  - Closed under finite union and complements
    - Entails other binary operation: union, diff, etc.
  - Contains the empty event and Ω

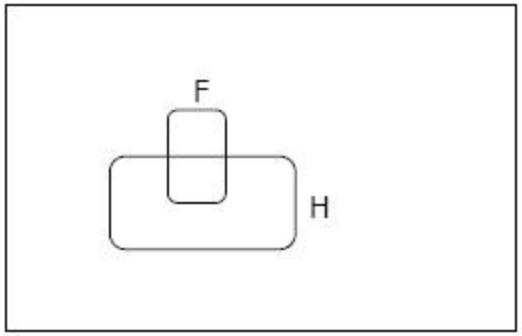
Probability Measure

- Defined over (Ω, S) s.t.
  - $P(\alpha) >= 0$ for all α in S
  - $P(\Omega) = 1$
  - If α, β are disjoint, then
    - $P(\alpha \cup \beta) = p(\alpha) + p(\beta)$
- We can deduce other axioms from the above ones
  - Ex: $P(\alpha \cup \beta)$ for non-disjoint event

    $P(\alpha \cup \beta) = p(\alpha) + p(\beta) - p(\alpha \cap \beta)$

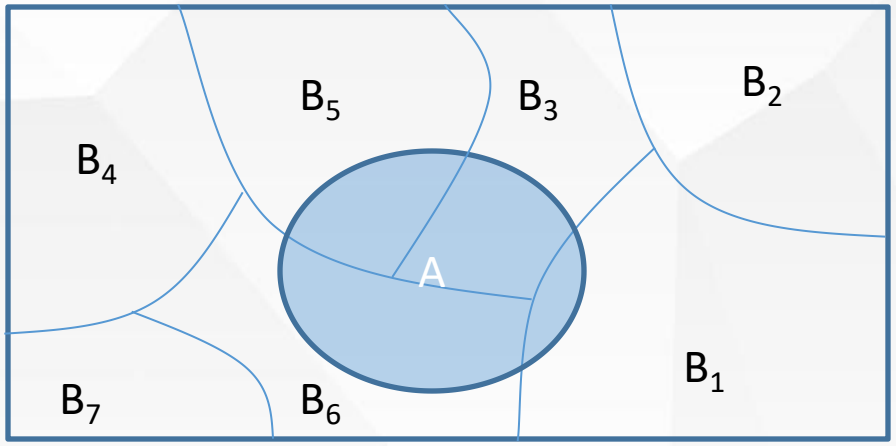
Conditional Probability

P(F|H) = Fraction of worlds in which H is true that also have F true



$$p(f \mid h) = \frac{p(F \cap H)}{p(H)}$$

Rule of total probability



$$p(A) = \sum P(B_i) P(A \mid B_i)$$

From Events to Random Variable

- Almost all the semester we will be dealing with RV
- Concise way of specifying attributes of outcomes
- Modeling students (Grade and Intelligence):
  - $\Omega =$ all possible students
  - What are events
    - Grade_A = all students with grade A
    - Grade_B = all students with grade B
    - Intelligence_High = … with high intelligence
  - Very cumbersome
  - We need "functions" that maps from Ω to an attribute space.
  - P(G = A) = P({student ϵ Ω : G(student) = A})

Discrete Random Variables

- Random variables (RVs) which may take on only a **countable** number of **distinct** values
  - E.g. the total number of tails X you get if you flip 100 coins
- X is a RV with arity *k* if it can take on exactly one value out of $\{x_1, \ldots, x_k\}$
  - E.g. the possible values that X can take on are 0, 1, 2, …, 100

Probability of Discrete RV

- Probability mass function (pmf): $P(X = x_i)$
- Easy facts about pmf
  - $\Sigma_i P(X = x_i) = 1$
  - $P(X = x_i \cap X = x_j) = 0$ if $i \neq j$
  - $P(X = x_i \cup X = x_j) = P(X = x_i) + P(X = x_j)$ if $i \neq j$
  - $P(X = x_1 \cup X = x_2 \cup \ldots \cup X = x_k) = 1$

Continuous Random Variables

- Probability density function (pdf) instead of probability mass function (pmf)

- A pdf is any function $f(x)$ that describes the probability density in terms of the input variable x.

Probability of Continuous RV

- Properties of pdf
  - $f(x) \geq 0, \forall x$
  - $\int_{-\infty}^{+\infty} f(x) = 1$
- Actual probability can be obtained by taking the integral of pdf
  - E.g. the probability of X being between 0 and 1 is

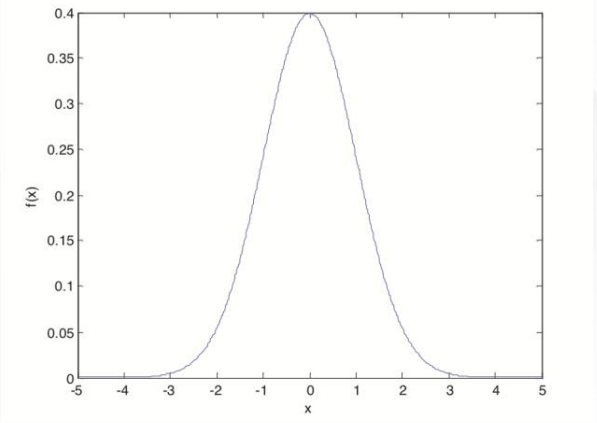
$$P(0 \leq X \leq 1) = \int_0^1 f(x)dx$$

Cumulative Distribution Function

- $F_X(v) = P(X \leq v)$
- Discrete RVs
  - $F_X(v) = \Sigma_{vi}\, P(X = v_i)$
- Continuous RVs
  - $F_X(v) = \int_{-\infty}^{v} f(x)dx$
  - $\frac{d}{dx} F_x(x) = f(x)$

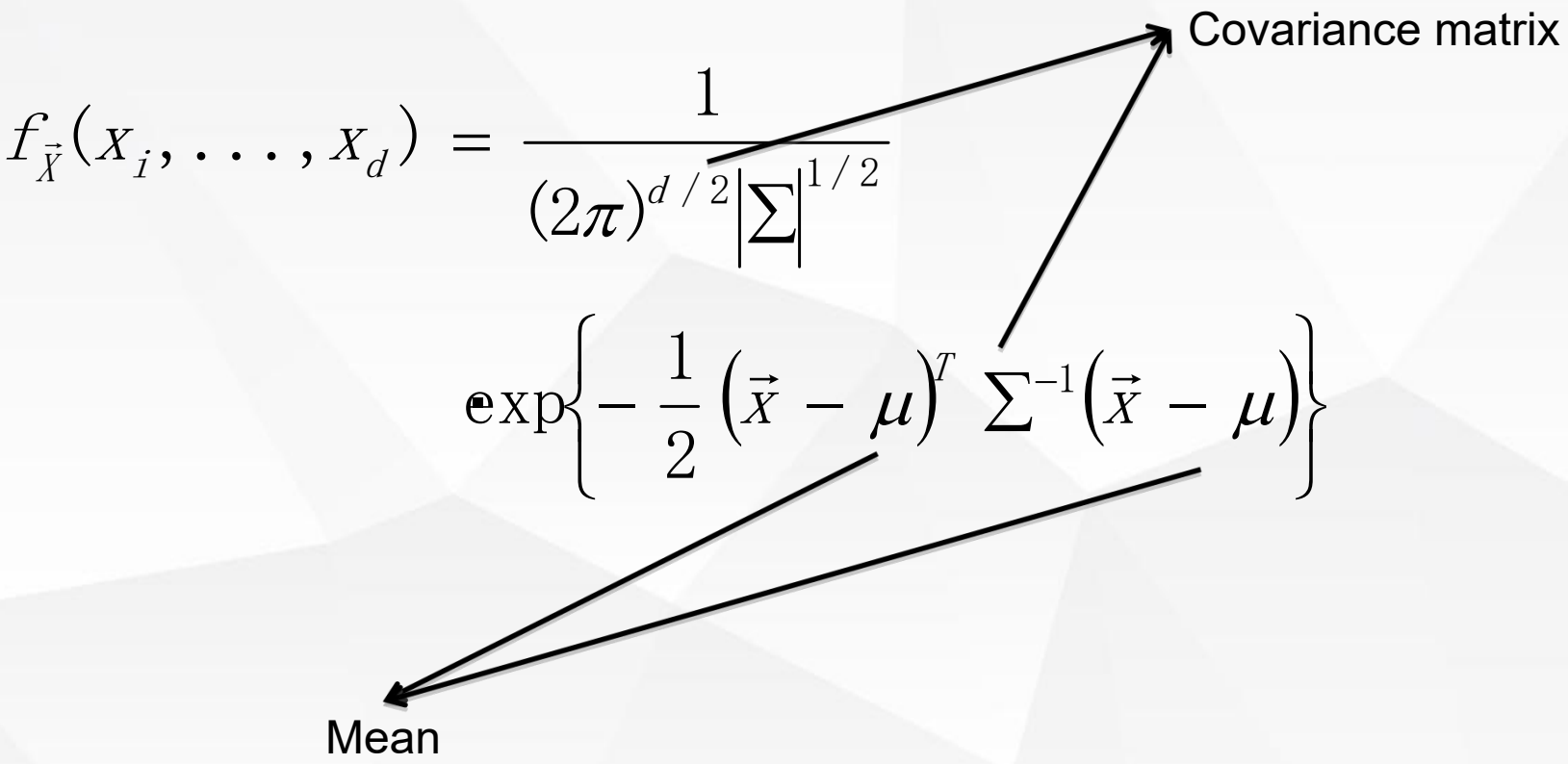
Common Distributions

- Normal X $N(\mu, \sigma^2)$
  - $f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$
  - E.g. the height of the entire population



Multivariate Normal

- Generalization to higher dimensions of the one-dimensional normal



Joint Probability Distribution

- Random variables encodes attributes
- Not all possible combination of attributes are equally likely
  - Joint probability distributions quantify this
- P(X= x, Y= y) = P(x, y)
  - Generalizes to N-RVs
  - $\sum_x \sum_y P(X = x, Y = y) = 1$
  - $\int_x \int_y f_{X,Y}(x, y)\,dx\,dy = 1$

Chain Rule

- Always true
  - $P(x, y, z) = p(x)\, p(y|x)\, p(z|x, y)$

    $= p(z)\, p(y|z)\, p(x|y, z)$

    $= \ldots$

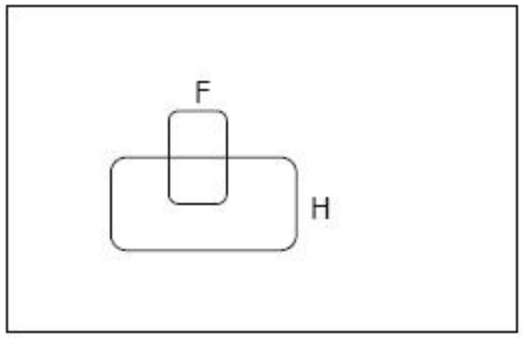
Conditional Probability

$$P(X = x \mid Y = y) = \frac{P(X = x \cap Y = y)}{P(Y = y)}$$

events

But we will always write it this way:

$$P(x \mid y) = \frac{p(x, y)}{p(y)}$$

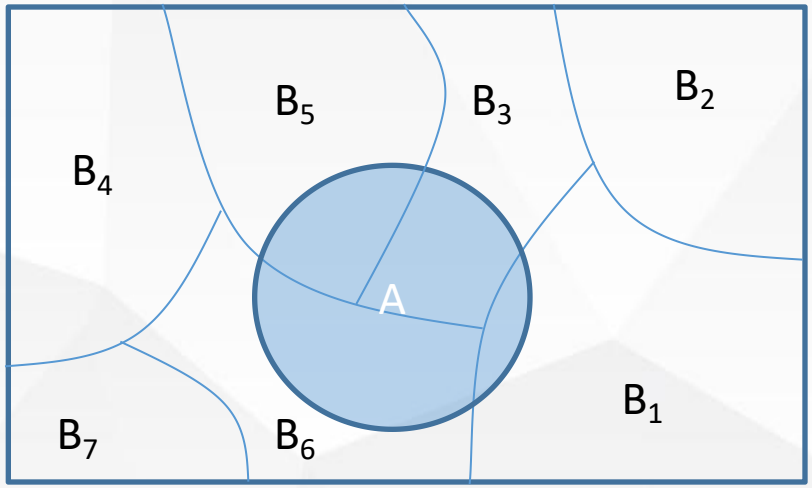


Marginalization

- We know p(X, Y), what is P(X=x)?
- We can use the low of total probability, why?

$$p(x) = \sum_y P(x, y)$$

$$= \sum_y P(y) P(x \mid y)$$



Marginalization Cont.

- Another example

$$p(x) = \sum_{y,z} P(x, y, z)$$

$$= \sum_{z,y} P(y, z) P(x \mid y, z)$$

Bayes Rule

- We know that P(rain) = 0.5
  - If we also know that the grass is wet, then how this affects our belief about whether it rains or not?

$$P(rain \mid wet) = \frac{P(rain)P(wet \mid rain)}{P(wet)}$$

$$P(x \mid y) = \frac{P(x)P(y \mid x)}{P(y)}$$

Bayes Rule cont.

- You can condition on more variables

$$P(x \mid y,z) = \frac{P(x \mid z)P(y \mid x,z)}{P(y \mid z)}$$

Independence

- X is independent of Y means that knowing Y does not change our belief about X.
  - $P(X|Y=y) = P(X)$
  - $P(X=x, Y=y) = P(X=x)\ P(Y=y)$
  - The above should hold for all x, y
  - It is symmetric and written as $X \perp Y$

Independence

- $X_1, \ldots, X_n$ are independent if and only if

$$P(X_1 \in A_1, \ldots, X_n \in A_n) = \prod_{i=1}^{n} P(X_i \in A_i)$$

- If $X_1, \ldots, X_n$ are independent and identically distributed we say they are *iid* (or that they are a random sample) and we write

$$X_1, \ldots, X_n \sim P$$

CI: Conditional Independence

- RV are rarely independent but we can still leverage local structural properties like Conditional Independence.
- $X \perp Y \mid Z$ if once Z is observed, knowing the value of Y does not change our belief about X
  - P(rain $\perp$ sprinkler's on | cloudy)
  - P(rain $\not\perp$ sprinkler's on | wet grass)

Conditional Independence

- $P(X=x \mid Z=z, Y=y) = P(X=x \mid Z=z)$
- $P(Y=y \mid Z=z, X=x) = P(Y=y \mid Z=z)$
- $P(X=x, Y=y \mid Z=z) = P(X=x \mid Z=z)\, P(Y=y \mid Z=z)$

We call these factors : very useful concept !!

Mean and Variance

- Mean (Expectation): $\mu = E(\mathrm{X})$
  - Discrete RVs: $E(\mathrm{X}) = \sum_{v_i} v_i \mathrm{P}(\mathrm{X} = v_i)$

$$E(g(X)) = \sum_{v_i} g(v_i) P(X = v_i)$$

  - Continuous RVs: $E(\mathrm{X}) = \int_{-\infty}^{+\infty} x f(x)\, dx$

$$E(g(X)) = \int_{-\infty}^{+\infty} g(x) f(x) dx$$

Mean and Variance

- Variance: $Var(X) = E((X-\mu)^2)$

$$Var(X) = E(X^2) - \mu^2$$

  - Discrete RVs: $V(\mathrm{X}) = \sum_{v_i} (v_i - \mu)^2 \, \mathrm{P}(\mathrm{X} = v_i)$
  - Continuous RVs: $V(\mathrm{X}) = \int_{-\infty}^{+\infty} (x-\mu)^2 \, f(x) dx$

- Covariance:

$$Cov(X,Y) = E((X-\mu_x)(Y-\mu_y)) = E(XY) - \mu_x \mu_y$$

Mean and Variance

- Correlation:

$$\rho(X,Y) = Cov(X,Y)/\sigma_x\sigma_y$$

$$-1 \le \rho(X,Y) \le 1$$

Properties

- Mean
  - $E(\mathrm{X}+\mathrm{Y})=E(\mathrm{X})+E(\mathrm{Y})$
  - $E(a\mathrm{X})=aE(\mathrm{X})$
  - If X and Y are independent, $E(\mathrm{XY})=E(\mathrm{X})\cdot E(\mathrm{Y})$
- Variance
  - $V(a\mathrm{X}+b)=a^2V(\mathrm{X})$
  - If X and Y are independent, $V(\mathrm{X}+\mathrm{Y})=V(\mathrm{X})+V(\mathrm{Y})$

Some more properties

- The conditional expectation of Y given X when the value of X = x is:

$$E(Y \mid X = x) = \int y * p(y \mid x) dy$$

- The Law of Total Expectation or Law of Iterated Expectation:

$$E(Y) = E\big[E(Y \mid X)\big] = \int E(Y \mid X = x)\, p_X(x) dx$$

Some more properties

- The law of Total Variance:

$$Var(Y) = Var\left[E(Y \mid X)\right] + E\left[Var(Y \mid X)\right]$$

04

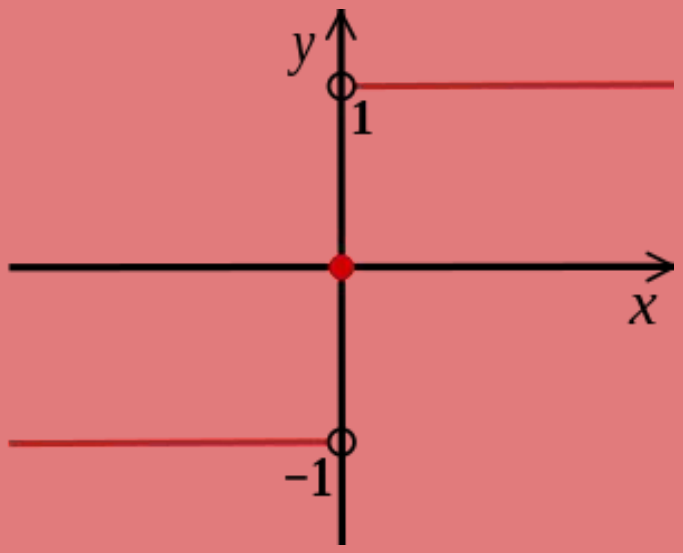
Logistic Regression

A basic supervised learning model

Using gradient ascent for linear classifiers

This decision function isn't differentiable:

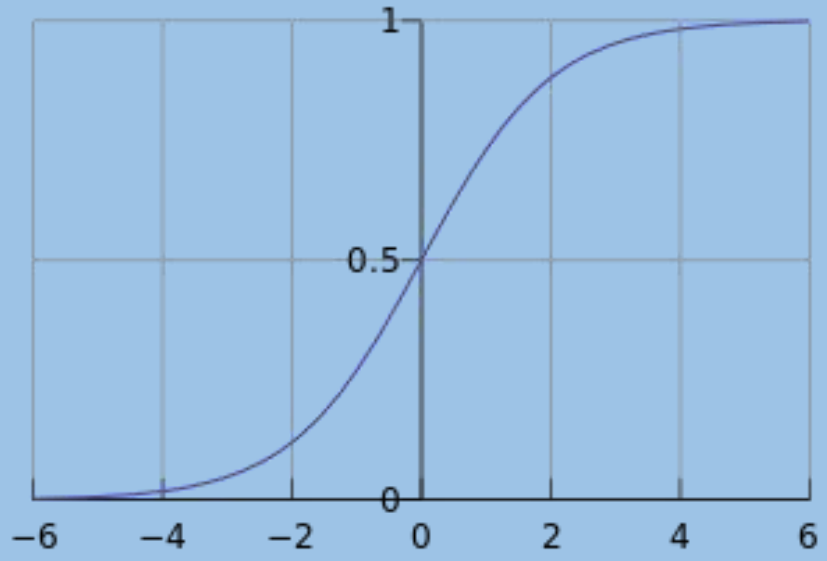
$$h(\mathbf{x}) = \text{sign}(\boldsymbol{\theta}^T \mathbf{x})$$



sign(x)

Use a differentiable function instead:

$$p_{\boldsymbol{\theta}}(y = 1|\mathbf{x}) = \frac{1}{1 + \exp(-\boldsymbol{\theta}^T \mathbf{x})}$$

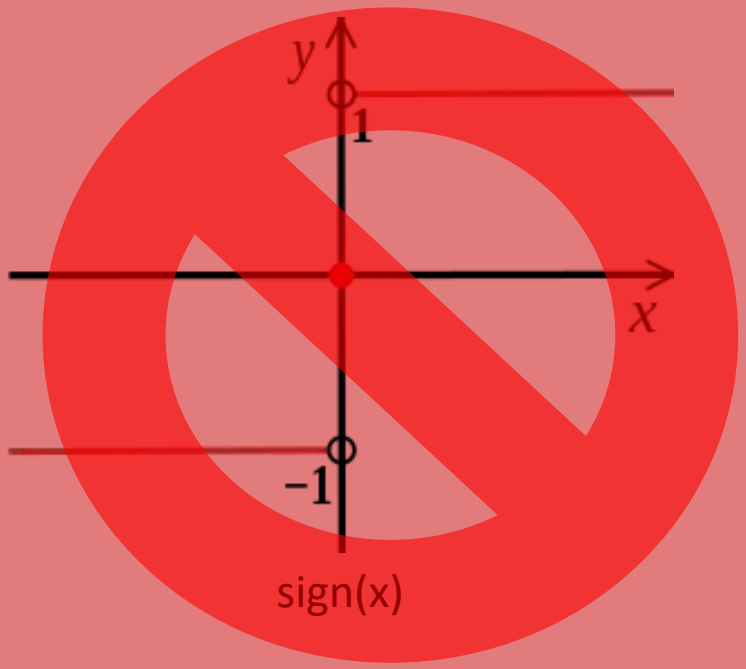


$$\text{logistic}(u) \equiv \frac{1}{1 + e^{-u}}$$

Using gradient ascent for linear classifiers

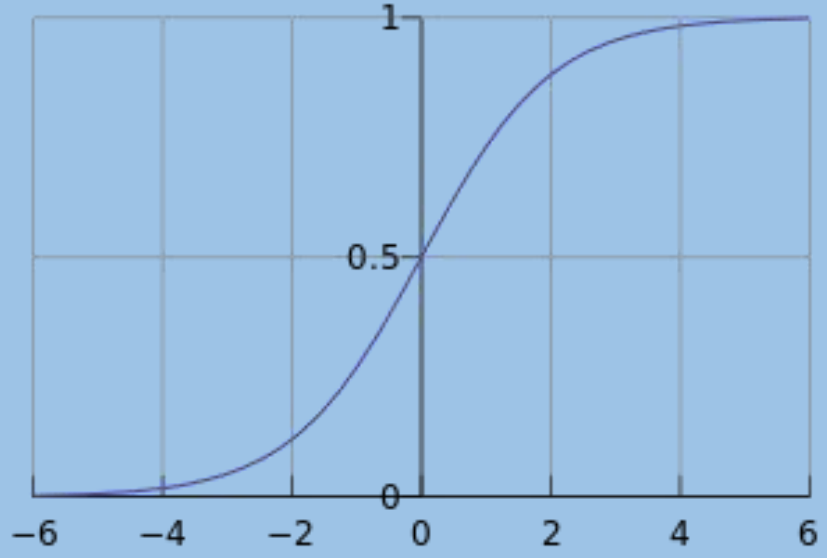
This decision function isn't differentiable:

$$h(\mathbf{x}) = \text{sign}(\boldsymbol{\theta}^T \mathbf{x})$$



Use a differentiable function instead:

$$p_{\boldsymbol{\theta}}(y = 1|\mathbf{x}) = \frac{1}{1 + \exp(-\boldsymbol{\theta}^T \mathbf{x})}$$



$$\text{logistic}(u) \equiv \frac{1}{1 + e^{-u}}$$

Logistic Regression

Data: Inputs are continuous vectors of length K. Outputs are discrete.

$$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^{N} \text{ where } \mathbf{x} \in \mathbb{R}^K \text{ and } y \in \{0, 1\}$$

Model: Logistic function applied to dot product of parameters with input vector.

$$p_{\boldsymbol{\theta}}(y = 1|\mathbf{x}) = \frac{1}{1 + \exp(-\boldsymbol{\theta}^T \mathbf{x})}$$

Learning: finds the parameters that minimize some objective function.

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}}\, J(\boldsymbol{\theta})$$

Prediction: Output is the most probable class.

$$\hat{y} = \underset{y \in \{0,1\}}{\operatorname{argmax}}\, p_{\boldsymbol{\theta}}(y|\mathbf{x})$$

A Recipe for Machine Learning

1. Given training data:

$$\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{N}$$

Face *Face* *Not a face*

2. Choose each of these:
   - Decision function

$$\hat{\boldsymbol{y}} = f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$$

   - Loss function

$$\ell(\hat{\boldsymbol{y}}, \boldsymbol{y}_i) \in \mathbb{R}$$

Examples: Linear regression, Logistic regression, Neural Network

Examples: Mean-squared error, Cross Entropy

The loss function

A loss function or cost function is a function that maps an event (mis-classification) onto a real value that corresponds to cost of such events.

In the case of a classifier, it inferred a cost for every wrong classification. Thus by minimizing the loss function, we are improving the model toward the direction of making fewer and fewer errors.

Loss functions that are differentiable can be minimized by gradient descend.

The choice of loss function is decided by the choice of models.

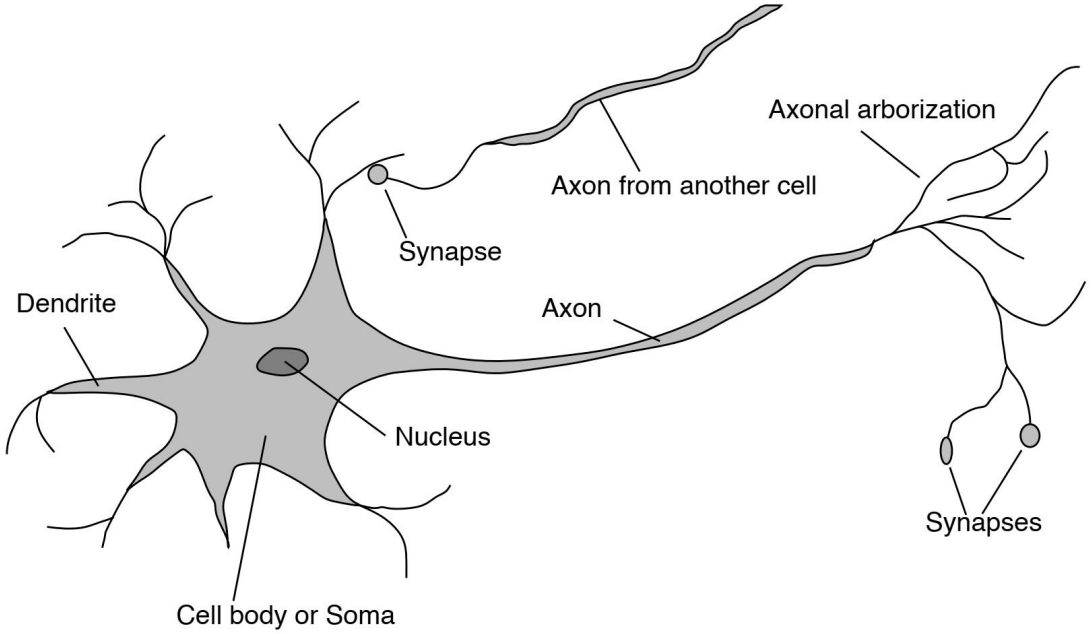
Lecture 3
CONTENTS

Neural Networks

Single Neuron Perceptron and MLP

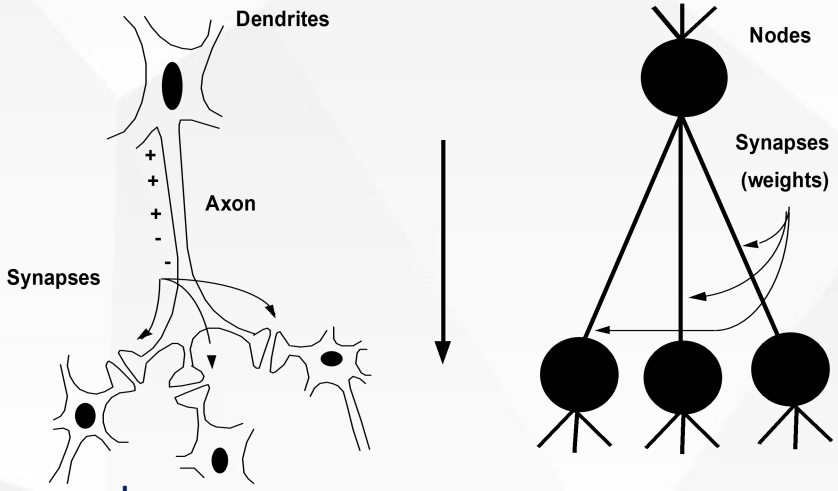
Brains

10^{11} neurons of > 20 types, 10^{14} synapses, 1ms–10ms cycle time
Signals are noisy "spike trains" of electrical potential



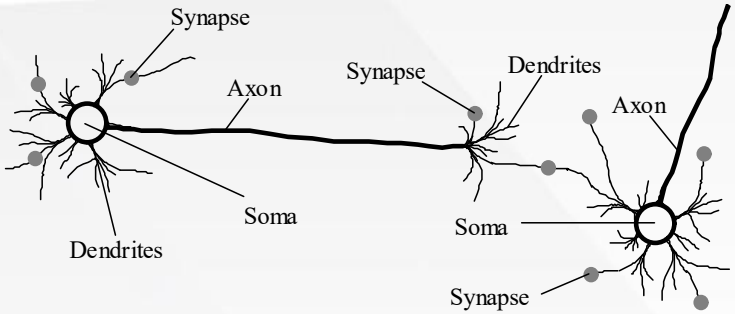
Background: Connectionist Models

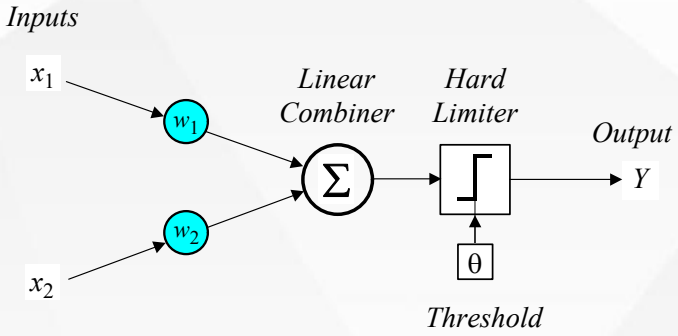
- Consider humans:
  - Neuron switching time
    ~ 0.001 second
  - Number of neurons
    ~ 10^{10}
  - Connections per neuron
    ~ 10^{4-5}
  - Scene recognition time
    ~ 0.1 second
  - 100 inference steps doesn't seem like enough
    → much parallel computation
- Properties of artificial neural nets (ANN)
  - Many neuron-like threshold switching units
  - Many weighted interconnections among units
  - Highly parallel, distributed processes



Overview: Perceptron and Neural Nets

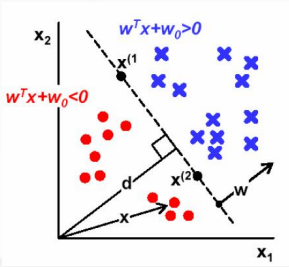
- From biological neuron to artificial neuron (perceptron)



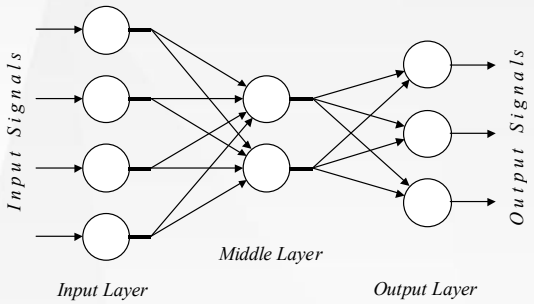


- Activation function

$$H = \sum_{i=1}^{n} X_i W_i \qquad Y = \begin{cases} +1, \text{ if } H \geq \omega_0 \\ -1, \text{ if } H < \omega_0 \end{cases}$$

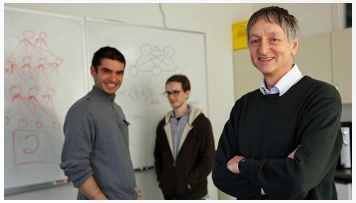


- Artificial neuron networks
  - supervised learning
  - gradient descent



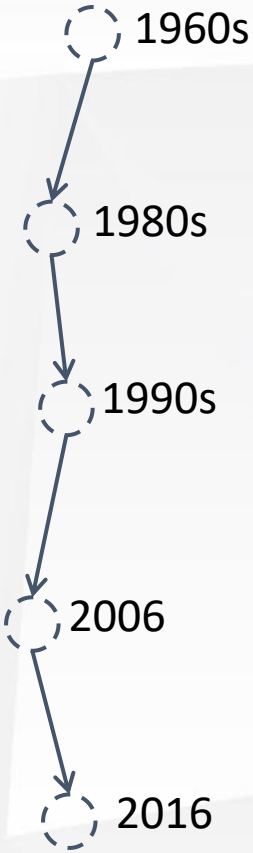
Why is everyone talking about Deep Learning?

- Because a lot of money is invested in it…
  - DeepMind: Acquired by Google for **$400 million**
  - DNNResearch: **Three person startup** (including Geoff Hinton) acquired by Google for unknown price tag
  - Enlitic, Ersatz, MetaMind, Nervana, Skylab: Deep Learning startups commanding **millions of VC dollars**
- Because it made the **front page** of the New York Times



The New York Times

Why is everyone talking about Deep Learning?



Deep learning:

- Has won numerous pattern recognition competitions
- Does so with minimal feature engineering

This wasn't always the case!

Since 1980s: Form of models hasn't changed much, but lots of new tricks…

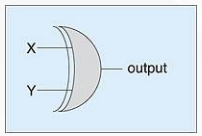
- More hidden units
- Better (online) optimization
- New nonlinear functions (ReLUs)
- Faster computers (CPUs and GPUs)

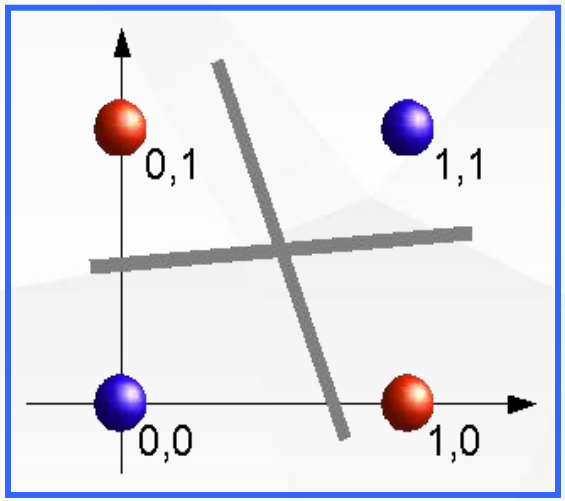
Learning highly non-linear functions

f: X → Y

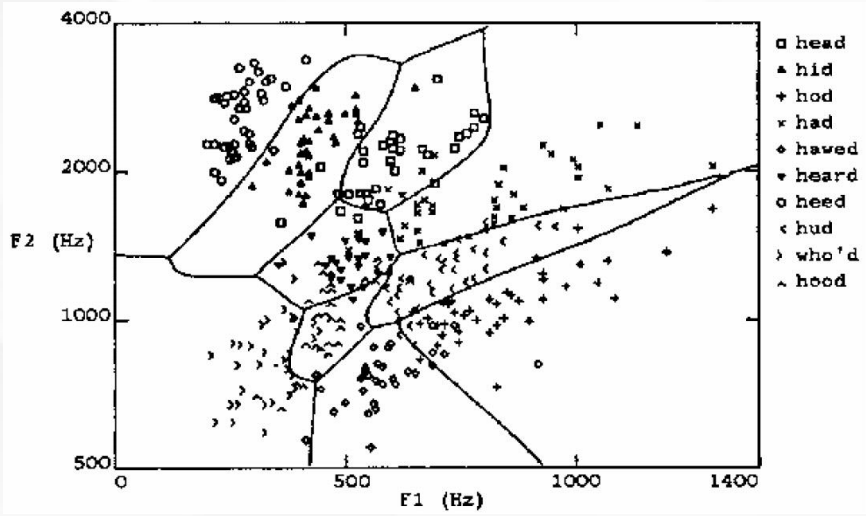
- f might be non-linear function
- X (vector of) continuous and/or discrete vars
- Y (vector of) continuous and/or discrete vars

The XOR gate

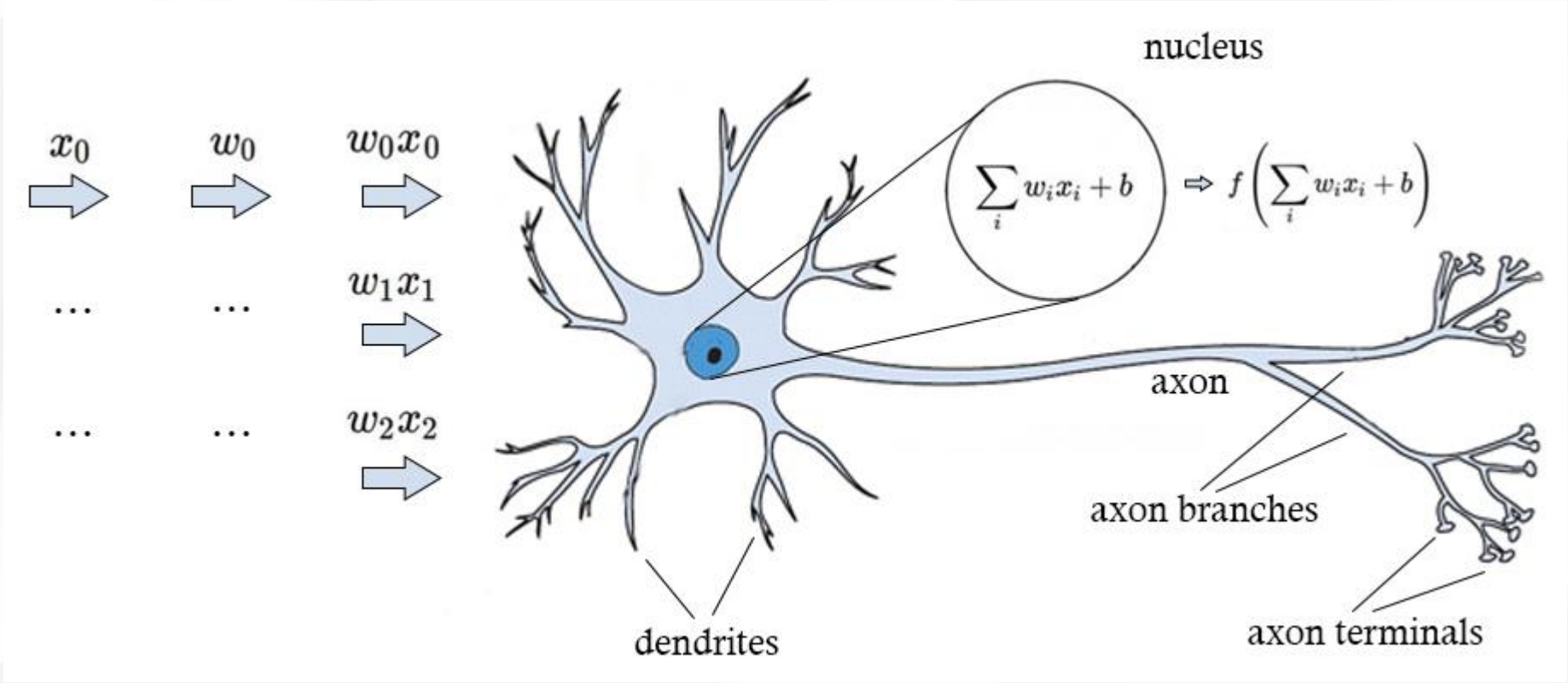




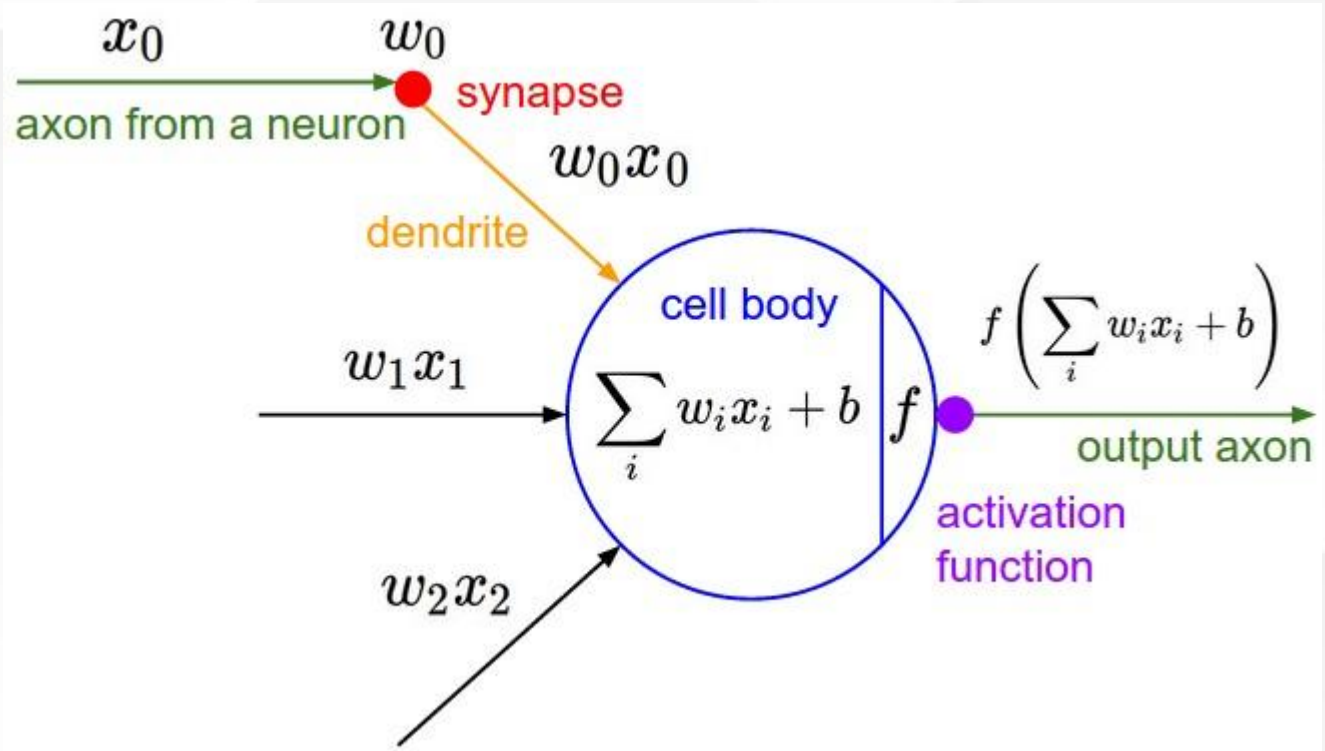
Speech recognition



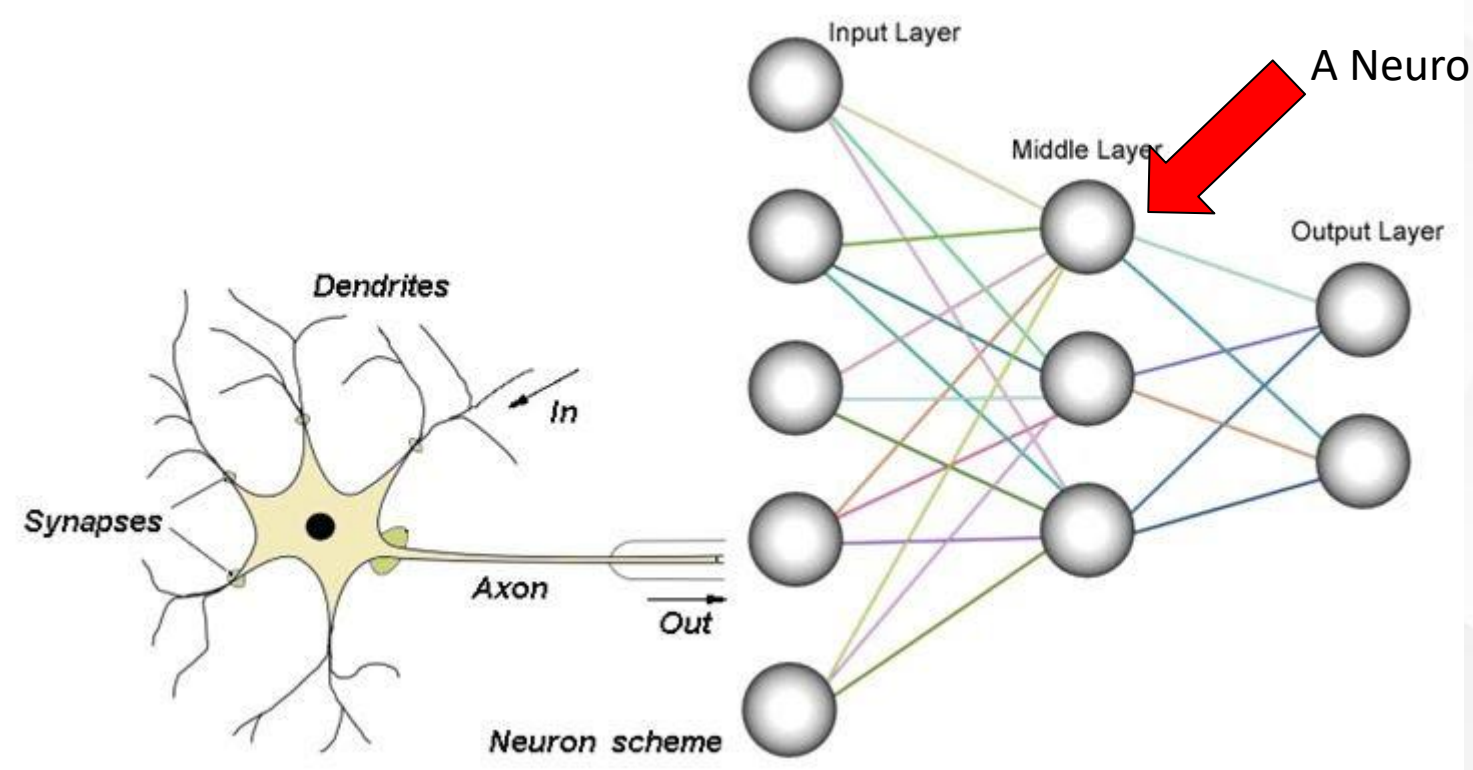
A Illustration of a Neuron



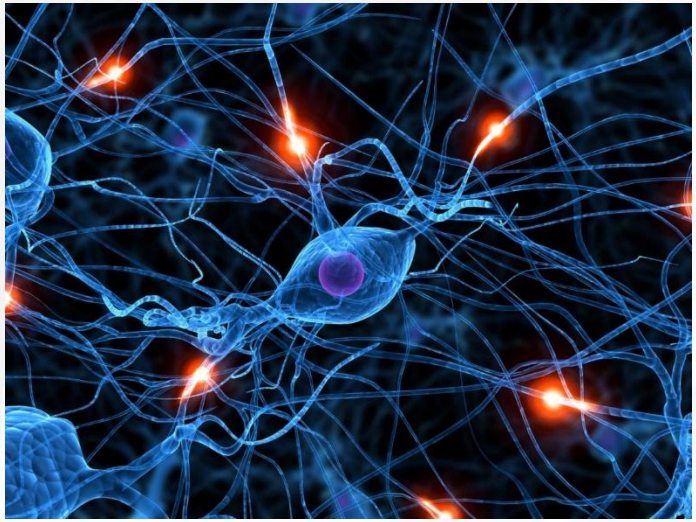
Within a Single Neuron

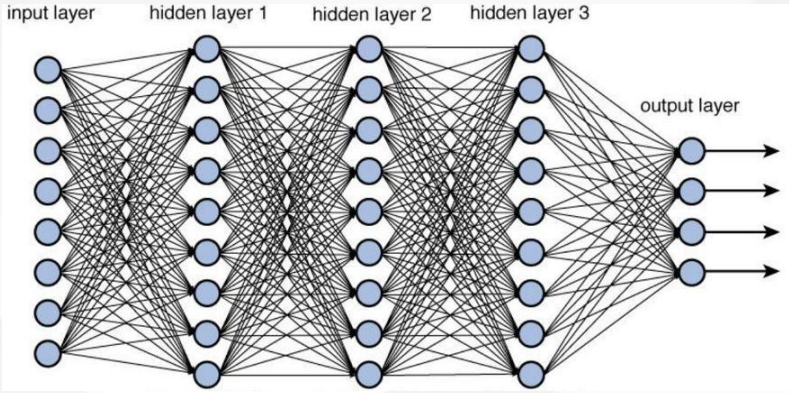


A Network of Neurons



From neurons to neuron network model

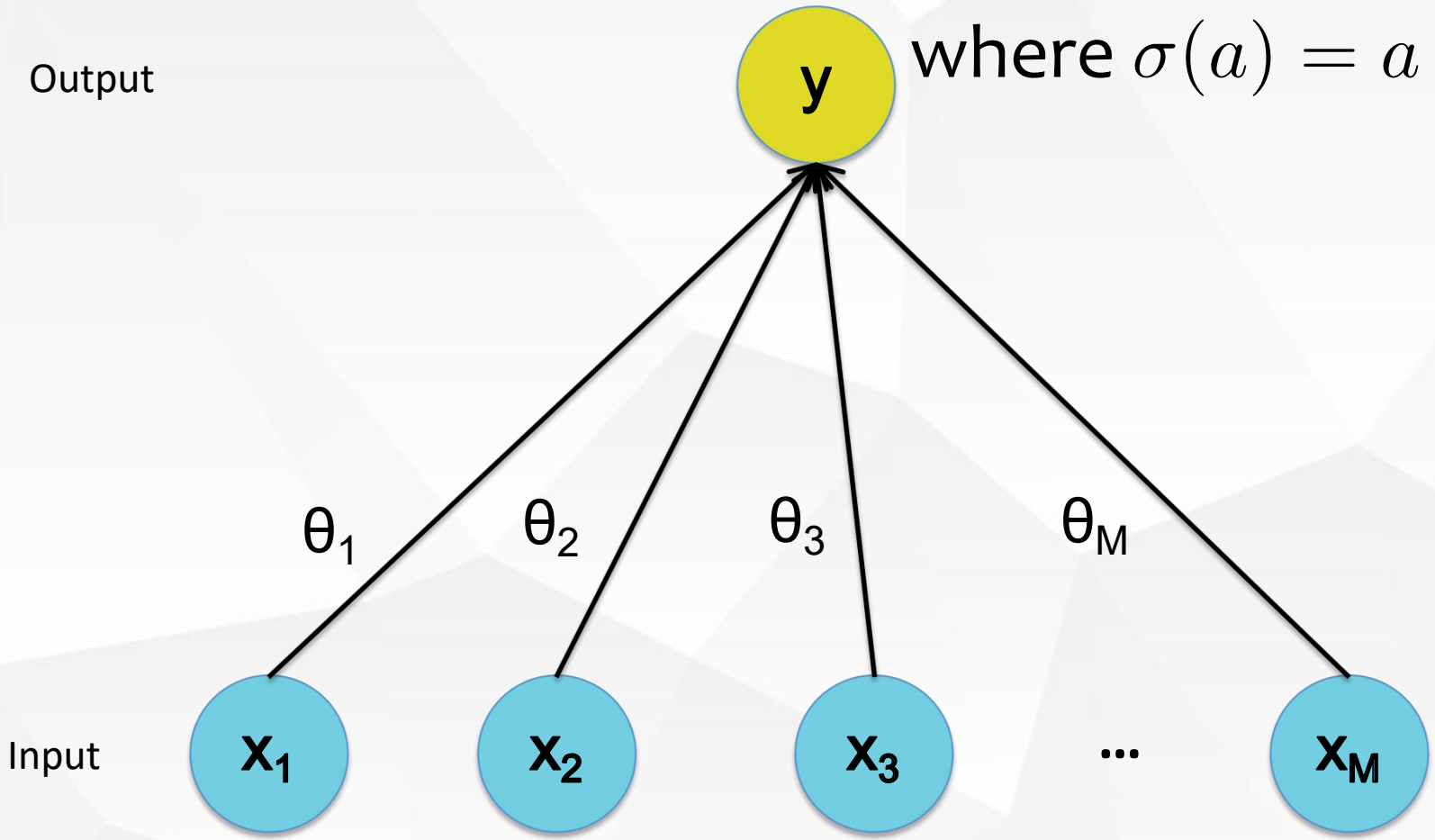




Linear Regression

$$y = h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sigma(\boldsymbol{\theta}^T \boldsymbol{x})$$

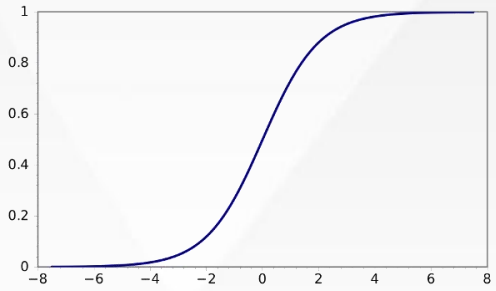
where $\sigma(a) = a$



Recall: Logistic Regression

$$y = h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sigma(\boldsymbol{\theta}^T \boldsymbol{x})$$

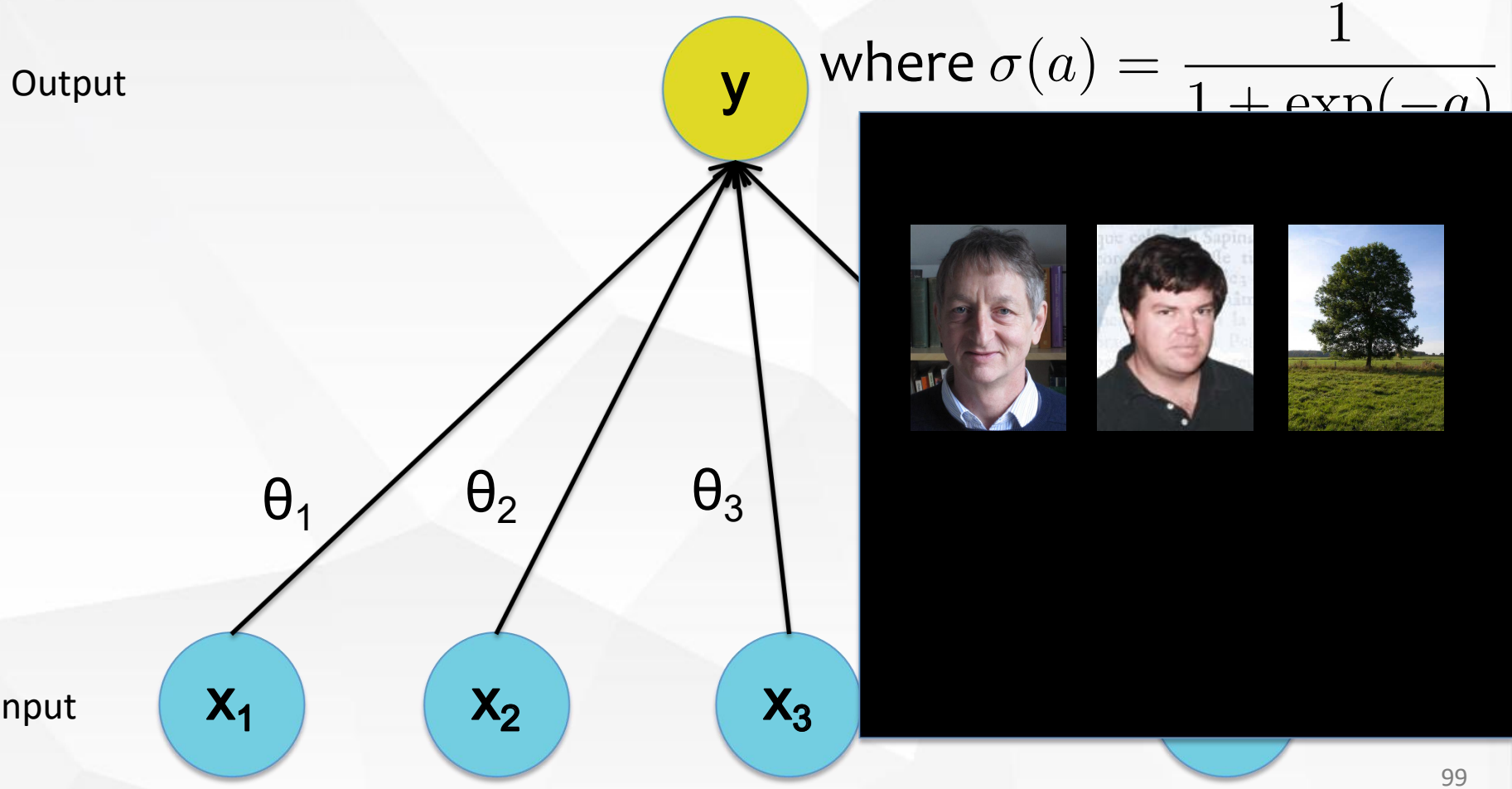
$$\text{where } \sigma(a) = \frac{1}{1 + \exp(-a)}$$



Recall: Logistic Regression

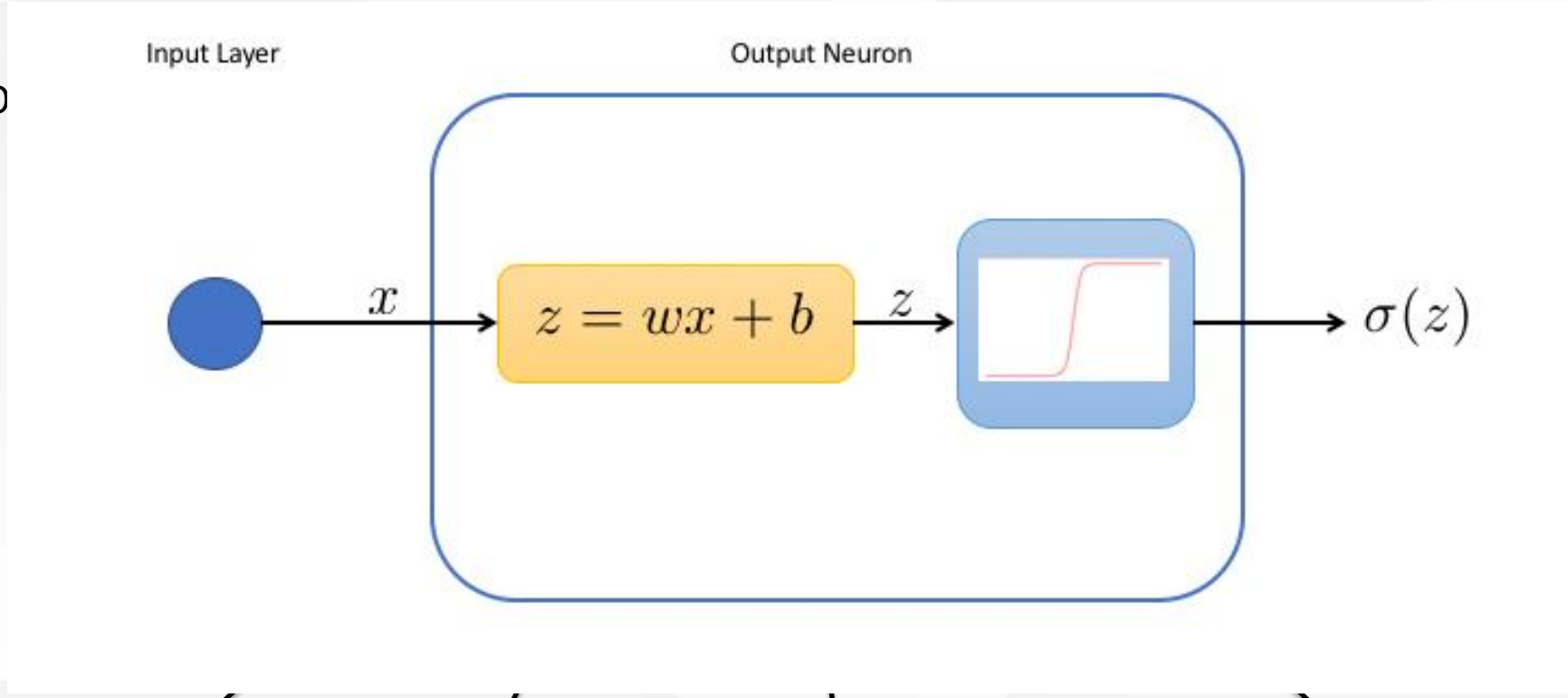
$$y = h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sigma(\boldsymbol{\theta}^T \boldsymbol{x})$$

where $\sigma(a) = \dfrac{1}{1 + \exp(-a)}$

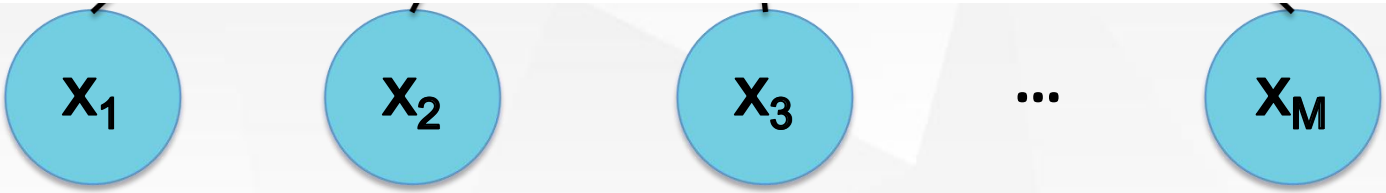


Logistic Regression as a Single Neuron

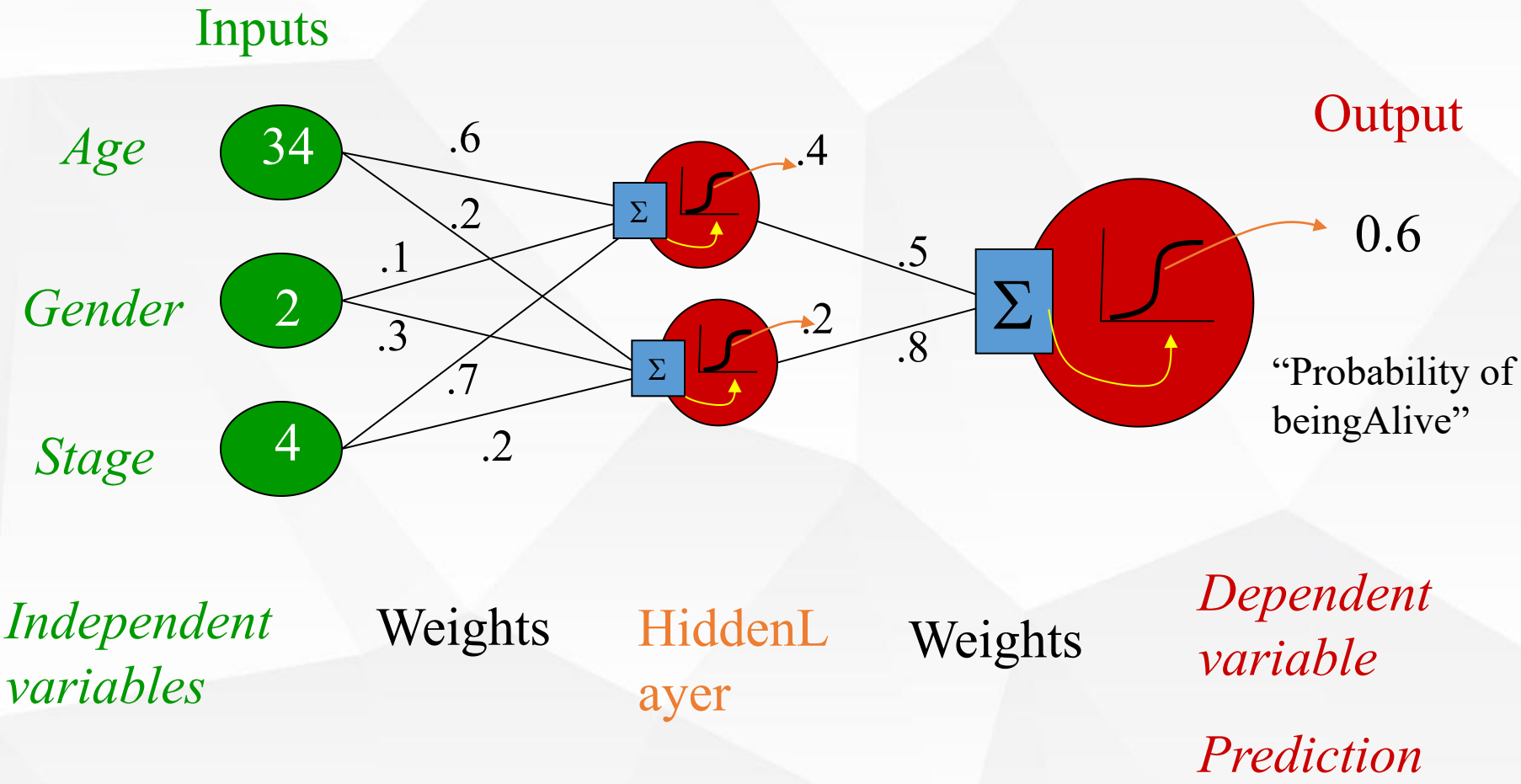
$$y = h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sigma(\boldsymbol{\theta}^T \boldsymbol{x})$$



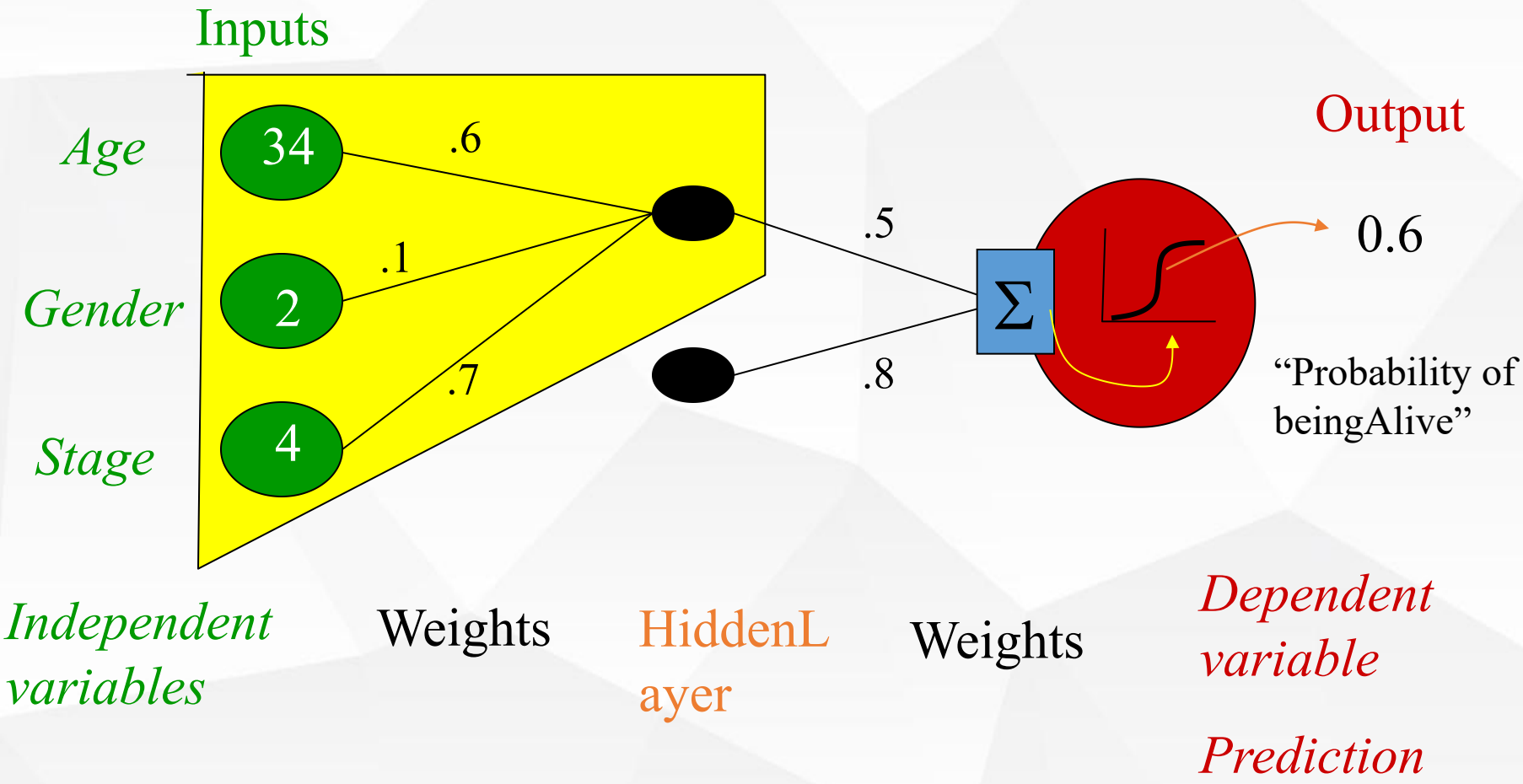
Input



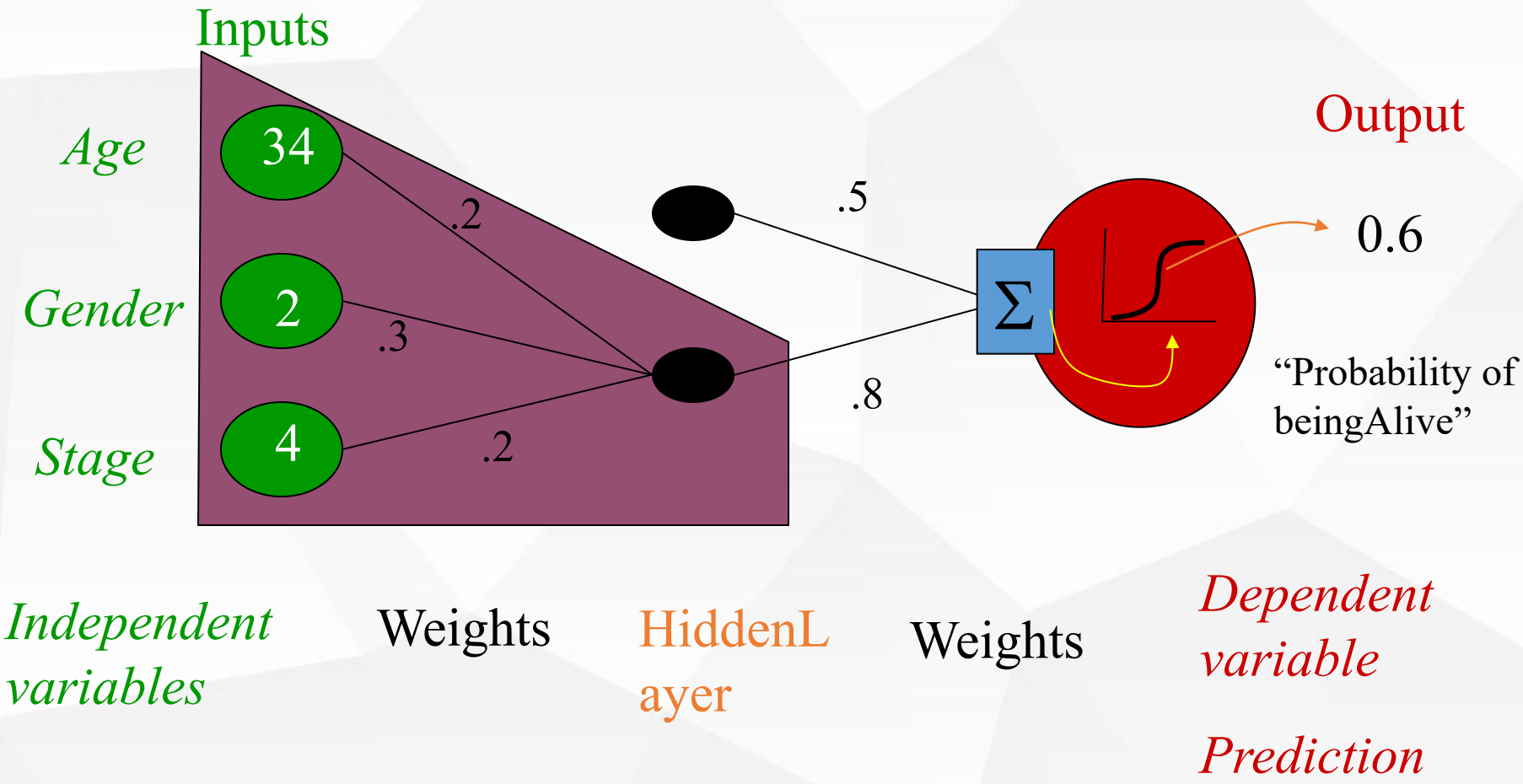
Stacking Neurons



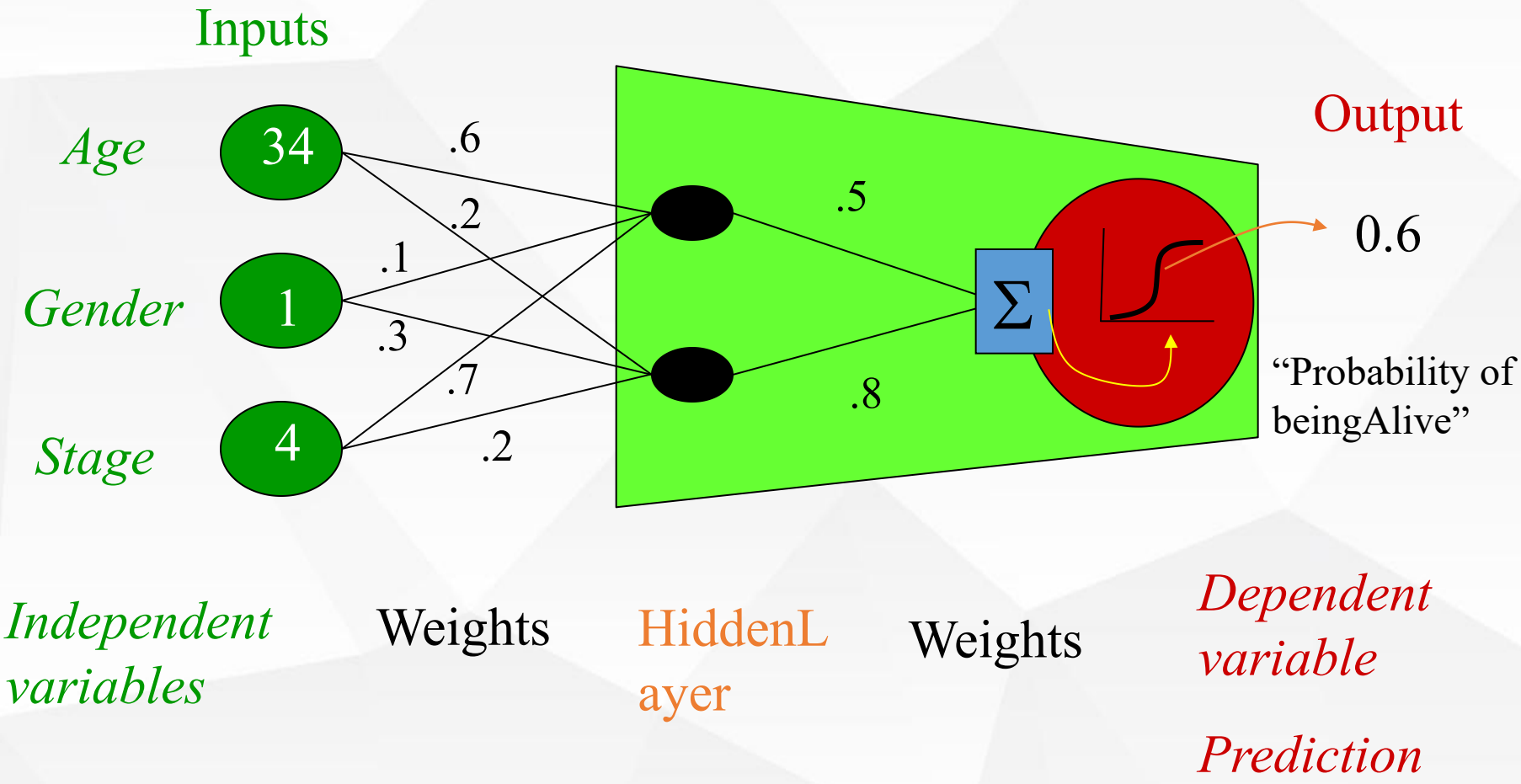
"Combined logistic models"



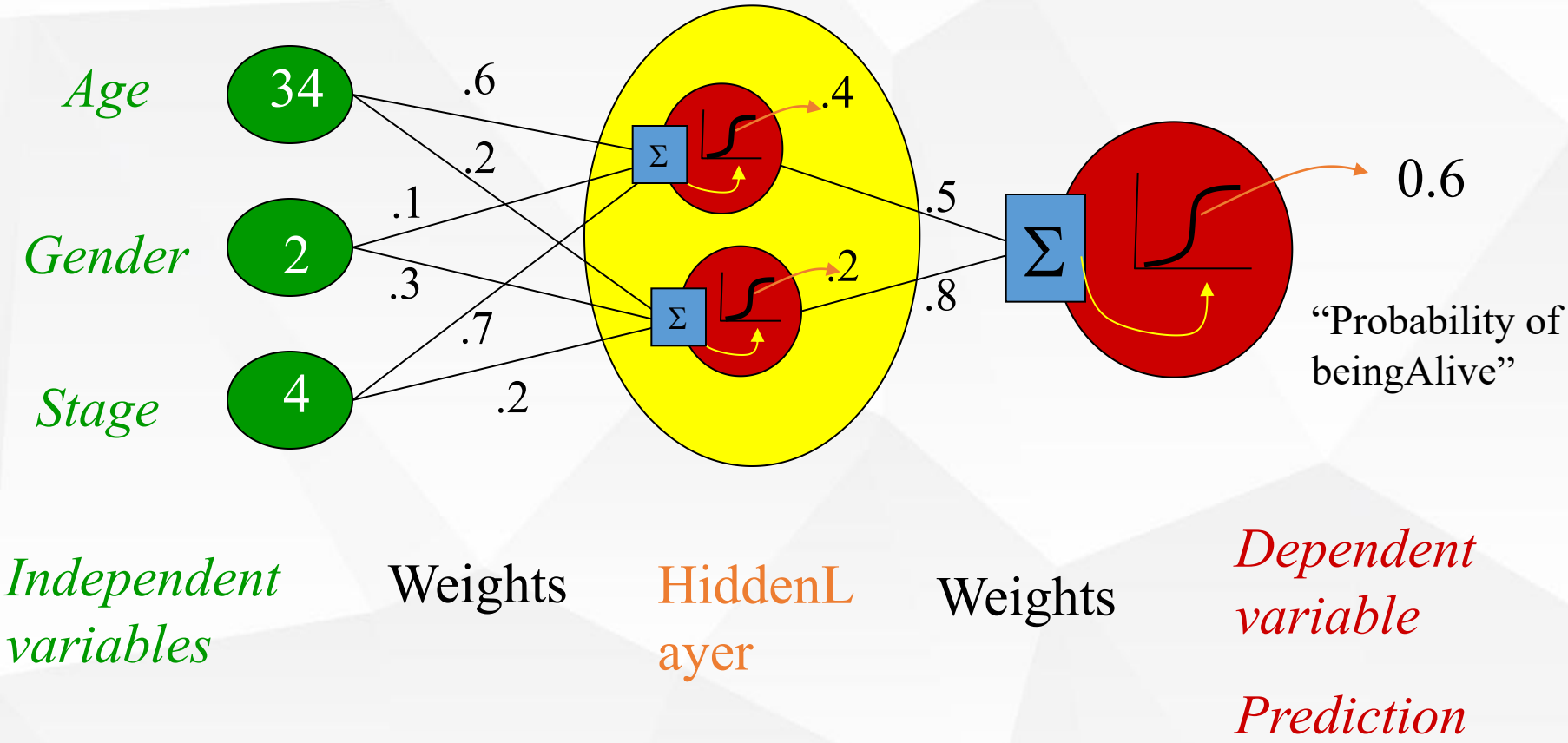
"Combined logistic models"



"Combined logistic models"



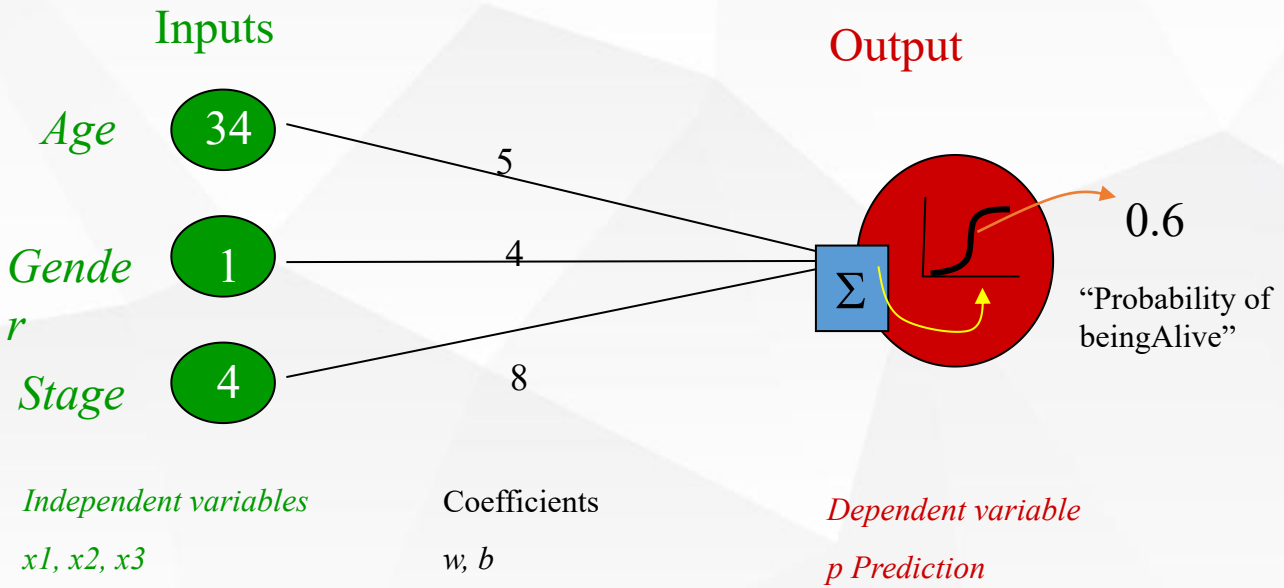
Not really, no target for hidden units…



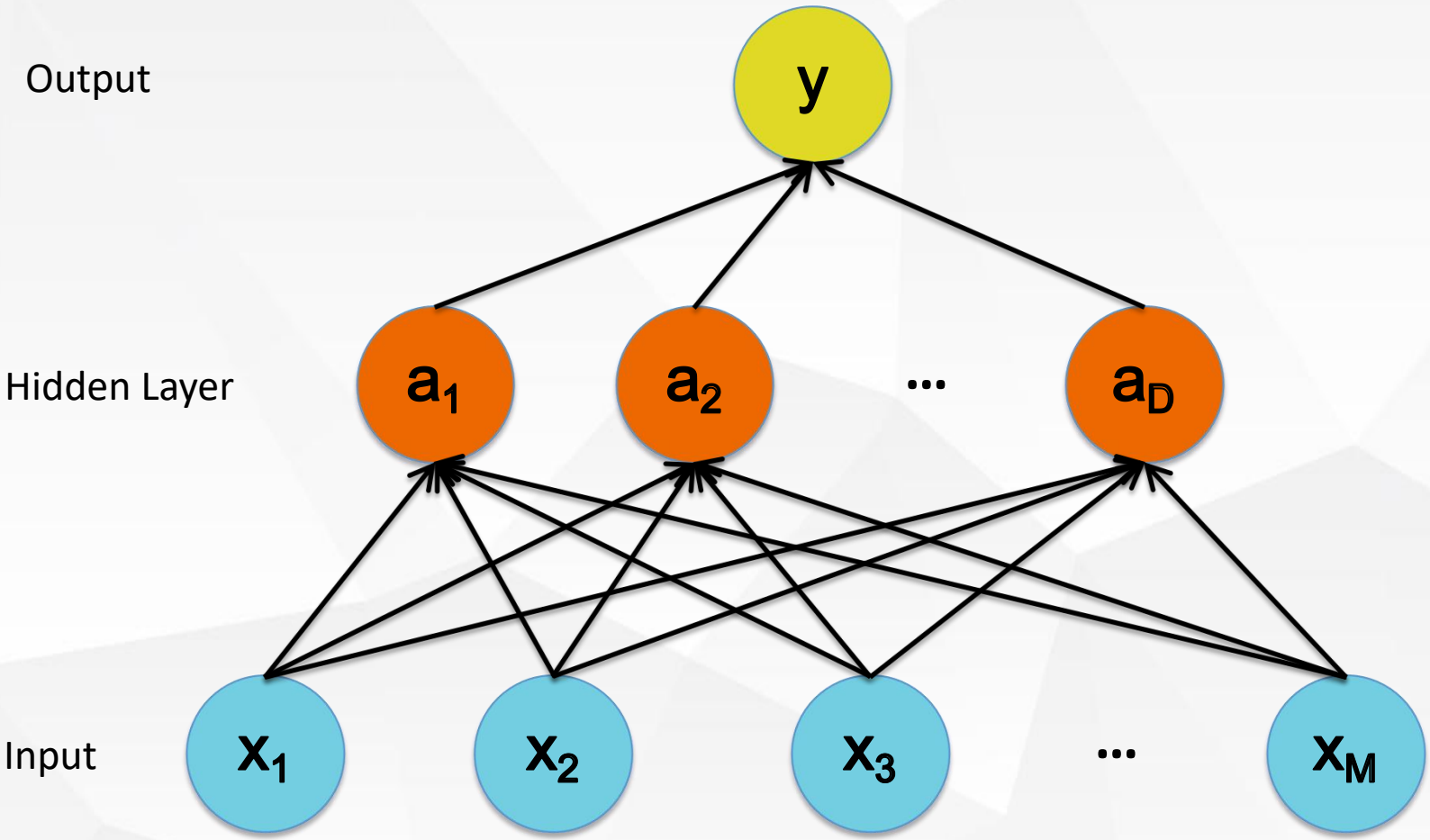
Jargons

- Independent variable = input variable
- Dependent variable = output variable
- Coefficients = “weights”
- Estimates = “targets”

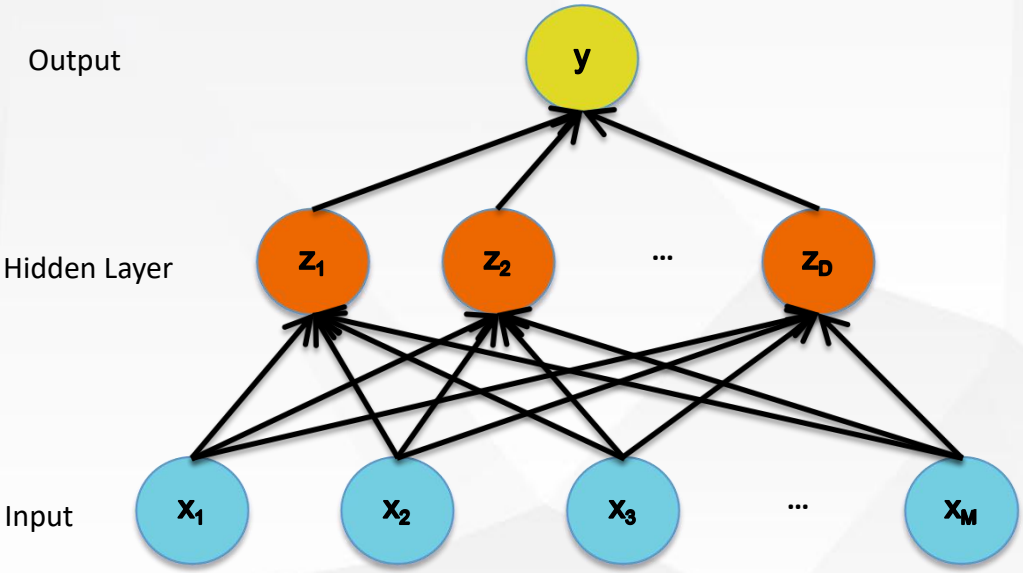
Logistic Regression Model (the sigmoid unit)



Neural Network



Neural Network



(E) **Output (sigmoid)**

$$y = \frac{1}{1+\exp(-b)}$$

(D) **Output (linear)**

$$b = \sum_{j=0}^{D} \beta_j z_j$$

(C) **Hidden (sigmoid)**

$$z_j = \frac{1}{1+\exp(-a_j)}, \ \forall j$$

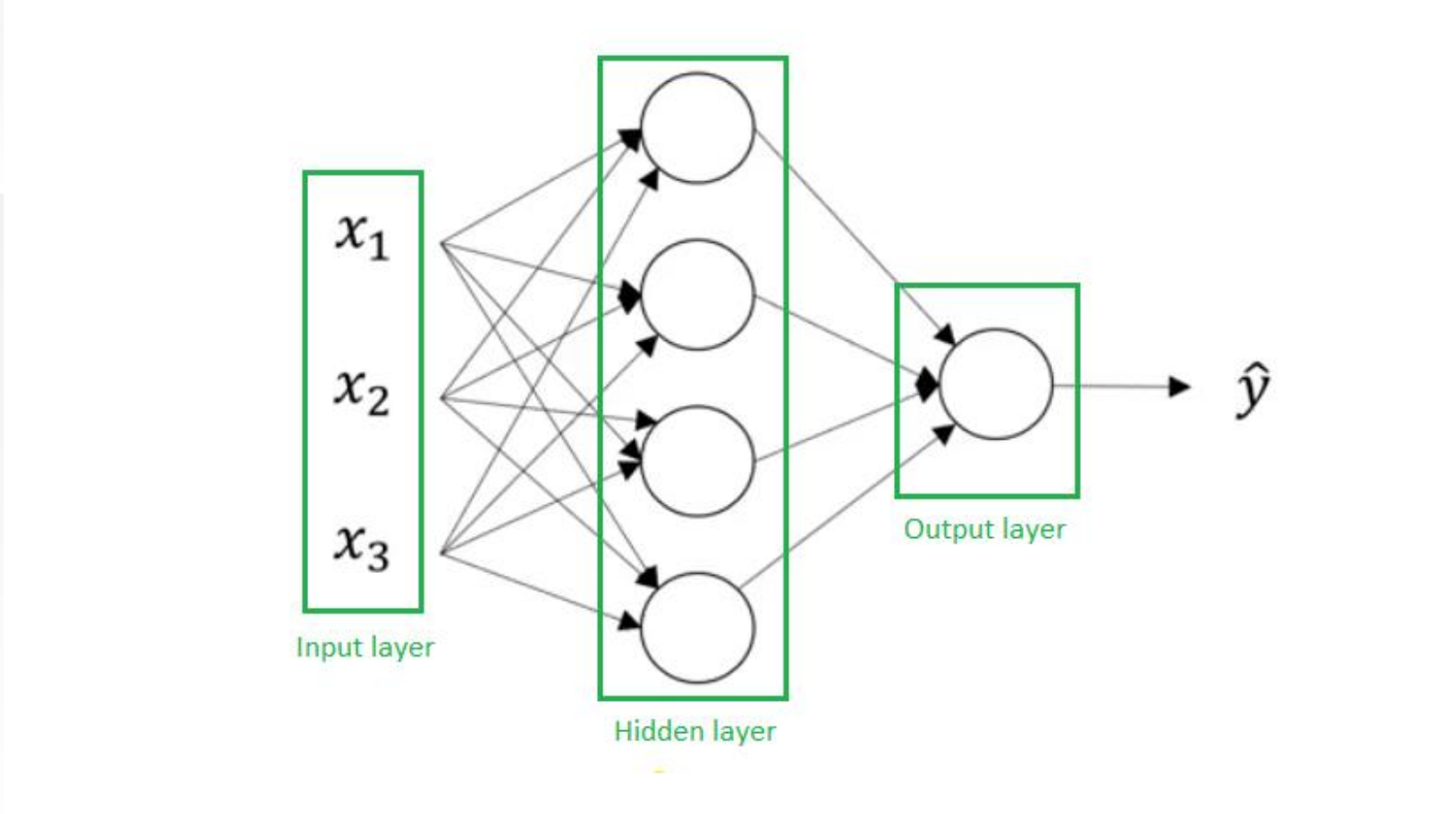
(B) **Hidden (linear)**

$$a_j = \sum_{i=0}^{M} \alpha_{ji} x_i, \ \forall j$$

(A) **Input**

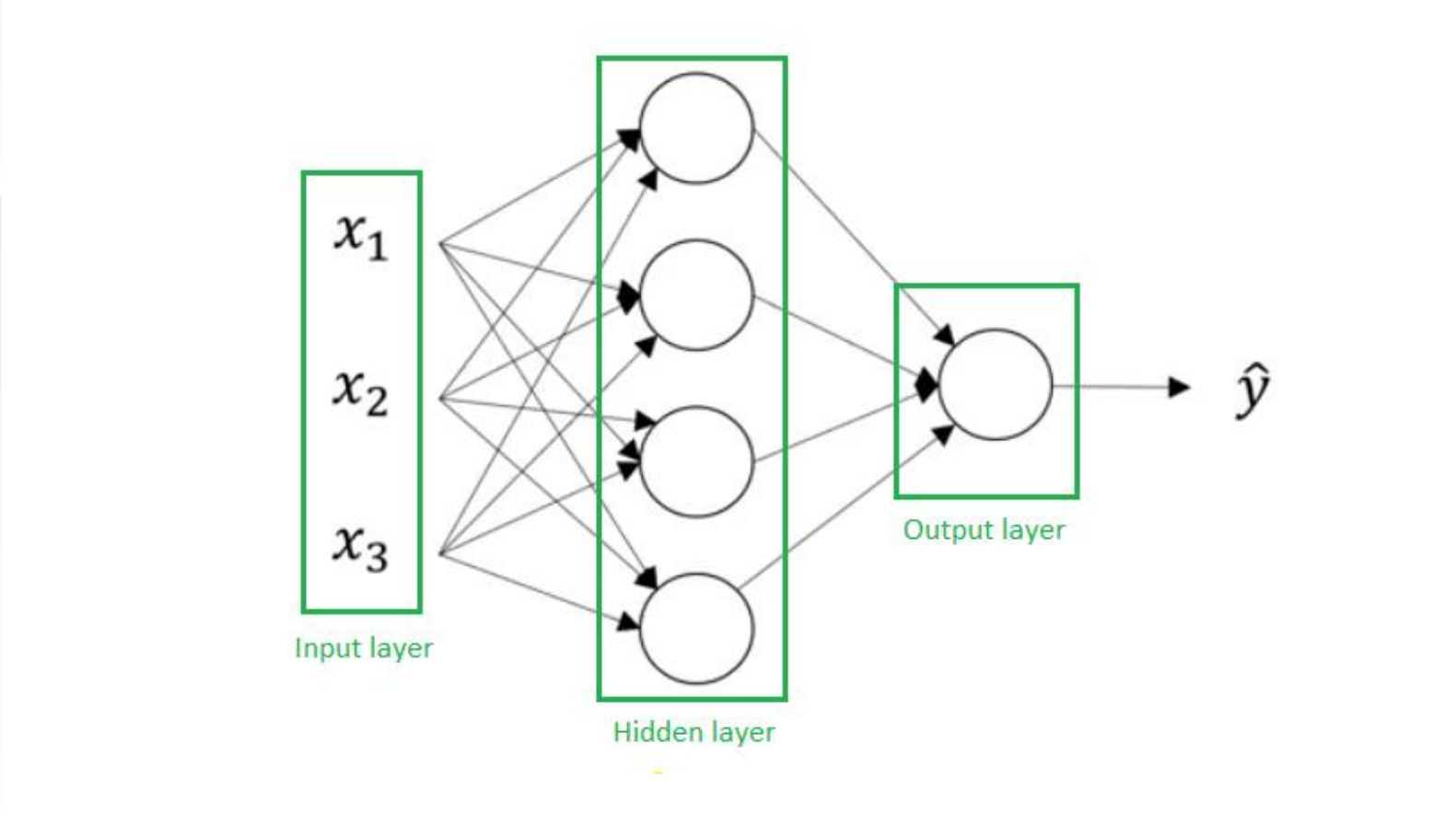
Given $x_i, \ \forall i$

Symbols for Neural Net



$x \; or \; a^{[0]}$

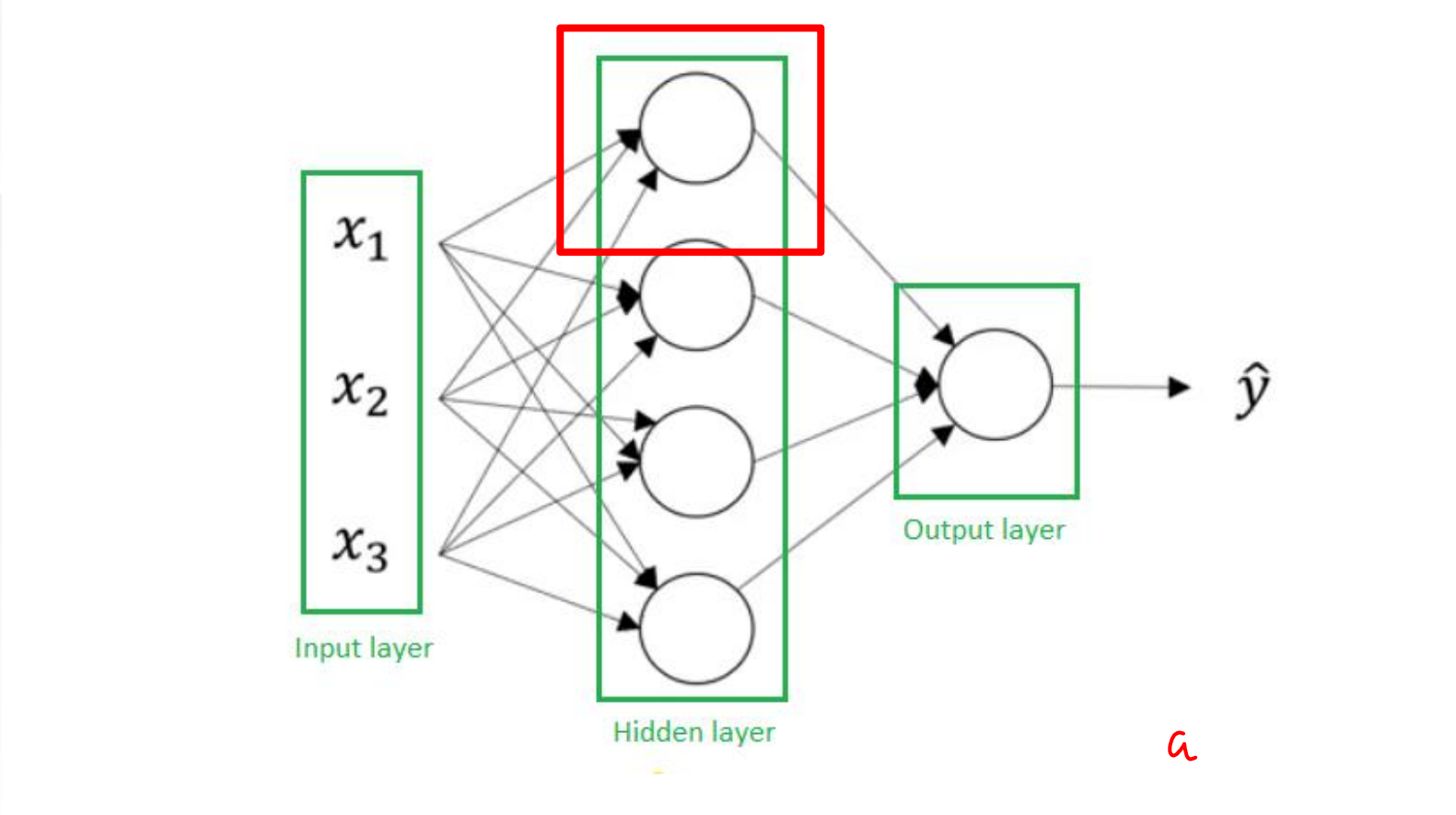
Symbols for Neural Net



x or $a^{[0]}$ $a^{[1]}$ $a^{[2]}$

Depth =2

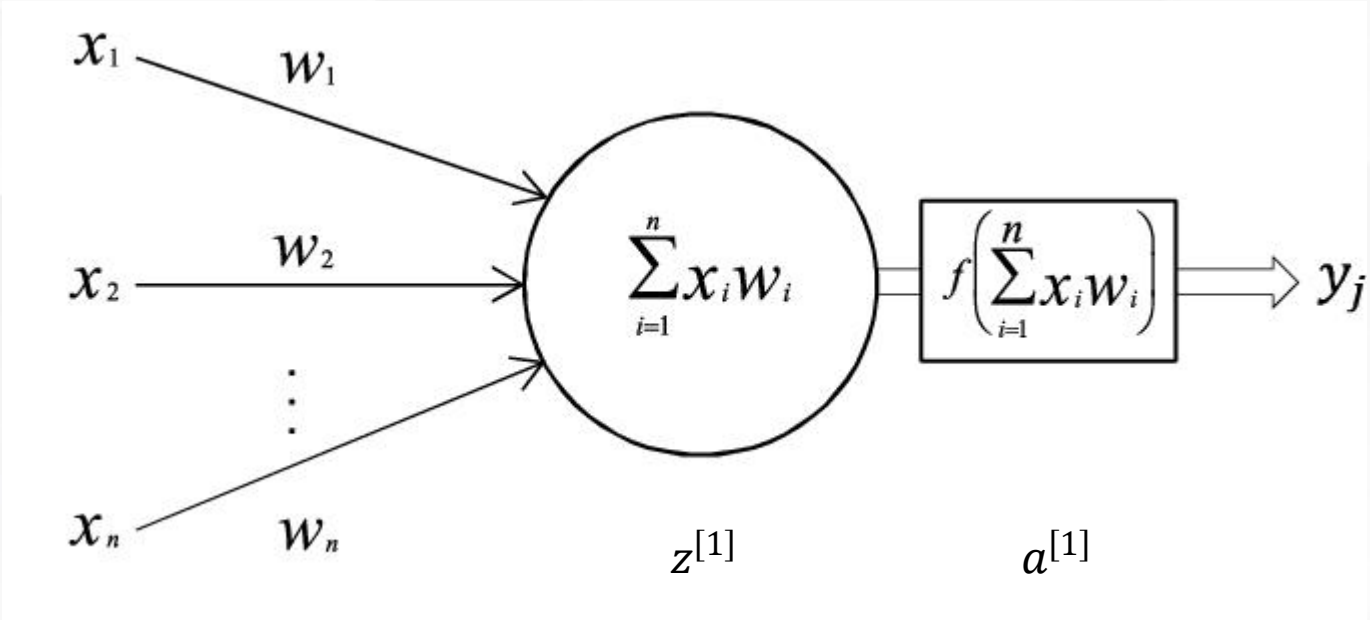
Symbols for Neural Net



$x \text{ or } a^{[0]}$ $a^{[1]}$ $y \text{ or } a^{[2]}$

Depth =2

Zoomed out



Universal Approximation Theorem: Bounded Depth

Universal approximation theorem — Let $C(X, \mathbb{R}^m)$ denote the set of continuous functions from a subset X of a Euclidean $\mathbb{R}^n$ space to a Euclidean space $\mathbb{R}^m$. Let $\sigma \in C(\mathbb{R}, \mathbb{R})$. Note that $(\sigma \circ x)_i = \sigma(x_i)$, so $\sigma \circ x$ denotes σ applied to each component of x.

Then σ is not polynomial if and only if for every $n \in \mathbb{N}$, $m \in \mathbb{N}$, compact $K \subseteq \mathbb{R}^n$, $f \in C(K, \mathbb{R}^m), \varepsilon > 0$ there exist $k \in \mathbb{N}$, $A \in \mathbb{R}^{k \times n}$, $b \in \mathbb{R}^k$, $C \in \mathbb{R}^{m \times k}$ such that

$$\sup_{x \in K} \|f(x) - g(x)\| < \varepsilon$$

where $g(x) = C \cdot (\sigma \circ (A \cdot x + b))$

Optional Contents

Proof sketch

It suffices to prove the case where $m = 1$, since uniform convergence in $\mathbb{R}^m$ is just uniform convergence in each coordinate.

Let F_σ be the set of all one-hidden-layer neural networks constructed with σ. Let $C_0(\mathbb{R}^d, \mathbb{R})$ be the set of all $C(\mathbb{R}^d, \mathbb{R})$ with compact support.

If the function is a polynomial of degree d, then F_σ is contained in the closed subspace of all polynomials of degree d, so its closure is also contained in it, which is not all of $C_0(\mathbb{R}^d, \mathbb{R})$.

Otherwise, we show that F_σ's closure is all of $C_0(\mathbb{R}^d, \mathbb{R})$. Suppose we can construct arbitrarily good approximations of the ramp function $r(x) = \begin{cases} -1 & \text{if } x < -1 \\ x & \text{if } |x| \leq 1 \\ 1 & \text{if } x > 1 \end{cases}$ then it can be combined to construct arbitrary compactly-supported continuous function to arbitrary precision. It remains to approximate the ramp function.

Any of the commonly used activation functions used in machine learning can obviously be used to approximate the ramp function, or first approximate the ReLU, then the ramp function.

if σ is "squashing", that is, it has limits $\sigma(-\infty) < \sigma(+\infty)$, then one can first affinely scale down its x-axis so that its graph looks like a step-function with two sharp "overshoots", then make a linear sum of enough of them to make a "staircase" approximation of the ramp function. With more steps of the staircase, the overshoots smooth out and we get arbitrarily good approximation of the ramp function.

The case where σ is a generic non-polynomial function is harder, and the reader is directed to.[14]

Optional Contents

Universal Approximation Theorem: Arbitrary Depth

Universal approximation theorem *(L1 distance, ReLU activation, arbitrary depth, minimal width).* For any Bochner–Lebesgue p-integrable function $f : \mathbb{R}^n \to \mathbb{R}^m$ and any $\varepsilon > 0$, there exists a fully connected ReLU network F of width exactly $d_m = \max\{n+1, m\}$, satisfying

$$\int_{\mathbb{R}^n} \|f(x) - F(x)\|^p \, \mathrm{d}x < \varepsilon.$$

Optional Contents

Thank you!