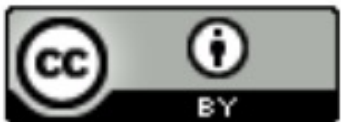


Artificial Neurons

Danna Gurari

University of Colorado Boulder

Spring 2022



Review

- Last lecture:
 - Deep learning applications
 - History of neural networks and deep learning
 - How does a machine learn?
 - Course logistics
- Assignments (Canvas):
 - Problem set 1 due next week
- Questions?

Today's Topics

- Binary classification applications
- Evaluating classification models
- Biological neurons: inspiration
- Artificial neuron: Perceptron

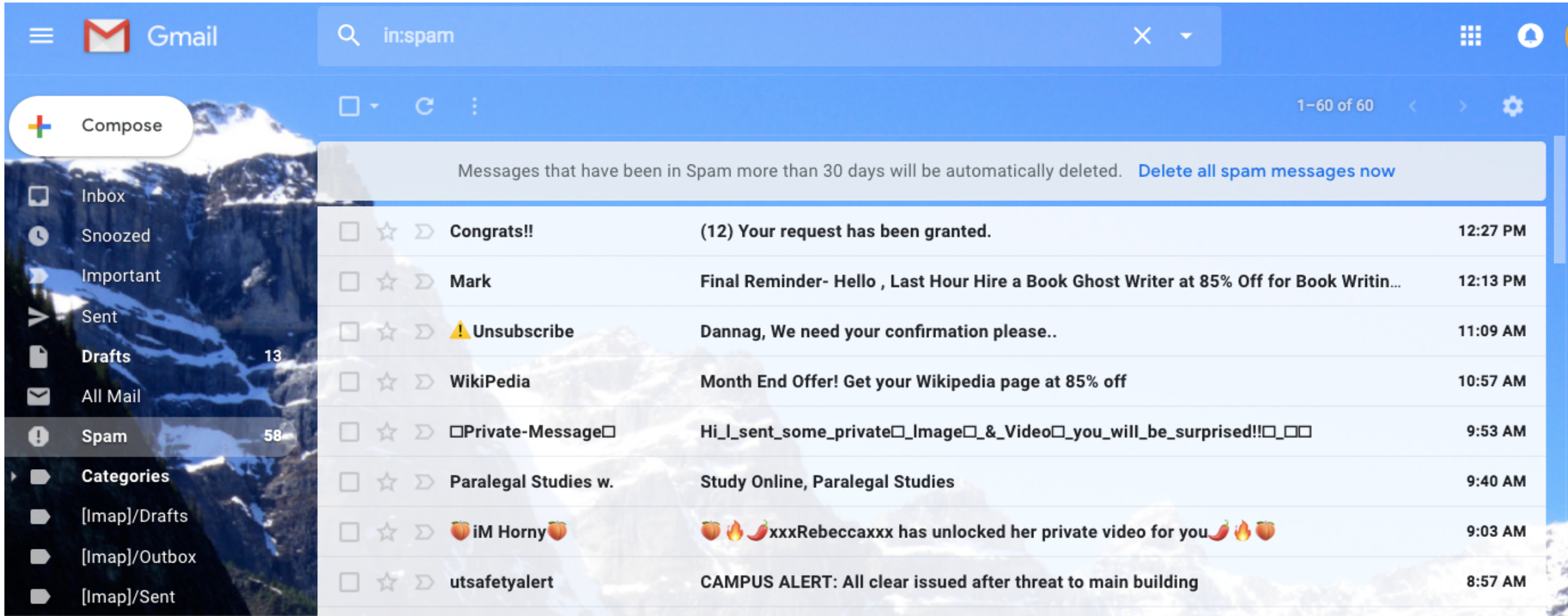
Today's Topics

- Binary classification applications
- Evaluating classification models
- Biological neurons: inspiration
- Artificial neuron: Perceptron

Today's Scope: Binary Classification

Distinguish 2 classes

Binary Classification: Spam Detection



The image shows a screenshot of a Gmail inbox with the search filter 'in:spam' applied. The interface includes a left sidebar with navigation options like 'Compose', 'Inbox', 'Spam', and 'Categories'. The main area displays a list of spam messages with their subjects and senders. A notification at the top of the message list states: 'Messages that have been in Spam more than 30 days will be automatically deleted. [Delete all spam messages now](#)'.

| Message Icon | Star | More | Sender | Subject | Time |
|--------------------------|------|------|----------------------|--|----------|
| <input type="checkbox"/> | ☆ | ⋮ | Congrats!! | (12) Your request has been granted. | 12:27 PM |
| <input type="checkbox"/> | ☆ | ⋮ | Mark | Final Reminder- Hello , Last Hour Hire a Book Ghost Writer at 85% Off for Book Writin... | 12:13 PM |
| <input type="checkbox"/> | ☆ | ⋮ | ! Unsubscribe | Dannag, We need your confirmation please.. | 11:09 AM |
| <input type="checkbox"/> | ☆ | ⋮ | WikiPedia | Month End Offer! Get your Wikipedia page at 85% off | 10:57 AM |
| <input type="checkbox"/> | ☆ | ⋮ | Private-Message | Hi_I_sent_some_private_Image_&_Video_you_will_be_surprised!! | 9:53 AM |
| <input type="checkbox"/> | ☆ | ⋮ | Paralegal Studies w. | Study Online, Paralegal Studies | 9:40 AM |
| <input type="checkbox"/> | ☆ | ⋮ | iM Horny | xxxRebeccaxxx has unlocked her private video for you | 9:03 AM |
| <input type="checkbox"/> | ☆ | ⋮ | utsafetyalert | CAMPUS ALERT: All clear issued after threat to main building | 8:57 AM |

Binary Classification: Resume Pre-Screening



Prime Talent Chain
Hiring Decentralized

AI AND BLOCKCHAIN STAFFING & RECRUITMENT

- ◆ Decentralizing and simplifying the staffing industry
- ◆ Eliminate the intermediaries between job seeker, through an open ecosystem of hiring managers by using Blockchain, AI, and other technologies, that ultimately makes hiring more cost-effective

[BUY COIN](#) [WHITE PAPER](#)

[Book demo](#) [Free trial](#)

[ABOUT](#) [PLATFORM](#) [ROADMAP](#) [COIN](#) [TEAM](#) [WHITEPAPER](#) [SIGN IN](#) [BUY COIN](#)

DCI Skillate [HOME](#) [PRODUCT](#) [ABOUT US](#) [BLOG](#) [REQUEST A DEMO](#)

The diagram illustrates the resume screening and matching process. On the left, a stack of resume icons is shown with a large arrow pointing to a single resume icon with a green checkmark, representing the screening process. On the right, a stack of resume icons is shown with a large arrow pointing to a single resume icon with a green checkmark, representing the matching process. The background features a stylized globe with green lines.

Automatically screen resumes

Trained with over 20 million diverse profiles, Skillate's AI algorithm helps to screen and shortlist resume with the click of a button. Seamlessly integrate with all external channels and ATS to source resume directly

Matching algorithm and candidate recommendation

Skillate's matching engine maps all the relevant profiles with the job requirements - be it skills, education or experience and recommends the best candidate

Binary Classification: Cancer Diagnosis



The image shows a screenshot of the PathAI website. At the top left is the PathAI logo, which consists of a red hexagon with a white stylized 'P' and 'A' inside, followed by the text 'PathAI'. To the right of the logo are navigation links: 'What we do', 'About us', 'News', 'Careers', 'Pathologists', and 'Partner with Us'. In the top right corner, there is a red rectangular button with the text 'Partner Login' in white. Below the navigation bar is a large circular image of a histological slide showing tissue with various colors (pink, purple, green, blue) indicating different cellular structures. To the right of this image is the main headline and subtext.

PathAI

What we do About us News Careers Pathologists Partner with Us

Partner Login

Pathology Evolved.

Advanced learning toward faster, more accurate diagnosis of disease.


Binary Classification: Cognitive Impairment Recognition by Apple App Usage



Image Credit: <https://www.techradar.com/news/the-10-best-phones-for-seniors>
https://www.technologyreview.com/f/615032/the-apps-you-use-on-your-phone-could-help-diagnose-your-cognitive-health/?utm_medium=tr_social&utm_campaign=site_visitor.unpaid.engagement&utm_source=Twitter#EchoBox=1579899156

Binary Classification: Sentiment Analysis


CRITIC REVIEWS FOR *MULAN*

 Its cast, its attitude, its overall eagerness to please -- all benefits, one would think -- don't add up to a good movie. They add up to a blueprint of the movie this ought to be.

October 23, 2020 | Rating: 2.5/5 | [Full Review...](#)



K. Austin Collins
Rolling Stone
★ **TOP CRITIC**

 While glorious to look at, the movie still feels slightly hollow. All the right pieces are there, but an emotional connection to the characters is lacking.

September 10, 2020 | Rating: 6.8/10 | [Full Review...](#)



Amy Amatangelo
Paste Magazine
★ **TOP CRITIC**

Binary Classification: Food Quality Control



Machine Learning: Using Algorithms to Sort Fruit

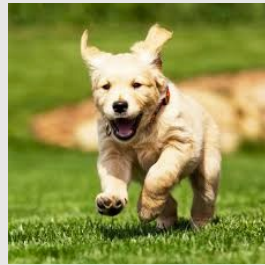
Demo: <https://www.youtube.com/watch?v=Bl3XzBWpZbY>

Today's Topics

- Binary classification applications
- **Evaluating classification models**
- Biological neurons: inspiration
- Artificial neuron: Perceptron

Goal: Design Models that **Generalize Well** to New, Previously Unseen Examples

Example:



Label:

Hairy

Hairy

Not Hairy



Hairy



Goal: Design Models that **Generalize Well** to New, Previously Unseen Examples

1. Split data into a “**training set**” and “**test set**”

Training Data

Test Data

Example:



Label:

Hairy

Hairy

Not Hairy



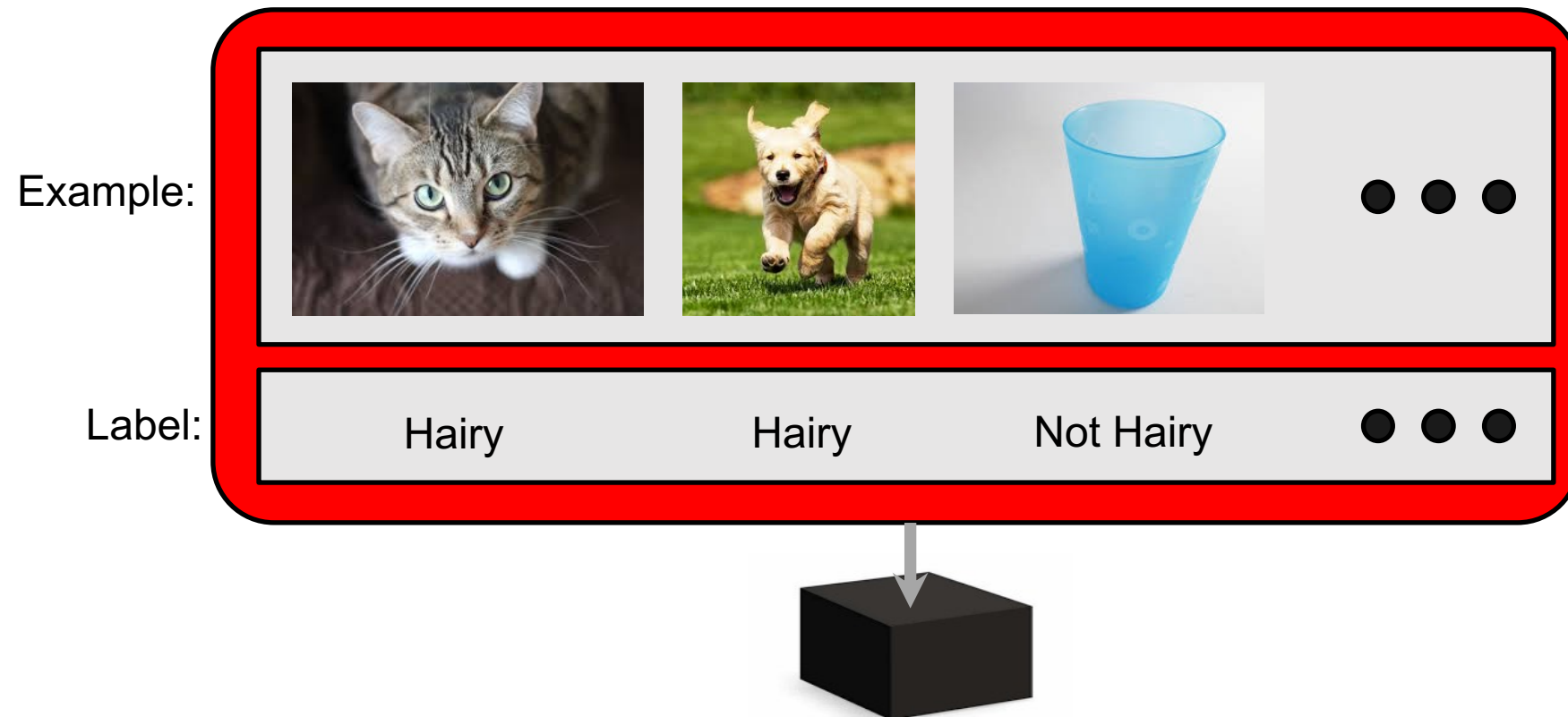
Hairy



Goal: Design Models that **Generalize Well** to New, Previously Unseen Examples

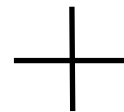
2. Train model on “**training set**” to try to minimize prediction error on it

Training Data

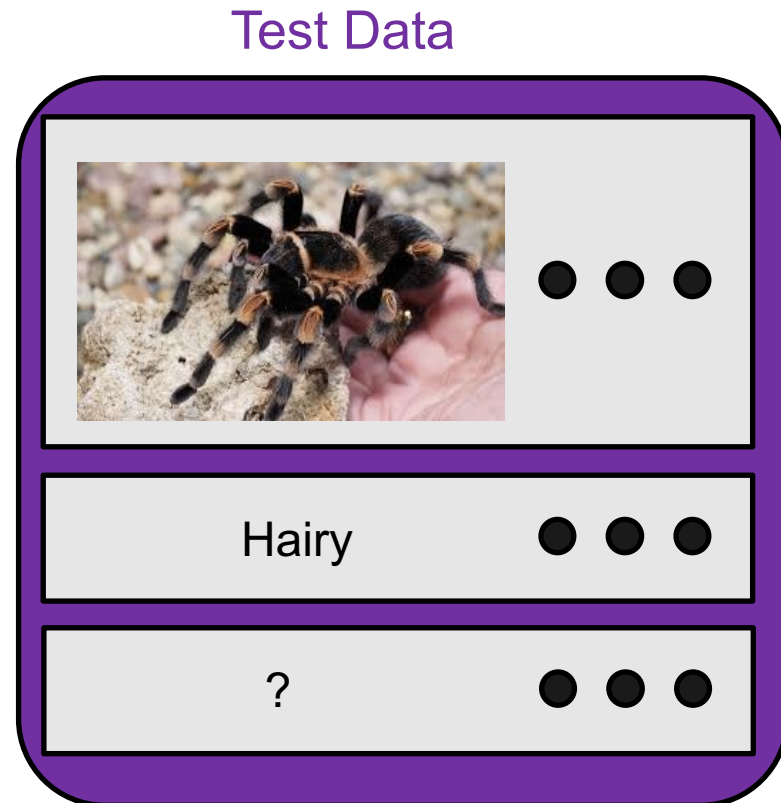


Goal: Design Models that **Generalize Well** to New, Previously Unseen Examples

3. Apply trained model on “**test set**” to measure generalization error

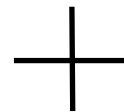


Example:

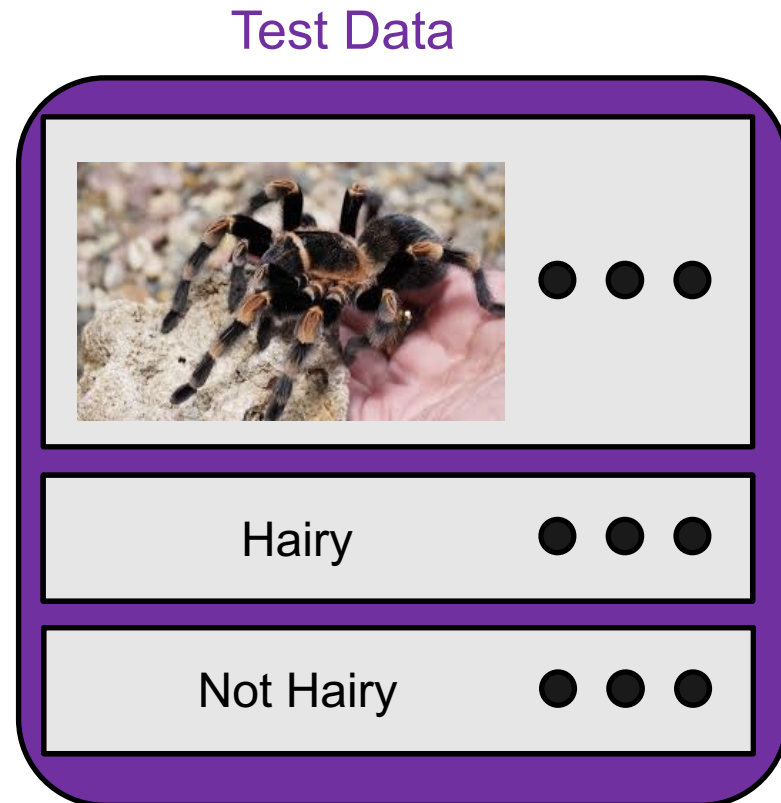


Goal: Design Models that **Generalize Well** to New, Previously Unseen Examples

3. Apply trained model on “**test set**” to measure generalization error



Example:

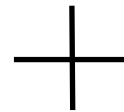


Label:

Predicted Label:


Goal: Design Models that **Generalize Well** to New, Previously Unseen Examples

3. Apply trained model on “**test set**” to measure generalization error



Example:

Test Data

| | |
|---|-------|
|  | ● ● ● |
| Hairy | ● ● ● |
| Not X airy | ● ● ● |

Label:

Predicted Label:

Evaluation Methods: Confusion Matrix

| | | Actual | |
|-----------|---------|--------|---------|
| | | Spam | Trusted |
| Predicted | Spam | TP | FP |
| | Trusted | FN | TN |

TP = true positive

TN = true negative

FP = false positive

FN = false negative

Evaluation Methods : Descriptive Statistics

Commonly-used statistical descriptions:

e.g.,

| | | Actual | |
|-----------|---------|--------|---------|
| | | Spam | Trusted |
| Predicted | Spam | 50 | 10 |
| | Trusted | 15 | 100 |

- How many **actual spam** results are there? - 65
- How many **actual trusted** results are there? - 110
- How many **correctly classified instances**? - 150/175 ~ 86%
- How many **incorrectly classified instances**? - 25/175 ~ 14%

- What is the **precision**? $\frac{TP}{TP + FP}$
 - 50/(50+10) ~ 83%

- What is the **recall**? $\frac{TP}{TP + FN}$
 - 50/(50+15) ~ 77%

Group Discussion

- Which of these evaluation metrics would you use versus not use and why?
 - Accuracy (number of correctly classified examples)
 - Precision
 - Recall
- Scenario 1: Medical test for a rare disease affecting one in every million people.
- Scenario 2: Deciding which emails to flag as spam.

Today's Topics

- Binary classification applications
- Evaluating classification models
- **Biological neurons: inspiration**
- Artificial neuron: Perceptron

Inspiration: Animal's Computing Machinery

Neuron

- basic unit in the nervous system for receiving, processing, and transmitting information; e.g., messages such as...

“hot”



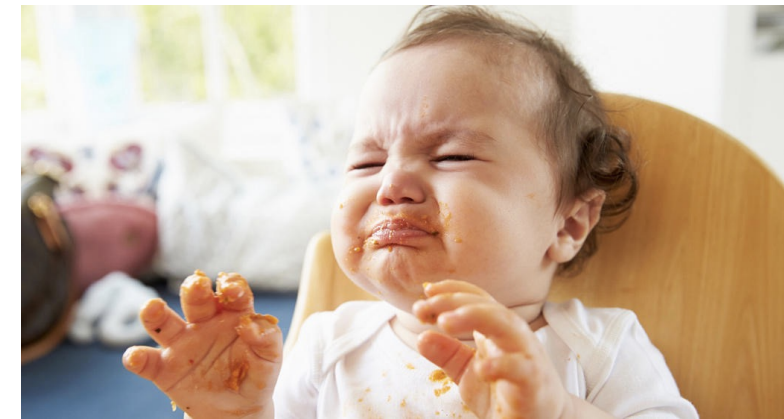
<https://www.clipart.email/clipart/dont-touch-hot-stove-clipart-73647.html>

“loud”



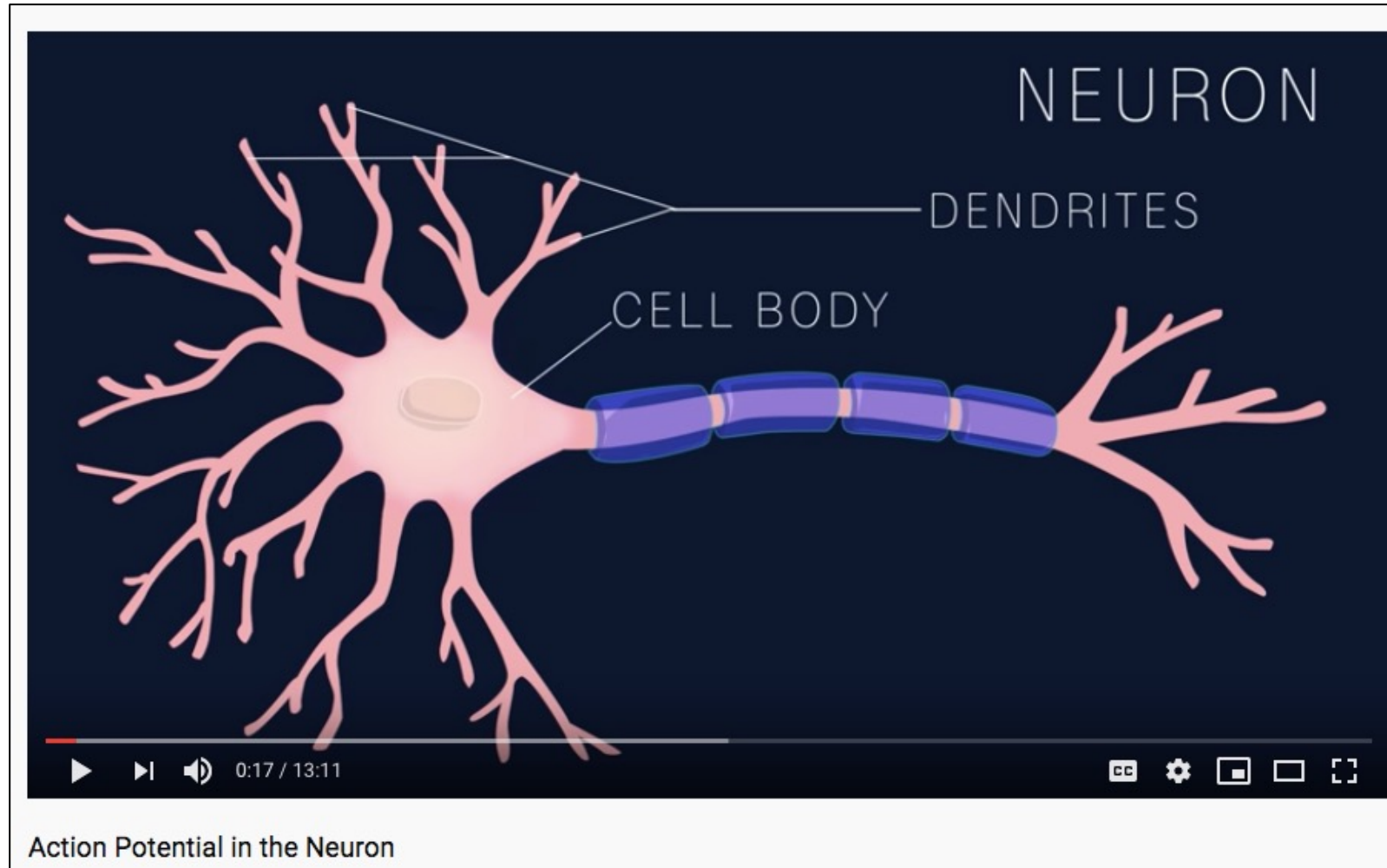
<https://kisselpaso.com/if-the-sun-city-music-fest-gets-too-loud-there-is-a-phone-number-you-can-call-to-complain/>

“spicy”



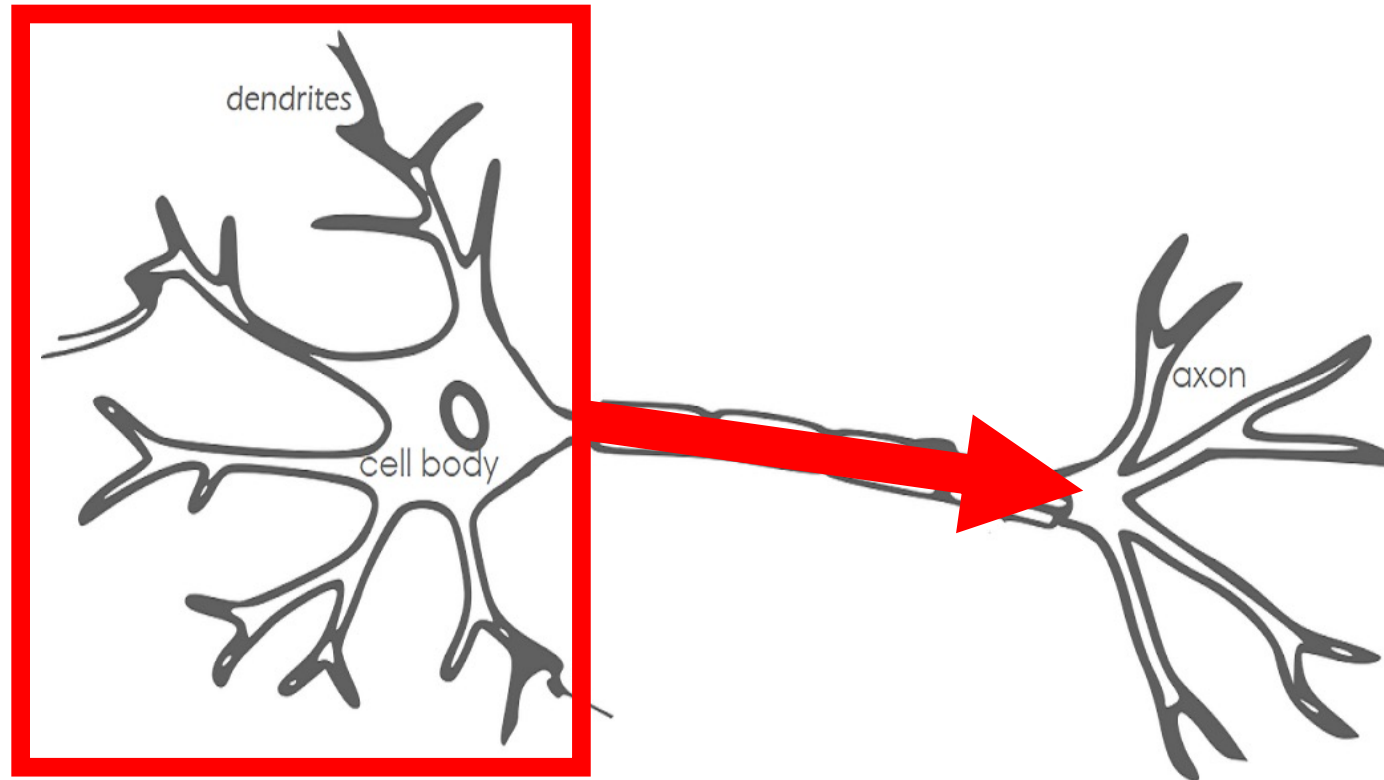
https://www.babycenter.com/404_when-can-my-baby-eat-spicy-foods_1368539.bc

Inspiration: Animal's Computing Machinery



Demo (0-1:14): <https://www.youtube.com/watch?v=oa6rvUJlg7o>

Inspiration: Neuron “Firing”



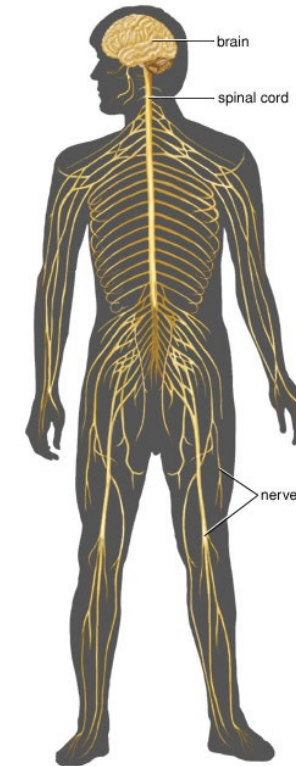
- When the input signals exceed a certain threshold within a short period of time, a neuron “fires”
- Neuron “firing” (outputs signal) is an “all-or-none” process

Inspiration: Animal's Computing Machinery



<https://en.wikipedia.org/wiki/Nematode#/media/File:CelegansGoldsteinLabUNC.jpg>

Nematode worm: 302 neurons



© 2006 Encyclopedia Britannica, Inc.

<https://www.britannica.com/science/human-nervous-system>

Human: ~100,000,000,000 neurons

Today's Topics

- Binary classification applications
- Evaluating classification models
- Biological neurons: inspiration
- **Artificial neuron: Perceptron**

Historical Context: Artificial Neurons

First mathematical
model of neuron

1943

1945

1950

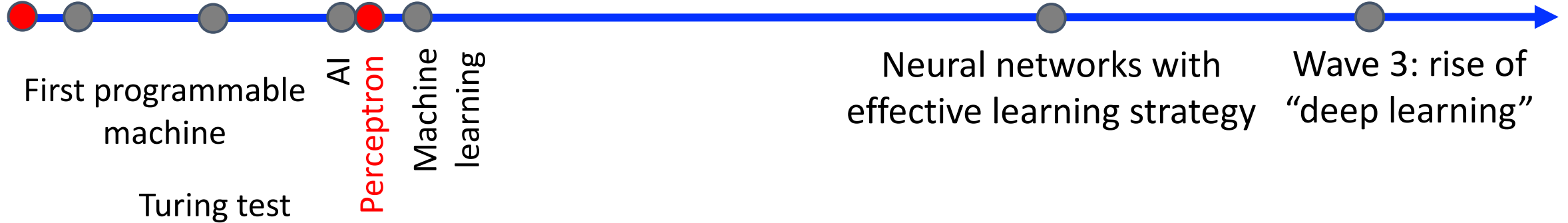
1956

1957

1959

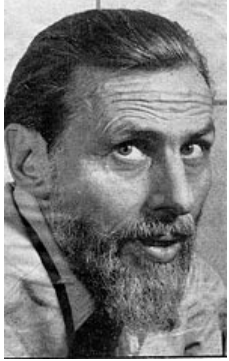
1986

2012



Recall: modern deep learning algorithms rely on techniques developed over the past 65 years.

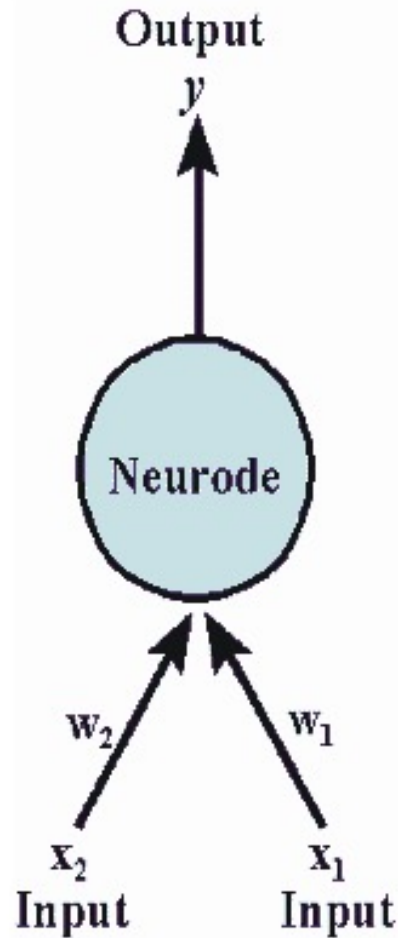
Artificial Neuron: McCulloch-Pitts Neuron



Warren McCulloch
(Neurophysiologist)



Walter Pitts
(Mathematician)



Note: NO learning involved

- weights (W) and threshold (T) values are fixed
- inputs and weights can be 0 or 1
- fires when combined input exceeds threshold

https://en.wikipedia.org/wiki/Walter_Pitts

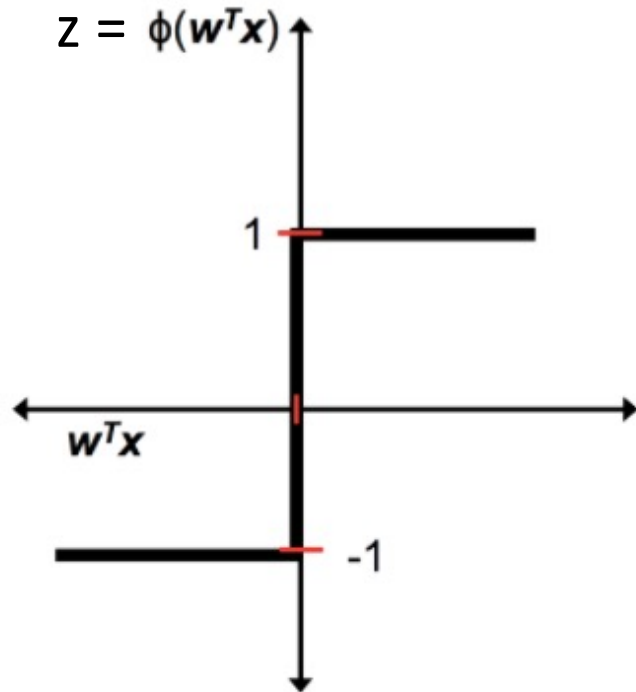
http://web.csulb.edu/~cwallis/artificialn/warren_mcculloch.html

Figure Source: <https://web.csulb.edu/~cwallis/artificialn/History.htm>

Warren McCulloch and Walter Pitts, A Logical Calculus of Ideas Immanent in Nervous Activity, 1943

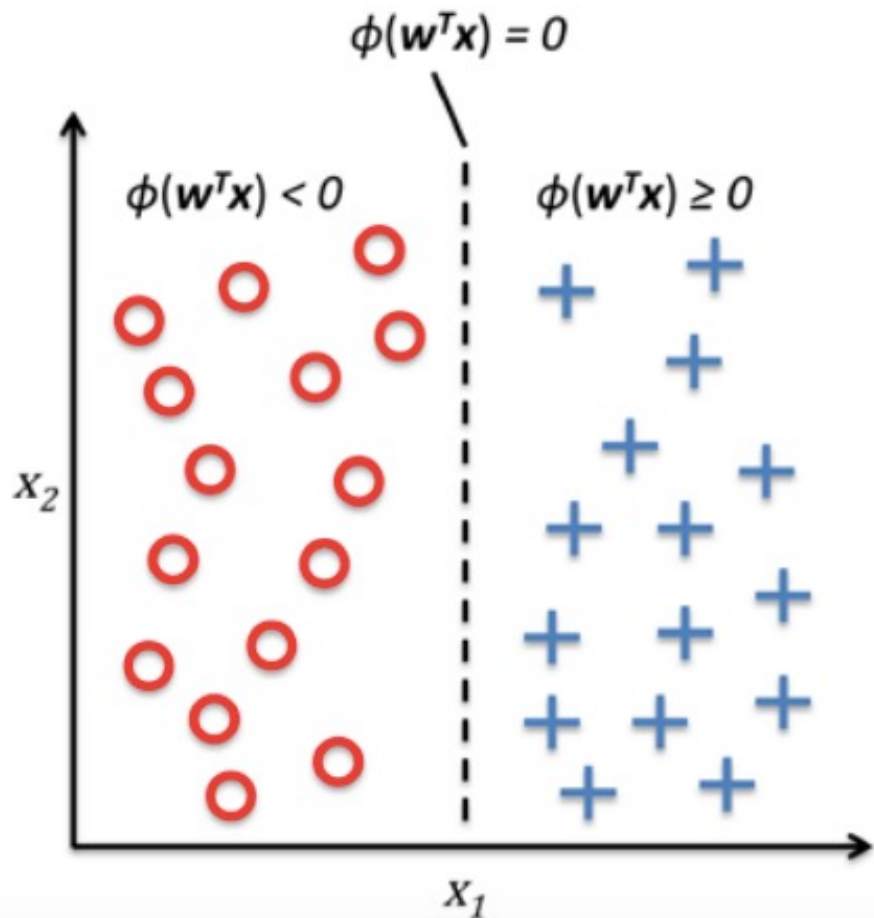
Artificial Neuron: McCulloch-Pitts Neuron

- Mathematical definition: $z = \begin{cases} 1 & \text{if } z \geq \theta \\ -1 & \text{otherwise} \end{cases}$
 - “fire” or “do not fire”
 - mimics human brain

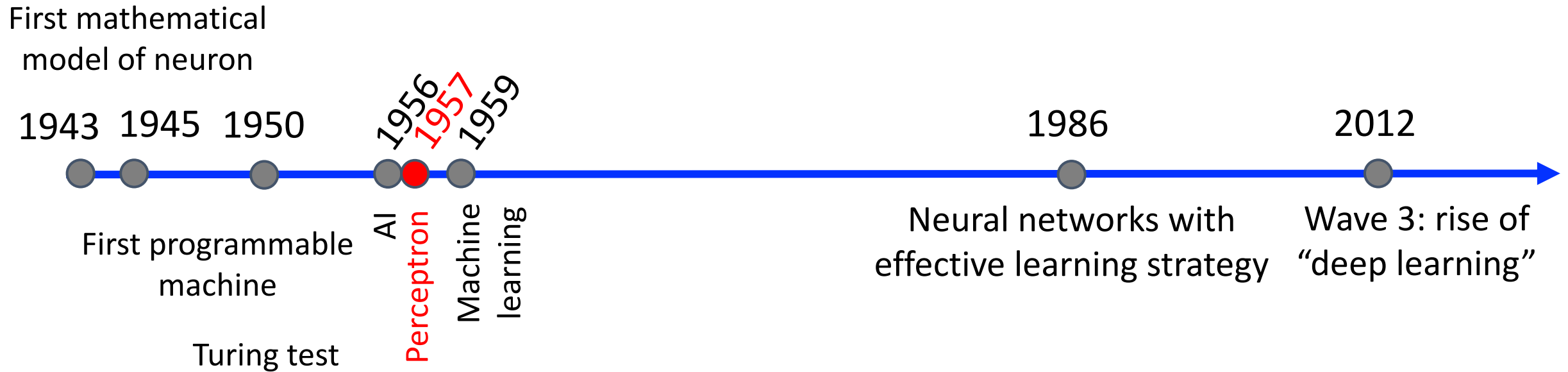


Artificial Neuron: McCulloch-Pitts Neuron

- Mathematical definition: $z = \begin{cases} 1 & \text{if } z \geq \theta \\ -1 & \text{otherwise} \end{cases}$



Historical Context: Artificial Neurons



Recall: modern deep learning algorithms rely on techniques developed over the past 65 years.

Perceptron: Innovator and Vision



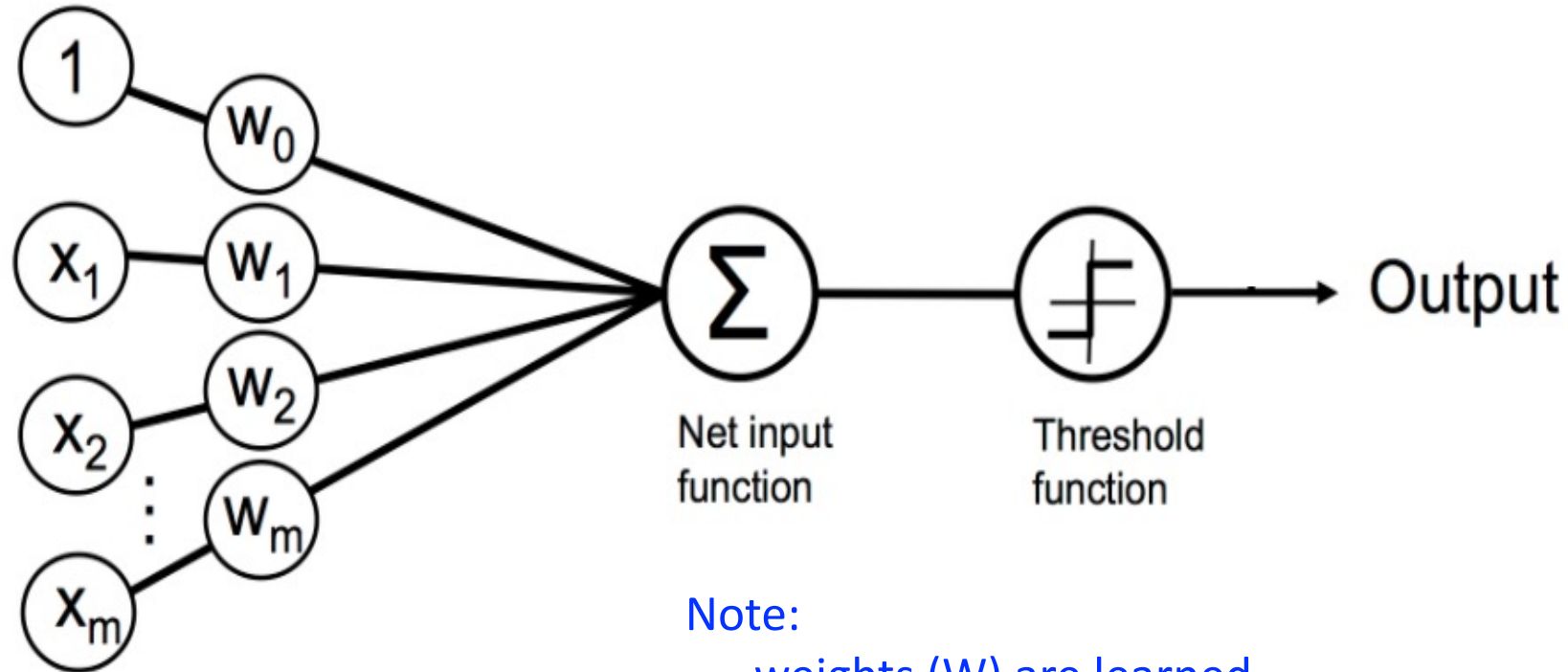
Frank Rosenblatt
(Psychologist)

“[The perceptron is] the embryo of an electronic computer that [the Navy] expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence.... [It] is expected to be finished in about a year at a cost of \$100,000.”

1958 New York Times article: <https://www.nytimes.com/1958/07/08/archives/new-navy-device-learns-by-doing-psychologist-shows-embryo-of.html>

https://en.wikipedia.org/wiki/Frank_Rosenblatt

Perceptron: Model (Linear Threshold Unit)



Note:

- weights (W) are learned
- inputs and weights can be any value
- fires when combined input exceeds threshold

Perceptron: Model (Linear Threshold Unit)

- Fires when a function exceeds threshold:

$$\phi(z) = \begin{cases} 1 & \text{if } z \geq \theta \\ -1 & \text{otherwise} \end{cases}$$

- Rewriting model:

$$\phi(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ -1 & \text{otherwise} \end{cases}$$

- Where:

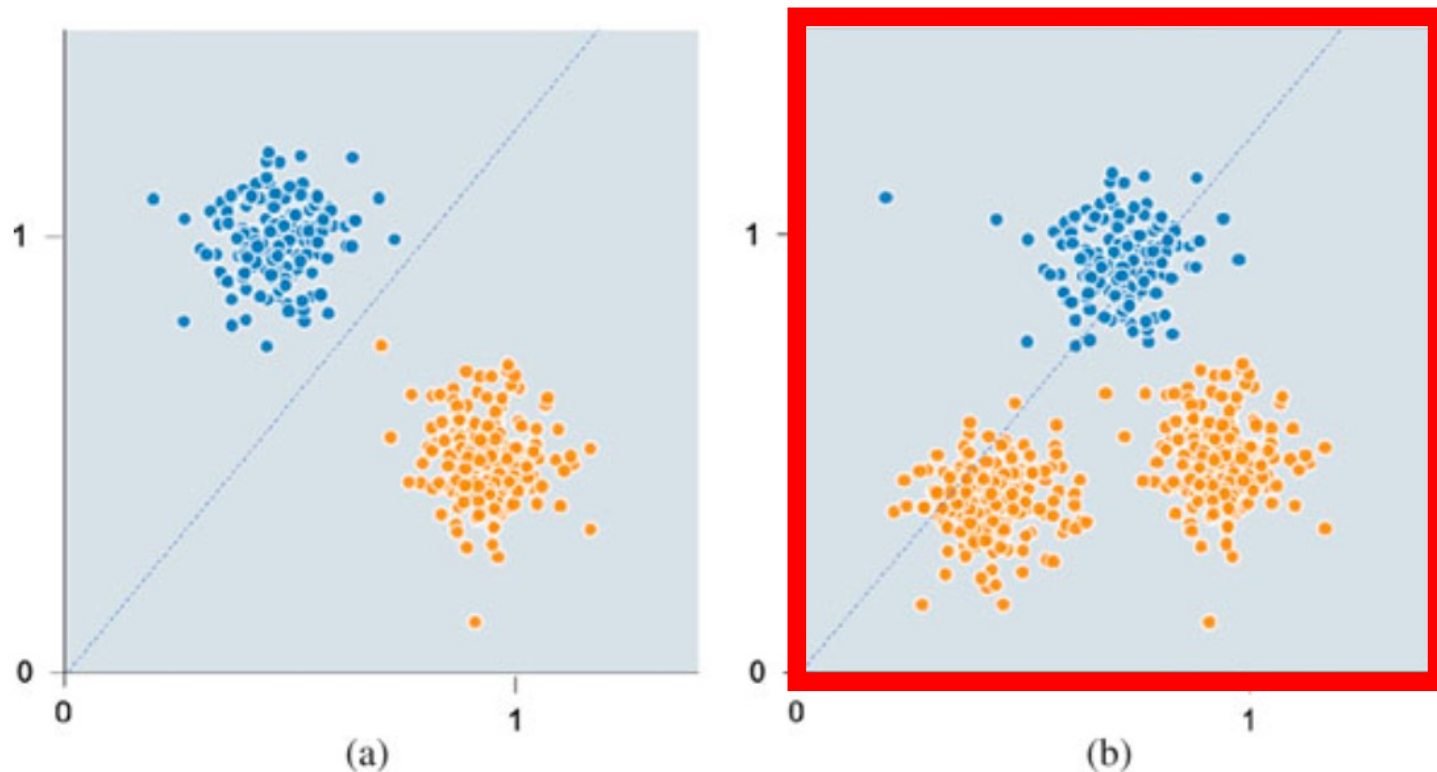
$$z = w_0 x_0 + w_1 x_1 + \dots + w_m x_m = \mathbf{w}^T \mathbf{x}$$

Bias

$$\begin{matrix} \boxed{-\theta} & \boxed{1} \end{matrix}$$

Perceptron: Model (Linear Threshold Unit)

Motivation for bias: without it, model must go through origin



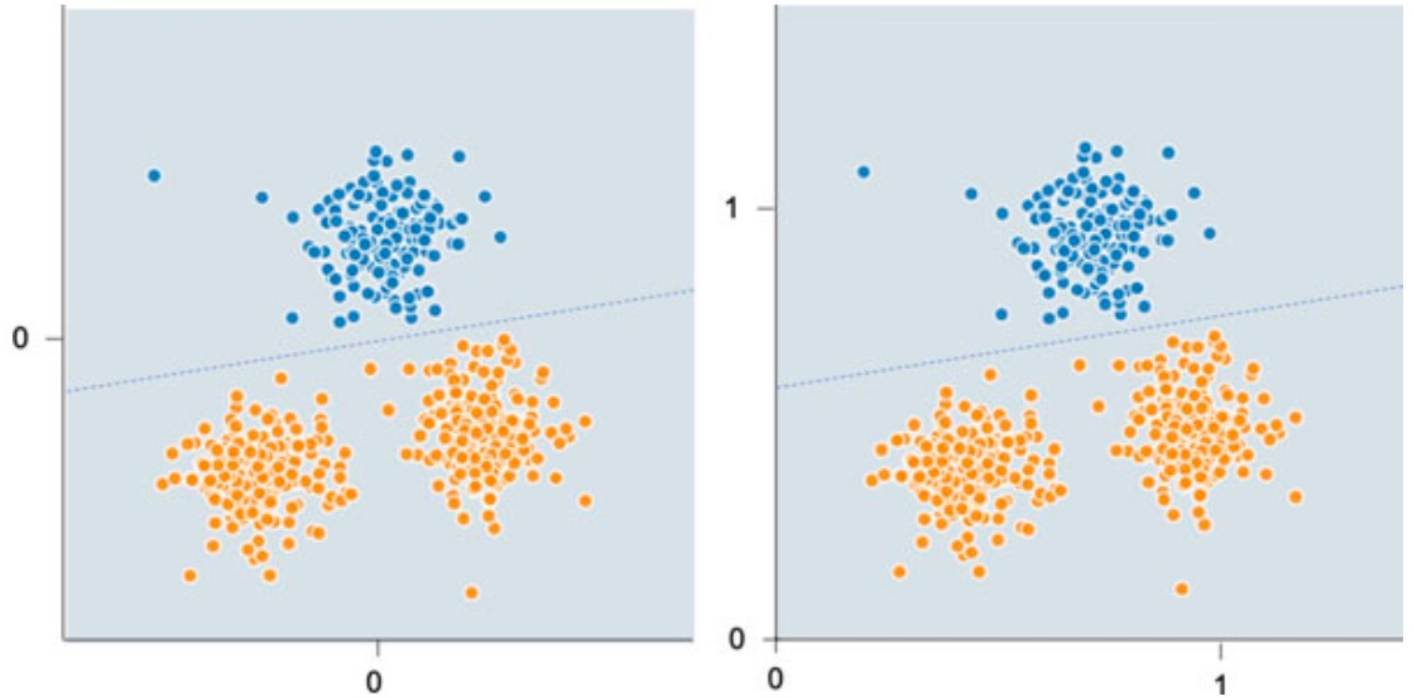
$$z = w_0 x_0 + w_1 x_1 + \dots + w_m x_m = \mathbf{w} \cdot \mathbf{x}$$

Bias

$$\begin{matrix} \boxed{-\theta} & \boxed{1} \end{matrix}$$

Perceptron: Model (Linear Threshold Unit)

Motivation for bias: with it, model does not have to go through origin

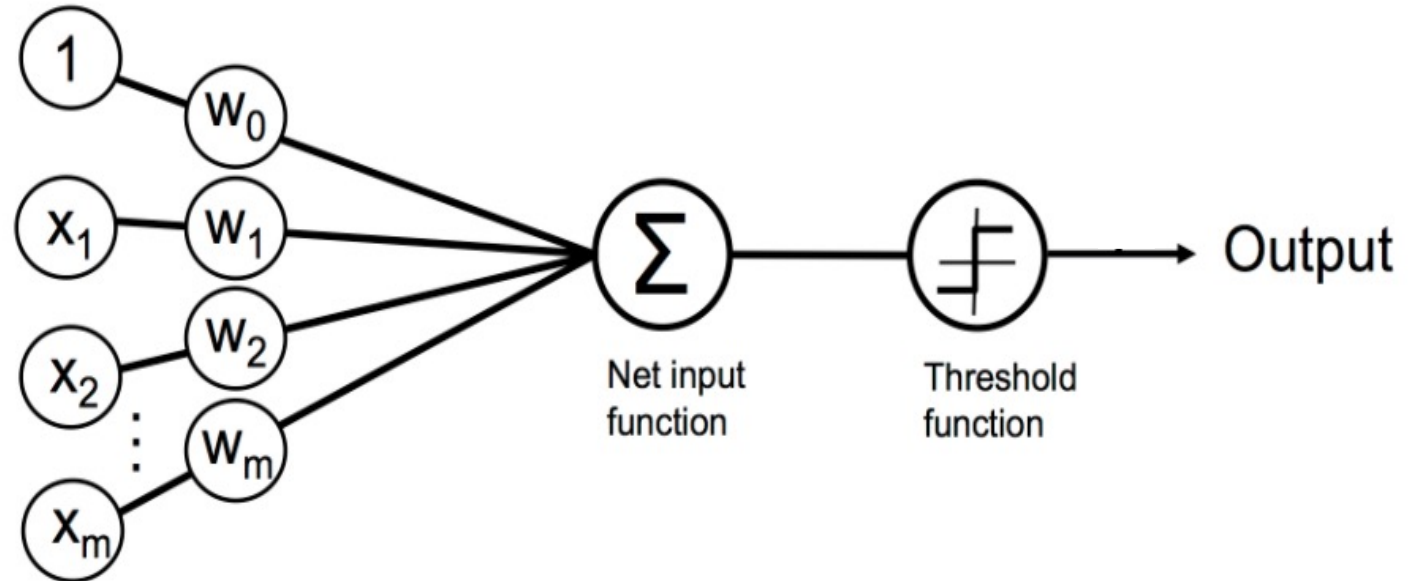


$$z = w_0 x_0 + w_1 x_1 + \dots + w_m x_m = \mathbf{w}^T \mathbf{x}$$

Bias

$$\begin{matrix} \boxed{-\theta} & \boxed{1} \end{matrix}$$

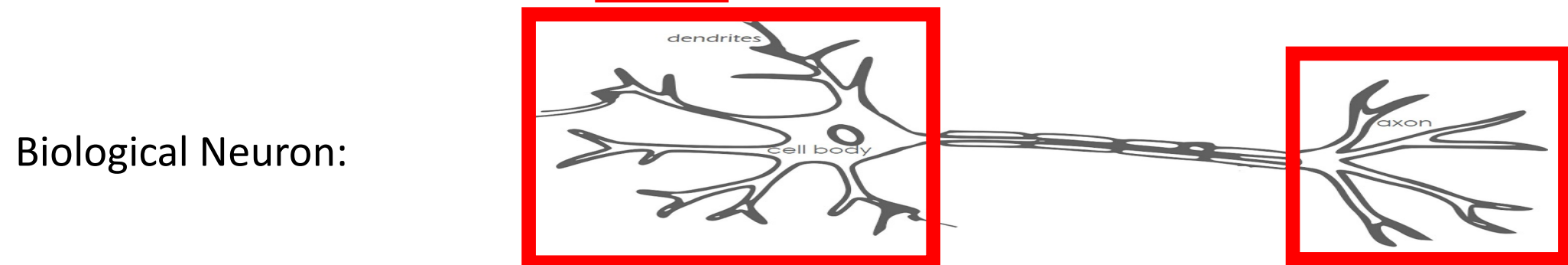
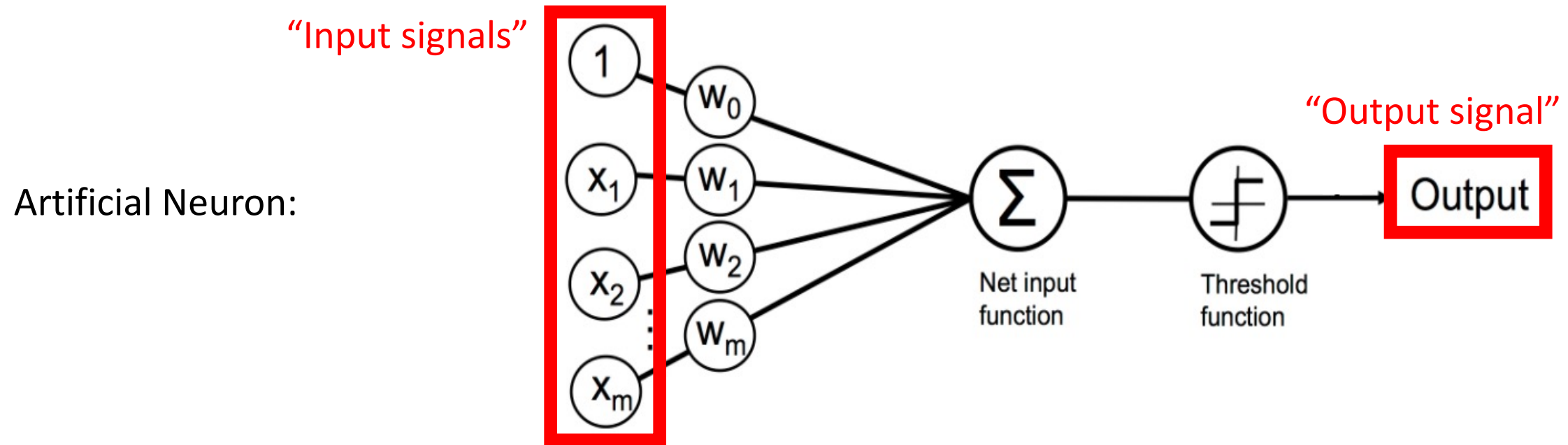
Perceptron: Model (Linear Threshold Unit)



$$\phi(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ -1 & \text{otherwise} \end{cases}$$

$$z = w_0x_0 + w_1x_1 + \dots + w_mx_m = \mathbf{w}^T \mathbf{x}$$

Perceptron: Model (Linear Threshold Unit)

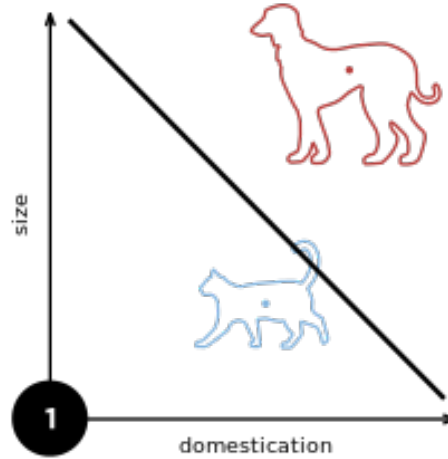


Python Machine Learning; Raschka & Mirjalili

Image Source: <https://becominghuman.ai/introduction-to-neural-networks-bd042ebf2653>

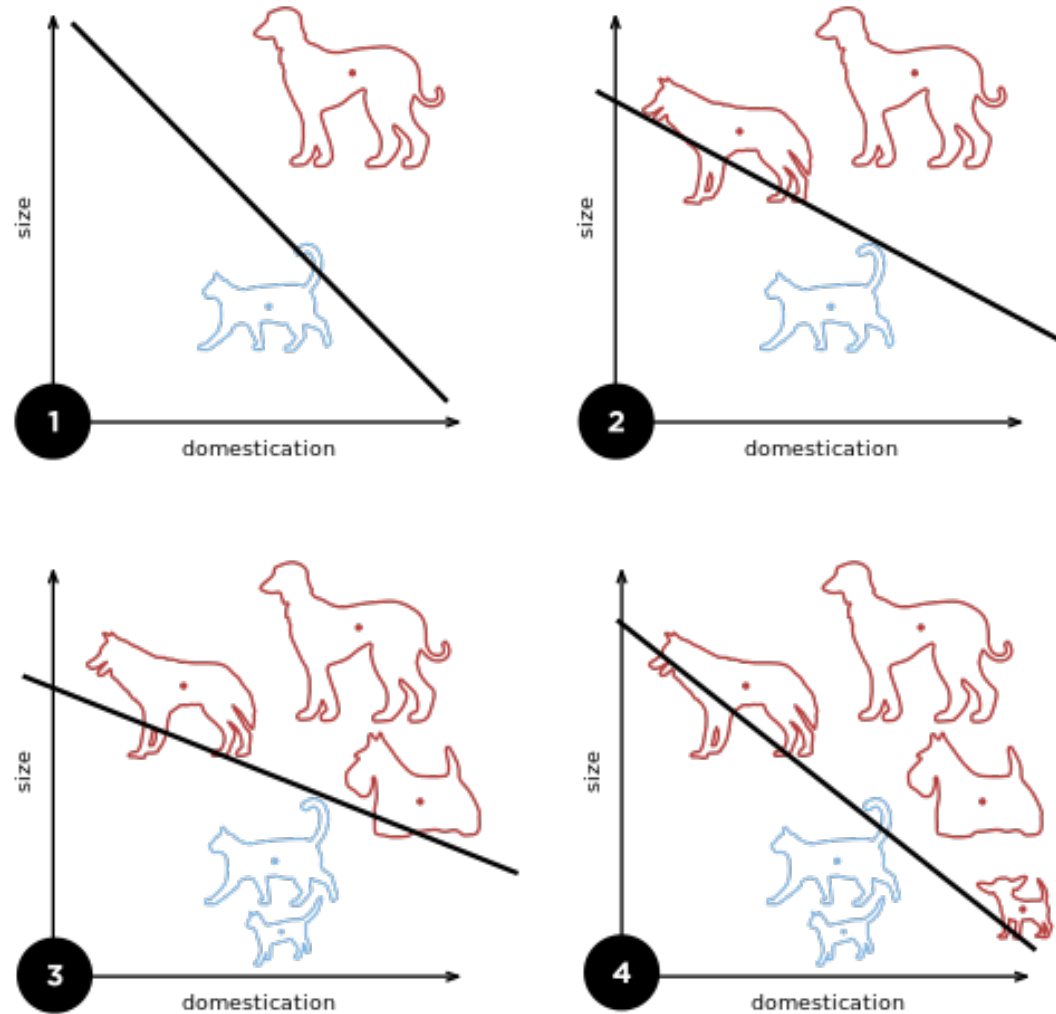
Perceptron: Learning Algorithm

Iteratively update linear boundary with observation of each additional example:



Perceptron: Learning Algorithm

Iteratively update linear boundary with observation of each additional example:



Perceptron: Learning Algorithm

1. Initialize weights to 0 or small random numbers
2. For each training sample:

1. Compute output value: $\sum_{j=0}^m \mathbf{x}_j \mathbf{w}_j = \mathbf{w}^T \mathbf{x}$

2. Update weights with the following definition: $w_j := w_j + \Delta w_j$

$$\Delta w_j = \eta (\text{target}^{(i)} - \text{output}^{(i)}) x_j^{(i)}$$

Learning Rate

True Class Label

Predicted Class Label

Perceptron: Learning Algorithm - What Happens to Weights When It Predicts Correct Class Label?

1. Initialize weights to 0 or small random numbers
2. For each training sample:

1. Compute output value: $\sum_{j=0}^m \mathbf{x}_j \mathbf{w}_j = \mathbf{w}^T \mathbf{x}$

2. Update weights with the following definition: $w_j := w_j + \Delta w_j$

equals 0, so no weight update

$$\Delta w_j = \eta (\text{target}^{(i)} - \text{output}^{(i)}) x_j^{(i)}$$

Learning Rate

True Class Label

Predicted Class Label

Perceptron: Learning Algorithm - What Happens to Weights When It Predicts Wrong Class Label?

1. Initialize weights to 0 or small random numbers
2. For each training sample:

1. Compute output value: $\sum_{j=0}^m \mathbf{x}_j \mathbf{w}_j = \mathbf{w}^T \mathbf{x}$

2. Update weights with the following definition: $w_j := w_j + \Delta w_j$

Equals positive or negative value, so weights change

$$\Delta w_j = \eta (\text{target}^{(i)} - \text{output}^{(i)}) x_j^{(i)}$$

Learning Rate

True Class Label

Predicted Class Label

Perceptron: Learning Algorithm (e.g., 2D dataset)

1. Initialize weights to 0 or small random numbers
2. For each training sample:

1. Compute output value: $\sum_{j=0}^m \mathbf{x}_j \mathbf{w}_j = \mathbf{w}^T \mathbf{x}$

2. Update weights with the following definition: $\mathbf{w}_j := \mathbf{w}_j + \Delta \mathbf{w}_j$

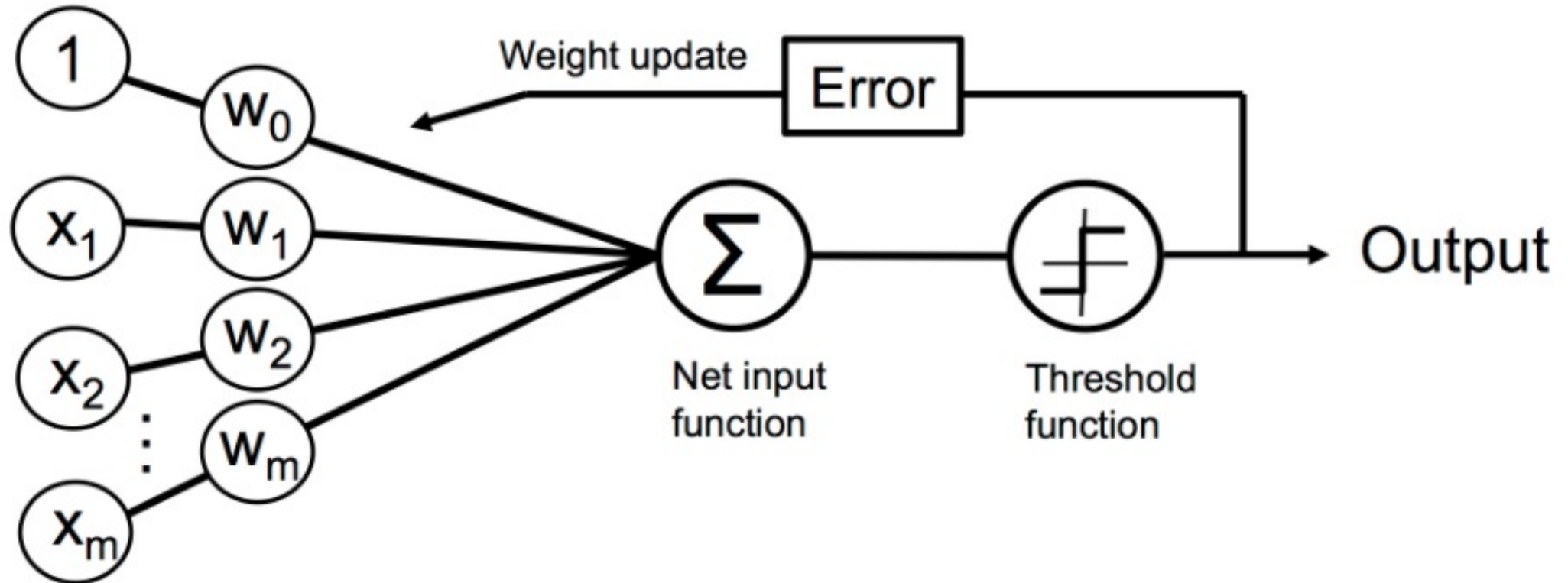
$$\Delta w_0 = \eta (\text{target}^{(i)} - \text{output}^{(i)})$$

$$\Delta w_1 = \eta (\text{target}^{(i)} - \text{output}^{(i)}) x_1^{(i)}$$

$$\Delta w_2 = \eta (\text{target}^{(i)} - \text{output}^{(i)}) x_2^{(i)}$$

All weights updated
simultaneously

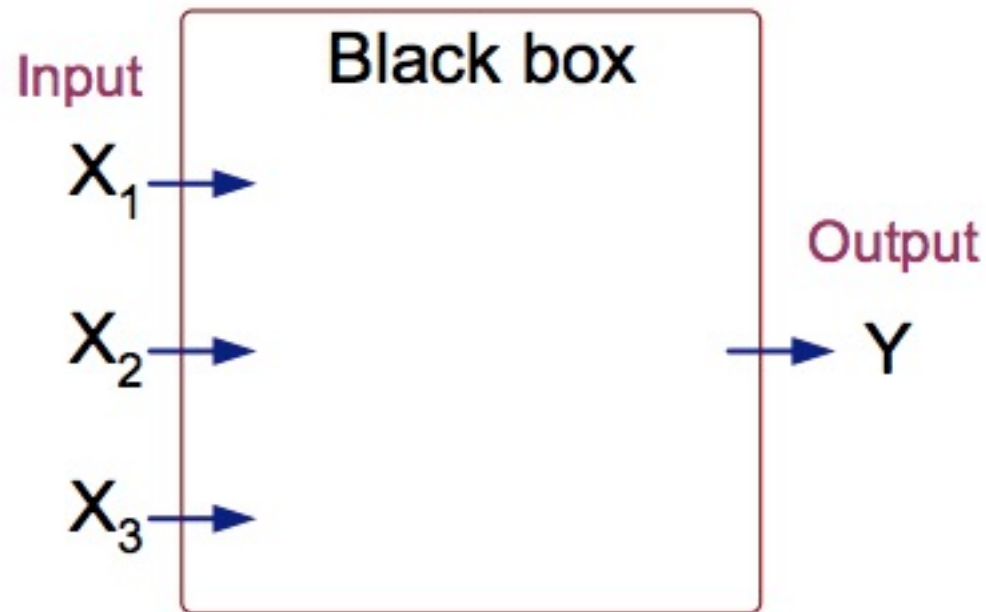
Perceptron: Learning Algorithm



Perceptron: Learning Algorithm Example

- True Model: Y is 1 if at least two of the three inputs are equal to 1.

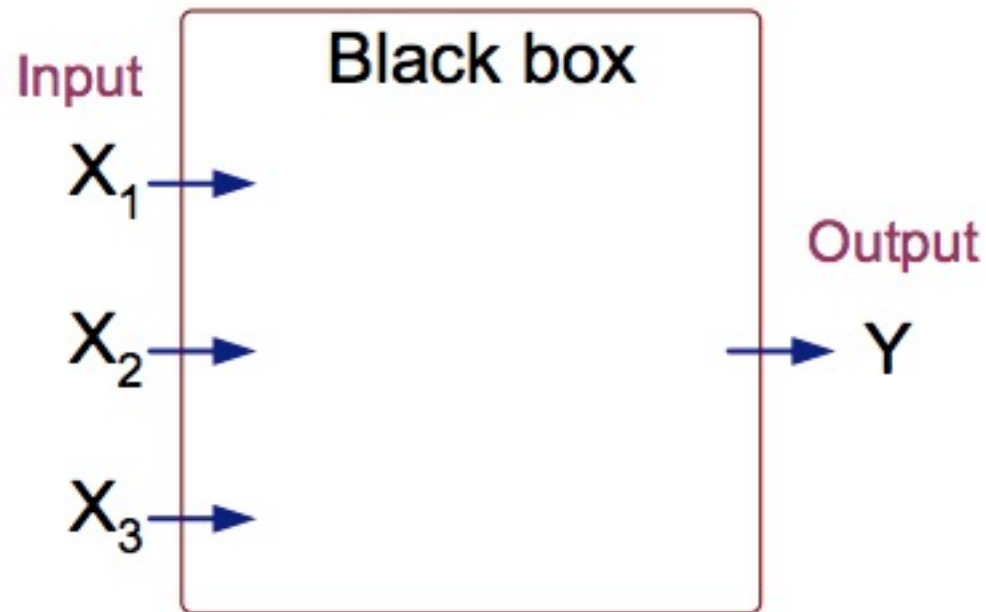
| X_1 | X_2 | X_3 | Y |
|-------|-------|-------|-----|
| 1 | 0 | 0 | |
| 1 | 0 | 1 | |
| 1 | 1 | 0 | |
| 1 | 1 | 1 | ? |
| 0 | 0 | 1 | |
| 0 | 1 | 0 | |
| 0 | 1 | 1 | |
| 0 | 0 | 0 | |



Perceptron: Learning Algorithm Example

- True Model: Y is 1 if at least two of the three inputs are equal to 1.

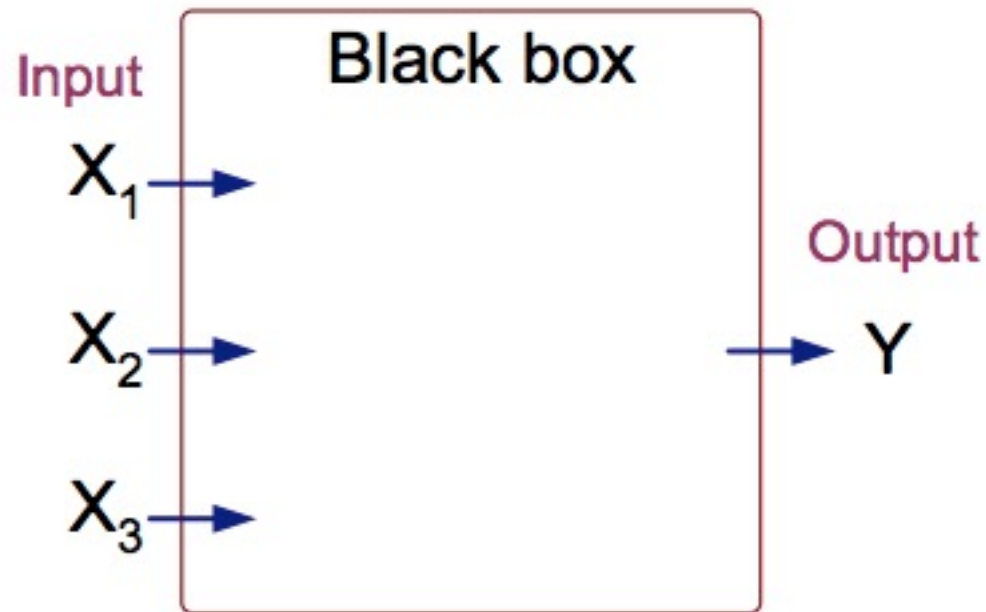
| X_1 | X_2 | X_3 | Y |
|-------|-------|-------|-----|
| 1 | 0 | 0 | -1 |
| 1 | 0 | 1 | |
| 1 | 1 | 0 | |
| 1 | 1 | 1 | ? |
| 0 | 0 | 1 | |
| 0 | 1 | 0 | |
| 0 | 1 | 1 | |
| 0 | 0 | 0 | |



Perceptron: Learning Algorithm Example

- True Model: Y is 1 if at least two of the three inputs are equal to 1.

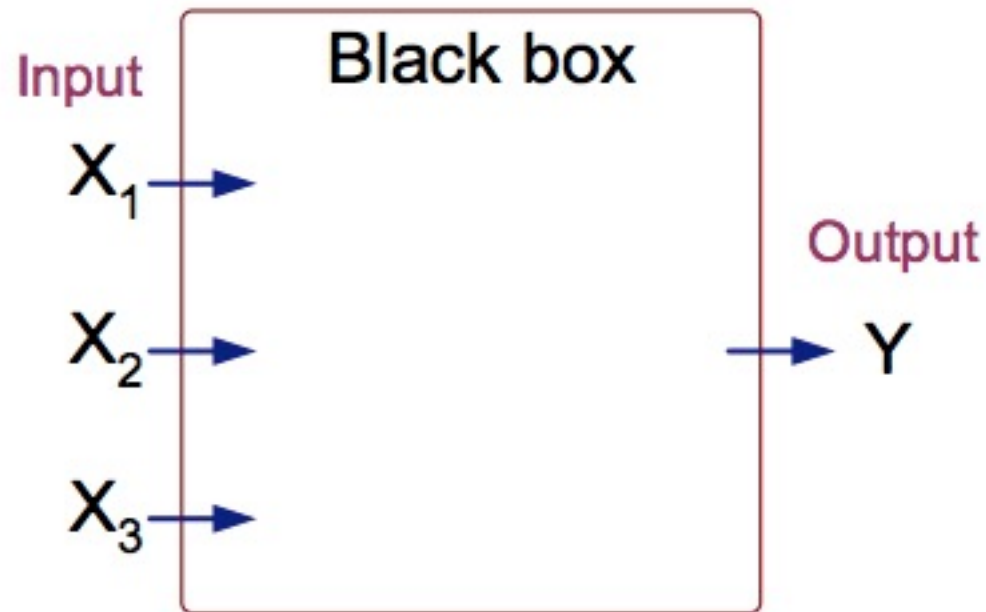
| X_1 | X_2 | X_3 | Y |
|-------|-------|-------|-----|
| 1 | 0 | 0 | -1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | ? |
| 1 | 1 | 1 | |
| 0 | 0 | 1 | |
| 0 | 1 | 0 | |
| 0 | 1 | 1 | |
| 0 | 0 | 0 | |
| 0 | 0 | 0 | |



Perceptron: Learning Algorithm Example

- True Model: Y is 1 if at least two of the three inputs are equal to 1.

| X_1 | X_2 | X_3 | Y |
|-------|-------|-------|-----|
| 1 | 0 | 0 | -1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | -1 |
| 0 | 1 | 0 | -1 |
| 0 | 1 | 1 | 1 |
| 0 | 0 | 0 | -1 |



Perceptron: Learning Algorithm Example

- First Sample

- Compute output value: $\sum_{j=0}^m x_j w_j = \mathbf{w}^T \mathbf{x}$; $\phi(\mathbf{w}^T \mathbf{x}) = \begin{cases} 1 & \text{if } \phi(\mathbf{w}^T \mathbf{x}) \geq 0 \\ -1 & \text{otherwise} \end{cases}$

| X_1 | X_2 | X_3 | Y | Predicted | w_0 | w_1 | w_2 | w_3 |
|-------|-------|-------|-----|-----------|-------|-------|-------|-------|
| 1 | 0 | 0 | -1 | ? | 0 | 0 | 0 | 0 |

Perceptron: Learning Algorithm Example

- First Sample

- Compute output value: $\sum_{j=0}^m x_j w_j = \mathbf{w}^T \mathbf{x}$; $\phi(\mathbf{w}^T \mathbf{x}) = \begin{cases} 1 & \text{if } \phi(\mathbf{w}^T \mathbf{x}) \geq 0 \\ -1 & \text{otherwise} \end{cases}$

| X_1 | X_2 | X_3 | Y |
|-------|-------|-------|-----|
| 1 | 0 | 0 | -1 |

| Predicted |
|-----------|
| 1 |

| w_0 | w_1 | w_2 | w_3 |
|-------|-------|-------|-------|
| 0 | 0 | 0 | 0 |

Perceptron: Learning Algorithm Example

- First Sample

- Update weights: $w_j = w_j + \eta (\text{target}^{(i)} - \text{output}^{(i)}) x_j^{(i)}$; learning rate = 0.1

| X_1 | X_2 | X_3 | Y |
|-------|-------|-------|-----|
| 1 | 0 | 0 | -1 |

| Predicted |
|-----------|
| 1 |

| w_0 | w_1 | w_2 | w_3 |
|-------|-------|-------|-------|
| 0 | 0 | 0 | 0 |
| ? | ? | ? | ? |

$$\Delta w_0 = \eta (\text{target}^{(i)} - \text{output}^{(i)})$$

$$\Delta w_1 = \eta (\text{target}^{(i)} - \text{output}^{(i)}) x_1^{(i)}$$

$$\Delta w_2 = \eta (\text{target}^{(i)} - \text{output}^{(i)}) x_2^{(i)}$$

$$\Delta w_3 = \eta (\text{target}^{(i)} - \text{output}^{(i)}) x_3^{(i)}$$

Perceptron: Learning Algorithm Example

- First Sample

- Update weights: $w_j = w_j + \eta (\text{target}^{(i)} - \text{output}^{(i)}) x_j^{(i)}$; learning rate = 0.1

| X_1 | X_2 | X_3 | Y |
|-------|-------|-------|-----|
| 1 | 0 | 0 | -1 |

| Predicted |
|-----------|
| 1 |

| w_0 | w_1 | w_2 | w_3 |
|-------|-------|-------|-------|
| 0 | 0 | 0 | 0 |
| ? | ? | ? | ? |

$$\Delta w_0 = 0.1(-1-1)*1 = -0.2$$

$$\Delta w_1 = 0.1(-1-1)*1 = -0.2$$

$$\Delta w_2 = 0.1(-1-1)*0 = 0$$

$$\Delta w_3 = 0.1(-1-1)*0 = 0$$

Perceptron: Learning Algorithm Example

- Second Sample

- Compute output value: $\sum_{j=0}^m x_j w_j = \mathbf{w}^T \mathbf{x}$; $\phi(\mathbf{w}^T \mathbf{x}) = \begin{cases} 1 & \text{if } \phi(\mathbf{w}^T \mathbf{x}) \geq 0 \\ -1 & \text{otherwise} \end{cases}$

| X_1 | X_2 | X_3 | Y |
|-------|-------|-------|-----|
| 1 | 0 | 0 | -1 |
| 1 | 0 | 1 | 1 |

| Predicted |
|-----------|
| 1 |
| ? |

| w_0 | w_1 | w_2 | w_3 |
|-------|-------|-------|-------|
| 0 | 0 | 0 | 0 |
| -0.2 | -0.2 | 0 | 0 |

Perceptron: Learning Algorithm Example

- Second Sample

- Compute output value: $\sum_{j=0}^m x_j w_j = \mathbf{w}^T \mathbf{x}$; $\phi(\mathbf{w}^T \mathbf{x}) = \begin{cases} 1 & \text{if } \phi(\mathbf{w}^T \mathbf{x}) \geq 0 \\ -1 & \text{otherwise} \end{cases}$

| X_1 | X_2 | X_3 | Y |
|-------|-------|-------|-----|
| 1 | 0 | 0 | -1 |
| 1 | 0 | 1 | 1 |

| Predicted |
|-----------|
| 1 |
| -1 |

| w_0 | w_1 | w_2 | w_3 |
|-------|-------|-------|-------|
| 0 | 0 | 0 | 0 |
| -0.2 | -0.2 | 0 | 0 |

Perceptron: Learning Algorithm Example

- Second Sample

- Update weights: $w_j = w_j + \eta (\text{target}^{(i)} - \text{output}^{(i)}) x_j^{(i)}$; learning rate = 0.1

| X_1 | X_2 | X_3 | Y |
|-------|-------|-------|-----|
| 1 | 0 | 0 | -1 |
| 1 | 0 | 1 | 1 |

| Predicted |
|-----------|
| 1 |
| -1 |

| w_0 | w_1 | w_2 | w_3 |
|-------|-------|-------|-------|
| 0 | 0 | 0 | 0 |
| -0.2 | -0.2 | 0 | 0 |
| ? | ? | ? | ? |

$$\Delta w_0 = \eta (\text{target}^{(i)} - \text{output}^{(i)})$$

$$\Delta w_1 = \eta (\text{target}^{(i)} - \text{output}^{(i)}) x_1^{(i)}$$

$$\Delta w_2 = \eta (\text{target}^{(i)} - \text{output}^{(i)}) x_2^{(i)}$$

$$\Delta w_3 = \eta (\text{target}^{(i)} - \text{output}^{(i)}) x_3^{(i)}$$

Perceptron: Learning Algorithm Example

- Second Sample

- Update weights: $w_j = w_j + \eta (\text{target}^{(i)} - \text{output}^{(i)}) x_j^{(i)}$; learning rate = 0.1

| X_1 | X_2 | X_3 | Y |
|-------|-------|-------|-----|
| 1 | 0 | 0 | -1 |
| 1 | 0 | 1 | 1 |

| Predicted |
|-----------|
| 1 |
| -1 |

| w_0 | w_1 | w_2 | w_3 |
|-------|-------|-------|-------|
| 0 | 0 | 0 | 0 |
| -0.2 | -0.2 | 0 | 0 |
| ? | ? | ? | ? |

$$\Delta w_0 = 0.1(1 - -1) * 1 = 0.2$$

$$\Delta w_1 = 0.1(1 - -1) * 1 = 0.2$$

$$\Delta w_2 = 0.1(1 - -1) * 0 = 0$$

$$\Delta w_3 = 0.1(1 - -1) * 1 = 0.2$$

Perceptron: Learning Algorithm Example

- One Epoch (All Examples)

- $w_j = w_j + \eta (\text{target}^{(i)} - \text{output}^{(i)}) x_j^{(i)}$; learning rate = 0.1

| X_1 | X_2 | X_3 | Y |
|-------|-------|-------|-----|
| 1 | 0 | 0 | -1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | -1 |
| 0 | 1 | 0 | -1 |
| 0 | 1 | 1 | 1 |
| 0 | 0 | 0 | -1 |

| | w_0 | w_1 | w_2 | w_3 |
|---|-------|-------|-------|-------|
| 0 | 0 | 0 | 0 | 0 |
| 1 | -0.2 | -0.2 | 0 | 0 |
| 2 | 0 | 0 | 0 | 0.2 |
| 3 | 0 | 0 | 0 | 0.2 |
| 4 | 0 | 0 | 0 | 0.2 |
| 5 | -0.2 | 0 | 0 | 0 |
| 6 | -0.2 | 0 | 0 | 0 |
| 7 | 0 | 0 | 0.2 | 0.2 |
| 8 | -0.2 | 0 | 0.2 | 0.2 |

Perceptron: Learning Algorithm Example

- Six Epochs (All Examples)

- $w_j = w_j + \eta (\text{target}^{(i)} - \text{output}^{(i)}) x_j^{(i)}$; learning rate = 0.1

| X_1 | X_2 | X_3 | Y |
|-------|-------|-------|-----|
| 1 | 0 | 0 | -1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | -1 |
| 0 | 1 | 0 | -1 |
| 0 | 1 | 1 | 1 |
| 0 | 0 | 0 | -1 |

| Epoch | w_0 | w_1 | w_2 | w_3 |
|-------|-------|-------|-------|-------|
| 0 | 0 | 0 | 0 | 0 |

Perceptron: Learning Algorithm Example

- Six Epochs (All Examples)

- $w_j = w_j + \eta (\text{target}^{(i)} - \text{output}^{(i)}) x_j^{(i)}$; learning rate = 0.1

| X_1 | X_2 | X_3 | Y |
|-------|-------|-------|-----|
| 1 | 0 | 0 | -1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | -1 |
| 0 | 1 | 0 | -1 |
| 0 | 1 | 1 | 1 |
| 0 | 0 | 0 | -1 |

| | w_0 | w_1 | w_2 | w_3 |
|---|-------|-------|-------|-------|
| 0 | 0 | 0 | 0 | 0 |
| 1 | -0.2 | -0.2 | 0 | 0 |
| 2 | 0 | 0 | 0 | 0.2 |
| 3 | 0 | 0 | 0 | 0.2 |
| 4 | 0 | 0 | 0 | 0.2 |
| 5 | -0.2 | 0 | 0 | 0 |
| 6 | -0.2 | 0 | 0 | 0 |
| 7 | 0 | 0 | 0.2 | 0.2 |
| 8 | -0.2 | 0 | 0.2 | 0.2 |

| Epoch | w_0 | w_1 | w_2 | w_3 |
|-------|-------|-------|-------|-------|
| 0 | 0 | 0 | 0 | 0 |
| 1 | -0.2 | 0 | 0.2 | 0.2 |
| 2 | -0.2 | 0 | 0.4 | 0.2 |
| 3 | -0.4 | 0 | 0.4 | 0.2 |
| 4 | -0.4 | 0.2 | 0.4 | 0.4 |
| 5 | -0.6 | 0.2 | 0.4 | 0.2 |
| 6 | -0.6 | 0.4 | 0.4 | 0.2 |

Perceptron: Learning Algorithm Choices

- Learning rate
- Number of epochs (passes over the dataset)

*** Note: Kamath textbook offers two representations of this algorithm with “Algorithm 4” (Ch. 2.5) matching that shown in this lecture and Ch. 4 using a 0.5 threshold instead; the threshold depends on the predicted values with 0 for $\{-1, 1\}$ and 0.5 for $\{0, 1\}$.**

Today's Topics

- Binary classification applications
- Evaluating classification models
- Biological neurons: inspiration
- Artificial neuron: Perceptron

A dark gray background with a white film strip border on the left and right sides. The film strip has rectangular sprocket holes. In the center, there is a faint, circular white glow. The text "The End" is written in a white, cursive script font with a slight drop shadow, centered within the glow.

The End

Credits

- Image of Boulder: <http://boulderrunning.com/where2run/five-trails-for-hill-running-and-mountain-training/>
- Stick person figure: <https://drawception.com/game/AsPNcppPND/draw-yourself-blindfolded-pio/>
- Figure: <https://www.quora.com/What-is-meant-by-gradient-descent-in-laymen-terms>
- Figure and great reference: https://beamandrew.github.io/deeplearning/2017/02/23/deep_learning_101_part1.html