10. Text Classification

LING-351 Language Technology and LLMs

Instructor: Hakyung Sung

September 30, 2025

*Acknowledgment: A part of course slides are based on materials from Dr. Kilho Shin @ Kyocera

Table of contents

Review

Review

- Exploring English corpora

Review

- Exploring English corpora
- Word distributions

Review

- Exploring English corpora
- Word distributions
    - Frequency is inversely proportional to rank (T/F)

Review

- Exploring English corpora
- Word distributions
    - Frequency is inversely proportional to rank (T/F)
    - More frequent words tend to be longer (T/F)

Review

- Exploring English corpora
- Word distributions
    - Frequency is inversely proportional to rank (T/F)
    - More frequent words tend to be longer (T/F)
    - As you read more tokens in a corpus, the rate of new words slows down (T/F)

Review

- Exploring English corpora
- Word distributions
  - Frequency is inversely proportional to rank (T/F)
  - More frequent words tend to be longer (T/F)
  - As you read more tokens in a corpus, the rate of new words slows down (T/F)
  - A topic modeling uses annotated labels to build a statiatical model (T/F)

Review

- Exploring English corpora
- Word distributions
    - Frequency is inversely proportional to rank (T/F)
    - More frequent words tend to be longer (T/F)
    - As you read more tokens in a corpus, the rate of new words slows down (T/F)
    - A topic modeling uses annotated labels to build a statiatical model (T/F)
    - Lemmatization is not necessary for topic modeling (T/F)

Review

Word vectors/embeddings

- Each word is turned into a list of numbers (a vector) (T/F)

Review

Word vectors/embeddings

- Each word is turned into a list of numbers (a vector) (T/F)
- Computers are good at understanding words as themselves, so NO transformation is necessary to build embeddings (T/F)

Review

Word vectors/embeddings

- Each word is turned into a list of numbers (a vector) (T/F)
- Computers are good at understanding words as themselves, so NO transformation is necessary to build embeddings (T/F)
- Algorithms can automatically learn these vectors from corpus data to build a word embedding (T/F)

Review

Word vectors/embeddings

- Each word is turned into a list of numbers (a vector) (T/F)
- Computers are good at understanding words as themselves, so NO transformation is necessary to build embeddings (T/F)
- Algorithms can automatically learn these vectors from corpus data to build a word embedding (T/F)
- king ≈ [0.52, -0.12, 0.34, ...]

Review

Word vectors/embeddings

- Each word is turned into a list of numbers (a vector) (T/F)
- Computers are good at understanding words as themselves, so NO transformation is necessary to build embeddings (T/F)
- Algorithms can automatically learn these vectors from corpus data to build a word embedding (T/F)
- king ≈ [0.52, -0.12, 0.34, ...]
- queen ≈ [0.51, -0.10, 0.36, ...]

Review

Word vectors/embeddings

- Each word is turned into a list of numbers (a vector) (T/F)
- Computers are good at understanding words as themselves, so NO transformation is necessary to build embeddings (T/F)
- Algorithms can automatically learn these vectors from corpus data to build a word embedding (T/F)
- king ≈ [0.52, -0.12, 0.34, ...]
- queen ≈ [0.51, -0.10, 0.36, ...]
- dog ≈ [-0.25, 0.80, -0.33, ...]

Review

Word vectors/embeddings

- Each word is turned into a list of numbers (a vector) (T/F)
- Computers are good at understanding words as themselves, so NO transformation is necessary to build embeddings (T/F)
- Algorithms can automatically learn these vectors from corpus data to build a word embedding (T/F)
- king ≈ [0.52, -0.12, 0.34, ...]
- queen ≈ [0.51, -0.10, 0.36, ...]
- dog ≈ [-0.25, 0.80, -0.33, ...]
- Words that have similar meanings get numbers that are close to each other (T/F)

Review

Word vectors/embeddings

- Each word is turned into a list of numbers (a vector) (T/F)
- Computers are good at understanding words as themselves, so NO transformation is necessary to build embeddings (T/F)
- Algorithms can automatically learn these vectors from corpus data to build a word embedding (T/F)
- king ≈ [0.52, -0.12, 0.34, ...]
- queen ≈ [0.51, -0.10, 0.36, ...]
- dog ≈ [-0.25, 0.80, -0.33, ...]
- Words that have similar meanings get numbers that are close to each other (T/F)
- This is not really helpful for the computer to see that *king* and *queen* are more related (T/F)

Lesson plan

Lesson plan

- ~~Review~~

Lesson plan

- ~~Review~~
- Text classification

Lesson plan

- ~~Review~~
- Text classification
- Artificial neural network

Lesson plan

- ~~Review~~
- Text classification
- Artificial neural network
- Perceptrons

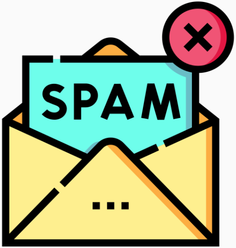
Lesson plan

- ~~Review~~
- Text classification
- Artificial neural network
- Perceptrons
- Multi-layer perceptrons

Text classification

Introduction

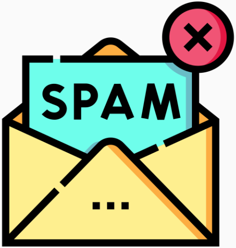
Every day, you probably receive hundreds of spam emails



- Unsolicited advertisements

Introduction

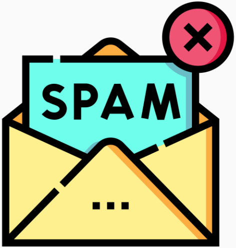
Every day, you probably receive hundreds of spam emails



- Unsolicited advertisements
- Financial scams

Introduction

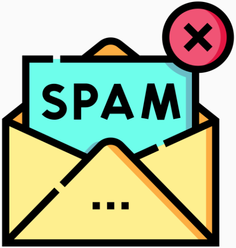
Every day, you probably receive hundreds of spam emails



- Unsolicited advertisements
- Financial scams
- Attempts to steal personal data

Introduction

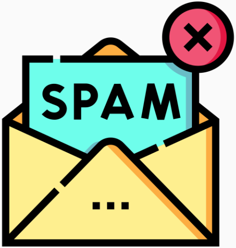
Every day, you probably receive hundreds of spam emails



- Unsolicited advertisements
- Financial scams
- Attempts to steal personal data

Introduction

Every day, you probably receive hundreds of spam emails

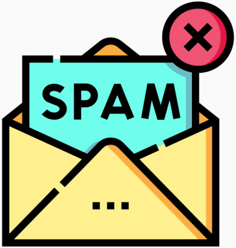


- Unsolicited advertisements
- Financial scams
- Attempts to steal personal data

Your email spam filter is one example of text classification:

- Automatically sorts texts into two or more **classes** (**labels**)

Every day, you probably receive hundreds of spam emails



- Unsolicited advertisements
- Financial scams
- Attempts to steal personal data

Your email spam filter is one example of text classification:

- Automatically sorts texts into two or more **classes** (**labels**)
- Q. *Who gives the labels?*

Other applications of text classification

Beyond spam detection, text classification is widely used:

- Sorting non-spam emails

Other applications of text classification

Beyond spam detection, text classification is widely used:

- Sorting non-spam emails
- Detecting hate speech, fake news

Other applications of text classification

Beyond spam detection, text classification is widely used:

- Sorting non-spam emails
- Detecting hate speech, fake news
- Sentiment analysis of product reviews (positive, negative, neutral)

Other applications of text classification

Beyond spam detection, text classification is widely used:

- Sorting non-spam emails
- Detecting hate speech, fake news
- Sentiment analysis of product reviews (positive, negative, neutral)
- *and what else*?

Brainstorm together

Questions:

- Can you think of one example of text classification used in real life?
- What labels would be needed in this case?
- What kind of corpus/corpora would be required to train such a model?
- What challenges might arise in building this system?

Why text classification matters?

- Assigns a label from a closed class of labels automatically - SAVE my time!

Why text classification matters?

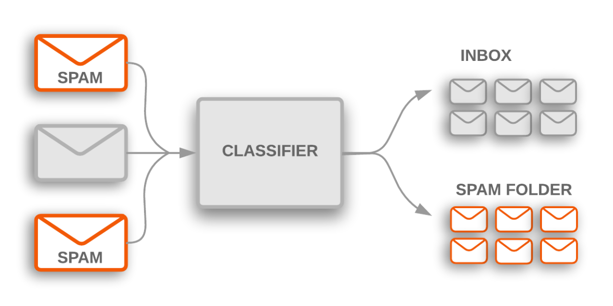
- Assigns a label from a closed class of labels automatically - SAVE my time!
- Any other advantages?

Why text classification matters?

- Assigns a label from a closed class of labels automatically - SAVE my time!
- Any other advantages?
- Any disadvantages?

Example

Let's focus on the spam task



Sourced from *https://developers.google.com/machine-learning/guides/text-classification*

Spam vs. Ham: Human Intuition

- Can you tell at a glance which emails are spam or ham?

Spam vs. Ham: Human Intuition

- Can you tell at a glance which emails are spam or ham?
- What clues did you notice?

Then how computers "Learn"?

- Computers must be trained to recognize spam vs. ham

Then how computers "Learn"?

- Computers must be trained to recognize spam vs. ham
- They extract features from the text (words, patterns, metadata)

Then how computers “Learn”?

- Computers must be trained to recognize spam vs. ham
- They extract features from the text (words, patterns, metadata)
- Learn classification rules from labeled data

Then how computers “Learn”?

- Computers must be trained to recognize spam vs. ham
- They extract features from the text (words, patterns, metadata)
- Learn classification rules from labeled data
- Apply these rules to new, unseen emails

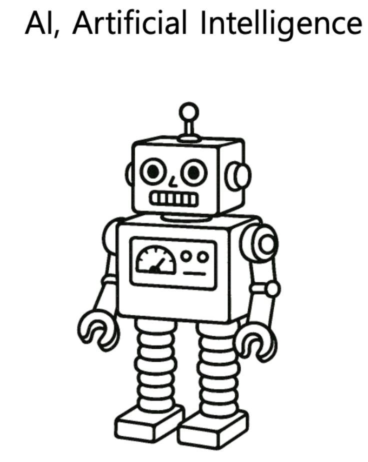
Then how computers “Learn”?

- For the rest of the class, we will explore the foundations of these classification models, focusing on neural networks.

Artificial neural network

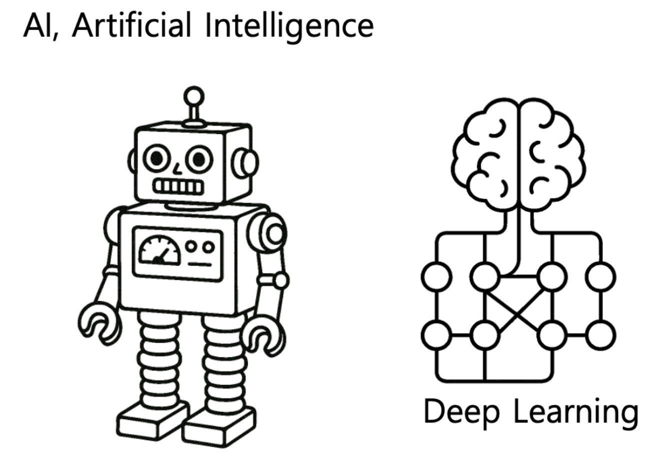
Artificial neural network

Creating machines that can think and communicate like humans has been a long-standing dream of humanity.



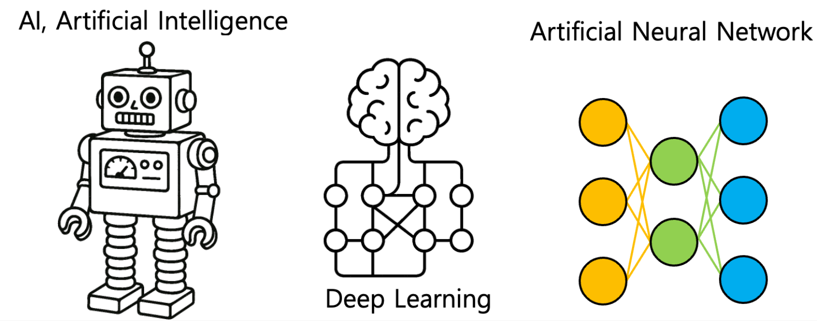
Artificial neural network

Today, artificial intelligence is largely based on machine learning, especially deep learning technologies.



Artificial neural network

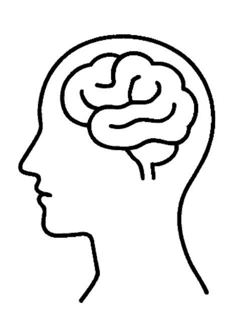
At the foundation of deep learning lies the artificial neural network, which serves as the starting point for understanding deep learning.



NLP: neural networks involve in word embeddings, recurrent neural networks, Transformer models

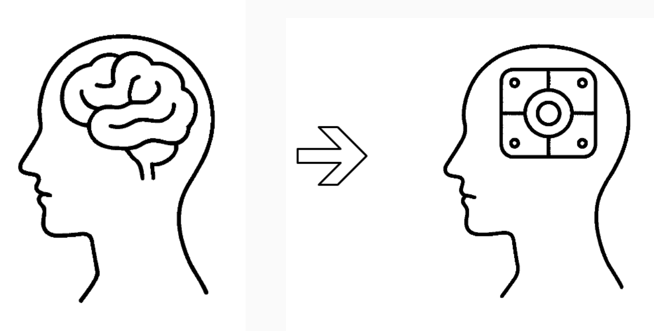
Understanding human brain

Artificial neural networks are computer programs designed to mimic the human brain.



Understanding human brain

Artificial neural networks are computer programs designed to mimic the human brain.



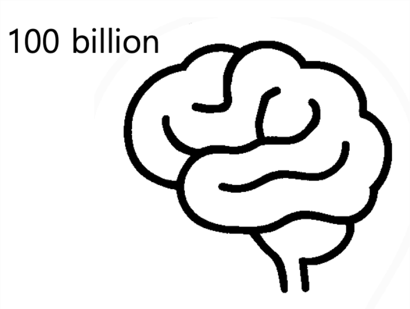
Therefore, understanding how the human brain works is the very first step.

Neuron and artificial neuron

The human brain is made up of about one hundred billion neurons,

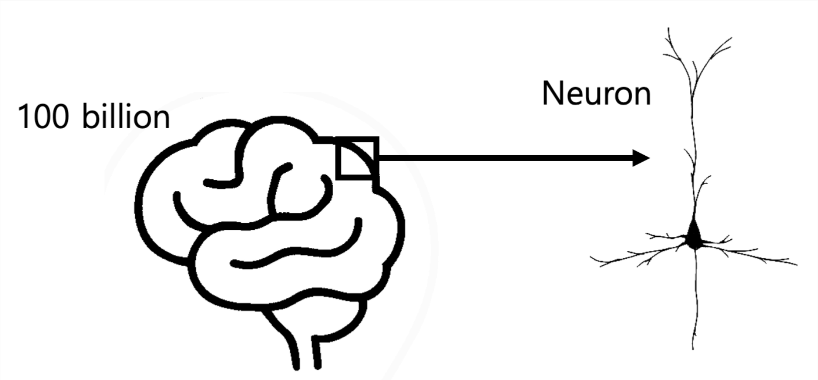
Neuron and artificial neuron

The human brain is made up of about one hundred billion neurons, and while its structure and functions are highly complex

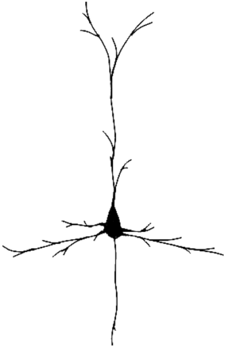


Neuron and artificial neuron

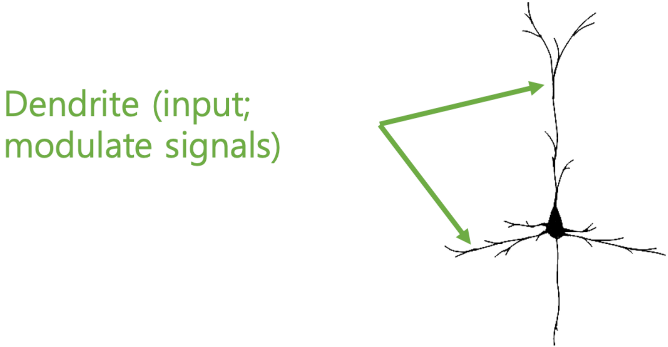
The basic unit that composes the brain is relatively simple.



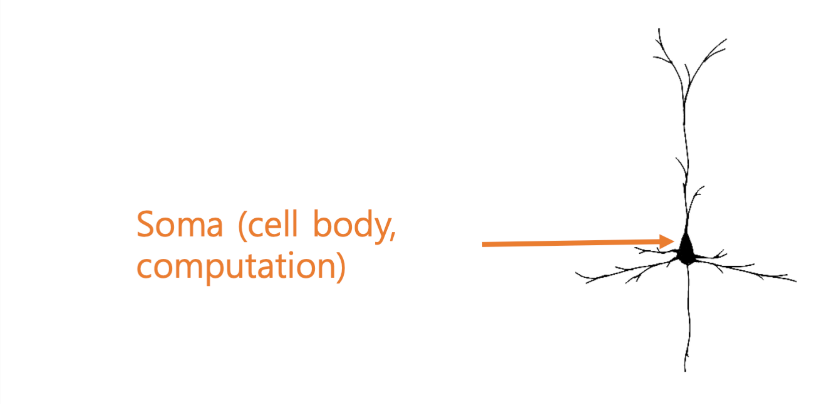
Neuron



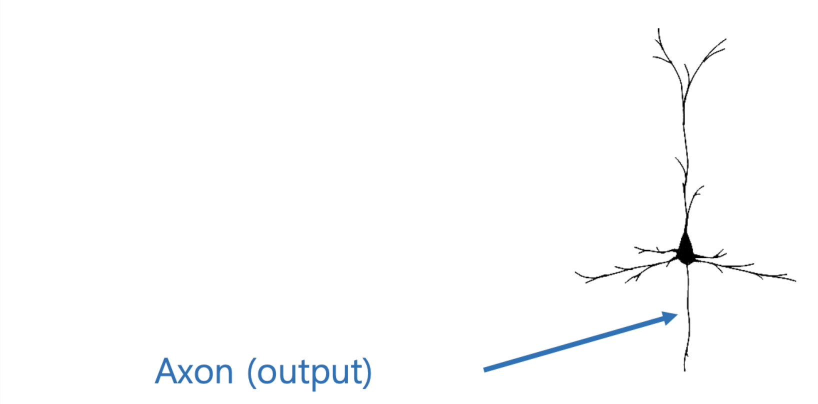
Neuron



Neuron

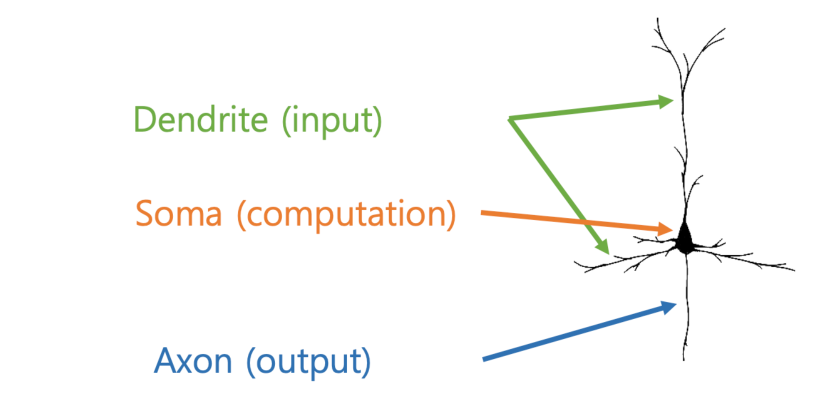


Neuron



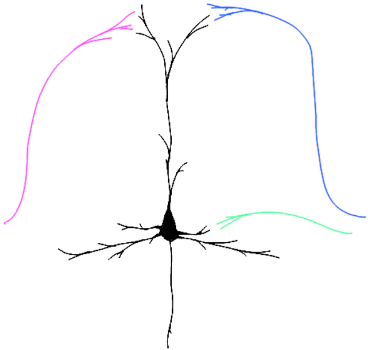
Neuron as an information processor

We can think of a neuron as an information-processing unit with three main functions: (1) input, (2) computation, and (3) output.



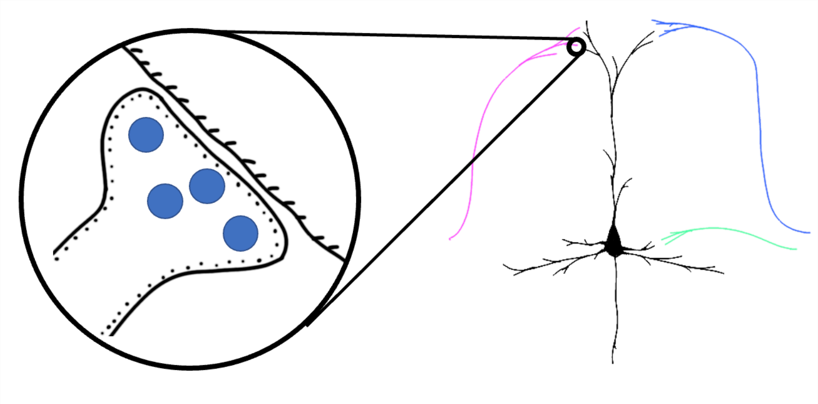
Connections between neurons

It connects to another neuron's axon



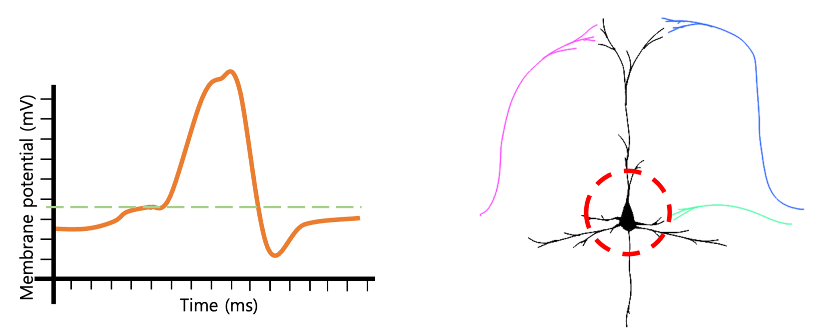
Connections between neurons

It connects to another neuron's axon through a **synapse**

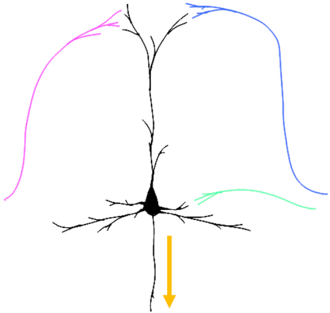


Firing of a neuron

In the soma (cell body), if the incoming signals exceed a certain threshold, the neuron fires an action potential.

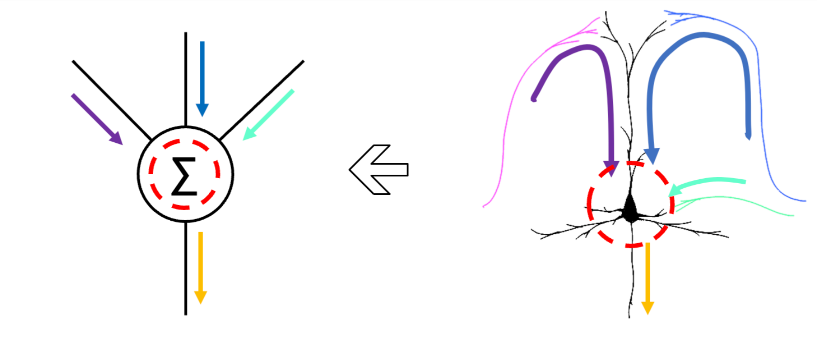


It allows the neuron to transfer information to the next neuron.



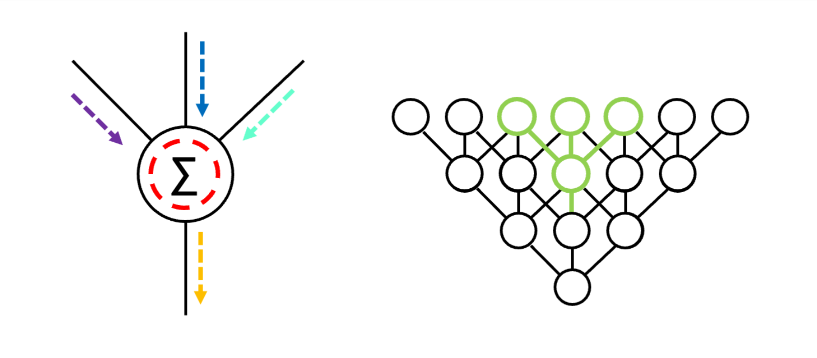
Neurons to artificial neurons

Artificial neurons are designed to mimic the information-processing mechanisms of biological neurons.



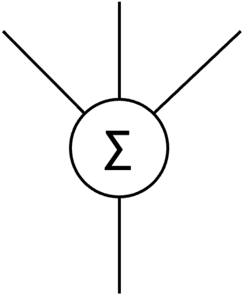
Neurons to artificial neurons

When combined, they form artificial neural networks. Then, let's try to understand how artificial neurons work.

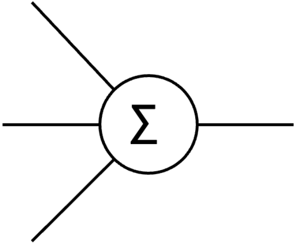


Perceptron

Artificial neurons

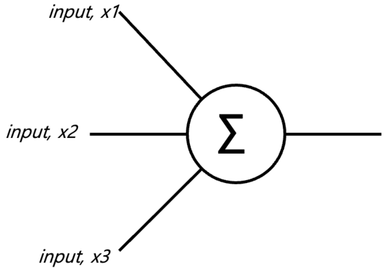


Artificial neurons



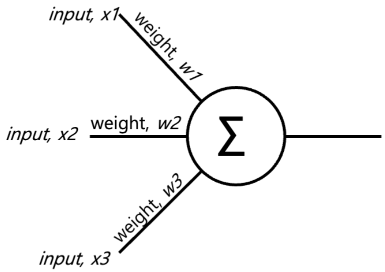
Artificial neurons

This cell receives **inputs** from three neurons, **calculates** whether the total input exceeds the **threshold**, and then produces an **output**.



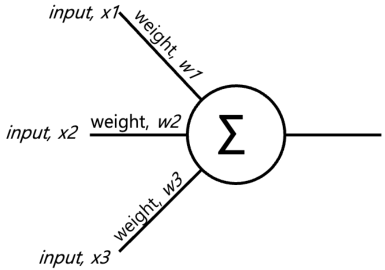
Artificial neurons

Each neuron provides an **input**, denoted as x_1, x_2, x_3.



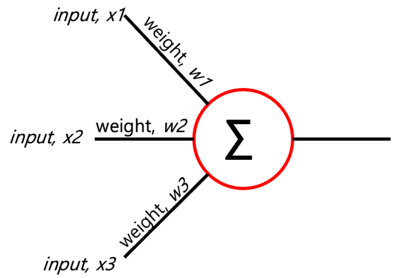
Artificial neurons

Each input x_1, x_2, x_3 is multiplied by a corresponding **weight** w_1, w_2, w_3 before being combined.



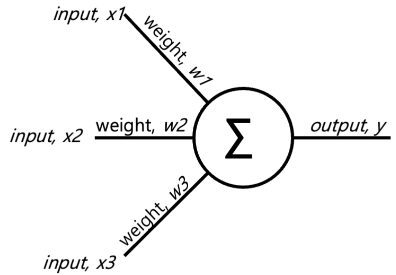
Artificial neurons

The circular unit is called a **node**. It receives the inputs from neurons, combines them with their weights, and calculates the node value.



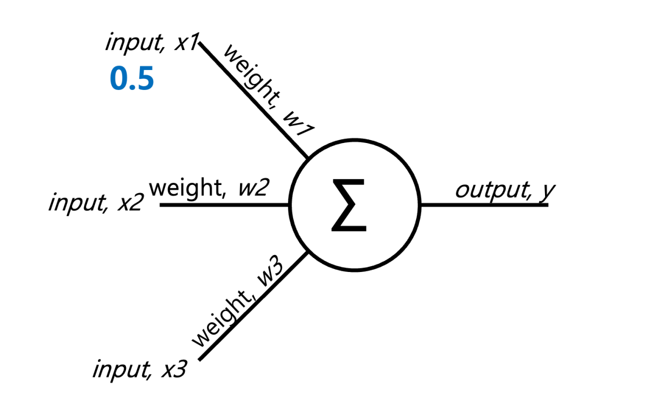
Artificial neurons

This computed output is then passed on to the next neuron.

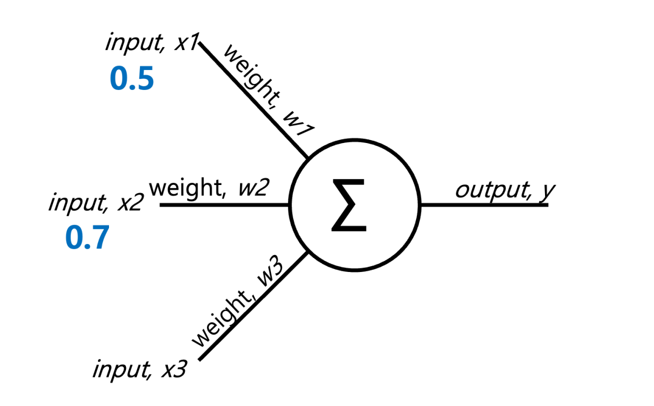


Artificial neurons

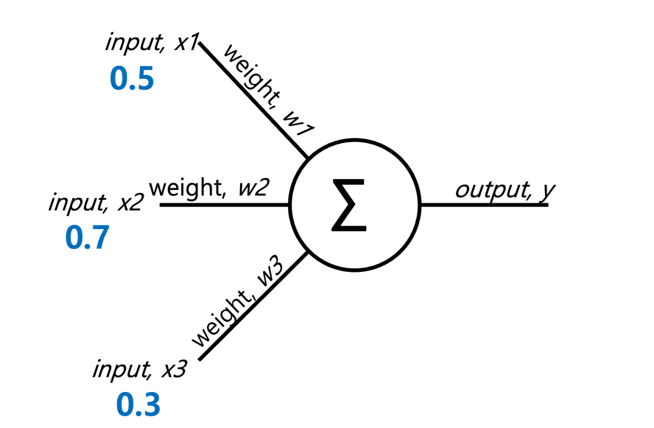
Now, let's put in actual values and calculate the output.



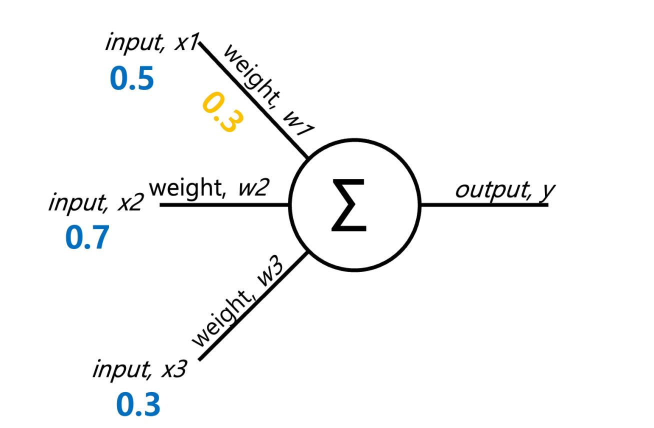
Now, let's put in actual values and calculate the output.



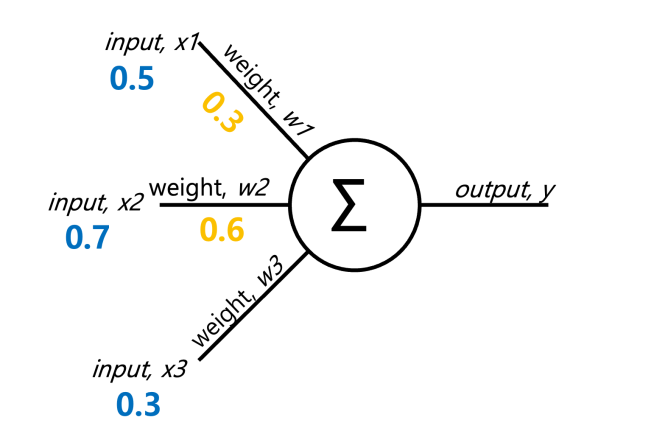
Now, let's put in actual values and calculate the output.



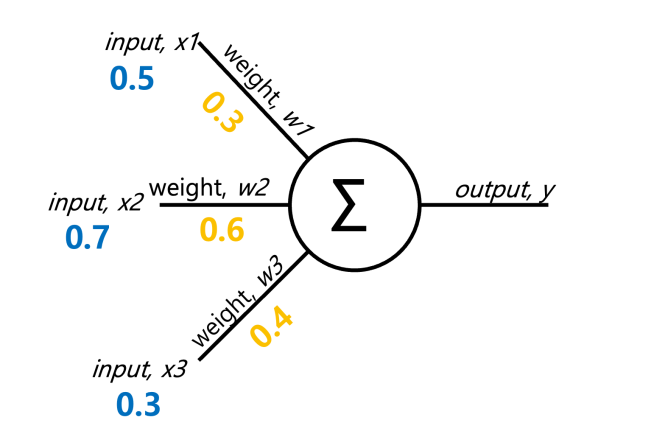
Now, let's put in actual values and calculate the output.



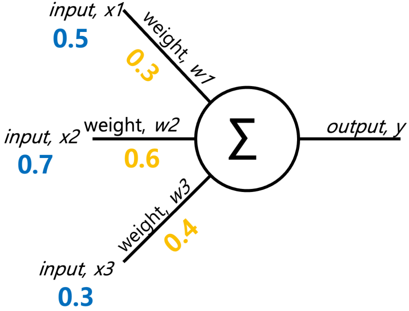
Now, let's put in actual values and calculate the output.



Now, let's put in actual values and calculate the output.



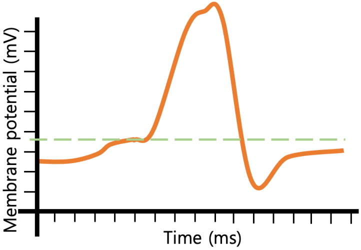
Now, let's put in actual values and calculate the output.

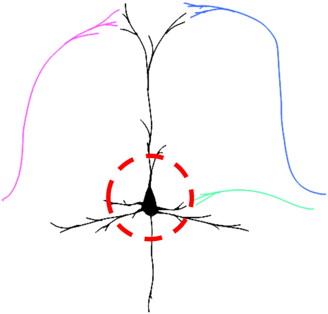


0.5 x 0.3 = 0.15
0.7 x 0.6 = 0.42
0.3 x 0.4 = 0.12

Artificial neurons: activation function

Just like the soma decides whether to fire based on the threshold, an **artificial neuron** computes a weighted sum of inputs and applies an **activation function**.

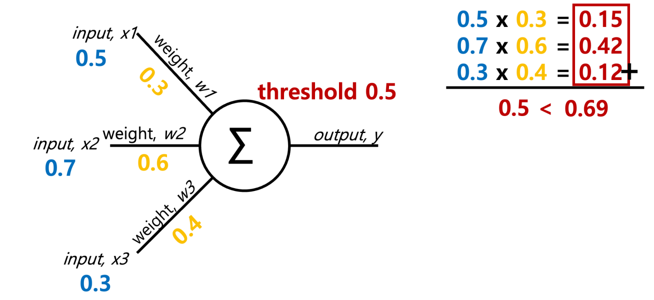




Artificial neurons: activation function

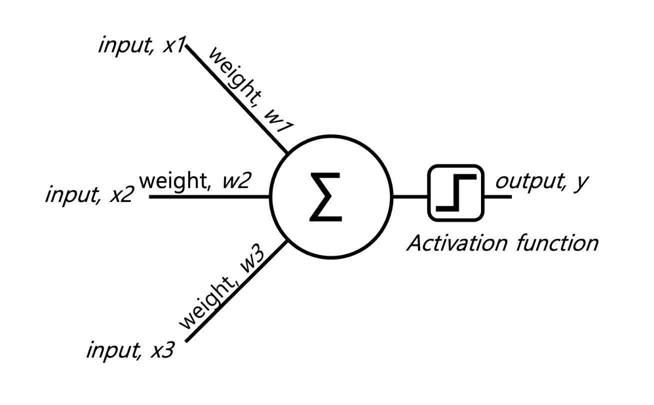
An **activation function** is applied to the weighted sum of inputs to determine the output. For simplicity, let's assume a step function as the activation function:

$$f(z) = \begin{cases} 1 & \text{if } z \geq 0.5 \\ 0 & \text{if } z < 0.5 \end{cases}$$



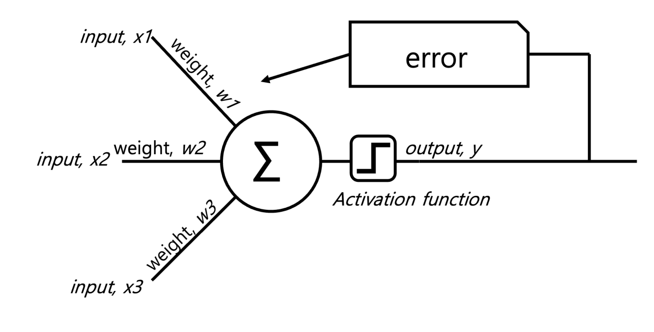
Perceptron

This is a basic structure of the **perceptron**.



Training perceptron

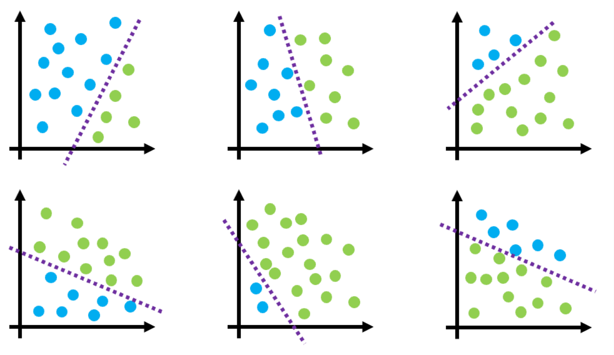
Key idea: To train a perceptron, we compare the predicted output with the actual output. The difference is the **error**, which is then used to adjust the weights so that the model improves over time.



Multi-layer perceptron

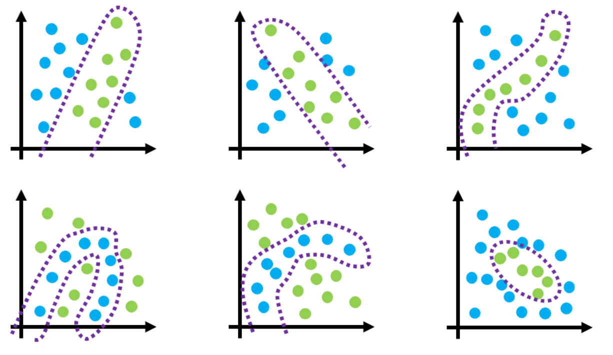
Limitation of a single-layer perceptron

A single-layer perceptron works well for **linearly separable data**. If the data points can be divided by a single straight line in a 2D plane, the perceptron can learn to adjust its weights to find that line and separate the classes.



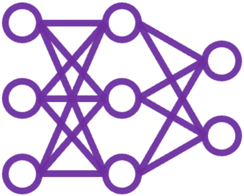
Limitation of a single-layer perceptron

However, a single-layer perceptron has clear **limitations**. It cannot solve problems where the data is **not linearly separable**.

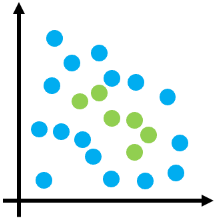


Introduction of an MLP

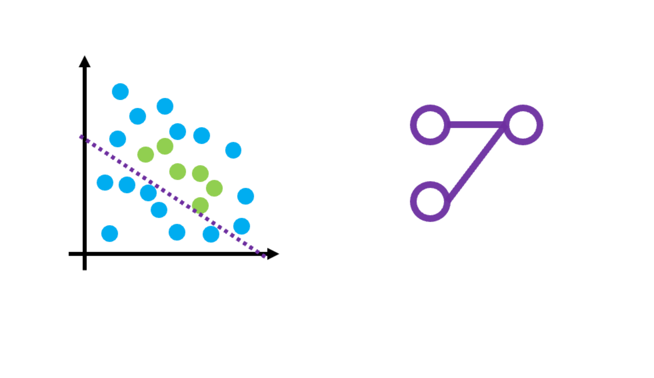
But if we allow **multiple lines**, there is a possibility to separate even non-linear data. This idea leads us to the **multi-layer perceptron (MLP)**.



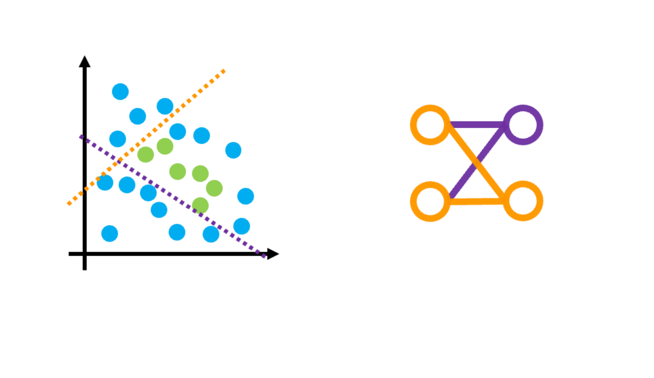
Let's assume we are given data in a complex form like this.



With a single perceptron, linear separation is not possible.

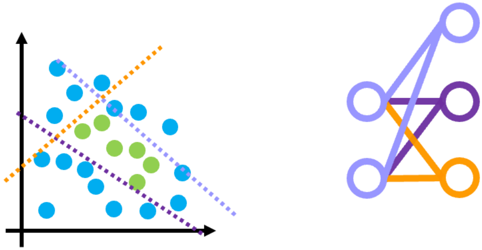


But if we add more lines, it becomes possible to separate further.



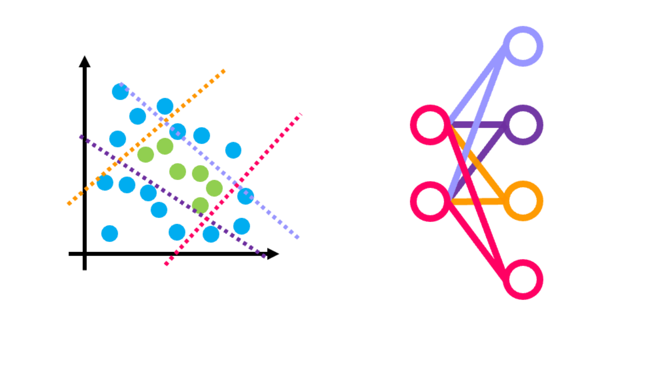
Introduction to an MLP

By adding several lines, the separation becomes more feasible.



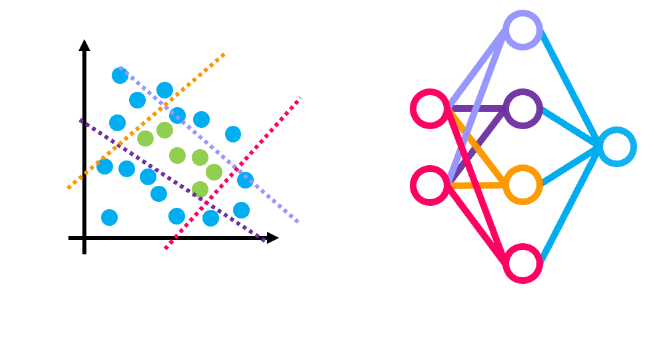
Introduction of an MLP

Four lines can be thought of as the outputs of **four perceptrons**.



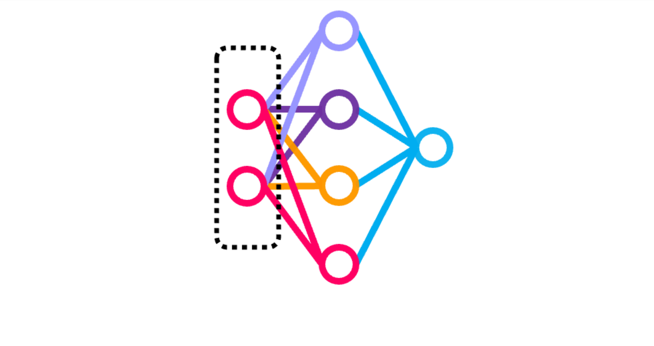
Introduction of an MLP

If we then connect another perceptron that takes these four outputs as its inputs, we can construct a **multi-layer neural network** capable of non-linear separation.



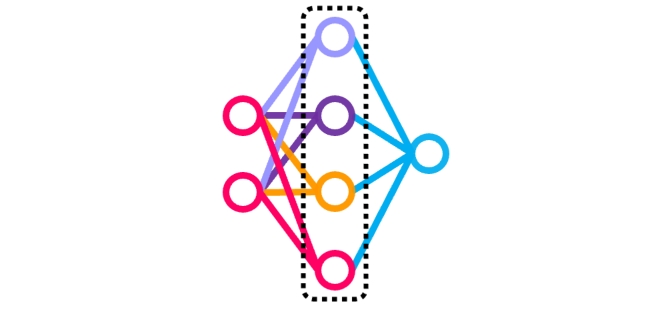
Structure of an MLP

So the MLP we build here consists of an input layer



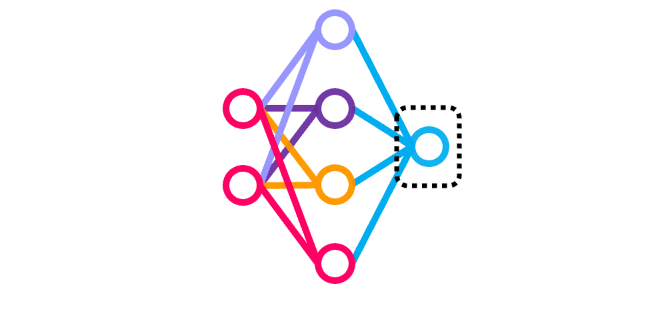
Structure of an MLP

a hidden layer



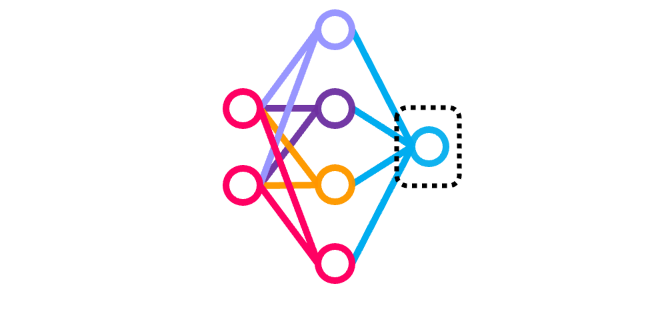
Structure of an MLP

and an output layer.



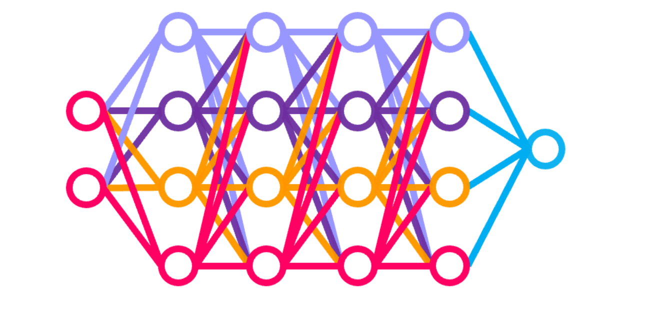
Structure of an MLP

As the number of layers increases, the model can handle more complex data.

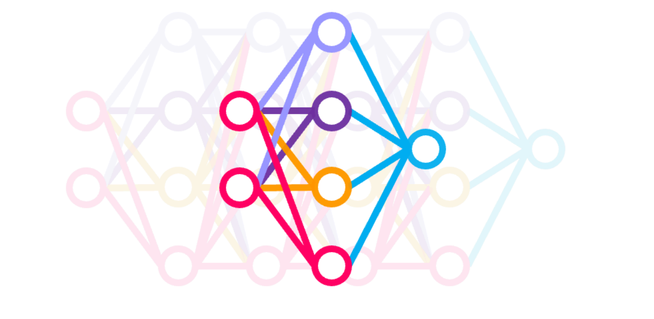


Structure of an MLP

When a network has many layers, we call it “deep.” This is where the term deep learning comes from.

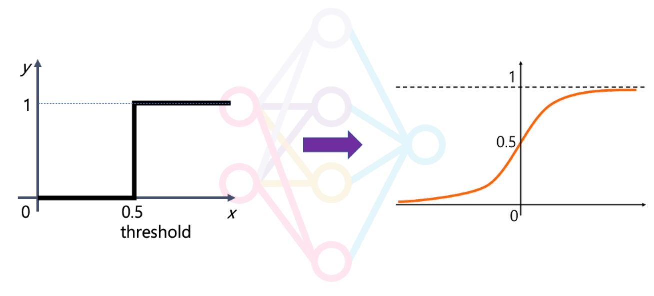


To understand how multilayer networks work, we need to look at a few more changes.



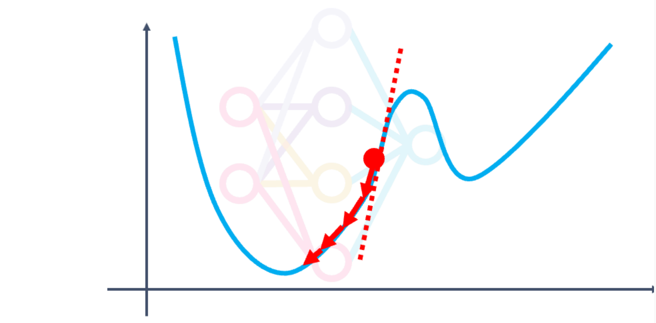
More changes: Activation function

More complex activation functions are used. For example, the *sigmoid function*.



More changes: Optimization

To reduce errors in multilayer networks, methods like gradient descent are used.

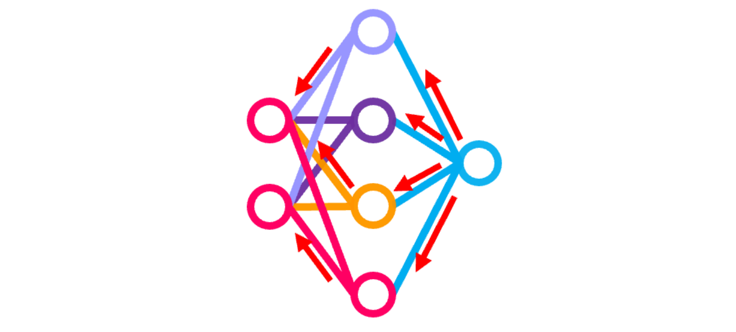


More changes: Optimization

- **Goal:** Learn good word vectors by minimizing a loss function (measures how *wrong* predictions are).
- **Idea:**
    - Start from random initial values
    - Compute the gradient of loss function (which tells us the slope)
    - Move a small step (learning rate)
    - Repeat many times until the loss becomes small

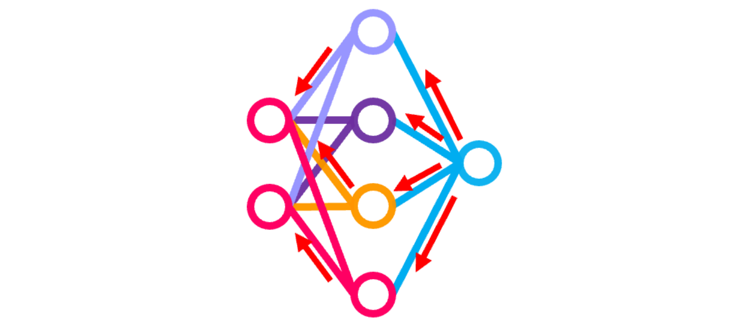
More changes: Backpropagation algorithm

A key algorithm in training neural networks is backpropagation (which might be beyond the scope of this class)



More changes: Backpropagation algorithm

A key algorithm in training neural networks is backpropagation (which might be beyond the scope of this class)



Wrap-up

Coming back to the very FIRST question

Then how computers "Learn"?

- Computers must be <u>trained</u> to recognize spam vs. ham

Coming back to the very FIRST question

Then how computers “Learn”?

- Computers must be trained to recognize spam vs. ham
    - Based on what we’ve learned in the class train is...

Coming back to the very FIRST question

Then how computers "Learn"?

- Computers must be trained to recognize spam vs. ham
  - Based on what we've learned in the class train is...
- They extract features from the text (words, patterns, metadata)

Coming back to the very FIRST question

Then how computers "Learn"?

- Computers must be <u>trained</u> to recognize spam vs. ham
  - Based on what we've learned in the class train is...
- They extract features from the text (words, patterns, metadata)
  - Extracted features become input given to the hidden layers

Coming back to the very FIRST question

Then how computers "Learn"?

- Computers must be <u>trained</u> to recognize spam vs. ham
  - Based on what we've learned in the class train is...
- They extract features from the text (words, patterns, metadata)
  - Extracted features become input given to the hidden layers
- Learn classification rules from <u>labeled data</u>

Coming back to the very FIRST question

Then how computers “Learn”?

- Computers must be trained to recognize spam vs. ham
  - Based on what we’ve learned in the class train is...
- They extract features from the text (words, patterns, metadata)
  - Extracted features become input given to the hidden layers
- Learn classification rules from labeled data
  - Compare the predicted values from the neural network to the true labels given by humans

Coming back to the very FIRST question

Then how computers “Learn”?

- Computers must be trained to recognize spam vs. ham
  - Based on what we’ve learned in the class train is...
- They extract features from the text (words, patterns, metadata)
  - Extracted features become input given to the hidden layers
- Learn classification rules from labeled data
  - Compare the predicted values from the neural network to the true labels given by humans
  - Recall the importance of building good dataset in the last class

Coming back to the very FIRST question

Then how computers "Learn"?

- Computers must be trained to recognize spam vs. ham
    - Based on what we've learned in the class train is...
- They extract features from the text (words, patterns, metadata)
    - Extracted features become input given to the hidden layers
- Learn classification rules from labeled data
    - Compare the predicted values from the neural network to the true labels given by humans
    - Recall the importance of building good dataset in the last class
- Apply these rules to new, unseen emails

On Thursday

We will do a hands-on activity on a text classification task.

Where we are at

6	9/30	Text classification	[LC] Ch. 5	
	10/2	Python tutorial 5		Student presentation topics submission
7	10/7	Searching; Midterm review	[LC] Ch. 6	
	10/9	Midterm		Midterm exam (Online)
8	10/14	**Fall break (No class)**		
	10/16	Presentation prep		