Class core values

1. Be **respect**ful to yourself and others
2. Be **confident** and believe in yourself
3. Always do your **best**
4. Be **cooperative**
5. Be **creative**
6. Have **fun**
7. Be **patient** with yourself while you learn
8. Don’t be shy to **ask “stupid” questions**
9. Be **inclusive** and **accepting**

Week 3, Lecture 1

A gentle introduction to a neuron

Learning Objectives

1. Describe the basic concept of perceptron
2. Explain an activation function and why it's needed
3. Describe the concept of gradient descent
4. Explain backpropagation and the basics of a dense neural net
5. Apply concepts like learning rate, and optimization

https://tinyurl.com/4xzn34as

Neural nets have been revolutionizing the field of protein structure prediction and design

Neural nets have been revolutionizing the field of protein structure prediction and design

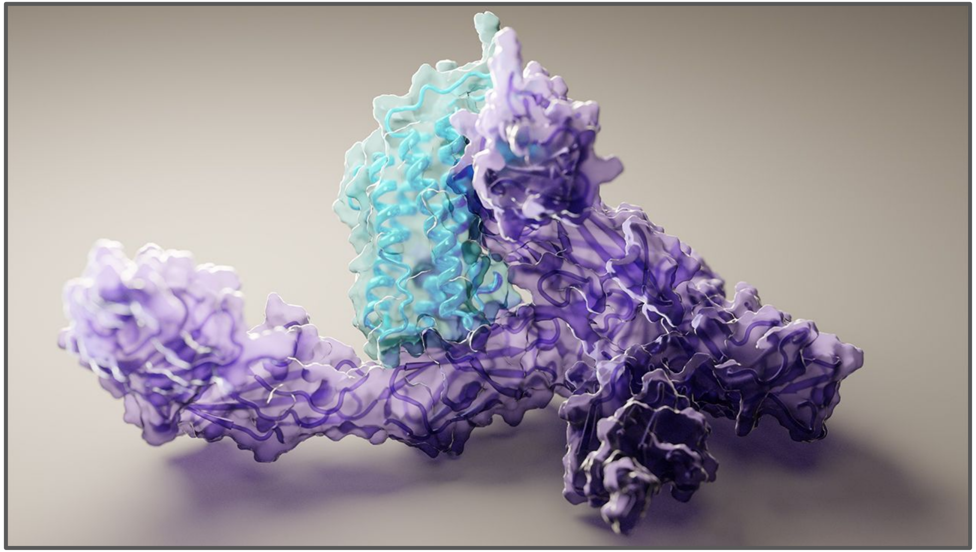


Image credit: Ian Haydon, Institute for Protein Design

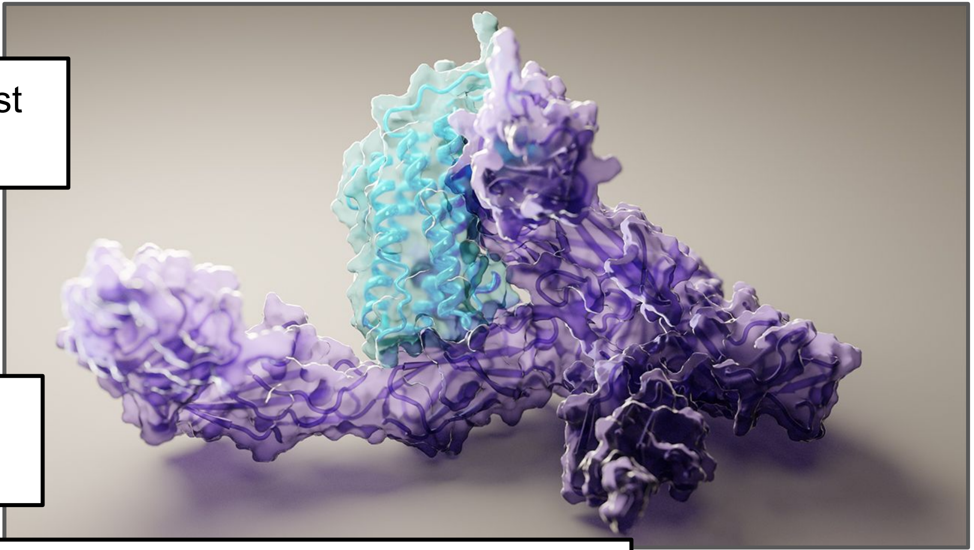
Neural nets have been revolutionizing the field of protein structure prediction and design

Nature 2020: “It will change everything!”

Forbes 2021: “ AlphaFold is the most important achievement in AI ever”

Deepmind 2020: “AlphaFold - a solution to a 50 year-old grand challenge in biology”

EMBL 2021: “Great expectations - The potential impact of AlphaFold DB”

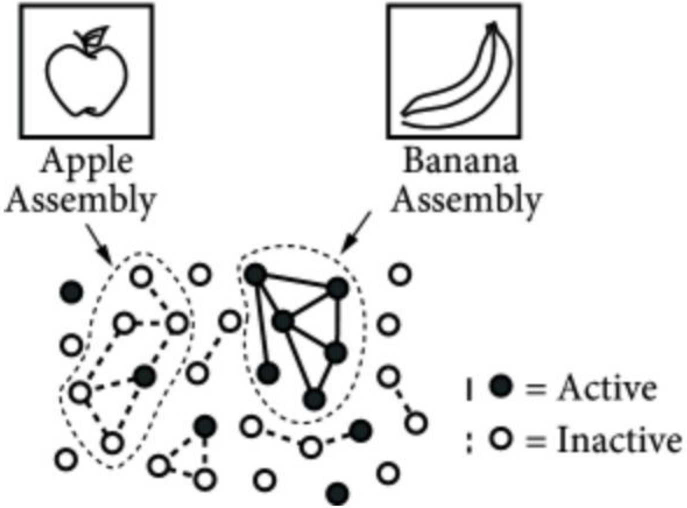


Science 2021: “Researchers unveil phenomenal new AI for predicting protein structures”

Image credit: Ian Haydon, Institute for Protein Design

Despite its rise in the last decade, neural nets and machine learning are old ideas

Despite its rise in the last decade, neural nets and machine learning are old ideas



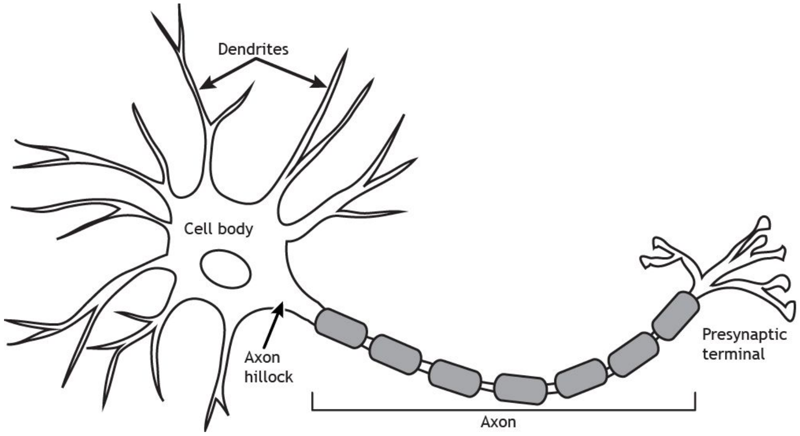
1940 D.O. Hebb **Hebbian learning**

The simplest building block of neural nets are called *perceptrons*

Hebbian learning

1958 Frank Rosenblatt **Perceptron**

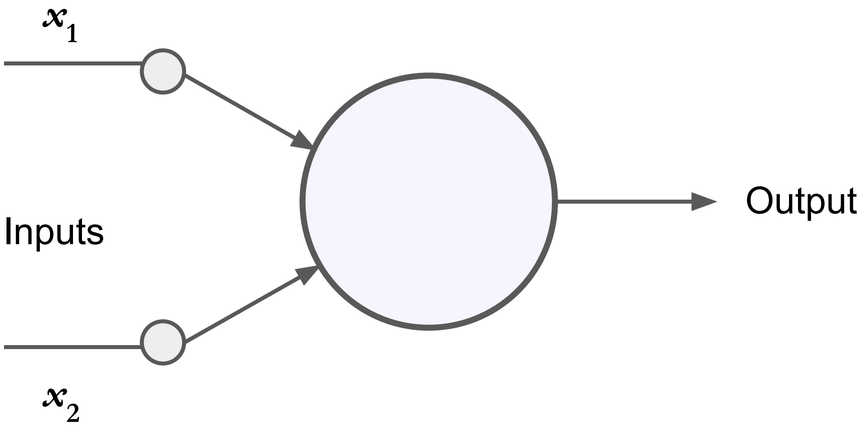
The simplest building block of neural nets are called *perceptrons*

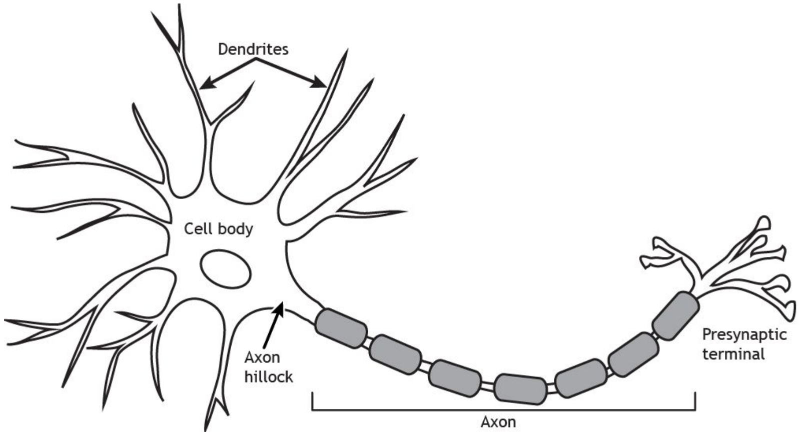


Hebbian learning

1958 Frank Rosenblatt **Perceptron**

The simplest building block of neural nets are called *perceptrons*

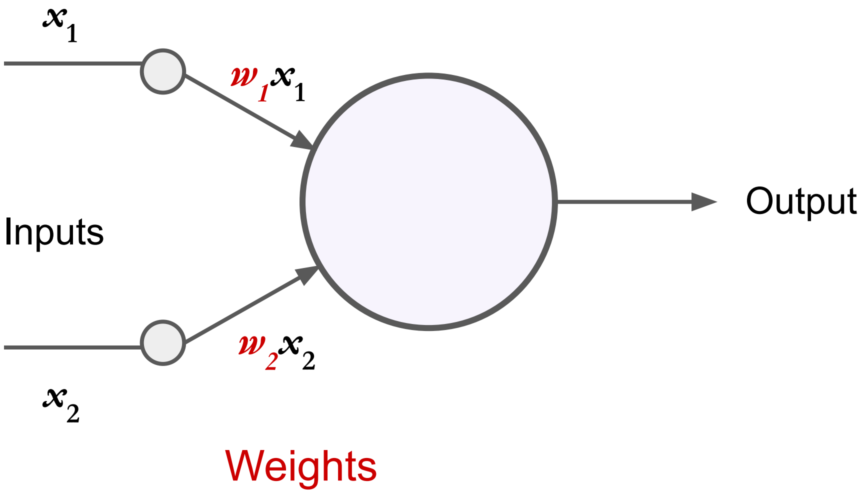




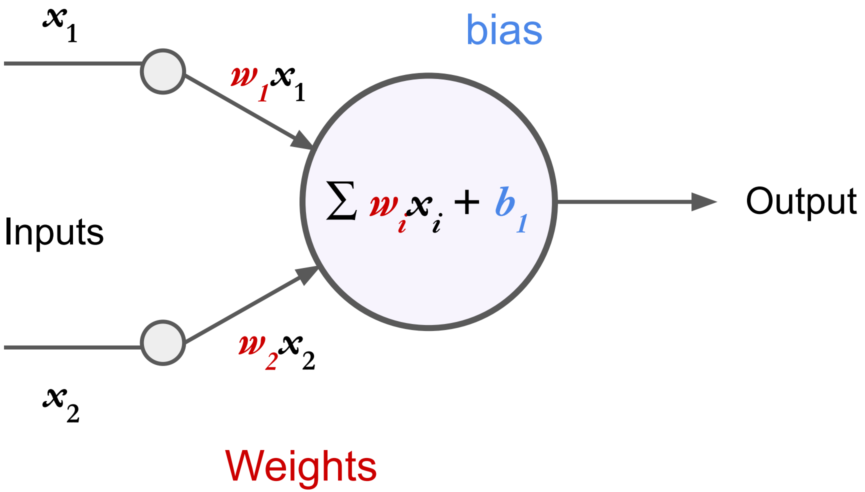
Hebbian learning

1958 Frank Rosenblatt **Perceptron**

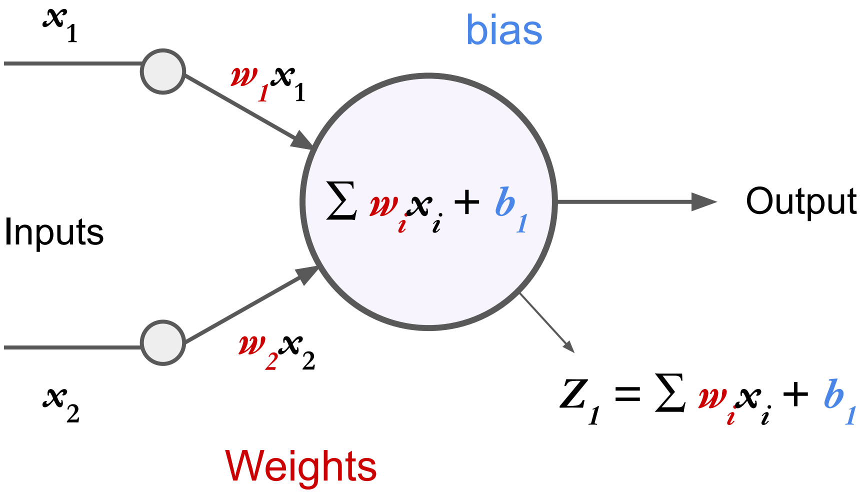
In a perceptron, each input "stimulus" gets a weight



Weights and bias are the *parameters* of the model that need to be initiated



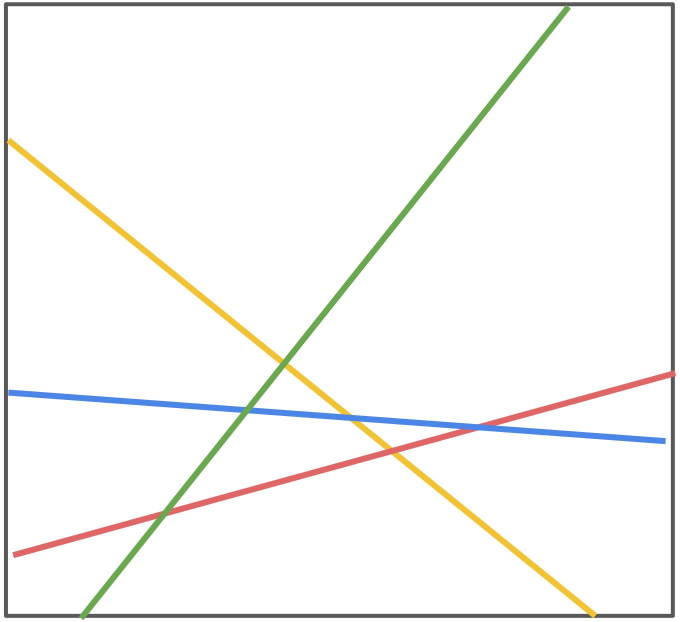
The overall input of each neuron (*z*) is the sum of the inputs with a *bias* term



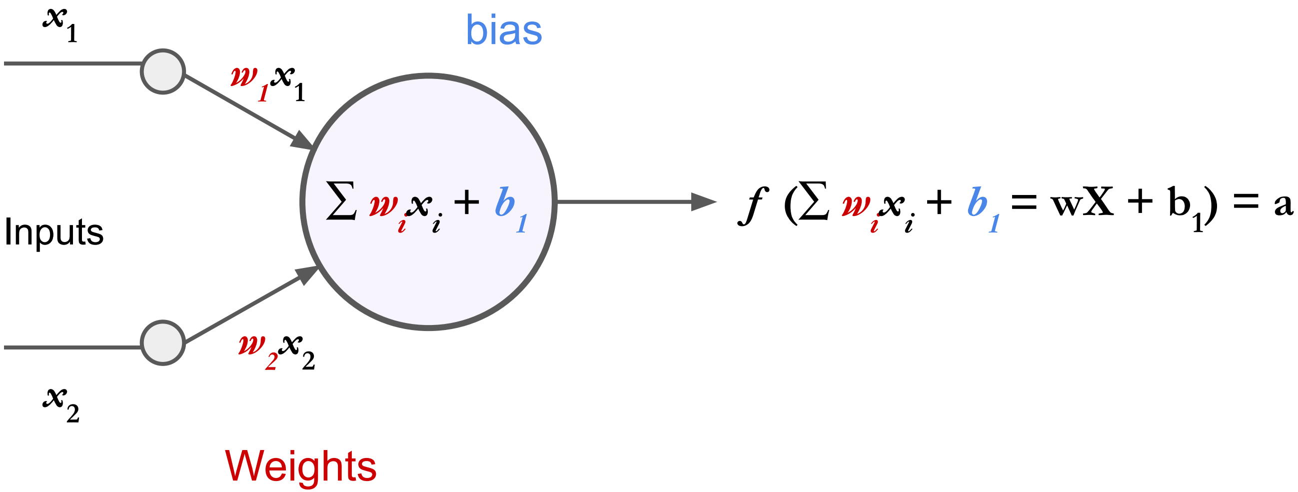
$$Z_1 = \sum w_i x_i + b_1 = \mathbf{w}\mathbf{X} + \mathbf{b}_1 \qquad [\,w1 \quad w2\,]\begin{pmatrix} x1 \\ x2 \end{pmatrix} + b_1$$

In-class activity

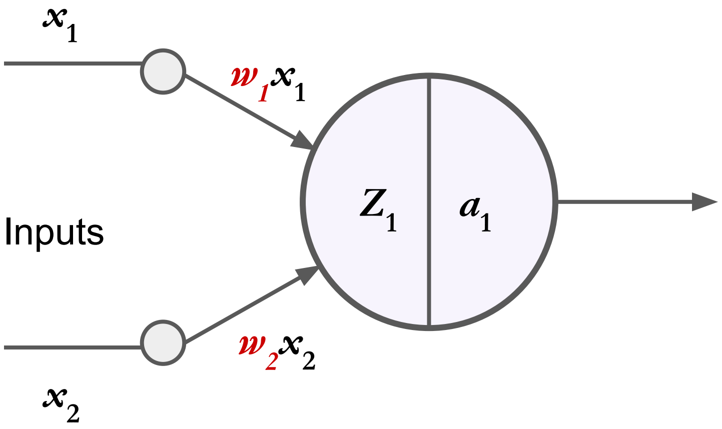
Sum of linear functions



An activation function allows neural nets to learn more complex patterns

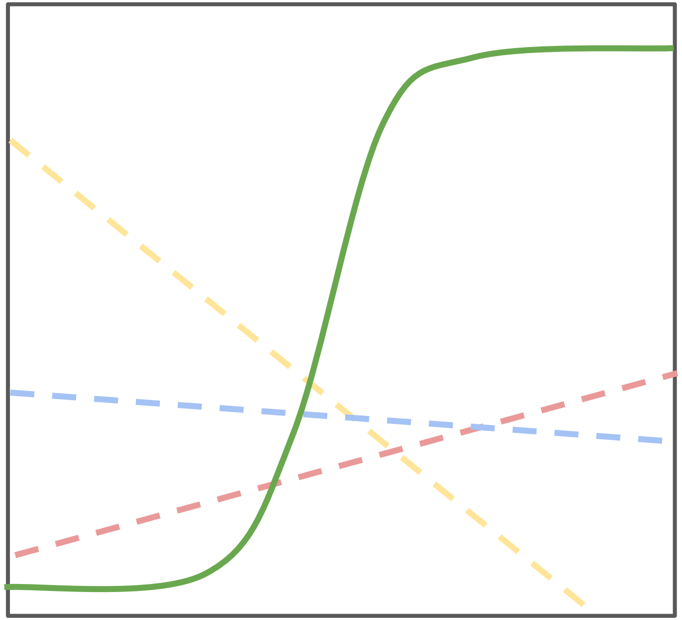


An activation function allows neural nets to learn more complex patterns



In-class activity

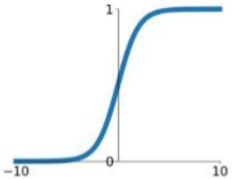
Exploring activation functions



Many activation functions can be used for neural nets

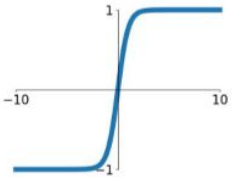
Sigmoid

$\sigma(x) = \frac{1}{1+e^{-x}}$



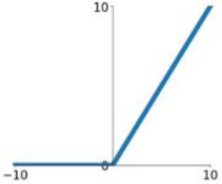
tanh

$\tanh(x)$



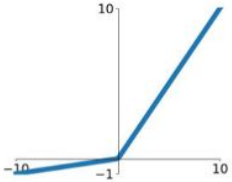
ReLU

$\max(0, x)$



Leaky ReLU

$\max(0.1x, x)$



Maxout

$\max(w_1^T x + b_1, w_2^T x + b_2)$

ELU

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

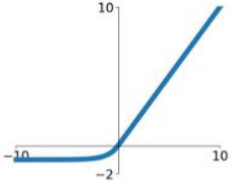


Image from medium: Shruti Jadon - Introduction to different activation functions

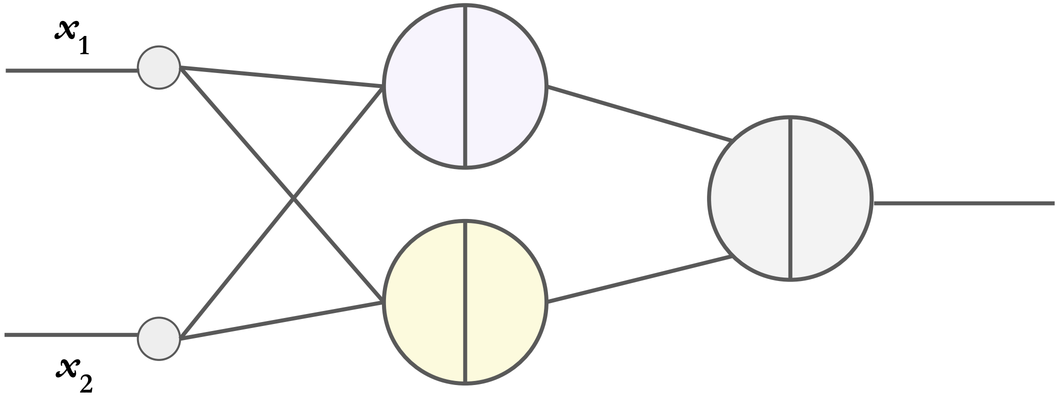
Adding more neurons help with getting more complicated functions

Hebbian learning

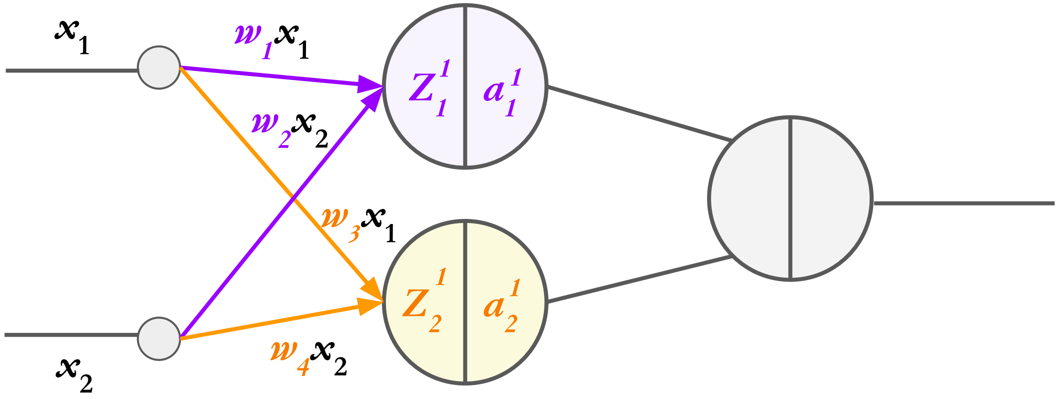
Perceptron

1965 Ivakhneko & Lapa First functional networks with many layers

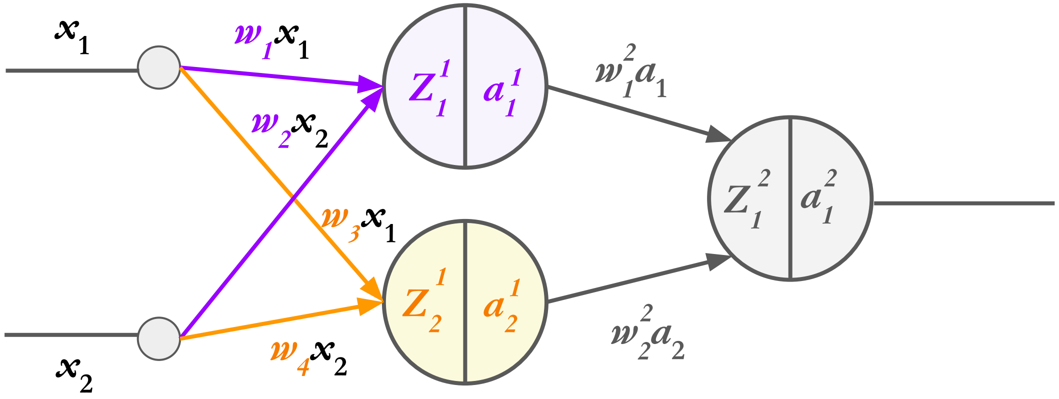
Adding more neurons help with getting more complicated functions



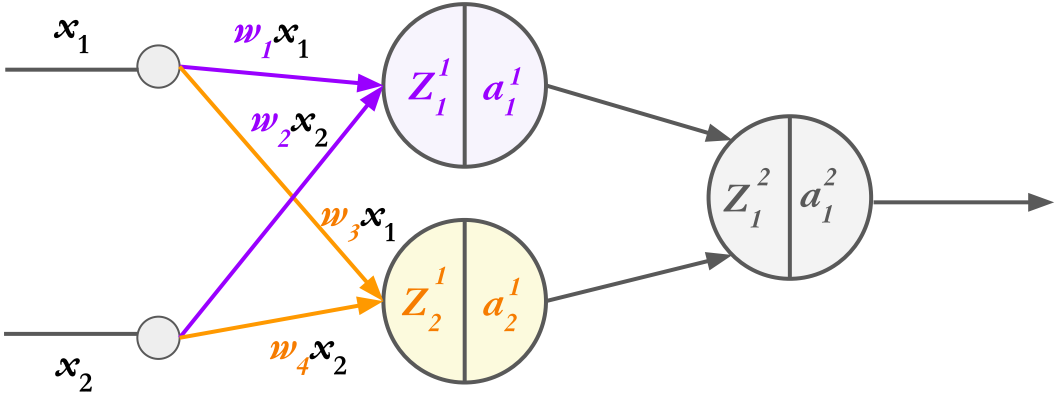
Adding more neurons help with getting more complicated functions



Adding more neurons help with getting more complicated functions



Feed-forward is the process of calculating the final output from the inputs



$$a^1_1 = sigmoid(Z_1)$$

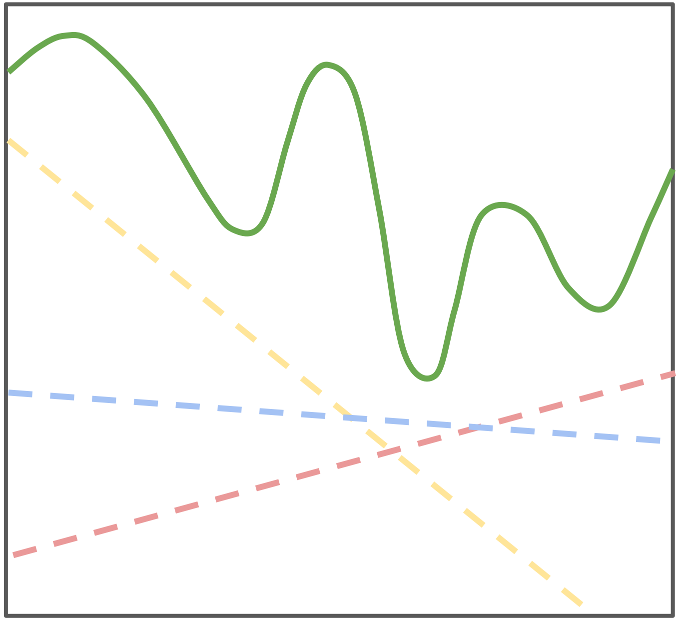
$$a^1_2 = sigmoid(Z_2)$$

$$Z^2_1 = \sum w^2_i a^1_i + b_2$$

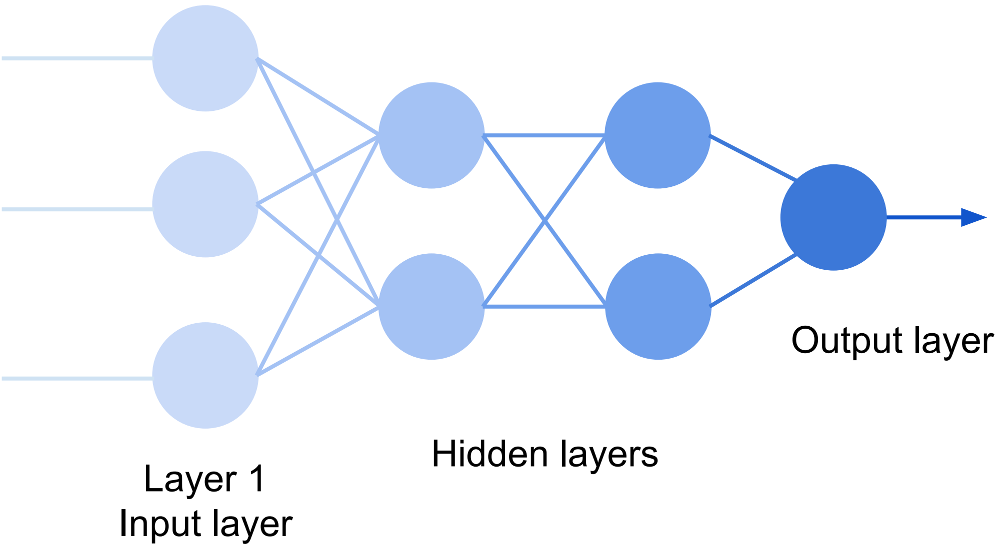
$$= w^2_1 * sigmoid(Z^1_1) + w^2_2 * sigmoid(Z^1_2) + b_2$$

In-class activity

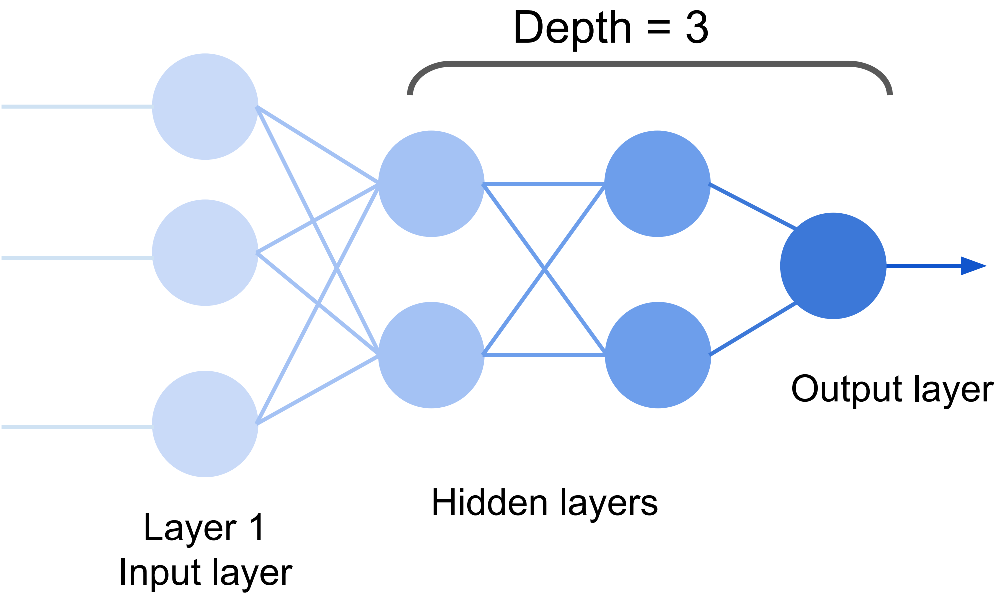
Effects of adding an additional layer



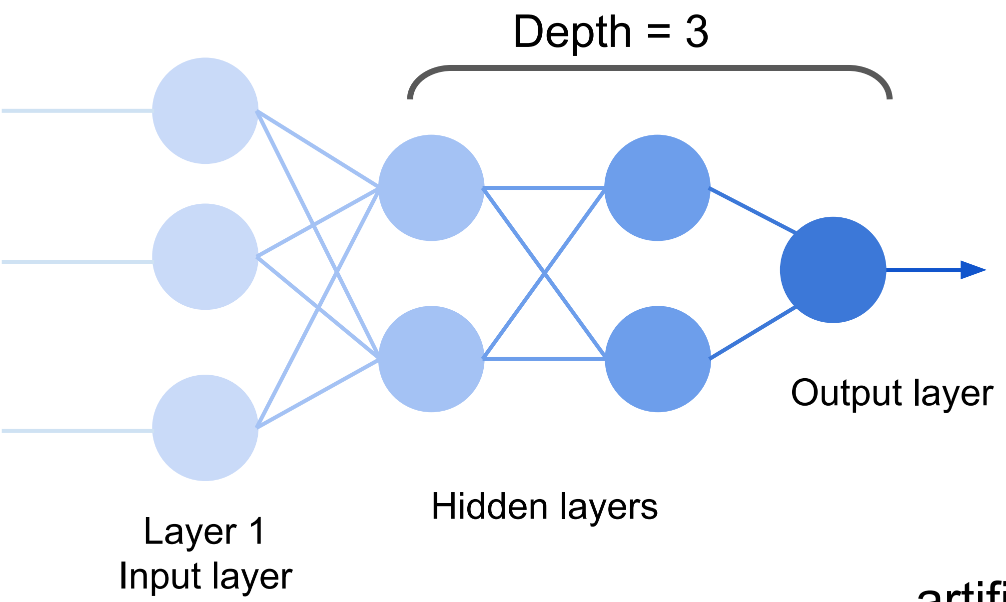
Deep neural nets are better than single layer neural nets for prediction



Deep neural nets are better than single layer neural nets for prediction

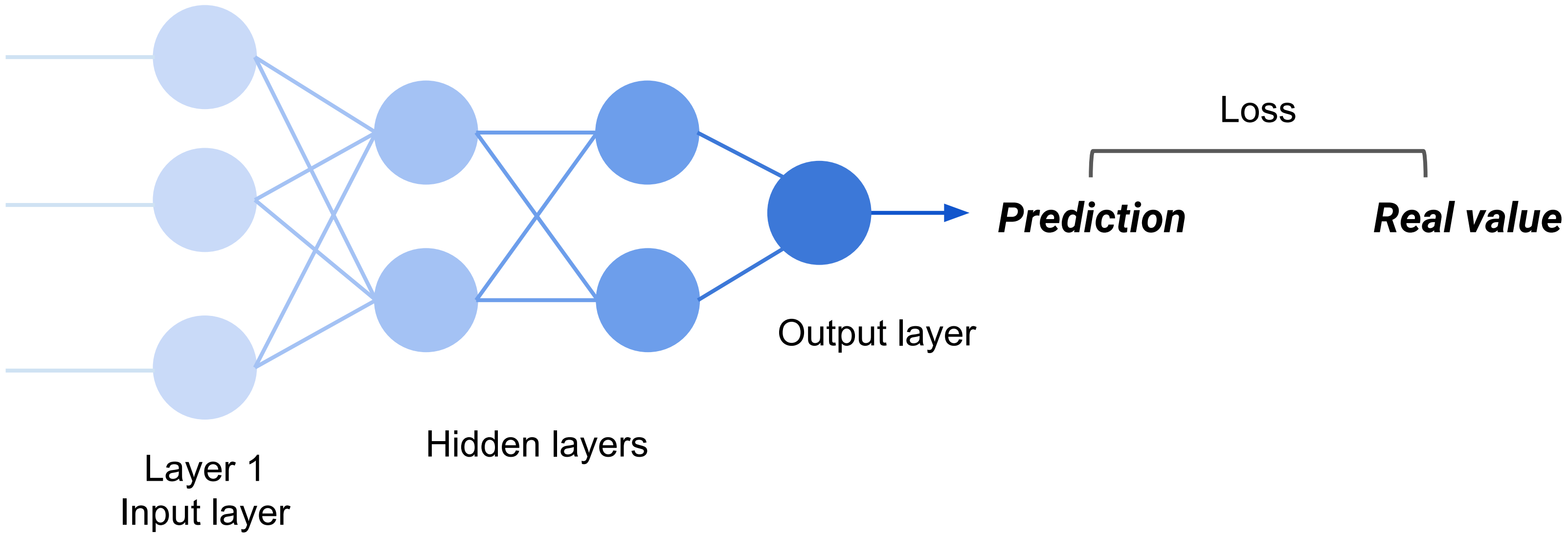


Deep neural nets are better than single layer neural nets for prediction



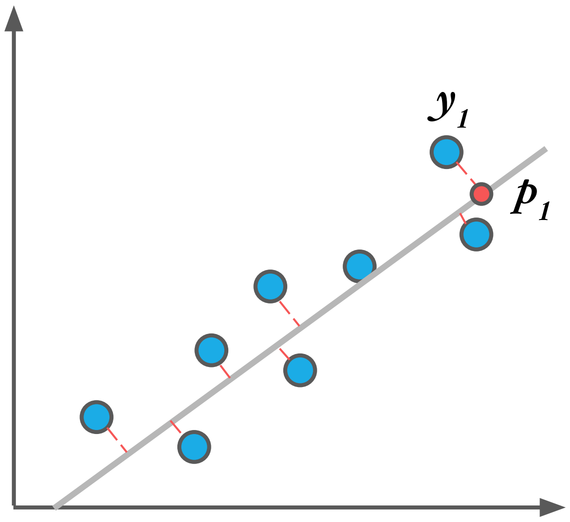
Fully connected
artificial neural network (ANN)

We can assess performance of the network through loss calculation



The choice of the loss function depends on the problem at hand

The choice of the loss function depends on the problem at hand



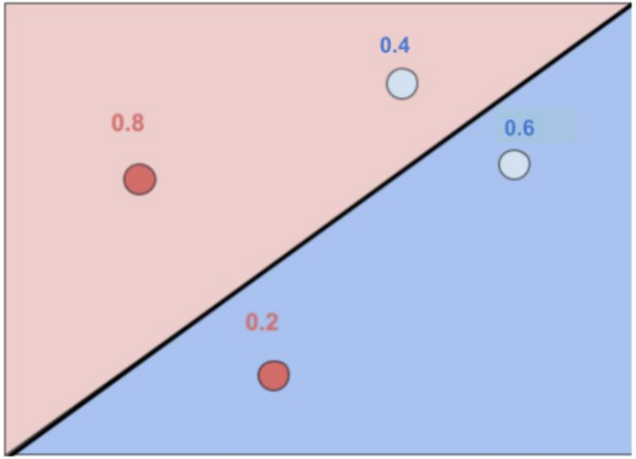
Regression

MSE (Mean Squared Error)

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

p_i

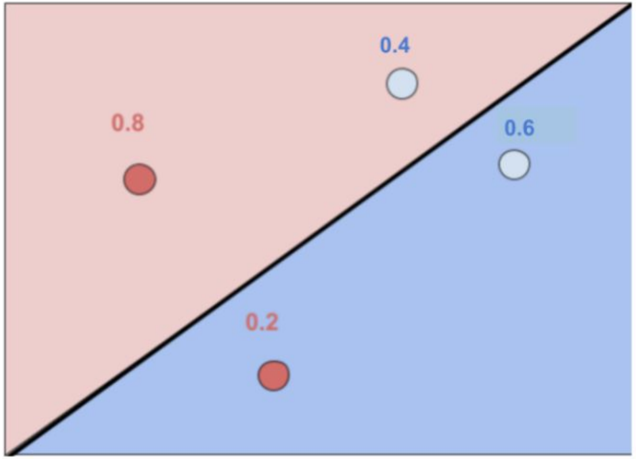
The choice of loss function depends on the problem at hand



Classification

Cross-entropy

The choice of loss function depends on the problem at hand

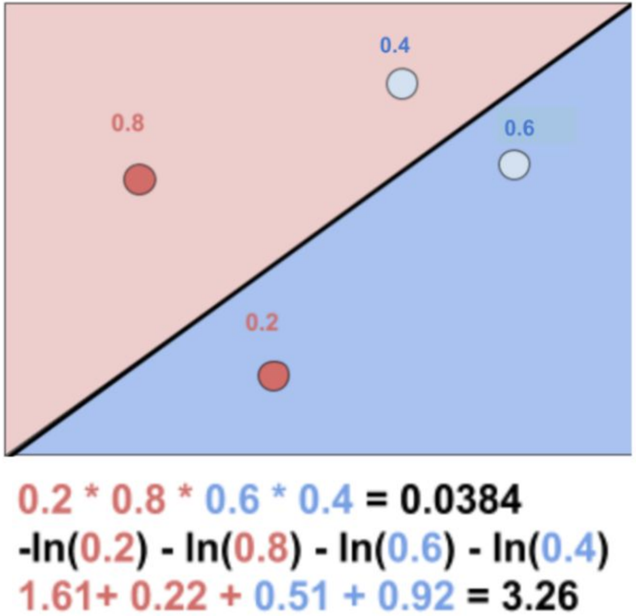


0.2 * 0.8 * 0.6 * 0.4 = 0.0384

Classification

Cross-entropy

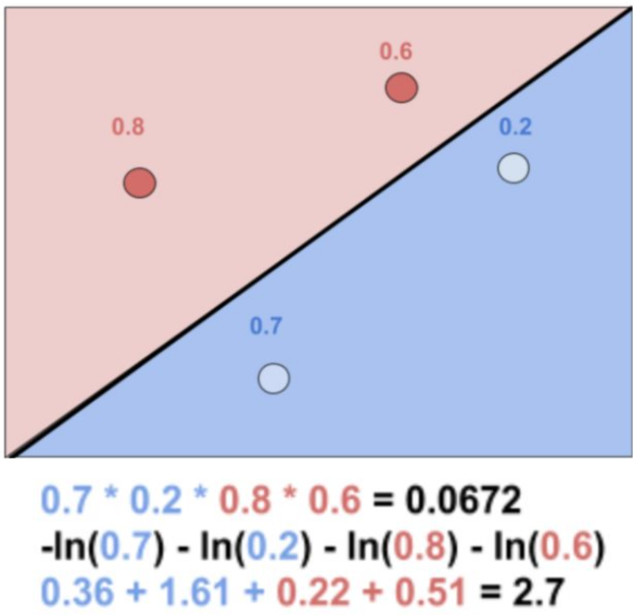
The choice of loss function depends on the problem at hand



Classification

Cross-entropy

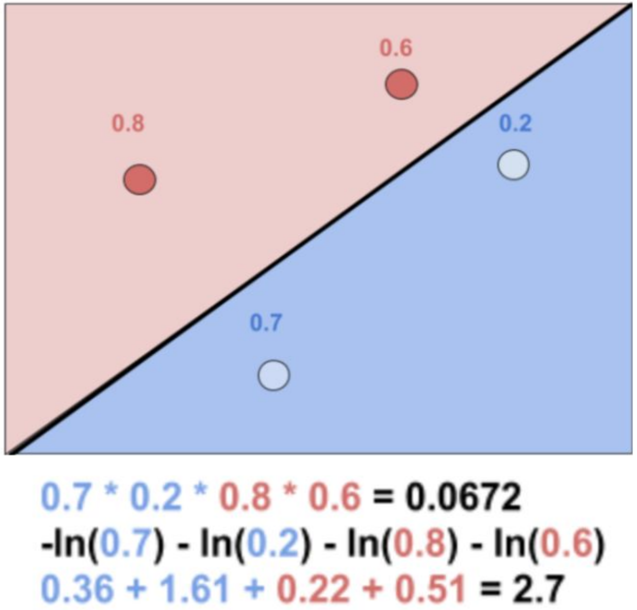
The choice of loss function depends on the problem at hand



Classification

Cross-entropy

The choice of loss function depends on the problem at hand

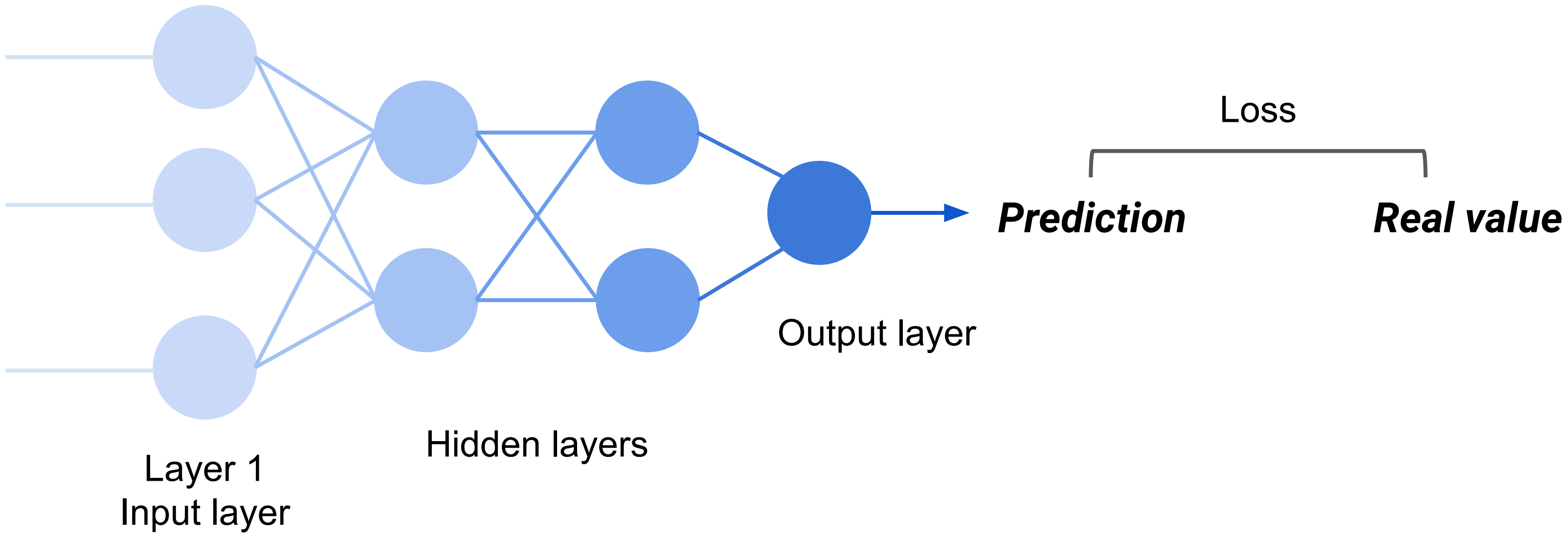


Classification

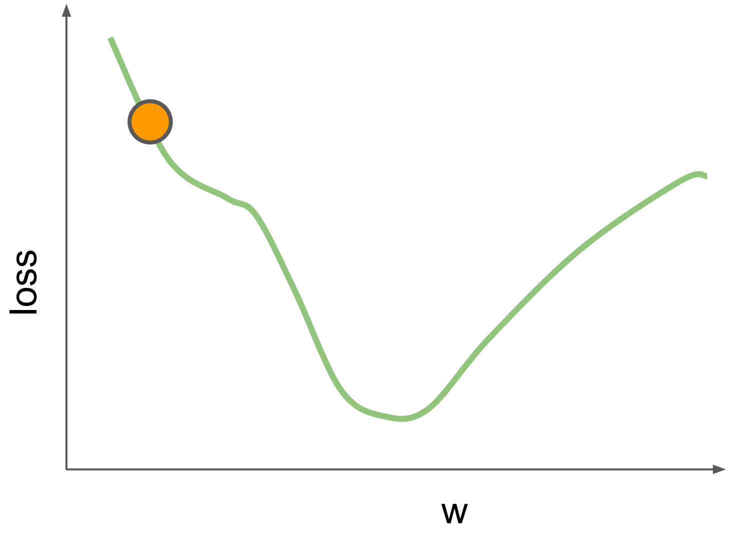
Cross-entropy
Categorical cross-entropy

Image from TDS - cross entropy

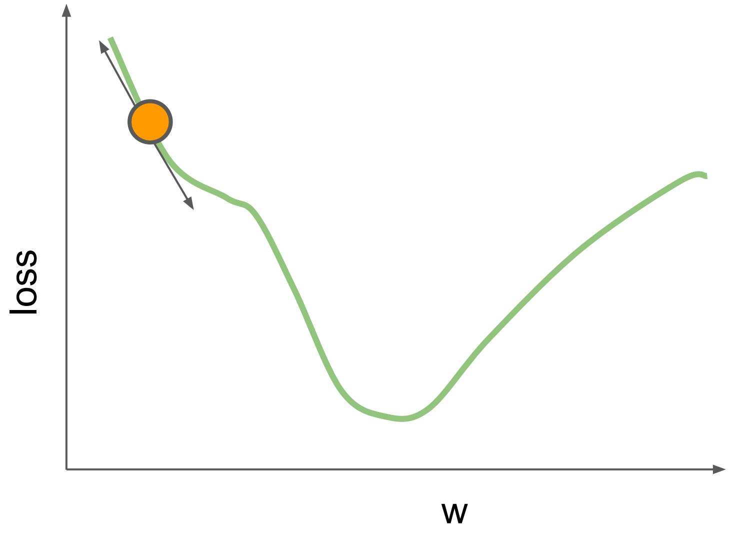
In order for the network to learn, it needs to use the loss to adjust itself



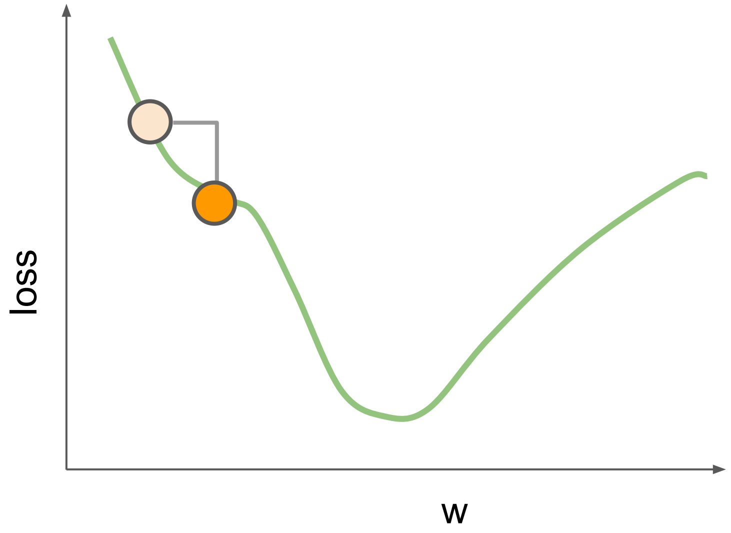
Loss function can guide the network to get better



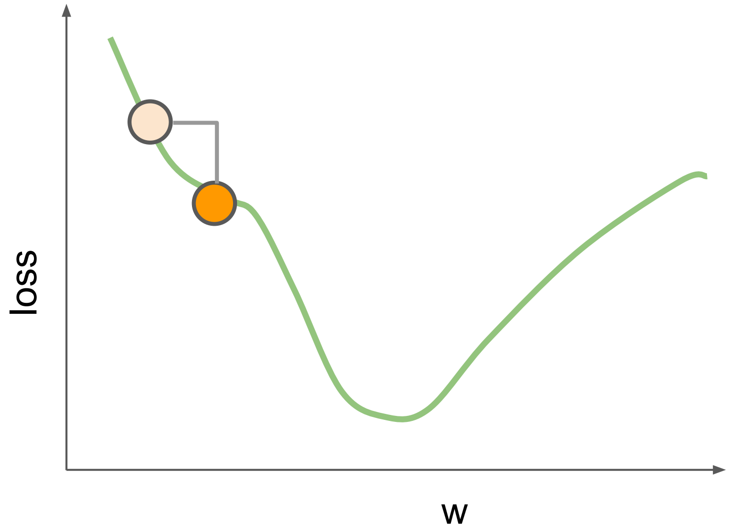
Loss function can guide the network to get better

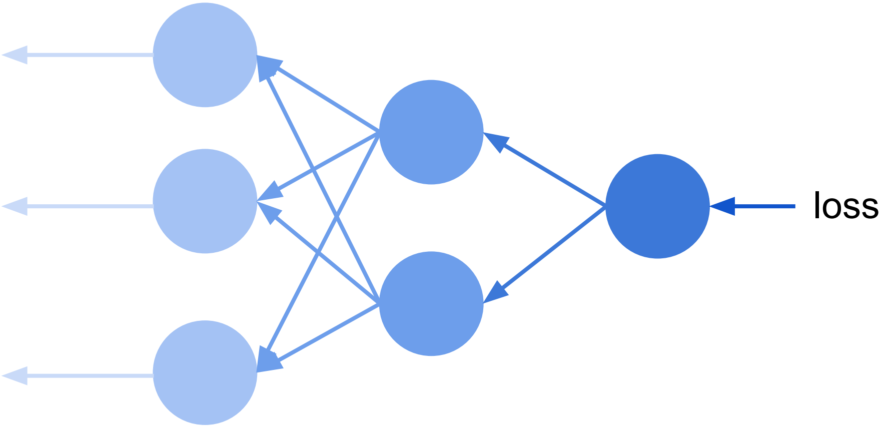


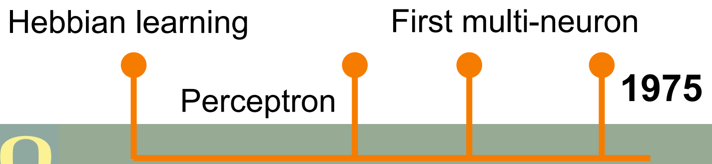
Loss function can guide the network to get better



Backpropagation is the process of using the value of the loss to adjust the weights in layers



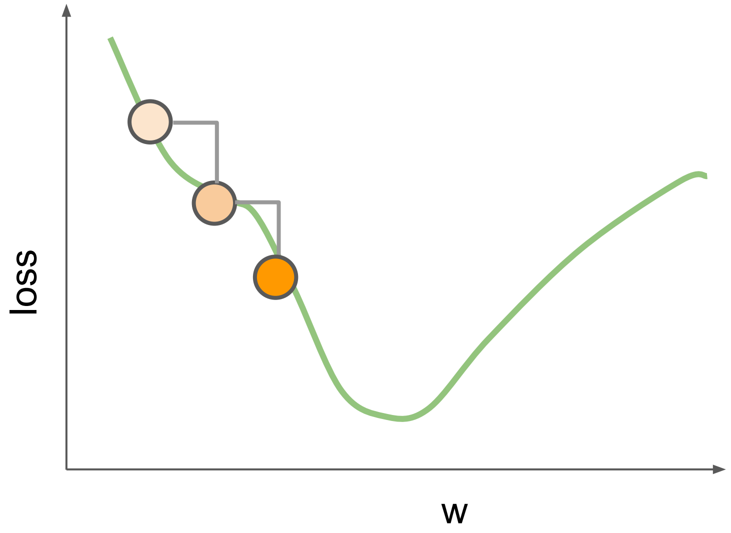




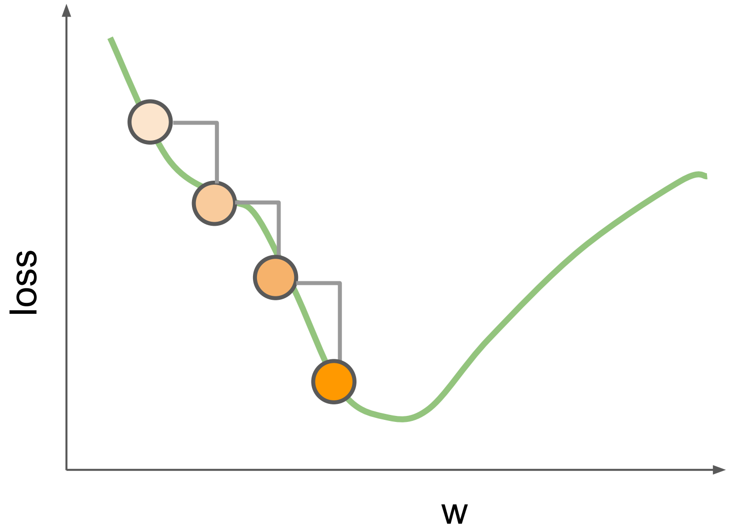
Werbos

Backpropagation

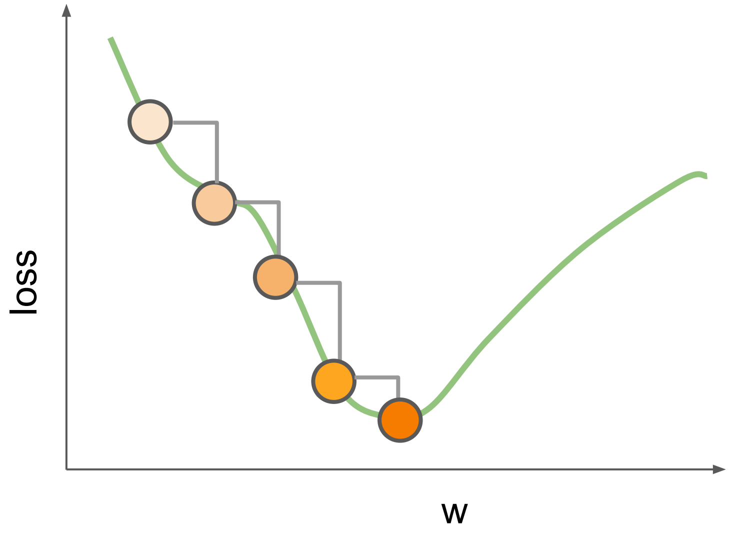
Gradient descent is used to find the best weights



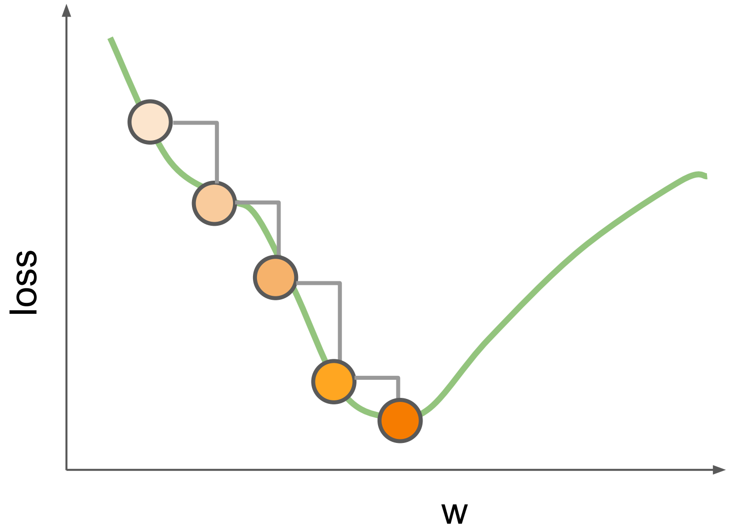
Gradient descent is used to find the best weights



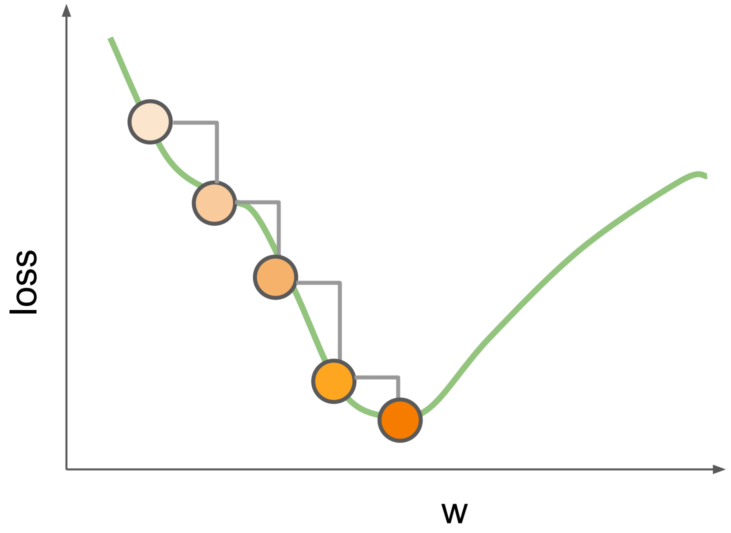
Gradient descent is used to find the best weights

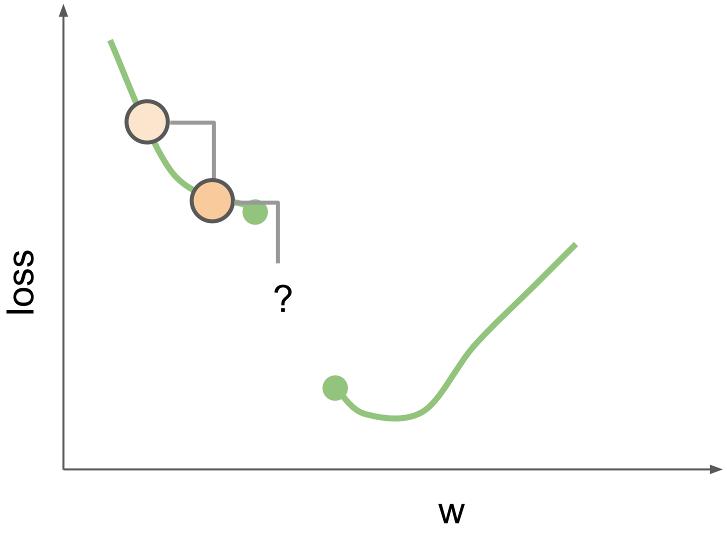


The direction of the move is defined by the slope of the function, aka its derivative

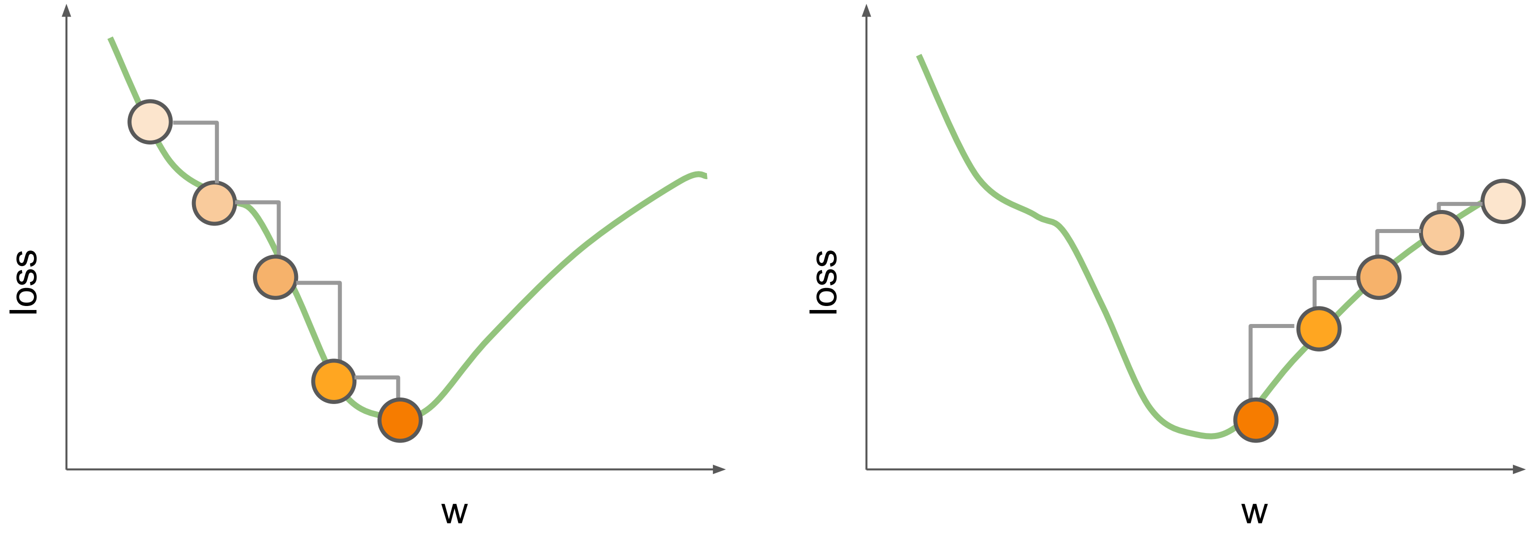


Loss function should be continuous

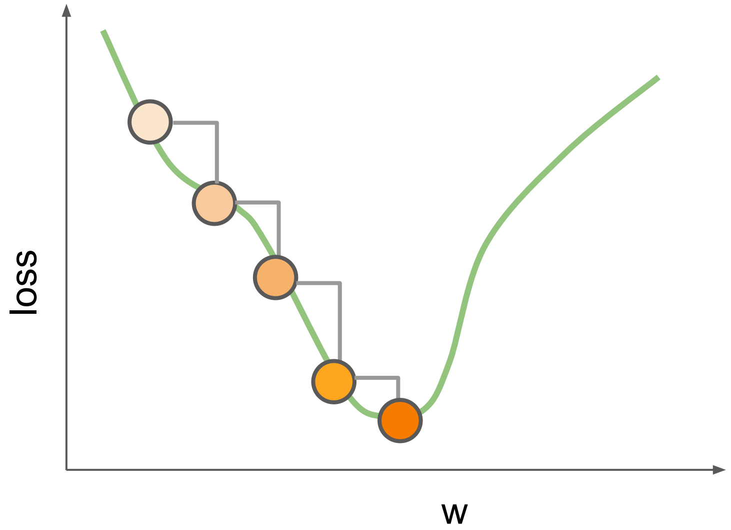




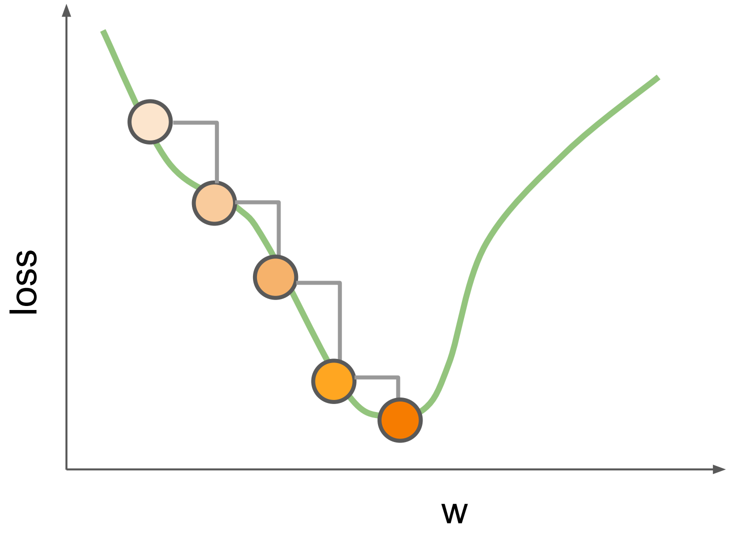
Starting with the right weights help with more efficient convergence

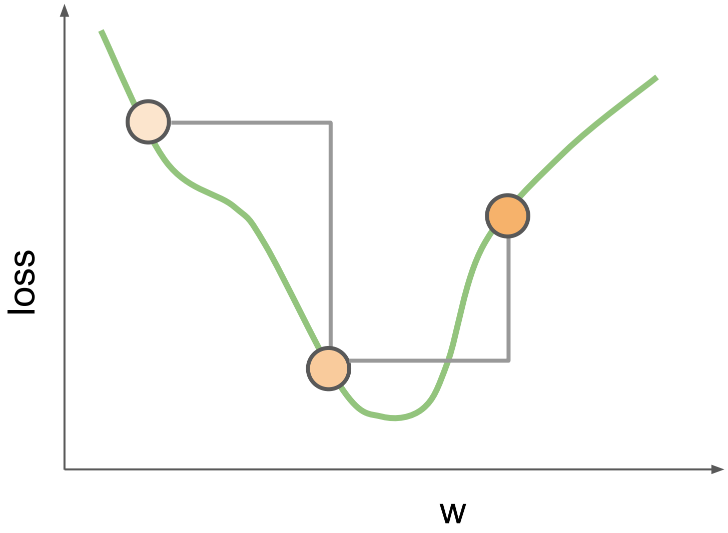


Step size is important in the success of gradient descent

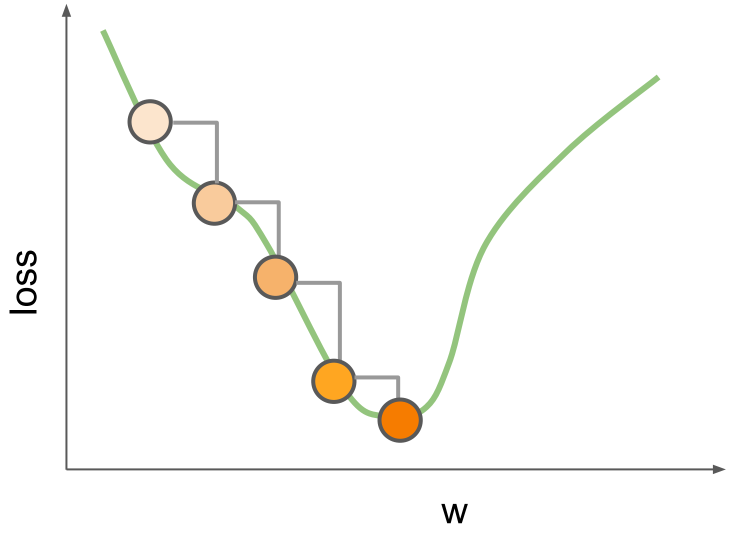


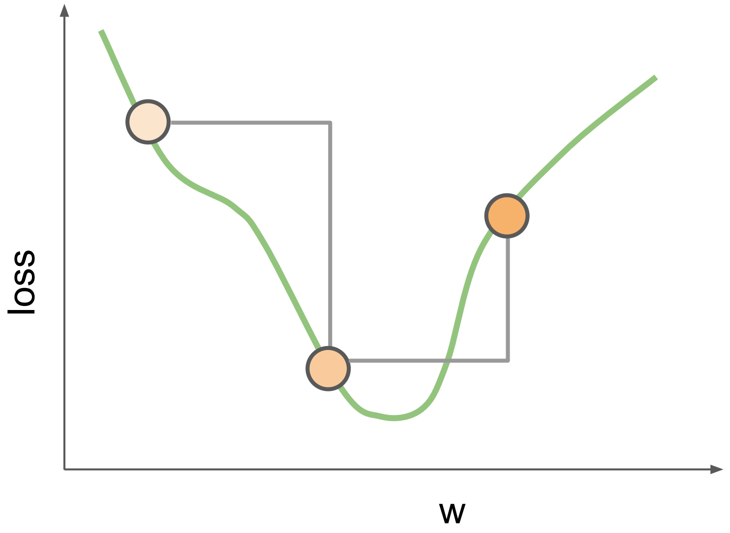
Step size is important in the success of gradient descent



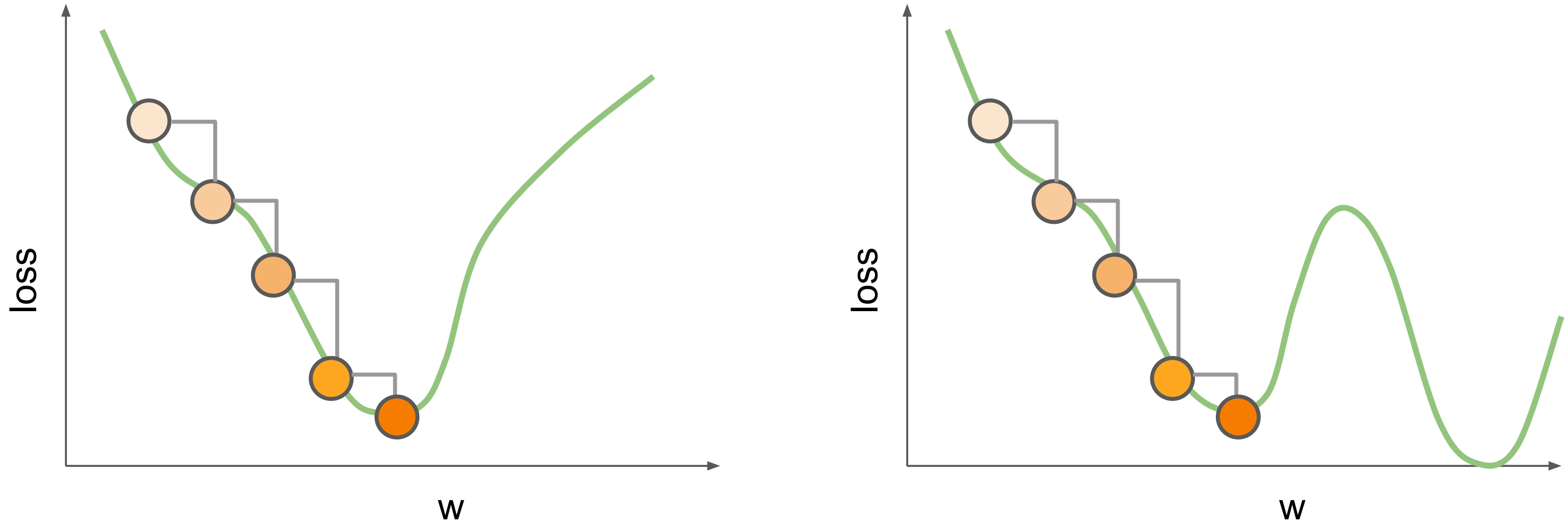


The parameter that describes this step is called *learning rate*





Finding the best answer is not always guaranteed



The importance of scaling the input data

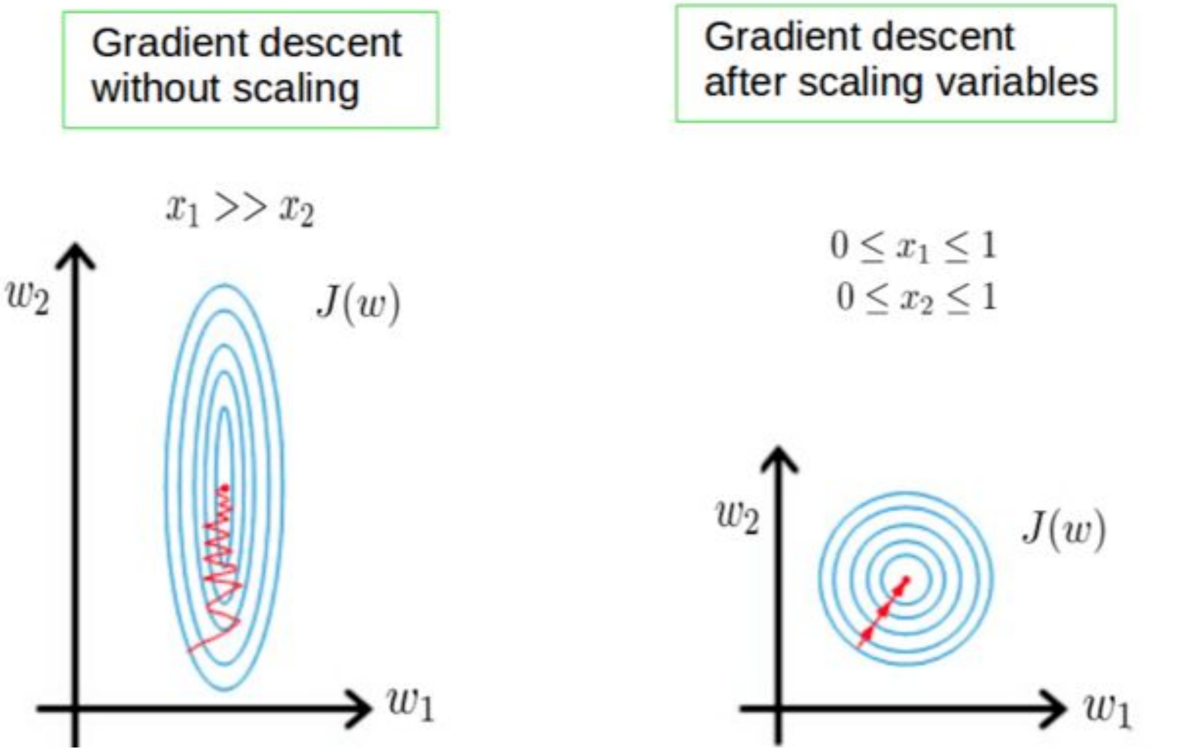
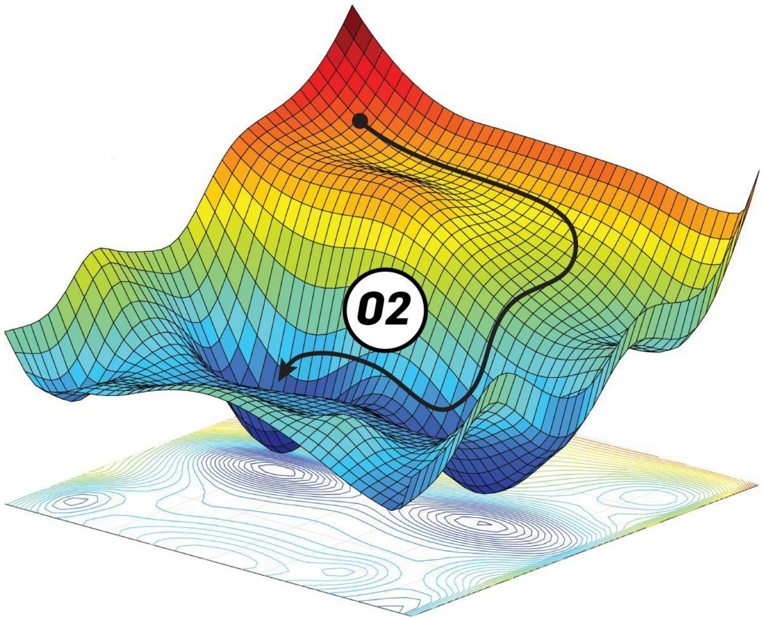


Image from TDS - all about feature scaling

In-class activity

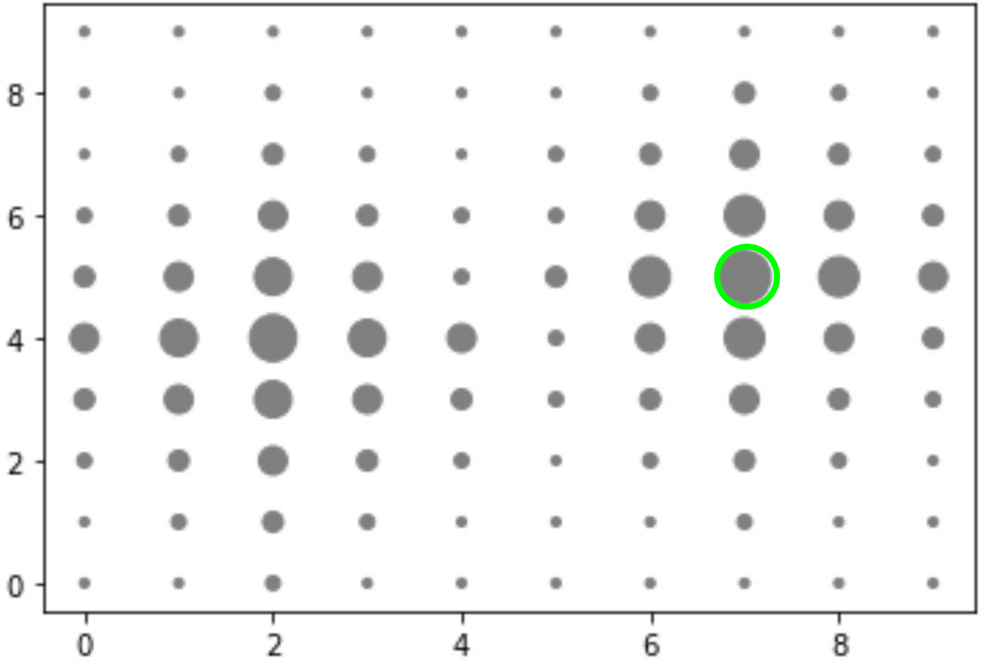
Gradient descent

https://tinyurl.com/2p82ey97



In-class activity

Gradient descent



Optimizers are mathematical functions that are developed to find the best parameters

1. Gradient descent

$$W_{new} = W_{old} - \alpha * \frac{\partial(Loss)}{\partial(W_{old})}$$

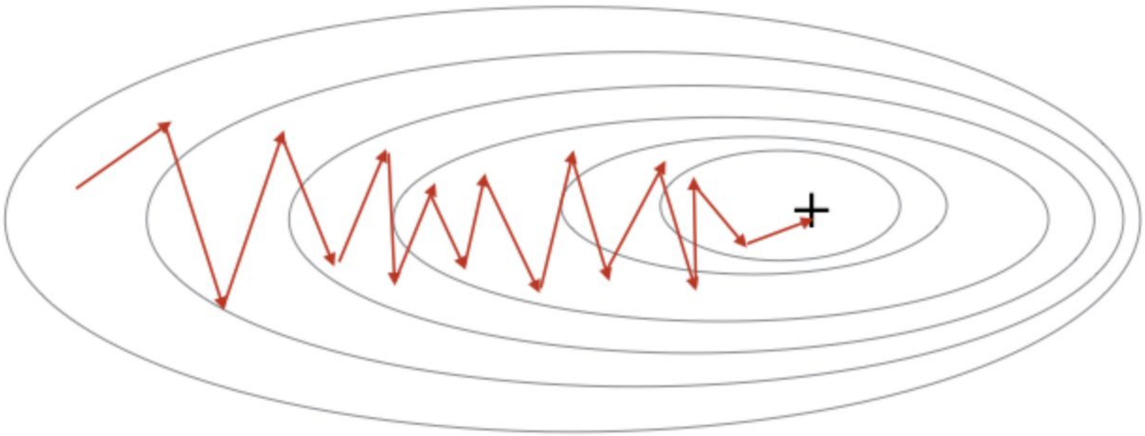
Optimizers are mathematical functions that are developed to find the best parameters

1. Gradient descent
   a. Pros:
      - Easy to understand
      - Easy to implement
   b. Cons:
      - Slow
      - Computationally expensive
      - Large memory

$$W_{new} = W_{old} - \alpha * \frac{\partial(Loss)}{\partial(W_{old})}$$

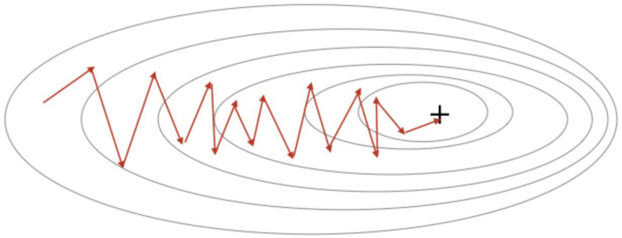
Optimizers are mathematical functions that are developed to find the best parameters

1. Gradient descent
2. Stochastic gradient descent (SGD)



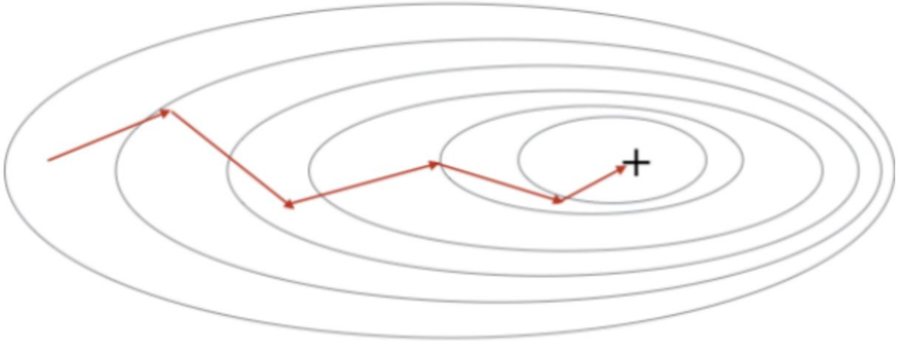
Optimizers are mathematical functions that are developed to find the best parameters

1. Gradient descent
2. Stochastic gradient descent (SGD)
   a. Pros:
      - Frequent update of parameters
      - Less memory
      - Can work in large datasets
   b. Cons:
      - Noisy gradients
      - Computationally expensive
      - High variance



Optimizers are mathematical functions that are developed to find the best parameters

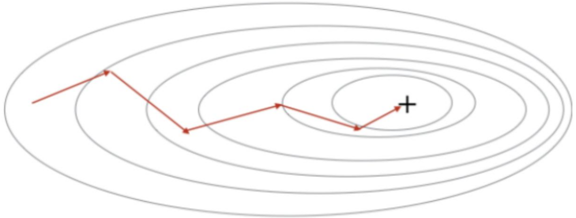
1. Gradient descent
2. Stochastic gradient descent (SGD)
3. Mini-batch gradient descent



Image/formula from medium - Optimizers in deep learning

Optimizers are mathematical functions that are developed to find the best parameters

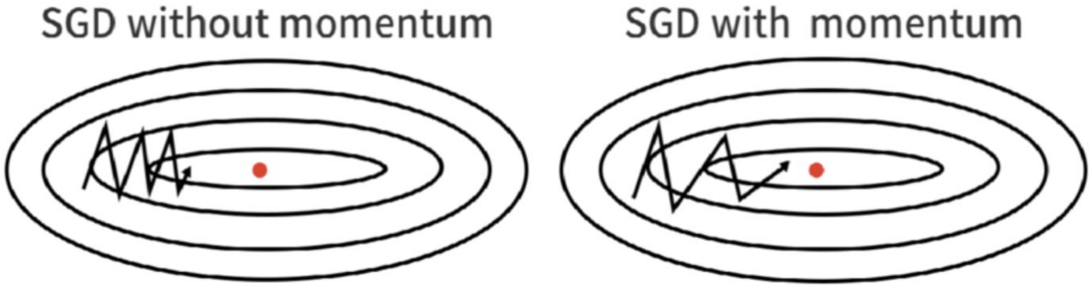
1. Gradient descent
2. Stochastic gradient descent (SGD)
3. Mini-batch gradient descent
    a. Pros:
        - More stable convergence
        - More efficient
        - Less memory
    b. Cons:
        - Does not guarantee good convergence
        - Very dependent on learning rate



Optimizers are mathematical functions that are developed to find the best parameters

1. Gradient descent
2. Stochastic gradient descent (SGD)
3. Mini-batch gradient descent
4. SGD with momentum

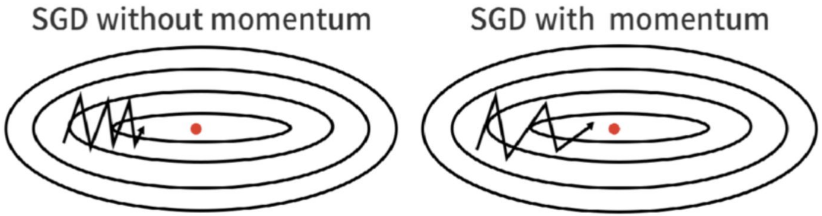
$$\nu_{new} = \eta * \nu_{old} - \alpha * \frac{\partial(Loss)}{\partial(W_{old})}$$



Image/formula from medium - Optimizers in deep learning

Optimizers are mathematical functions that are developed to find the best parameters

1. Gradient descent
2. Stochastic gradient descent (SGD)
3. Mini-batch gradient descent
4. SGD with momentum
    a. Pros:
        - Reduces the noise
        - Smoothens the curve
    b. Cons:
        - Extra hyper-parameter is added



Optimizers are mathematical functions that are developed to find the best parameters

1. Gradient descent
2. Stochastic gradient descent (SGD)
3. Mini-batch gradient descent
4. SGD with momentum
5. AdaGrad (Adaptive gradient descent)

$$W_{new} = W_{old} + \frac{\alpha}{\sqrt{cache_{new} + \epsilon}} * \frac{\partial(Loss)}{\partial(W_{old})}$$

Optimizers are mathematical functions that are developed to find the best parameters

$$W_{new} = W_{old} + \frac{\alpha}{\sqrt{cache_{new}} + \epsilon} * \frac{\partial(Loss)}{\partial(W_{old})}$$

1. Gradient descent
2. Stochastic gradient descent (SGD)
3. Mini-batch gradient descent
4. SGD with momentum
5. AdaGrad (Adaptive gradient descent)
   a. Pros:
      - Learning rate is updated adaptively
      - Can be used on sparse data
   b. Cons:
      - For very deep neural nets, the rate becomes very low → dead neurons

Optimizers are mathematical functions that are developed to find the best parameters

1. Gradient descent
2. Stochastic gradient descent (SGD)
3. Mini-batch gradient descent
4. SGD with momentum
5. AdaGrad (Adaptive gradient descent)
6. RMS-prop

Optimizers are mathematical functions that are developed to find the best parameters

1. Gradient descent
2. Stochastic gradient descent (SGD)
3. Mini-batch gradient descent
4. SGD with momentum
5. AdaGrad (Adaptive gradient descent)
6. RMS-prop
7. AdaDelta

Optimizers are mathematical functions that are developed to find the best parameters

1. Gradient descent
2. Stochastic gradient descent (SGD)
3. Mini-batch gradient descent
4. SGD with momentum
5. AdaGrad (Adaptive gradient descent)
6. RMS-prop
7. AdaDelta
8. Adam (Adaptive Momentum Optimization)

$$\mathrm{w}_{\mathrm{t}} = \mathrm{w}_{\mathrm{t}-1} - \frac{\eta}{\sqrt{\mathrm{S}_{\mathrm{dw}_{\mathrm{t}}} - \varepsilon}} * \mathrm{V}_{\mathrm{dw}_{\mathrm{t}}}$$

$$b_{\mathrm{t}} = b_{\mathrm{t}-1} - \frac{\eta}{\sqrt{\mathrm{S}_{db_{\mathrm{t}}} - \varepsilon}} * \mathrm{V}_{db_{\mathrm{t}}}$$

Optimizers are mathematical functions that are developed to find the best parameters

1. Gradient descent
2. Stochastic gradient descent (SGD)
3. Mini-batch gradient descent
4. SGD with momentum
5. AdaGrad (Adaptive gradient descent)
6. RMS-prop
7. AdaDelta
8. **Adam (Adaptive Momentum Optimization)**
   a. Easy to implement, efficient, little memory requirement

$$\mathrm{w}_{\mathrm{t}} = \mathrm{w}_{\mathrm{t}-1} - \frac{\eta}{\sqrt{\mathrm{S}_{\mathrm{dw}_{\mathrm{t}}} - \varepsilon}} * \mathrm{V}_{\mathrm{dw}_{\mathrm{t}}}$$

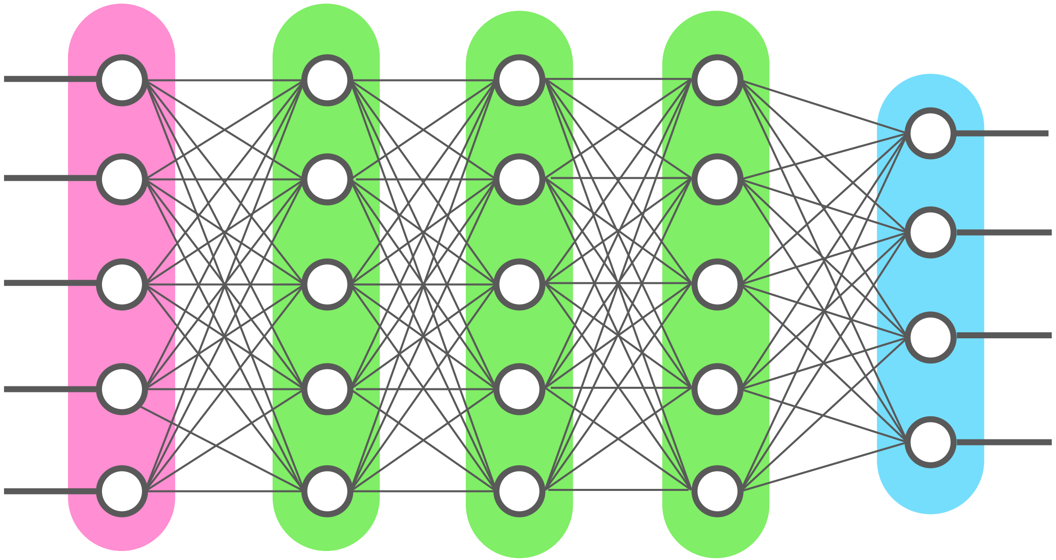
$$b_{\mathrm{t}} = b_{\mathrm{t}-1} - \frac{\eta}{\sqrt{\mathrm{S}_{db_{\mathrm{t}}} - \varepsilon}} * \mathrm{V}_{db_{\mathrm{t}}}$$

Optimizers are mathematical functions that are developed to find the best parameters

Image/formula from medium - Optimizers in deep learning

Next lecture:

Fully connected dense neural nets



Earth Day of Service	
What:	Volunteer opportunity for students to work alongside one of our many different community partners in the Eugene/Springfield area. All students are welcome, regardless of previous volunteering experience, and can receive community service hours that work toward any student org requirements!
When:	· Saturday, April 23rd · 9:00am to 12:00pm
Register:	· Click Here! · Or visit: holden.uoregon.edu/daysofservice to learn more
Deadline:	· Registration is open until April 21st at 11:59pm **OR** when max capacity is reached

Emerging Leadership Project[HW3]	
What:	Year-long, cohort-based program blending leadership development and community service. Students will spend the first half of the year developing and exploring their leadership skills, and the second half of the term implementing their own service project. Students will build connections with their fellow cohort members and have the opportunity to work closely with a professional mentor. The primary learning outcomes are centered around the following, · Leadership Skill Development · Direct Community Service Experience · Career Readiness · Community & Connections
When:	· 2022-2023 academic year, beginning fall term and ending in the spring.
Apply:	· **Click Here!** · Or visit: **holden.uoregon.edu/elp** for more information
Deadline:	· **Sunday, May 1st**