Tutorial 3
Neural Network

Kai CHEN

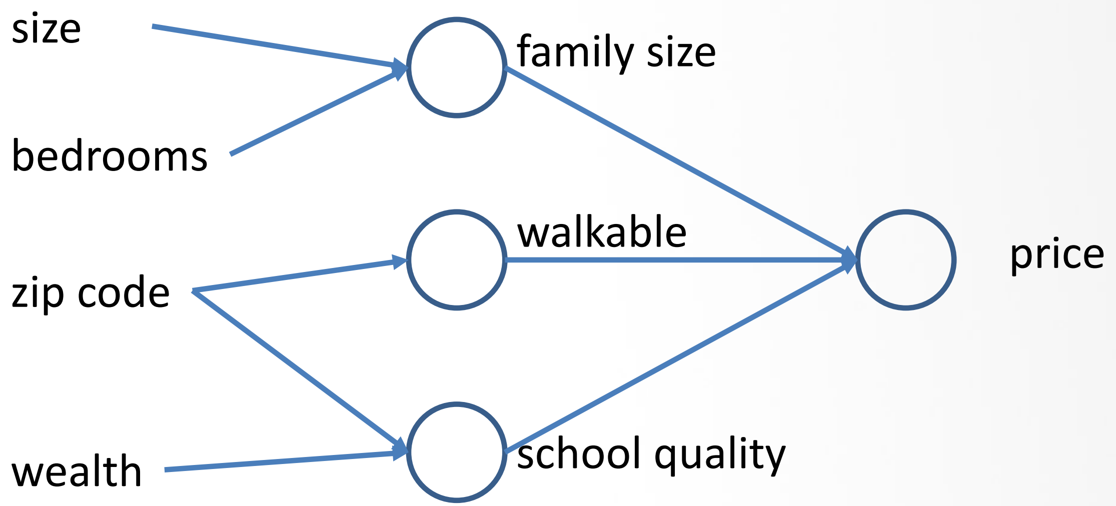
Overview

What is a neural network?

- an information processing paradigm inspired by the way biological nervous systems
- composed of a large number of highly interconnected processing elements (neurons)

Overview

A simple example - regression



size

bedrooms

zip code

wealth

family size

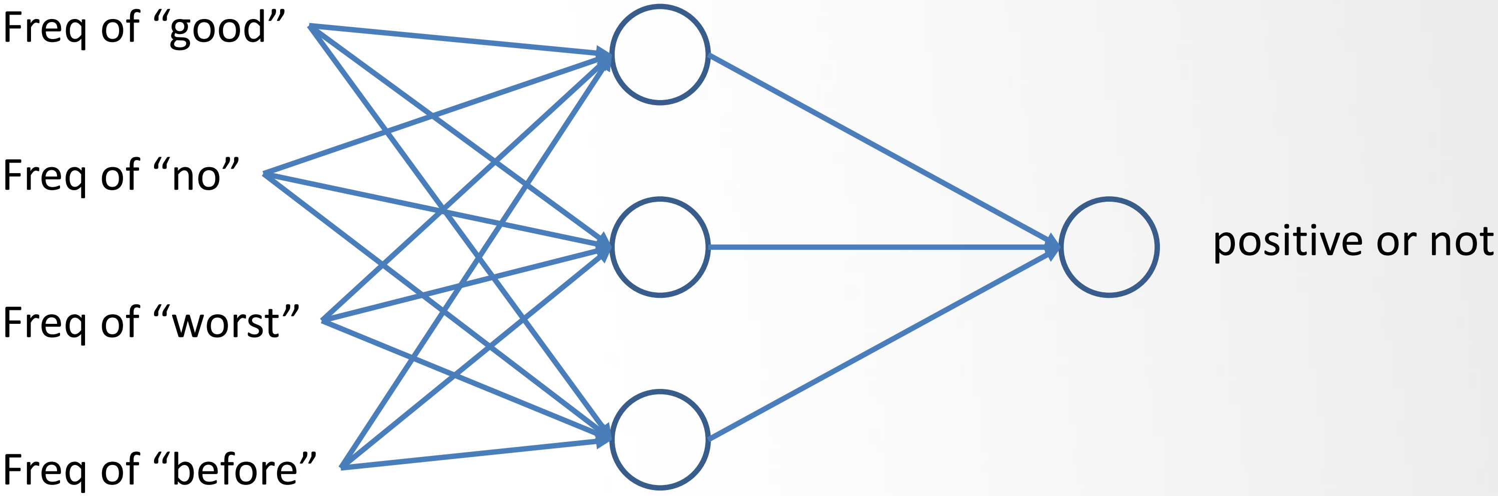
walkable

school quality

price

Overview

A simple example - classification



Freq of “good”

Freq of “no”

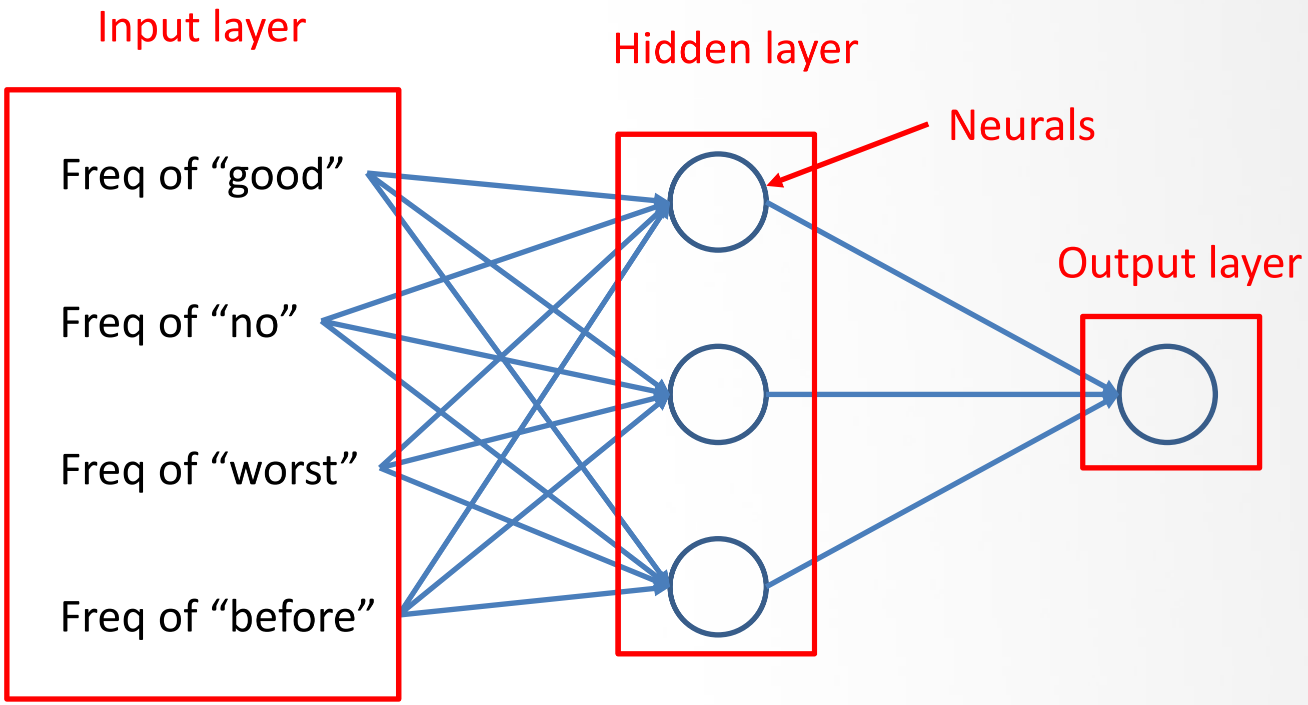
Freq of “worst”

Freq of “before”

positive or not

Overview

A simple example - classification



Input layer

Freq of “good”

Freq of “no”

Freq of “worst”

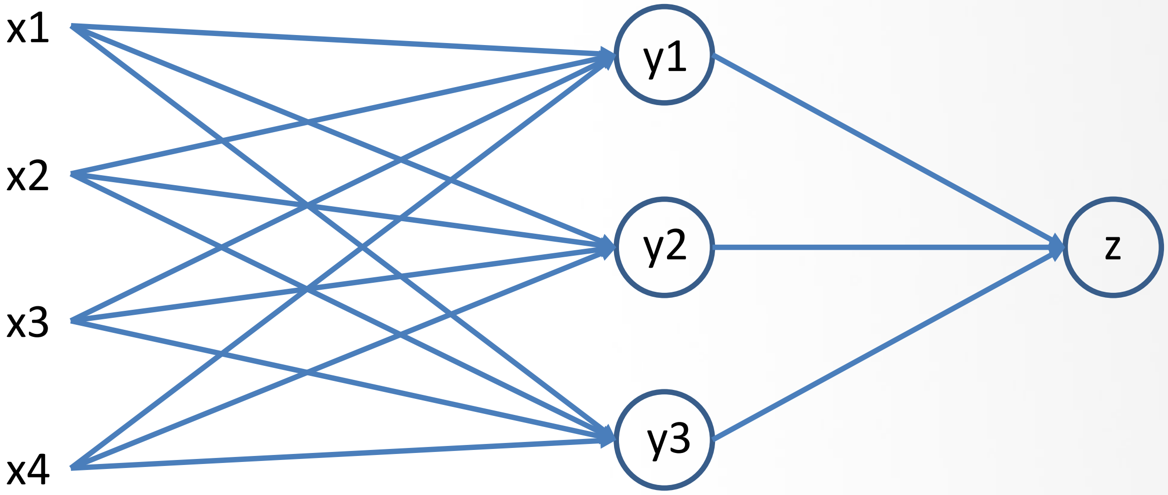
Freq of “before”

Hidden layer

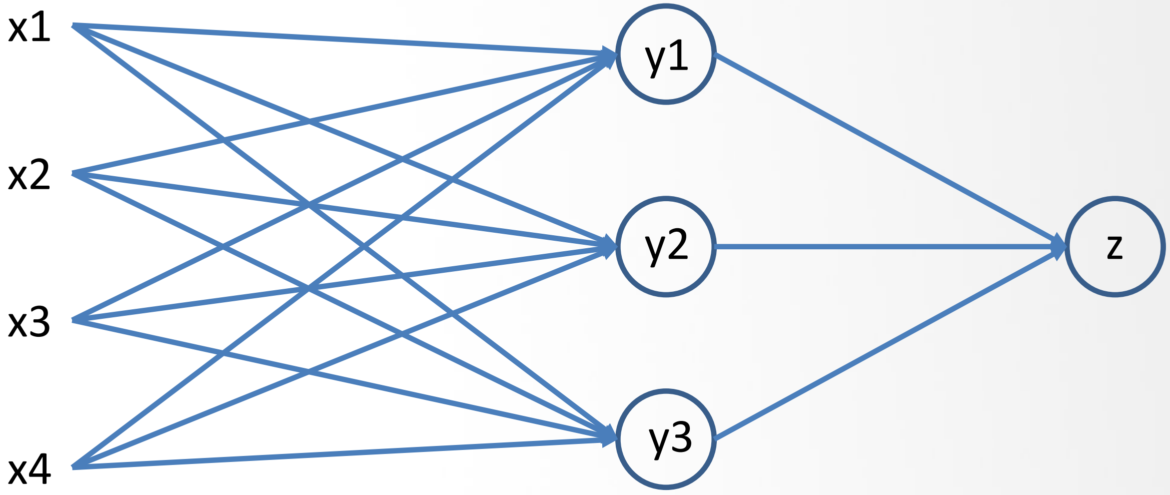
Neurals

Output layer

Representation



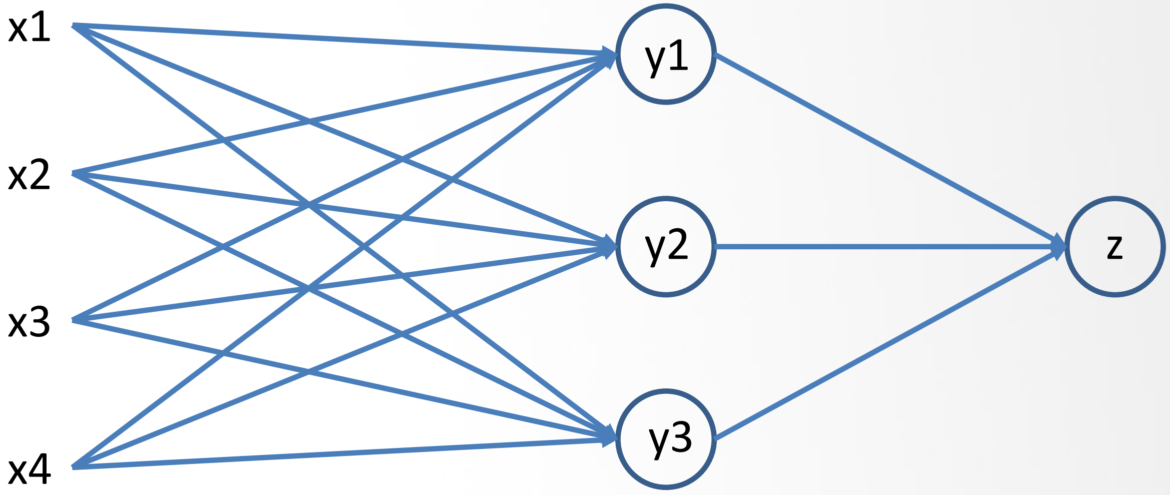
Representation



$$y_1 = g(w_{11}x_1 + w_{12}x_2 + w_{13}x_3 + w_{14}x_4)$$

g: activation function (non-linear)

Representation



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

$$g(x) = \max(x, 0)$$ ReLU

$$g(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$ tanh

Gradient Descent

Cost function

Cost function measures the difference/distance between predicted values and the target values.

The minimum point of the cost function means the least amount of prediction error.

Gradient Descent

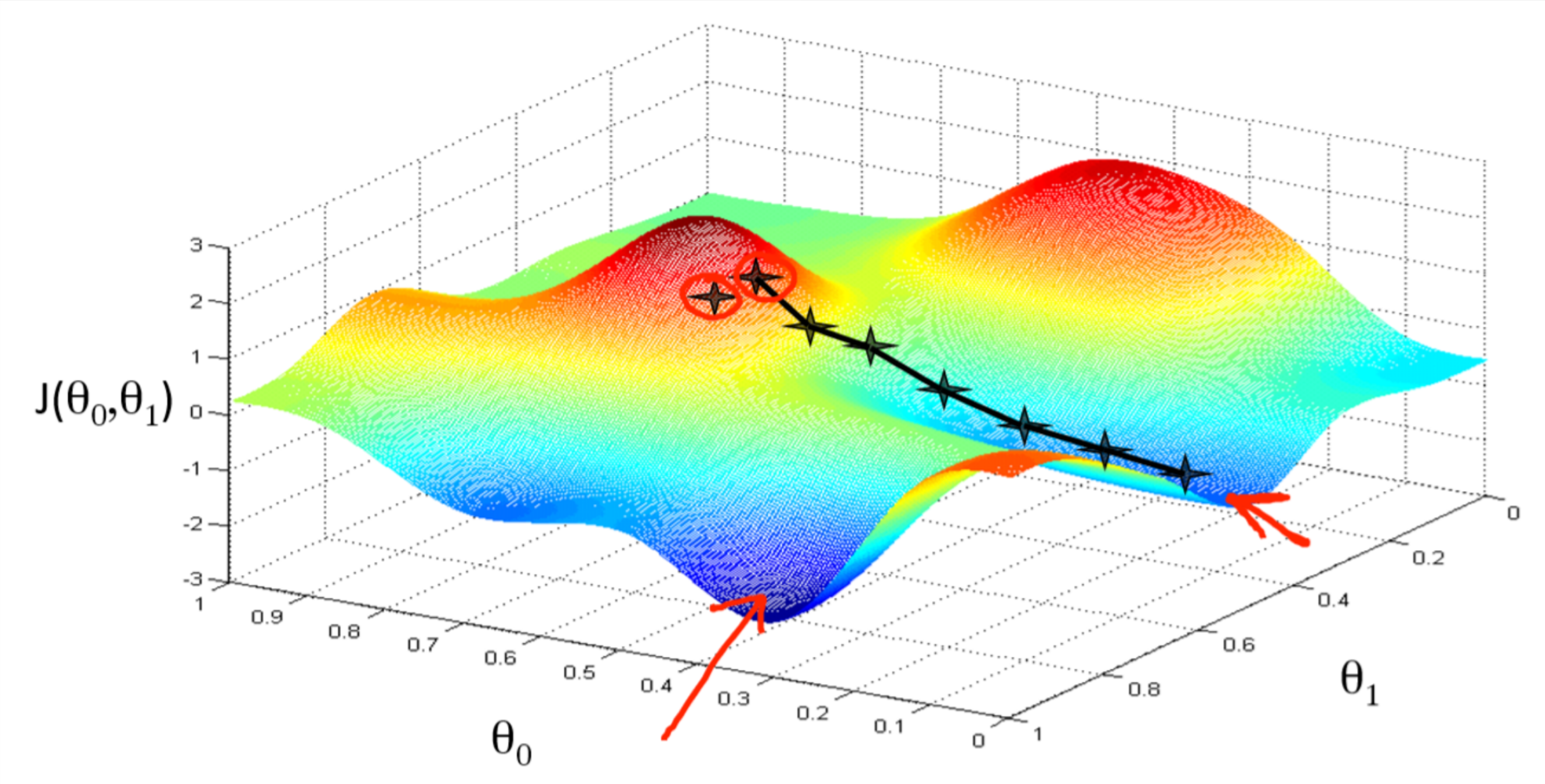
Gradient descent

Gradient descent is an optimization algorithm we use in machine learning to minimizes the cost function.

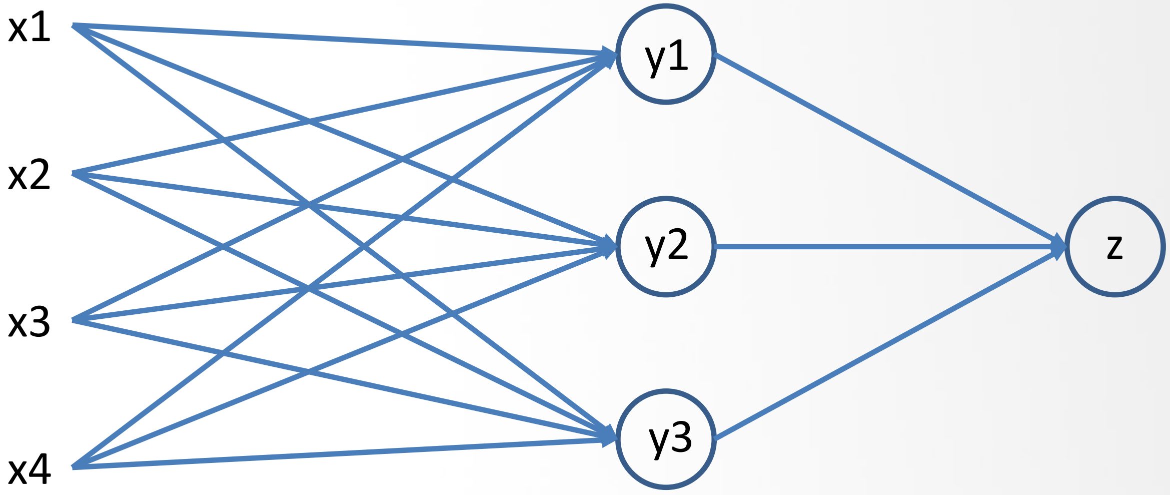
F(x) is differentiable in a neighborhood of a point a, then F(x) decreases fastest of one goes from a in the direction of the negative of gradient of F at a.

$$a_{n+1} = a_n - \gamma \nabla F(a_n)$$

Gradient Descent



Back Propagation



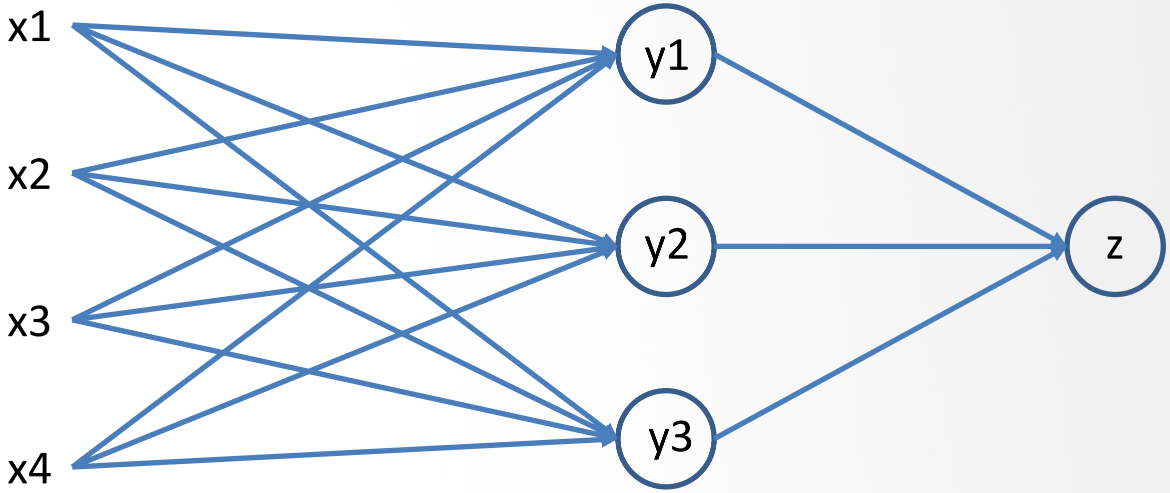
$Y = f(X; \Theta_1)$

$z = g(Y; \Theta_2)$

$J(\Theta) = L(z, \hat{z})$

$\Theta = (\Theta_1, \Theta_2)$ is the parameters of the network.
The cost function J is a function of Θ

Back Propagation



$$\frac{\partial J}{\partial \Theta_2} = \frac{\partial J}{\partial z} \cdot \frac{\partial z}{\partial \Theta_2}$$

$$\frac{\partial J}{\partial \Theta_1} = \frac{\partial J}{\partial z} \cdot \frac{\partial z}{\partial Y} \cdot \frac{\partial Y}{\partial \Theta_1}$$

chain rule + gradient descent

Back Propagation

Usually, we do not need to implement backpropagation by ourselves.

Modern deep learning frameworks like PyTorch and TensorFlow provides the autograd functionality.

Back Propagation

Resources:

CS229: Machine Learning
http://cs229.stanford.edu/notes/cs229-notes-deep_learning.pdf
CS231n: Convolutional Neural Networks for Visual Recognition

Thank you!