KAIST

# CS572 Lecture Note

11/07

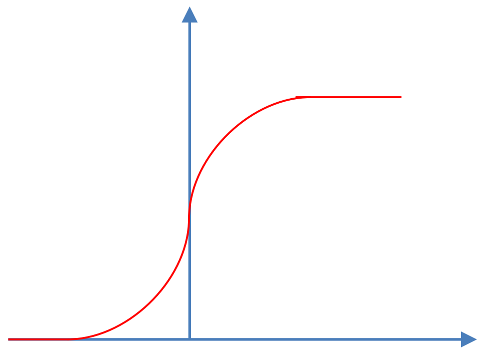
Group 1

Aydar
Chansoo Park
Seungwoo Yoon
Shinhyung Kim

## PPT section

- Classification

| Movie name | Mary's rating | John's rating | I like? |
|---|---|---|---|
| Lard~~~ | 1 | 5 | No |
| … | … | … | … |
| Star ~~ | 4.5 | 4 | Yes |
| Gravity | 3 | 3 | ? |



$H(x:\theta) = \theta 1x1 + \theta 2x2 + b$
$H(x:\theta) = g(\theta Tx + b)$, where $g(z) = 1/(1+\exp(-z))$

h(x(1):θ)=y(1), where x(1)is Mary's and John's rating for 1st movie

h(x(2):θ)=y(2), where x(2)is Mary's and John's rating for 2nd movie

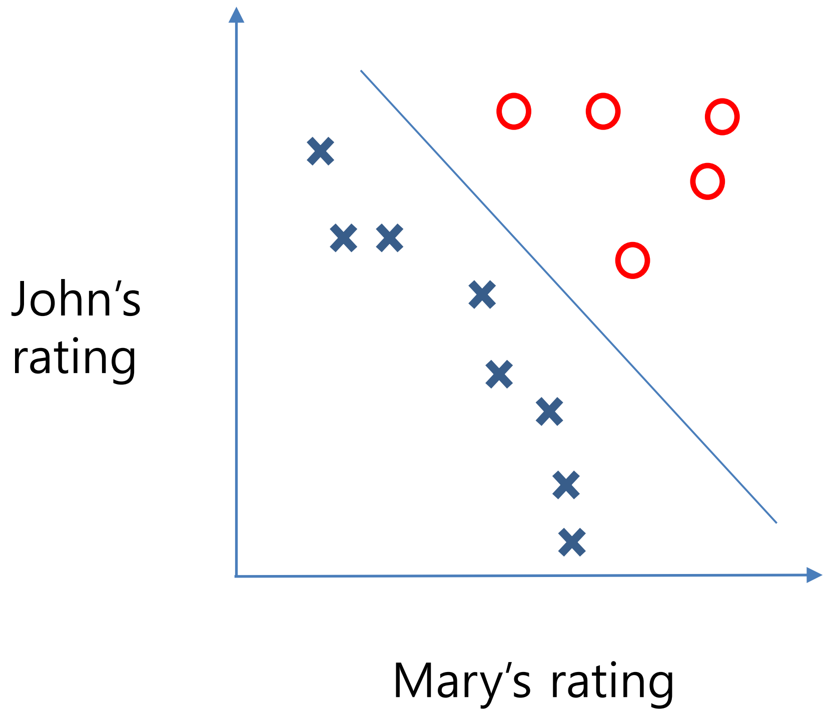
…

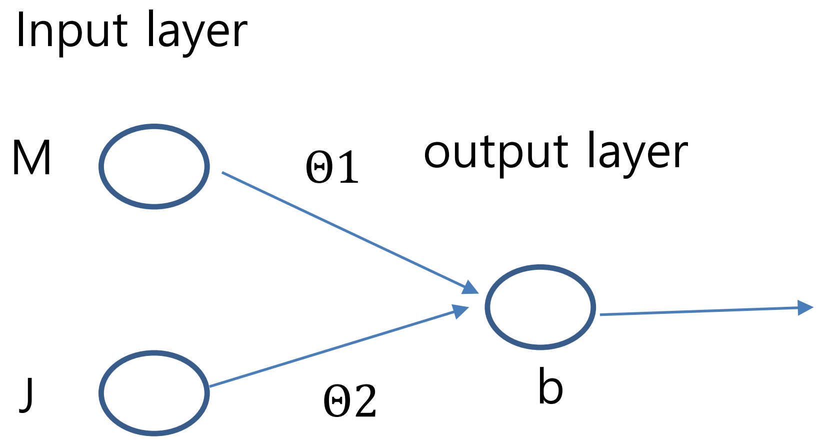
h(x(m):θ)=y(m), where x(m)is Mary's and John's rating for m-th movie

J(θ,b)=(h(x(1):θ,b))^2+(h(x(2):θ,b))^2+….+(h(x(m):θ,b))^2

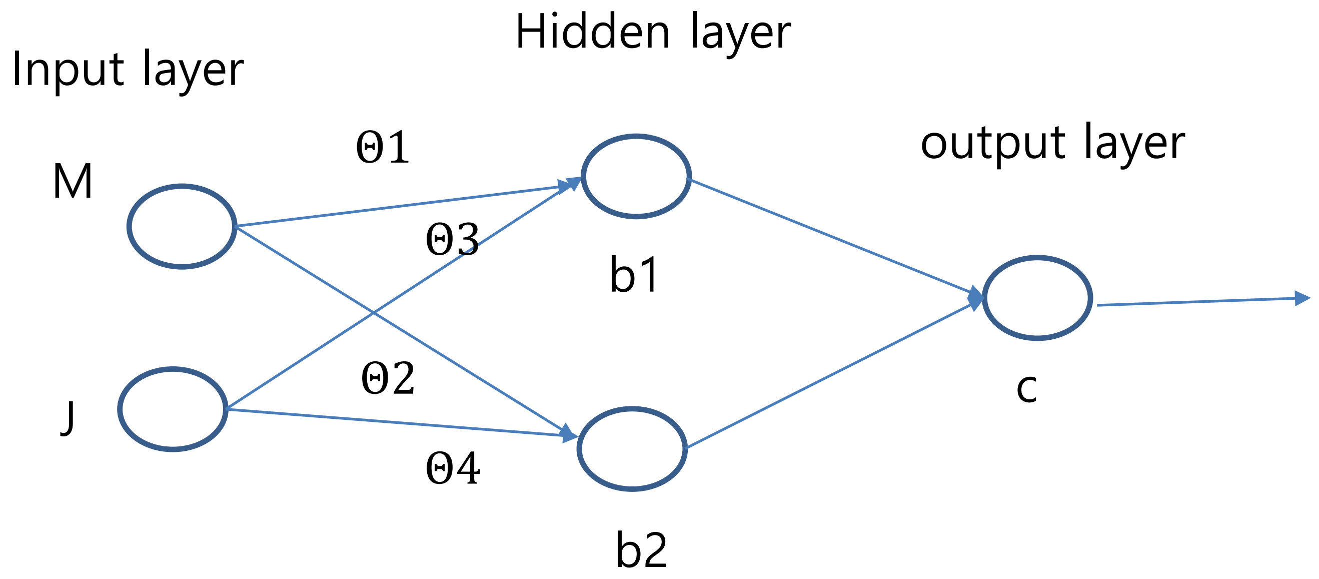
Loss function

To use stochastic Gradient Descent (SGD)

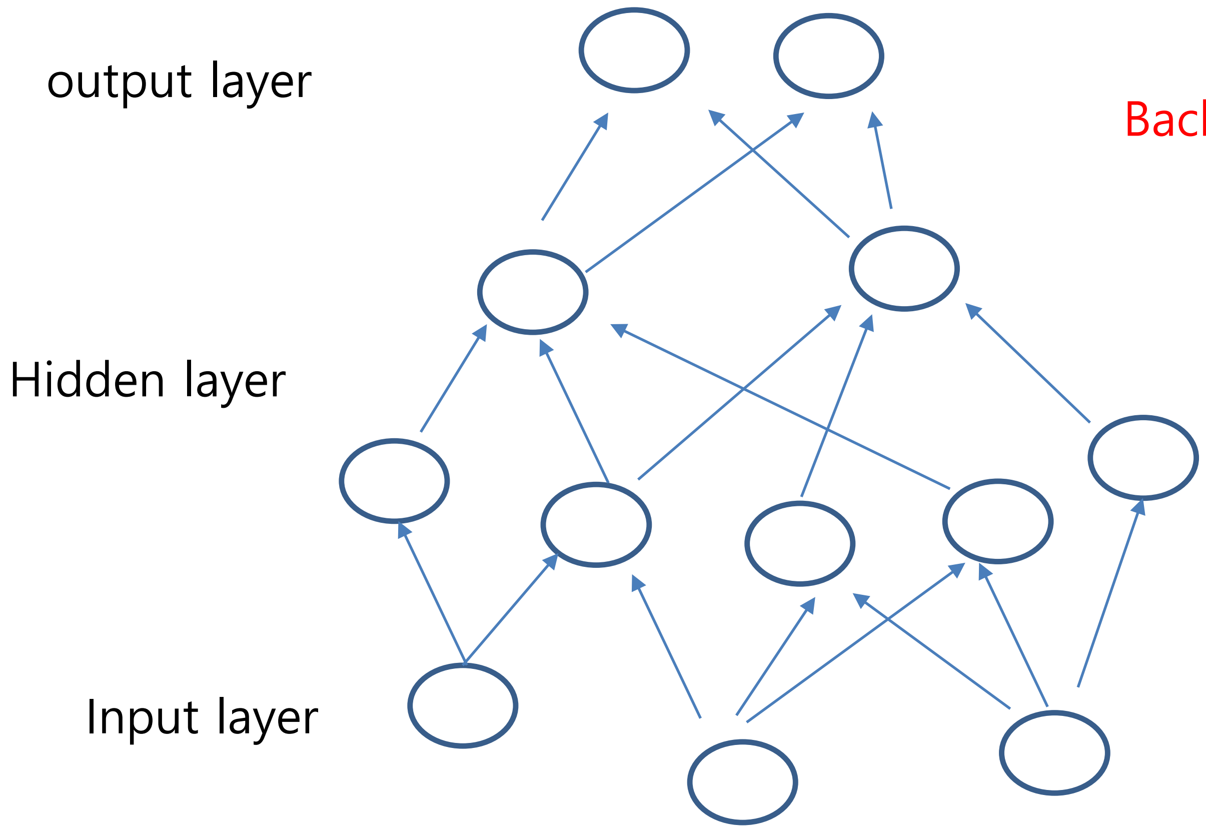




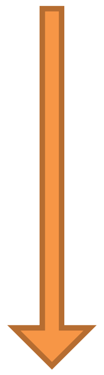
To use Gradient Descent Algorithm


Input layer
Hidden layer
output layer
M
J
ϴ1
ϴ3
ϴ2
ϴ4
b1
b2
c

## How to learn in the case of Deep Network?

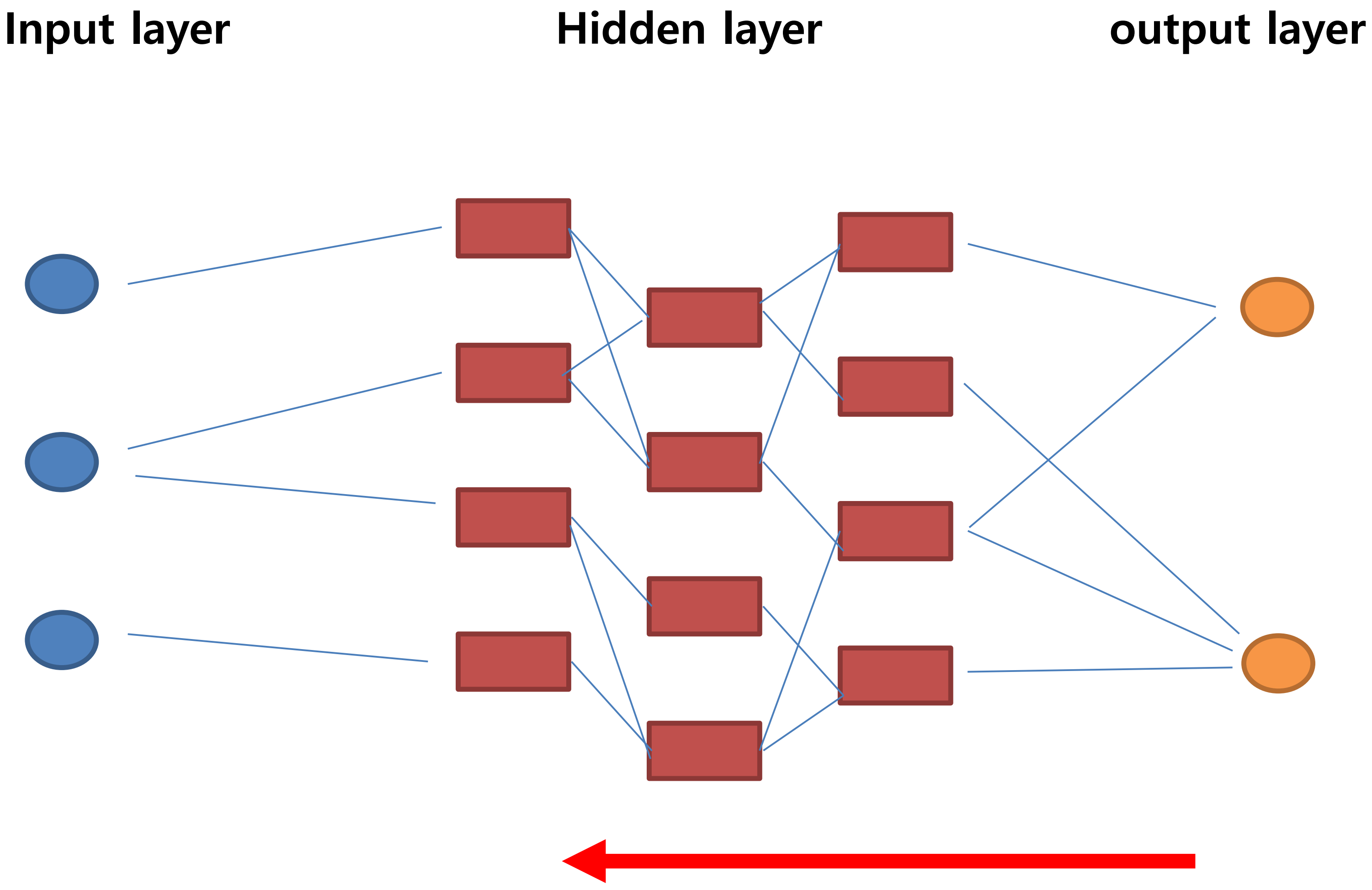


Backpropagation Algorithm



KAIST

## Blackboard section

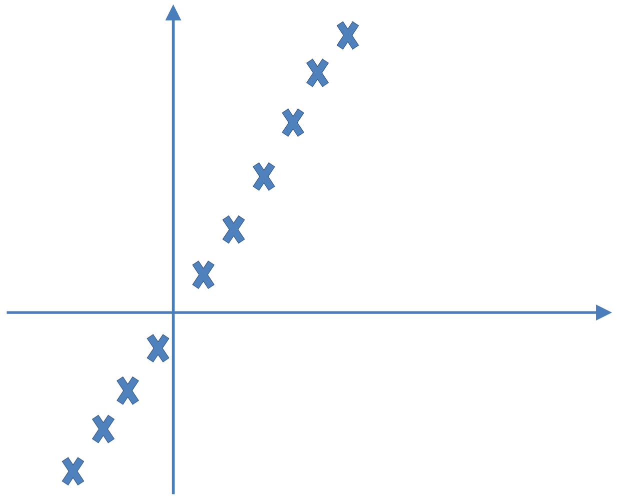


Back propagation (BP)
Back propagation through time (BPTT)

- Unsupervised Learning &data compression
  - ✓ Autoencoder which requires modification in loss fcn
- Translational invariance
  - ✓ Convolutional Neural Networks(CNN) which require in Networks
- Variable – sized sequence prediction
  - ✓ Recurrent Neural Networks(RNN)

- Autoencoders

: pertaining, weight in each layers with unsupervised learning algorithm



$$X_2 = 2X_1$$

$$\underset{\text{2D}}{(X_2, X_1)} \rightarrow \underset{\text{1D}}{X_1}$$

1. Encoding : in machine A, map data $X^{(i)}$to compressed $Z^{(i)}$
2. Sending : send $Z^{(i)}$to other machine B
3. Decoding : in B map from composed data $Z^{(i)}$ back to $X^{(i)}$ which approximate $X^{(i)}$

- Autoencoders steps

1. Encoding : in machine A, map data $X^{(i)}$to compressed $Z^{(i)}$
2. Sending : send $Z^{(i)}$to other machine B
3. Decoding : in B map from composed data $Z^{(i)}$ back to $X^{(i)}$ which approximate $X^{(i)}$

$$Z^{(i)}=W_1X^{(i)}+b_1$$
$$\tilde{X}^{(i)}=W_2Z^{(i)}+b_2$$

$$\begin{aligned}\text{E(w1, w2, b1, b2)}&=\textstyle\sum_{i=1}^{m}(\tilde{X}^{(i)}-X^{(i)})^2\\ &=\textstyle\sum_{i=1}^{m}(W_2\left(W_1X^{(i)}+b1\right)+b2-X^{(i)})^2\end{aligned}$$

$\longrightarrow$ stochastic gradient descent algorithm

# Additional Materials

## Back propagation

Backpropagation is shorthand for "the backward propagation of errors," since an error is computed at the output and distributed backwards throughout the network’s layers. It is commonly used to train deep neural networks, a term referring to neural networks with more than one hidden layer.

Backpropagation is a special case of a more general technique called automatic differentiation. In the context of learning, backpropagation is commonly used by the gradient descent optimization algorithm to adjust the weight of neurons by calculating the gradient of the loss function. This technique is also sometimes called backward propagation of errors, because the error is calculated at the output and distributed back through the network layers.

## Backpropagation Through Time

The training algorithm for updating network weights to minimize error. The goal of the backpropagation training algorithm is to modify the weights of a neural network in order to minimize the error of the network outputs compared to some expected output in response to corresponding inputs.
It is a supervised learning algorithm that allows the network to be corrected with regard to the specific errors made.

The general algorithm is as follows:

1. Present a training input pattern and propagate it through the network to get an output.
2. Compare the predicted outputs to the expected outputs and calculate the error.
3. Calculate the derivatives of the error with respect to the network weights.
4. Adjust the weights to minimize the error.
5. Repeat.

## Stochastic gradient descent

**Stochastic gradient descent** (often shortened to **SGD**), also known as **incremental** gradient descent, is an iterative method for optimizing a differentiable objective function, a stochastic approximation of gradient descent optimization.

Procedure

- Choose an initial vector of parameters and learning rate .
- Repeat until an approximate minimum is obtained:
    - Randomly shuffle examples in the training set.
    - Adjusting initial vector to use parameters

## References

https://en.wikipedia.org/wiki/Backpropagation
https://machinelearningmastery.com/gentle-introduction-backpropagation-time/
https://en.wikipedia.org/wiki/Stochastic_gradient_descent