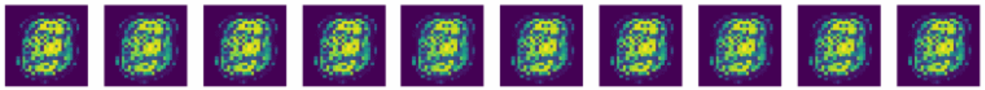
한림대학교 Summer Deep Learning Workshop

Neural Network

July 11, 2017

Seung-Chan Kim, Ph. D



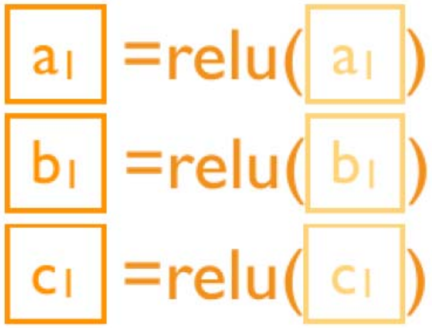
- 1. 머신러닝 개론 및 주요 개념의 이해. Tensorflow 시스템 설치 및 환경설정 (7/4 화)
- 2. Tensorflow 에 익숙해지기 실습 및 Regression의 이해 (7/6 목)
- **3. Neural Network 이해 및 tensorflow 를 이용한 구현 (7/11 화)**
- 4. 이미지 분류 이해 및 Tensorflow를 이용한 구현 (7/13 목)

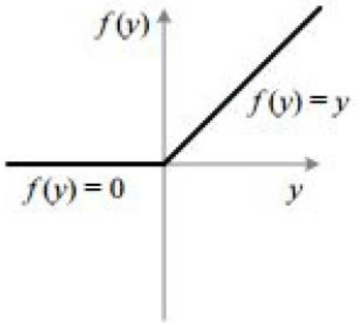
https://github.com/dalek7/DLWorkshop17Summer

복습

- Optimization
- Regression
- Rectified Linear Unit (ReLU)

Rectified Linear Unit (ReLU)



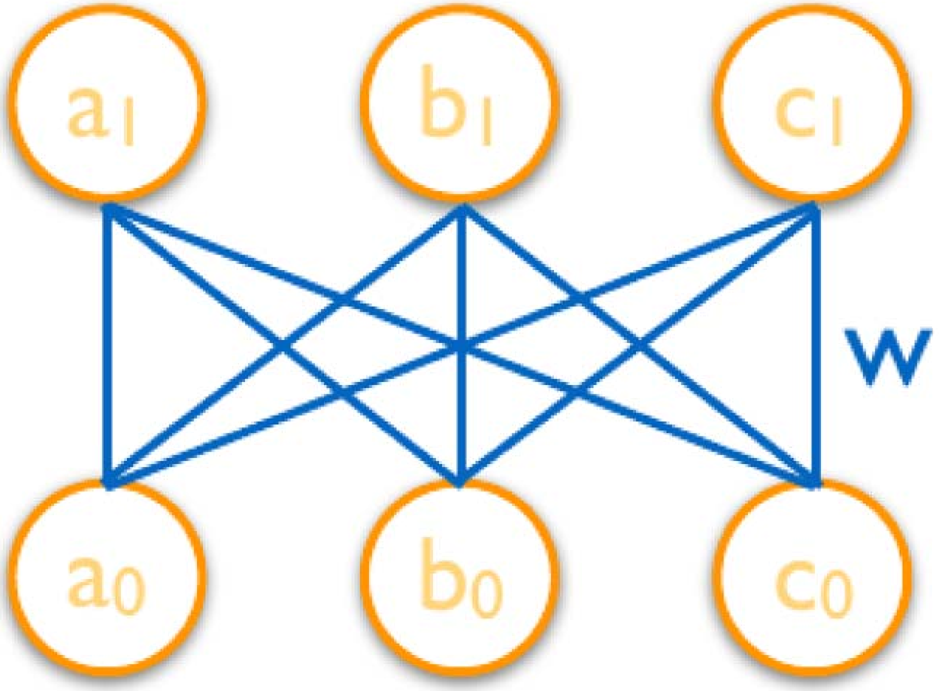


```
out = tf.nn.relu(y)
```

```
(-5, '-->',     )
(-4, '-->',     )
(-3, '-->',     )
(-2, '-->',     )
(-1, '-->',   ? )
( 0, '-->',     )
( 1, '-->',     )
( 2, '-->',     )
( 3, '-->',     )
( 4, '-->',     )
```

00-5-relutest.py 를 열어주세요

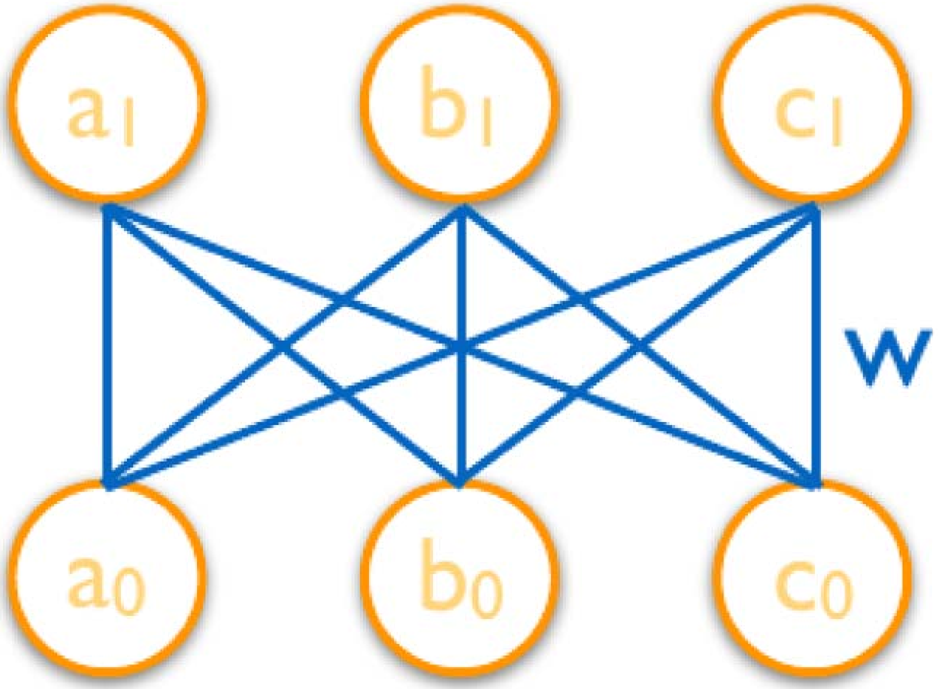
A simple Rectified Linear Unit (ReLU) network



$$\overset{x}{\begin{bmatrix} a_0 & b_0 & c_0 \end{bmatrix}} \cdot \overset{W}{\begin{bmatrix} W_{a,a} & W_{a,b} & W_{a,c} \\ W_{b,a} & W_{b,b} & W_{b,c} \\ W_{c,a} & W_{c,b} & W_{c,c} \end{bmatrix}} = \overset{y}{\begin{bmatrix} a_1 & b_1 & c_1 \end{bmatrix}}$$

```
y = tf.matmul(x,w)
```

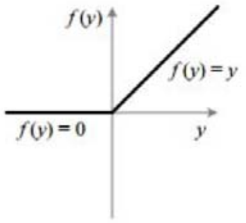
A simple Rectified Linear Unit (ReLU) network



$$\begin{array}{ccccc} x & & w & & \\ \begin{bmatrix} a_0 & b_0 & c_0 \end{bmatrix} & \cdot & \begin{bmatrix} w_{a,a} & w_{a,b} & w_{a,c} \\ w_{b,a} & w_{b,b} & w_{b,c} \\ w_{c,a} & w_{c,b} & w_{c,c} \end{bmatrix} & = & \begin{bmatrix} a_1 & b_1 & c_1 \end{bmatrix} \end{array}$$

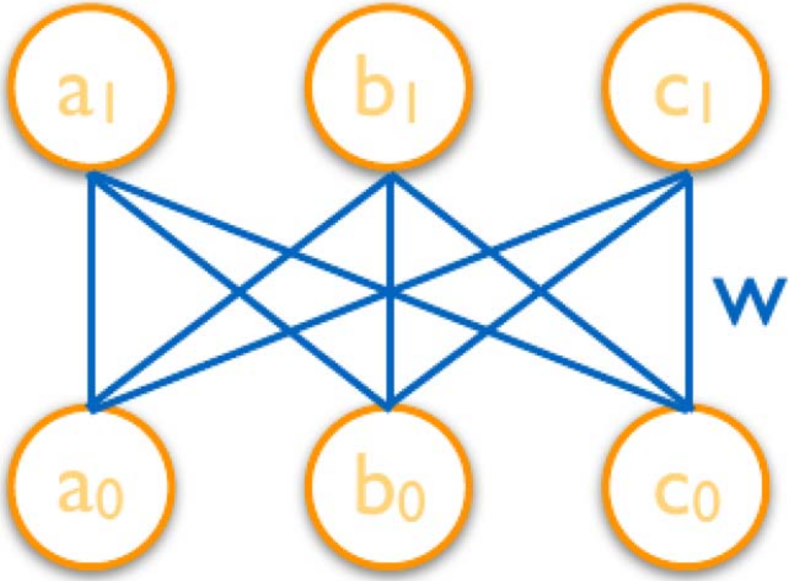
```
y = tf.matmul(x,w)
```

$$\begin{aligned} \boxed{a_1} &= \text{relu}(\boxed{a_1}) \\ \boxed{b_1} &= \text{relu}(\boxed{b_1}) \\ \boxed{c_1} &= \text{relu}(\boxed{c_1}) \end{aligned}$$



```
out = tf.nn.relu(y)
```

A simple Rectified Linear Unit (ReLU) network



```
import tensorflow as tf
sess = tf.Session()
x = tf.placeholder("float", [1, 3])
w = tf.Variable(tf.random_normal([3, 3]), name='w')
y = tf.matmul(x, w)
relu_out = tf.nn.relu(y)
```

01-3-simplenetwork.py 를 열어주세요

Why Neural Networks?

- Consider humans
  - Neuron switching time ~ .001 second
  - Number of neurons ~ 10^{10}
  - Connections per neuron ~ $10^{4\sim5}$
  - Scene recognition time ~ .1 second
  - 100 inference steps doesn't seem like enough
    $\Rightarrow$ massively parallel computation
- Properties of artificial neural nets (ANN)
  - Many neuron-like threshold switching
  - Many weighted interconnections among units
  - Highly parallel, distributed process
  - Emphasis on tuning weights automatically
- Other names: connectionism, parallel distributed processing, neural computation

When to Consider Neural Networks?

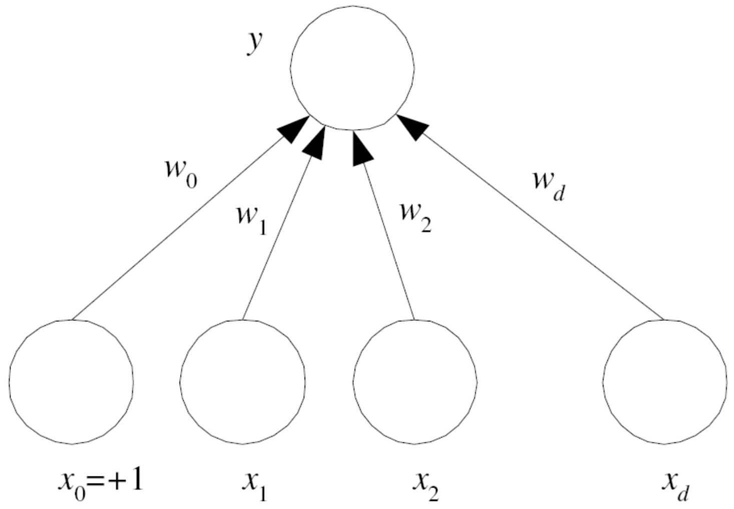
- Input is high-dimensional discrete or real-valued (e.g., raw sensor input)
- Output is discrete or real-valued
- Output is a vector of values
- Possibly noisy data
- Form of target function is unknown
- Human readability of result is unimportant

- Examples
  - Speech phoneme recognition
  - Image classification
  - Financial prediction

Single-Layer Perceptron

- Classification
  - Sigmoid activation function

$$y = \text{sigmoid}(in) = \frac{1}{1 + \exp[-\mathbf{w}^T \mathbf{x}]}$$

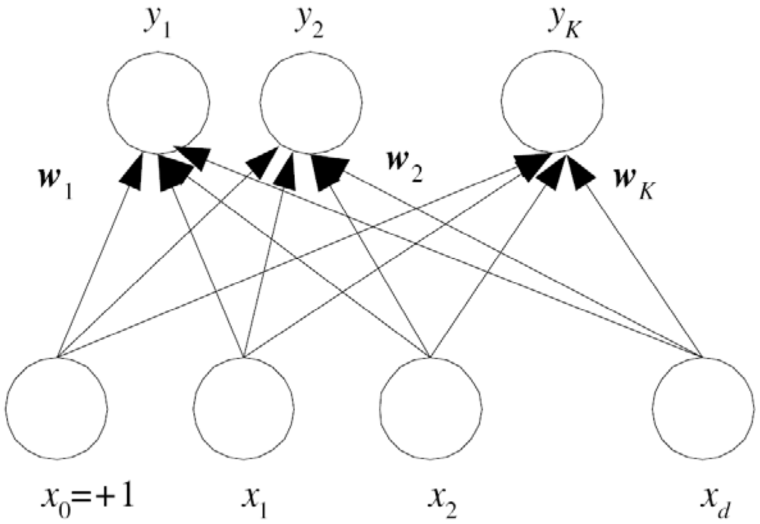


Single-Layer Perceptron with K Outputs

- Classification
  - Sigmoid activation function

$$y_i = \text{sigmoid}(in_i) = \frac{1}{1 + \exp[-\mathbf{w}_i^T \mathbf{x}]}$$

$$\text{choose } C_i \text{ if } y_i = \max_k y_k$$

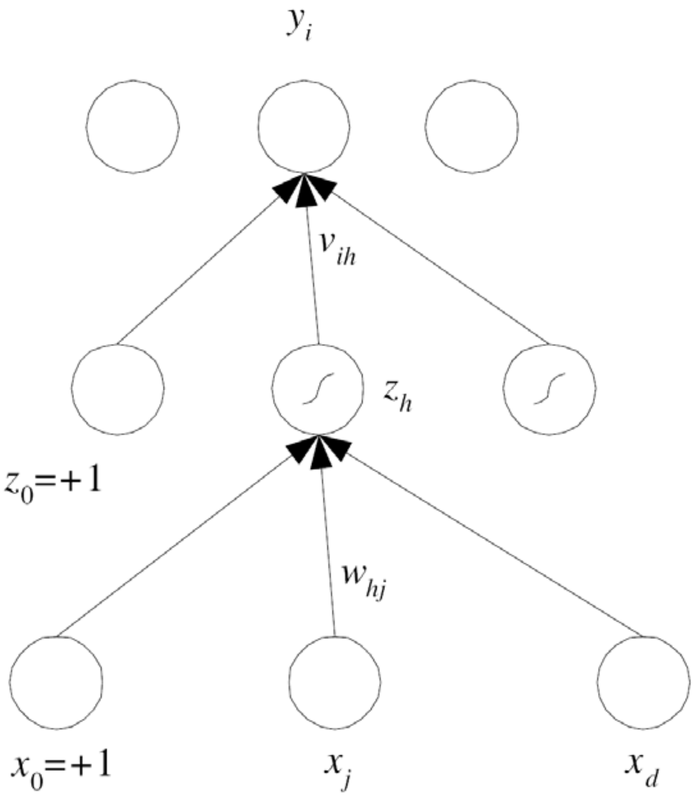


Multi-layer Perceptrons

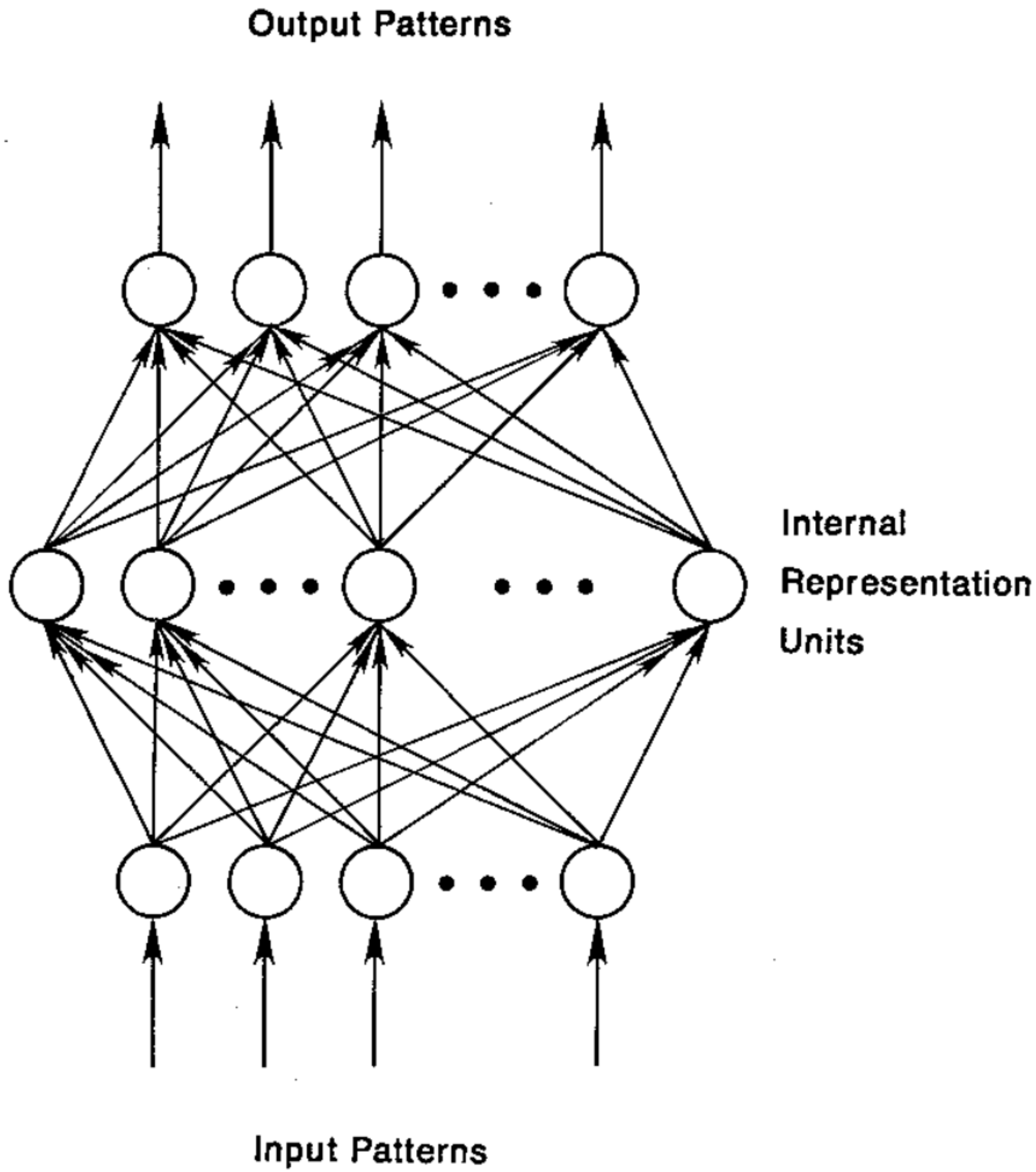
- [Rumelhart et al., 1986]

$$y_i = \mathbf{w}_i^T \mathbf{z} = \sum_{h \in H} w_{hi} z_h + w_{0i}$$

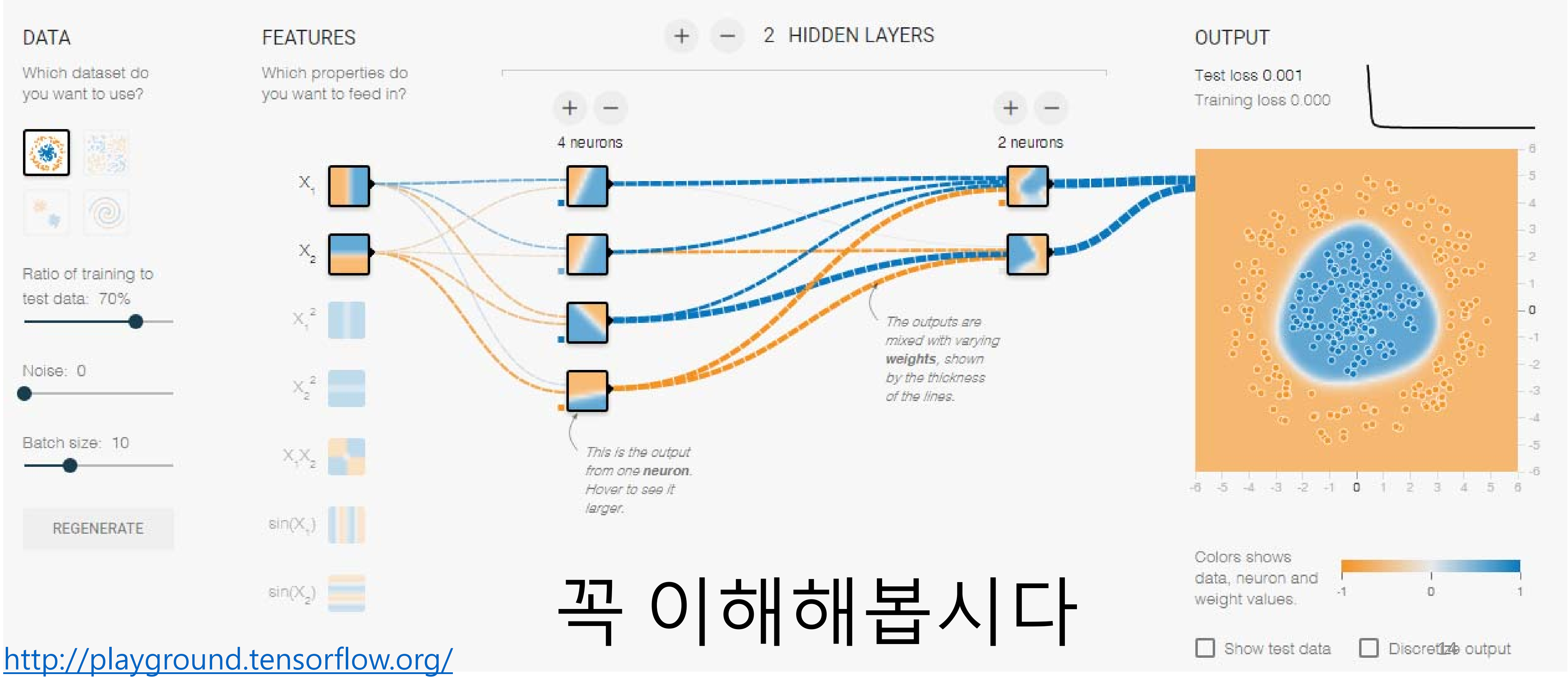
$$\begin{aligned} z_h &= a_h(\mathbf{w}_h^T \mathbf{x}) \\ &= \frac{1}{1 + \exp[-(\sum_{j=1}^{d} w_{jh} x_j + w_{0h})]} \end{aligned}$$



D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," Nature, 10.1038/323533a0 vol. 323, no. 6088, pp. 533-536, 10/09/print 1986.


Output Patterns
Internal
Representation
Units
Input Patterns

Neural Network : playground.tensorflow.org



DATA

Which dataset do you want to use?

Ratio of training to test data: 70%

Noise: 0

Batch size: 10

REGENERATE

FEATURES

Which properties do you want to feed in?

2 HIDDEN LAYERS

4 neurons

2 neurons

*This is the output from one **neuron**. Hover to see it larger.*

*The outputs are mixed with varying **weights**, shown by the thickness of the lines.*

OUTPUT

Test loss 0.001

Training loss 0.000

Colors shows data, neuron and weight values.

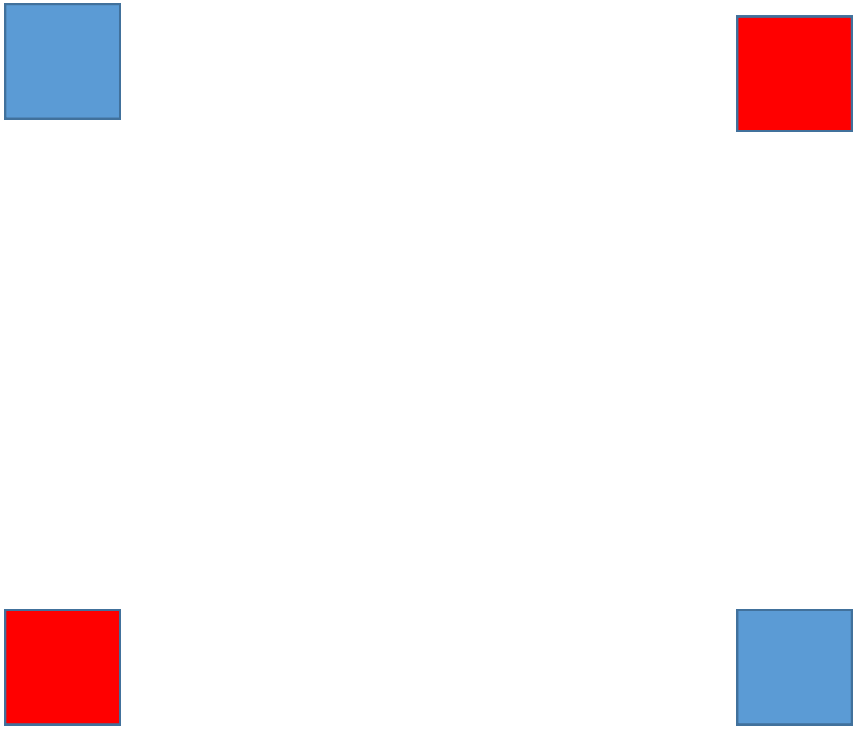
□ Show test data

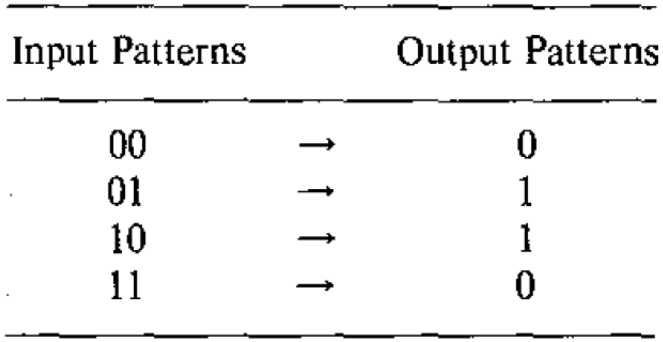
□ Discretize output

꼭 이해해봅시다

http://playground.tensorflow.org/

Recap - XOR 문제 !!



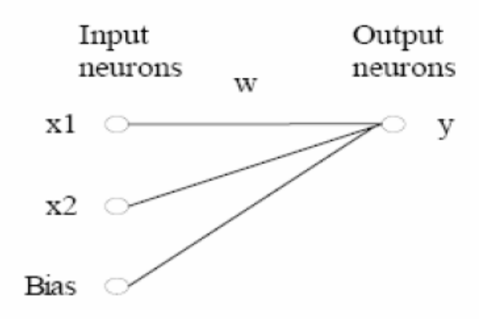


Input Patterns		Output Patterns
00	→	0
01	→	1
10	→	1
11	→	0

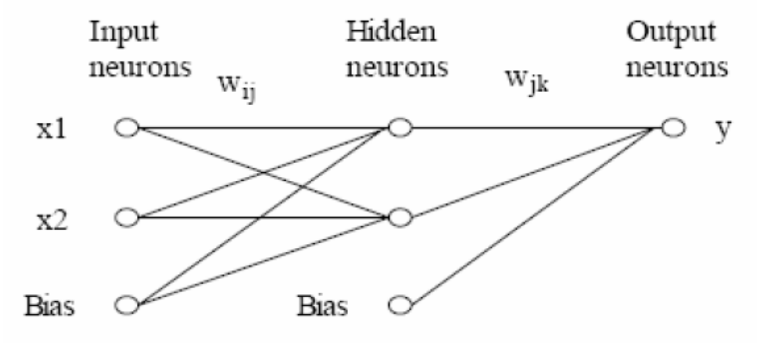
XOR classification using single and two layer neural networks

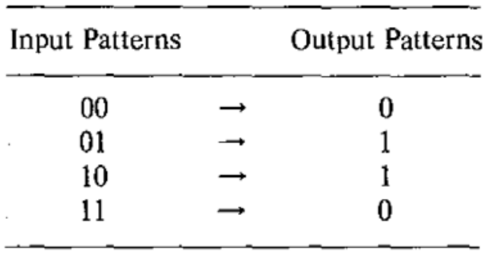
XOR classification using single and two layer neural networks

- **Single layer networks (ADALINE)**

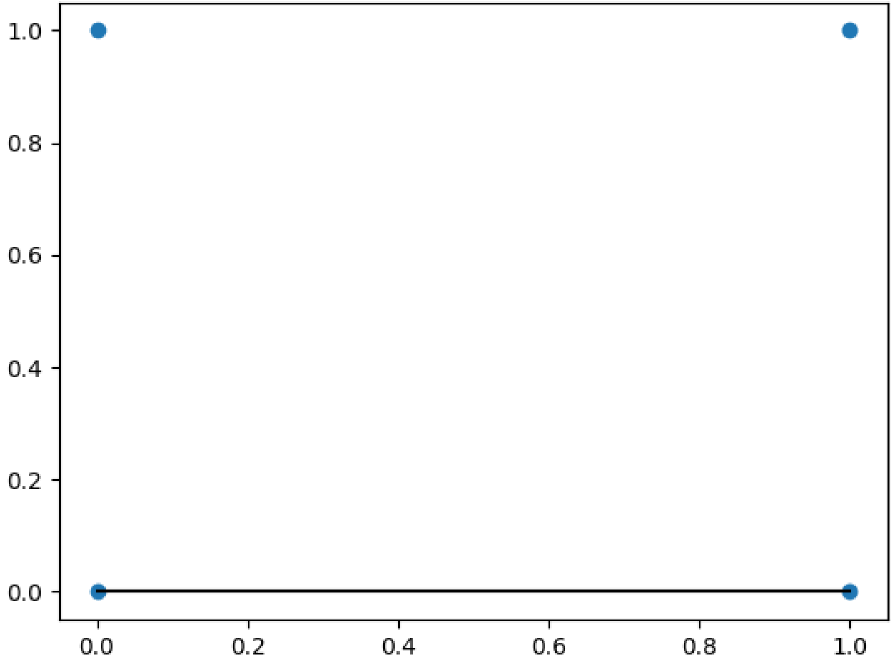


- **An example of two layer neural networks: 2-2-1 structure**

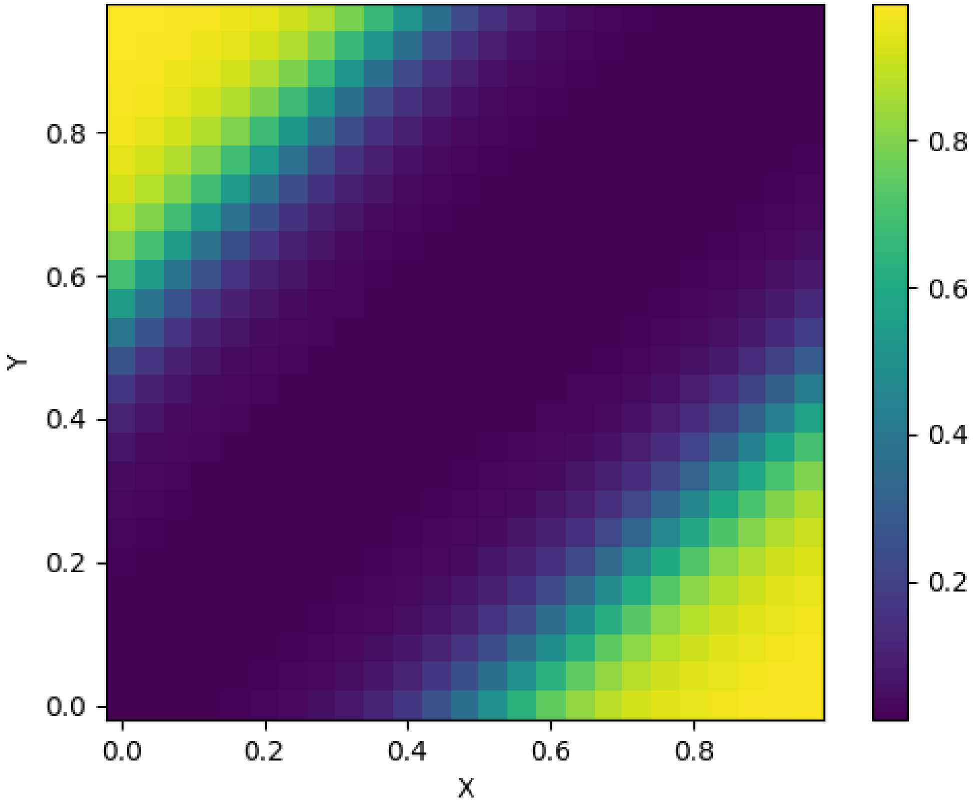




Input Patterns		Output Patterns
00	→	0
01	→	1
10	→	1
11	→	0



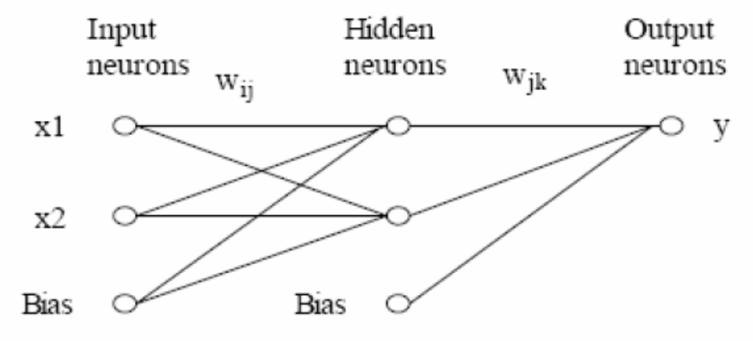
03-1-xor-simple.py 를 열어주세요



```
0 [[ 0.01338216]] [[ 0.]]
1 [[ 0.98166394]] [[ 1.]]
2 [[ 0.98809403]] [[ 1.]]
3 [[ 0.01135799]] [[ 0.]]
```

03-2-xor-layers.py 를 열어주세요

- **An example of two layer neural networks: 2-2-1 structure**



MNIST dataset

- MNIST Dataset
  - Handwritten digits, which has a training set of 60,000 examples and a test set of 10,000 examples.
  - Includes 28 x 28 gray-scaled image and labels for each image.

03-3-mnist-softmax.py를 열어주세요

Acknowledgement

universitätbonn ais

모두를 위한 머신러닝/딥러닝 강의

Seung-chan

Seung-chan

Seung-Chan

감사합니다.