

Artificial Neural Networks: Lecture 3

Statistical classification by deep networks

Objectives for today:

- The cross-entropy error is the optimal loss function for classification tasks
- The sigmoidal (softmax) is the optimal output unit for classification tasks
- Multi-class problems and '1-hot coding'
- Under certain conditions we may interpret the output as a probability
- The rectified linear unit (RELU) for hidden layers

Reading for this lecture:

Bishop 2006, Ch. 4.2 and 4.3

Pattern recognition and Machine Learning

or

Bishop 1995, Ch. 6.7 – 6.9

Neural networks for pattern recognition

or




Goodfellow et al., 2016 Ch. 5.5 and 3.13 of

Deep Learning

Artificial Neural Networks

Wulfram Gerstner

EPFL, Lausanne, Switzerland

1. Simple perceptrons for classification
2. Backprop and multilayer perceptron
3. Statistical Classification by deep networks  miniproject1
4. Deep learning:
regularization and tricks of the trade **TODAY**
5. Complements to deep learning
6. Sequence predictions and LSTMs  miniproject2
7. Convolutional networks
8. Reinforcement learning1: TD learning
9. Reinforcement learning2: SARSA  miniproject3
10. Reinforcement learning3: Policy Gradient
11. Deep Reinforcement learning
12. Applications

miniproject1

Handout TODAY

You will work with

- regularization methods
- cross-entropy error function
- sigmoidal (softmax) output
- rectified linear hidden units
- 1-hot coding for multiclass
- batch normalization
- Adam optimizer

(see last week)

This week

Next week

Artificial Neural Networks: Lecture 3

Statistical classification by deep networks

Objectives for today:

- The cross-entropy error is the optimal loss function for classification tasks
- The sigmoidal (softmax) is the optimal output unit for classification tasks
- Multi-class and '1-hot coding'
- Under certain conditions we may interpret the output as a probability
- The rectified linear unit (RELU) for hidden layers

Review: Data base for Supervised learning (single output)

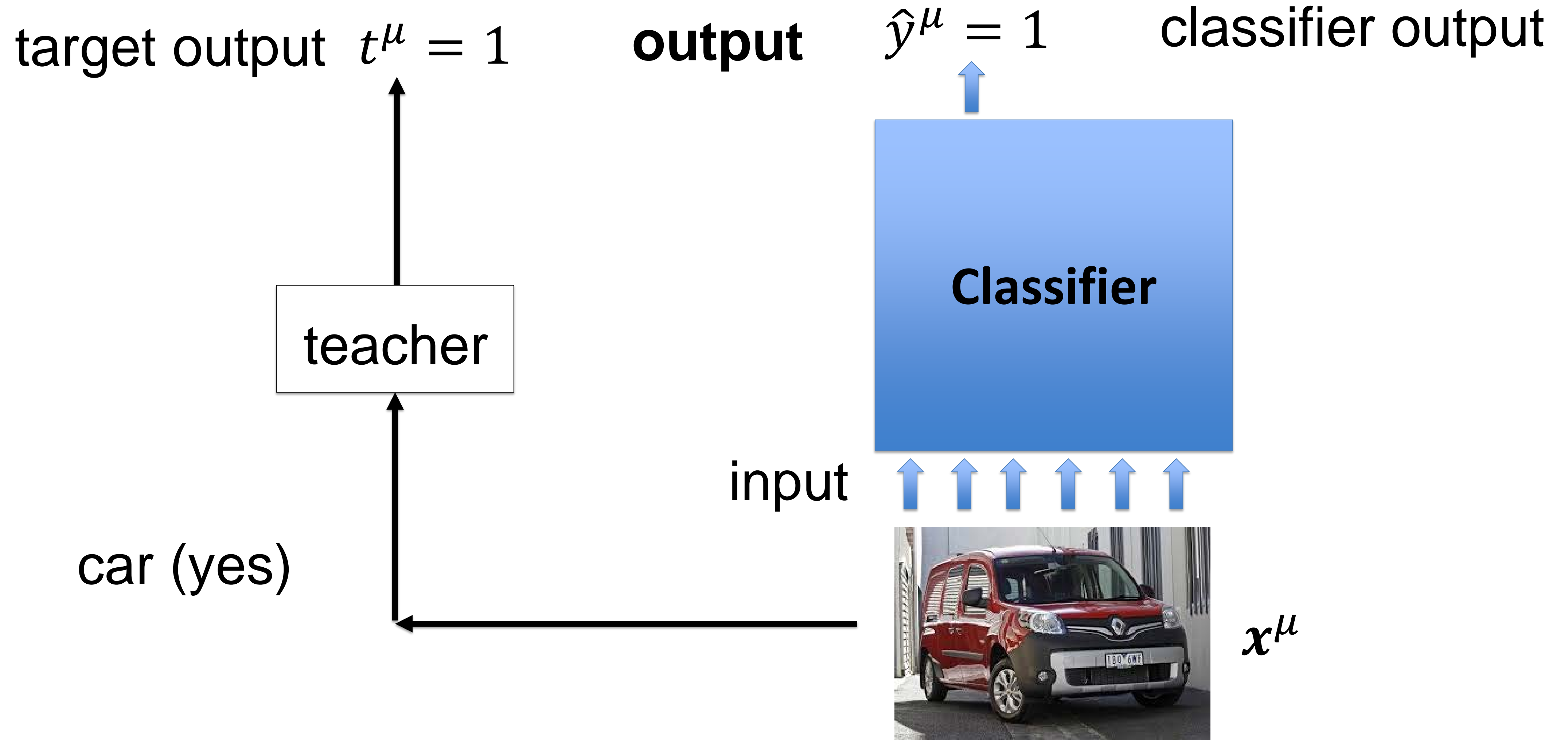
P data points $\{ (\mathbf{x}^\mu, t^\mu) , \quad 1 \leq \mu \leq P \};$


input target output

$t^\mu = 1$ car =yes

$t^\mu = 0$ car =no

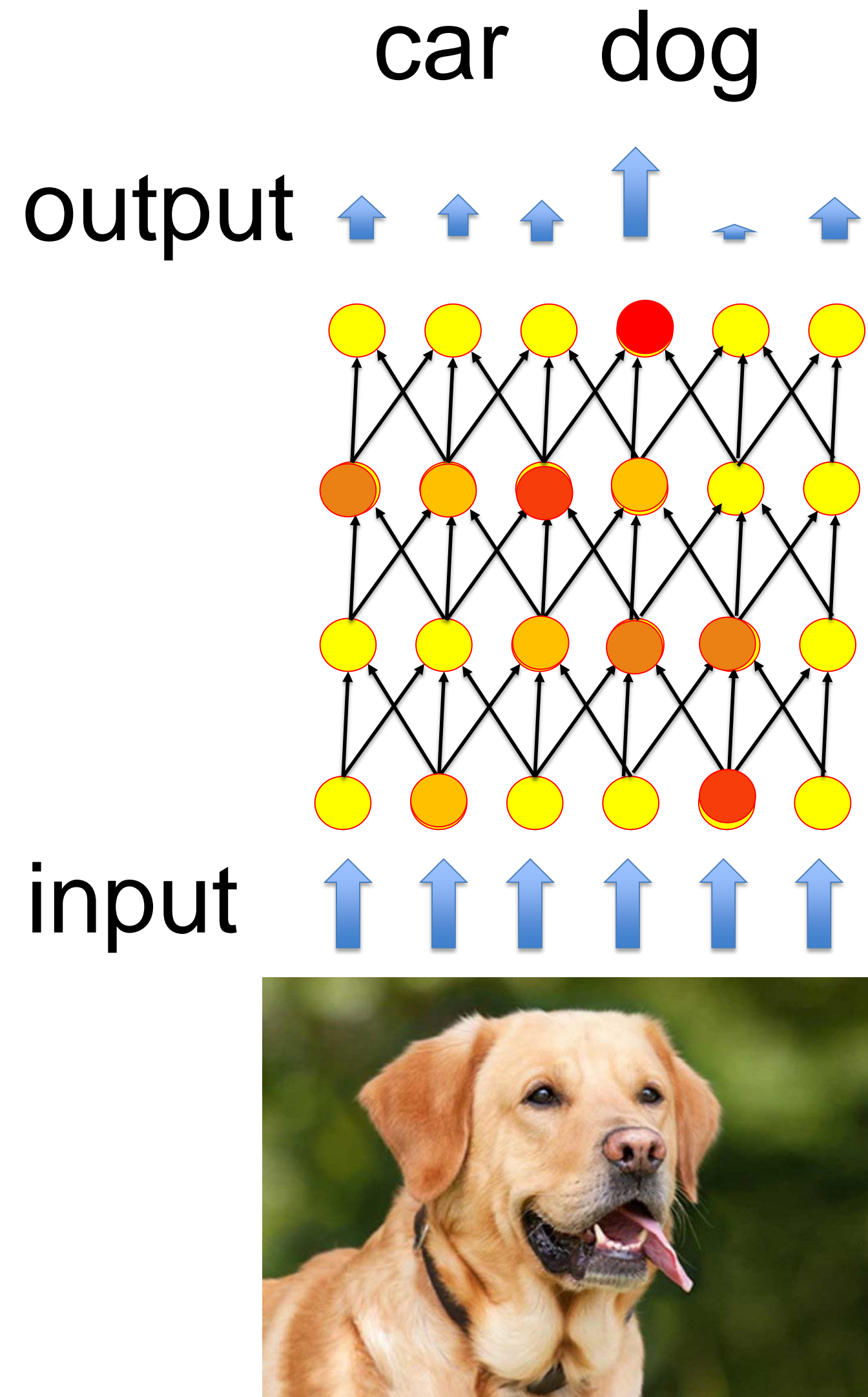
review: Supervised learning



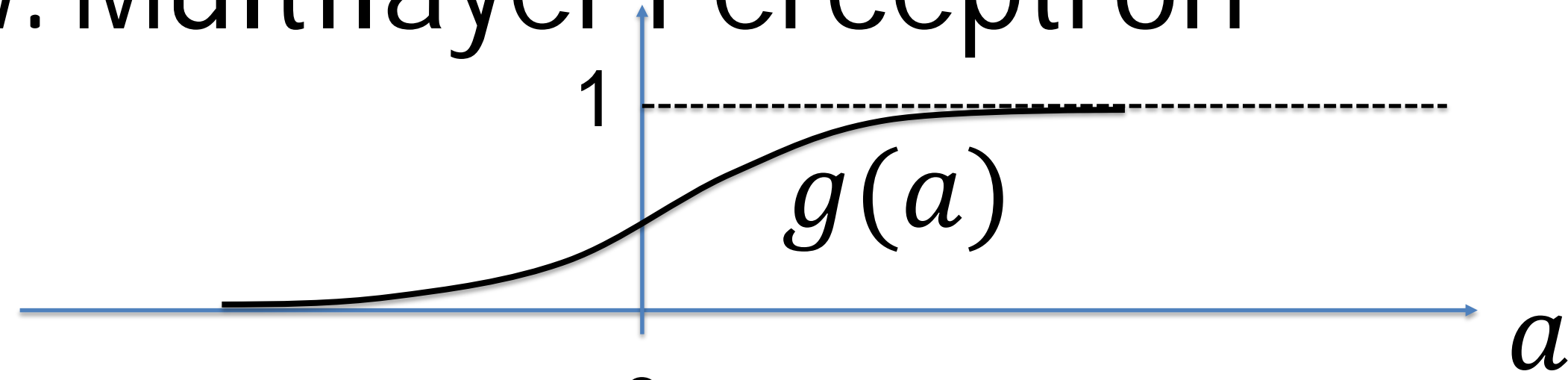
review: Artificial Neural Networks for classification

Aim of learning:

Adjust connections such
that output is correct
(for each input image,
even new ones)



Review: Multilayer Perceptron



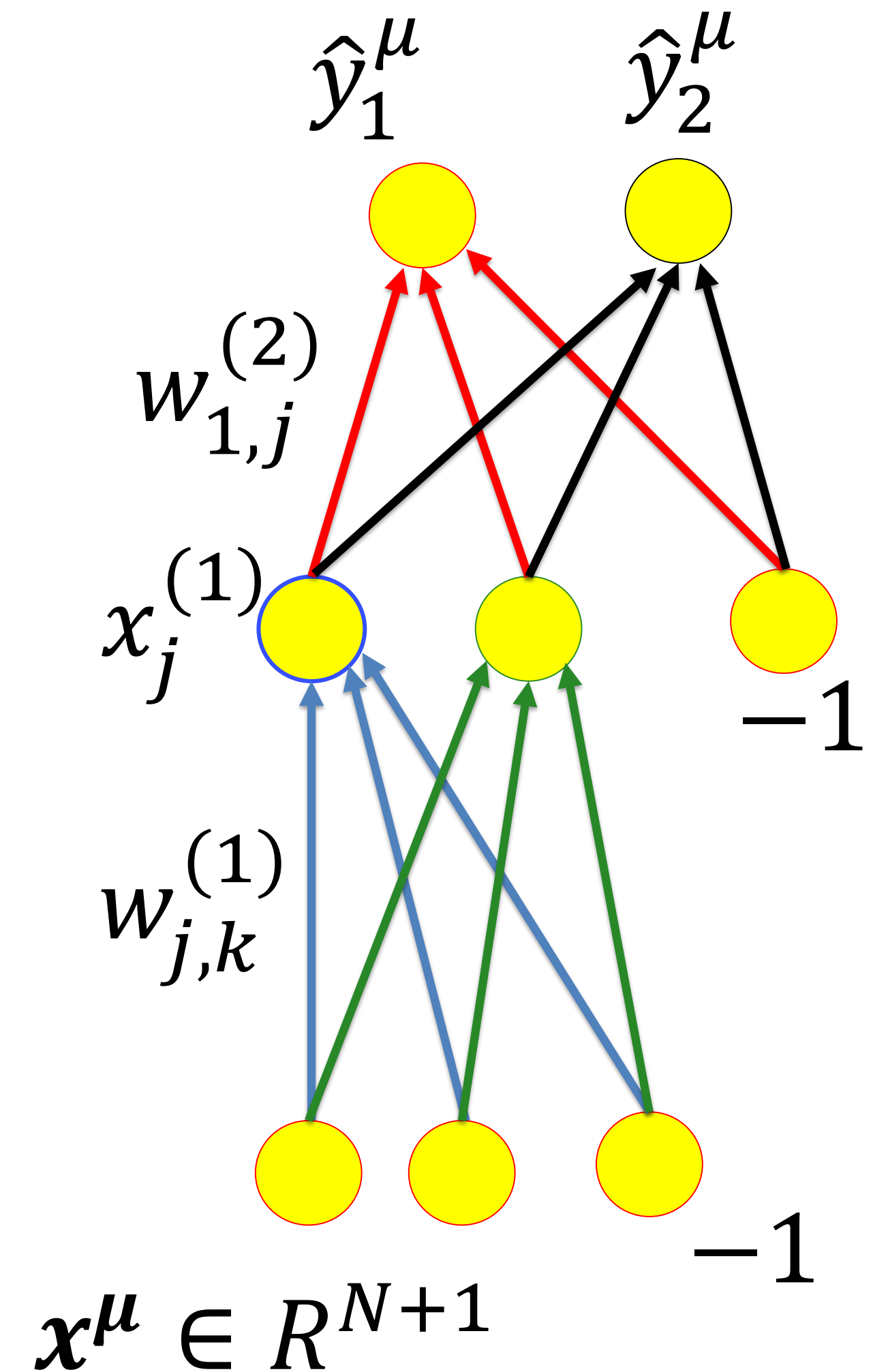
$$\hat{y}_i^\mu = x_i^{(2)} \quad (1)$$

$$= g^{(2)}[a_i^{(2)}] \quad (2)$$

$$= g^{(2)}[\sum_j w_{ij}^{(2)} x_j^{(1)}] \quad (3)$$

$$= g^{(2)}[\sum_j w_{ij}^{(2)} g^{(1)}(a_j^{(1)})] \quad (4)$$

$$= g^{(2)}[\sum_j w_{ij}^{(2)} g^{(1)}(\sum_k w_{jk}^{(1)} x_k^\mu)] \quad (5)$$



Review: Example MNIST



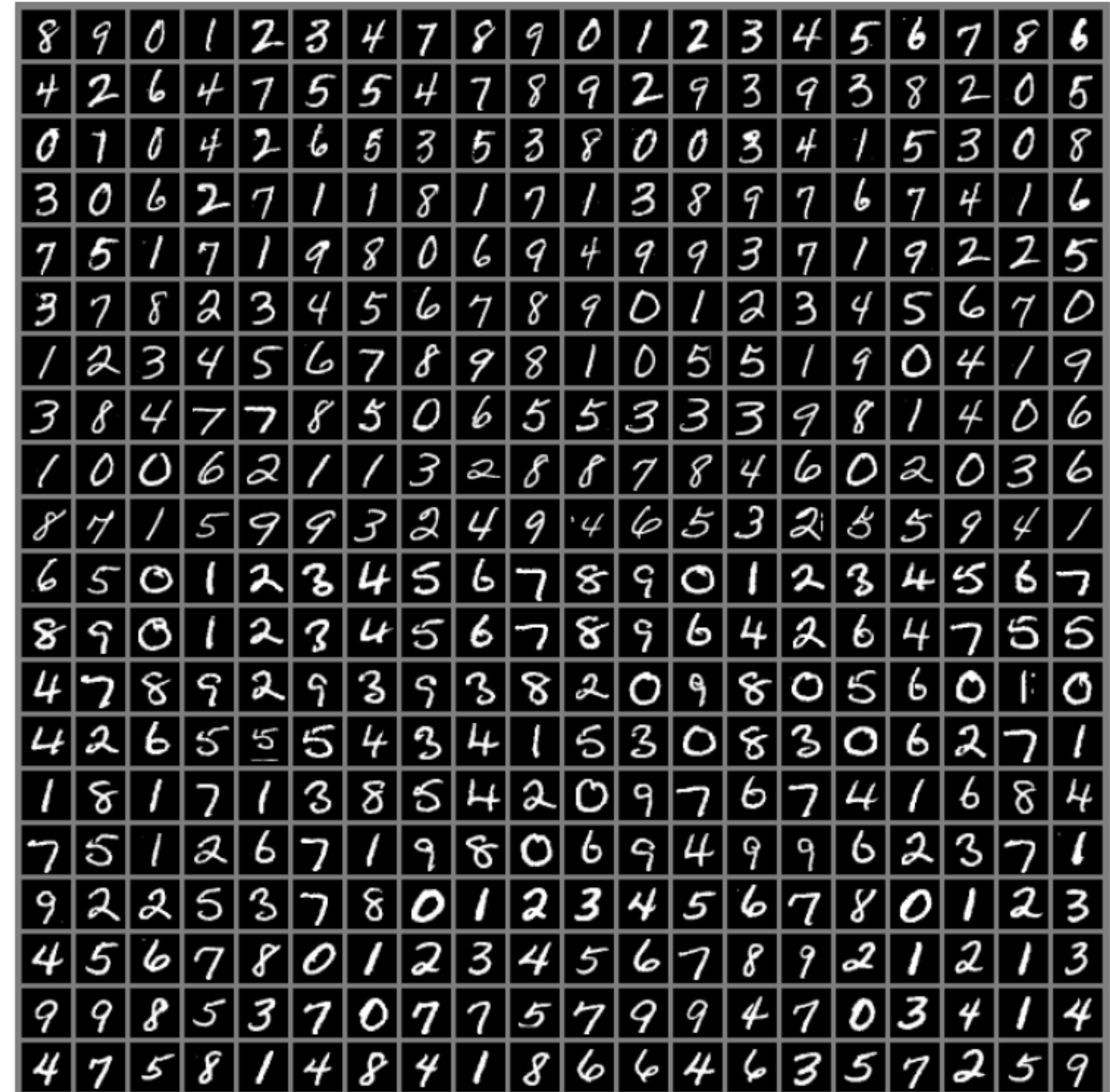
- images 28x28
- Labels: 0, ..., 9
- 250 writers
- 60 000 images in training set

Picture: Goodfellow et al, 2016

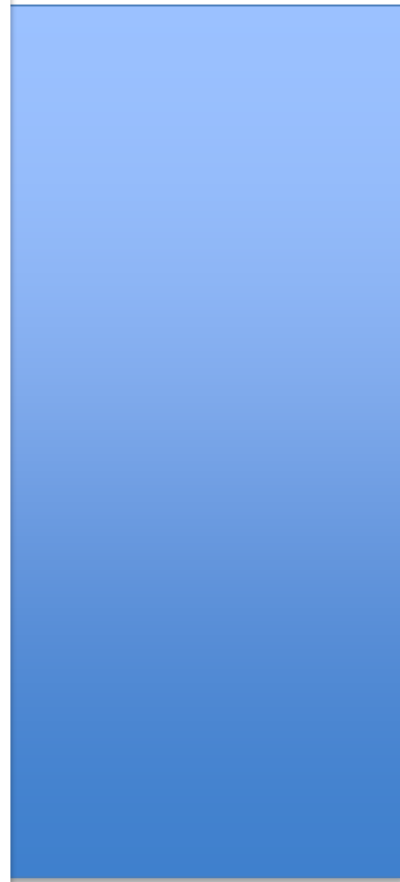
Data base:

<http://yann.lecun.com/exdb/mnist/>

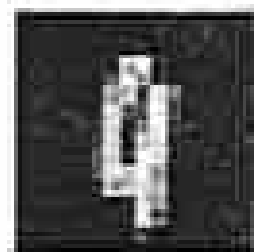
MNIST data samples



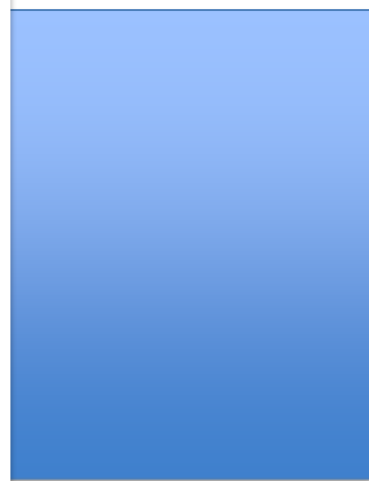
review: data base is noisy



9 or 4?



9 or 4?

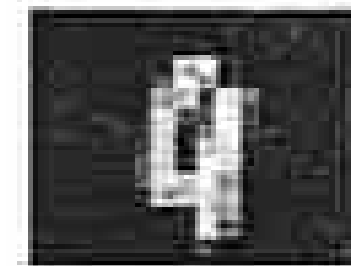
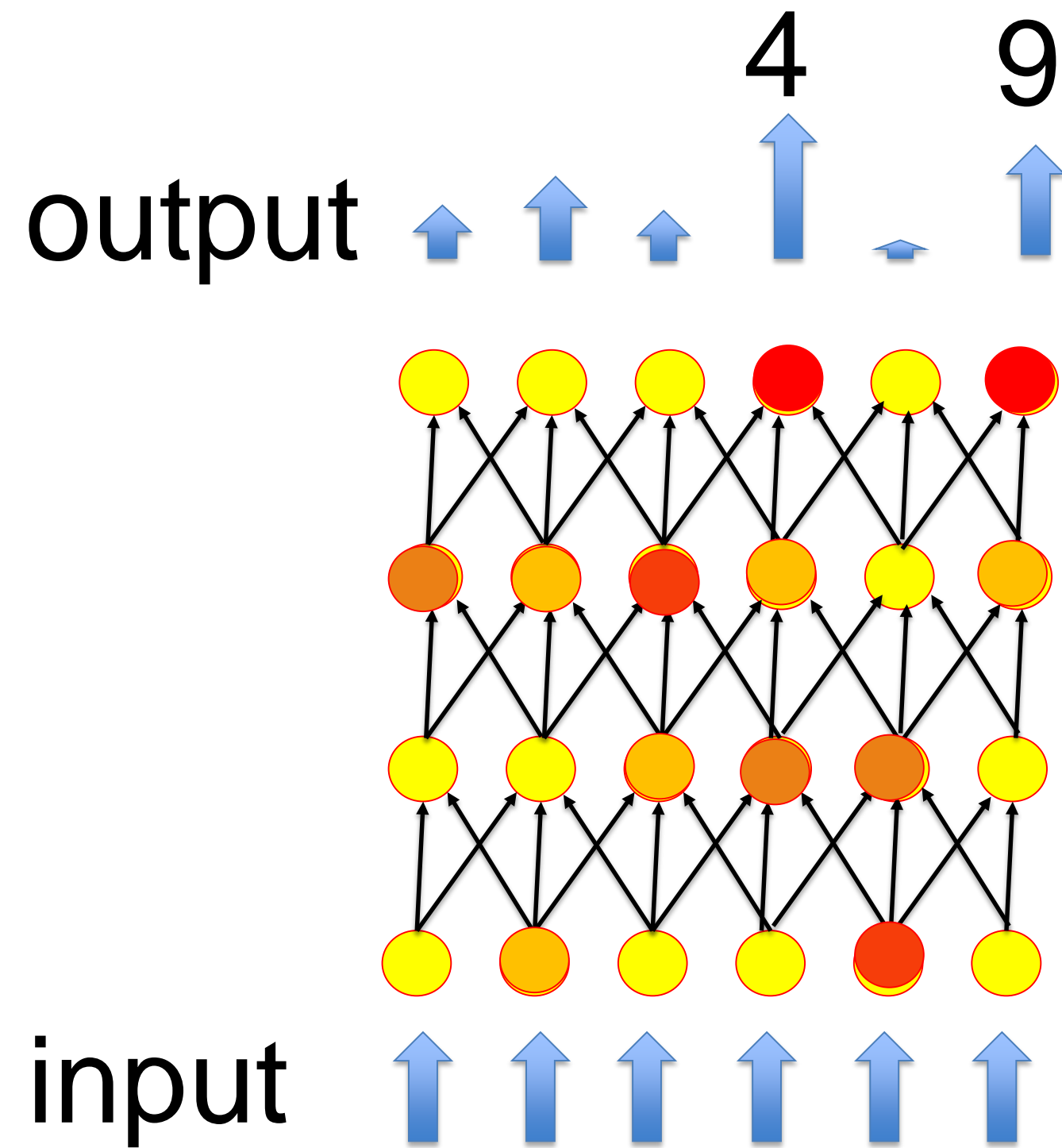


- training data is always noisy
- the future data has different noise
- Classifier must extract the essence
→ **do not fit the noise!!**

What might be a
9 for reader A
Might be a
4 for reader B

Question for today

May we interpret
the outputs
our network as
a probability?



Artificial Neural Networks: Lecture 3

Wulfram Gerstner

EPFL, Lausanne, Switzerland

Statistical Classification by Deep Networks

1. The statistical view: generative model

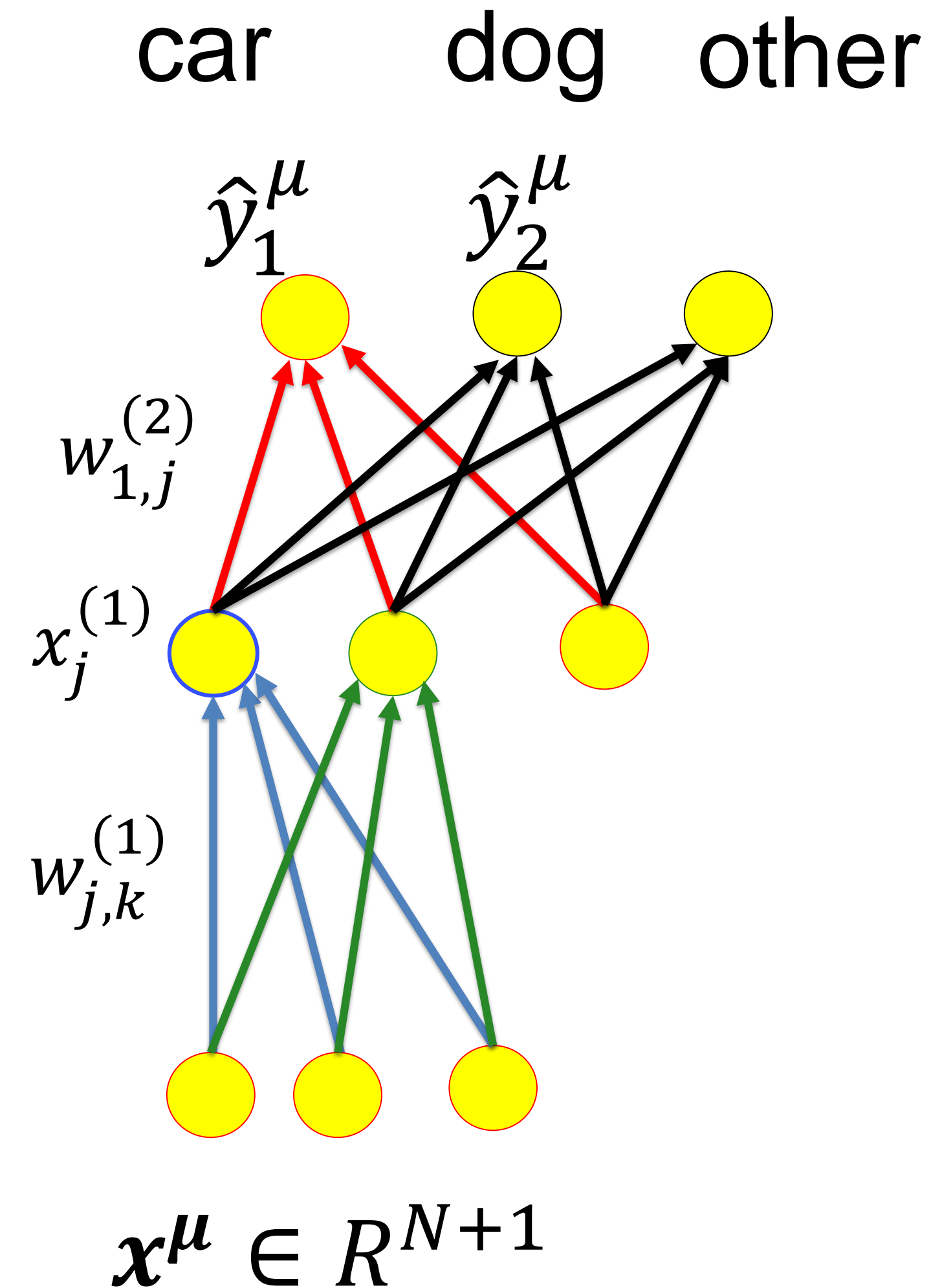
1. The statistical view

Idea:

interpret the output \hat{y}_k^μ
as the **probability** that
the novel input pattern \mathbf{x}^μ
should be classified
as class k

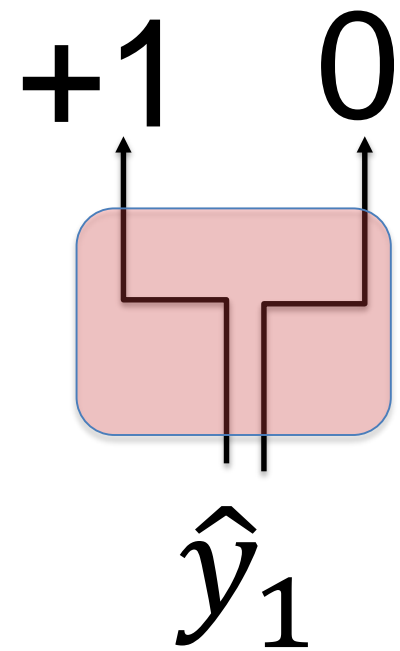
$$\hat{y}_k^\mu = P(C_k | \mathbf{x}^\mu) \quad \text{pattern from data base}$$

$$\hat{y}_k = P(C_k | \mathbf{x}) \quad \text{arbitrary novel pattern}$$



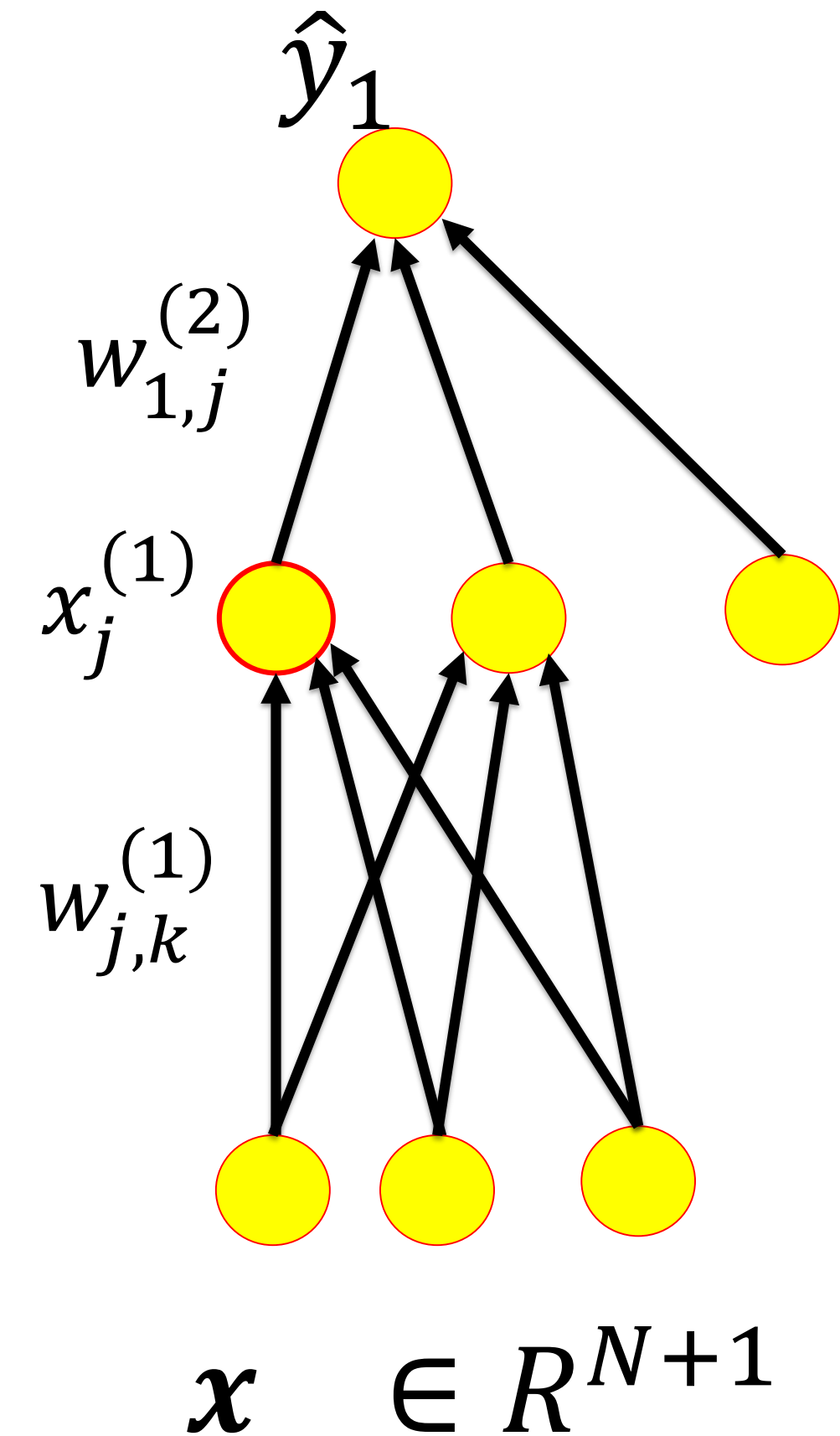
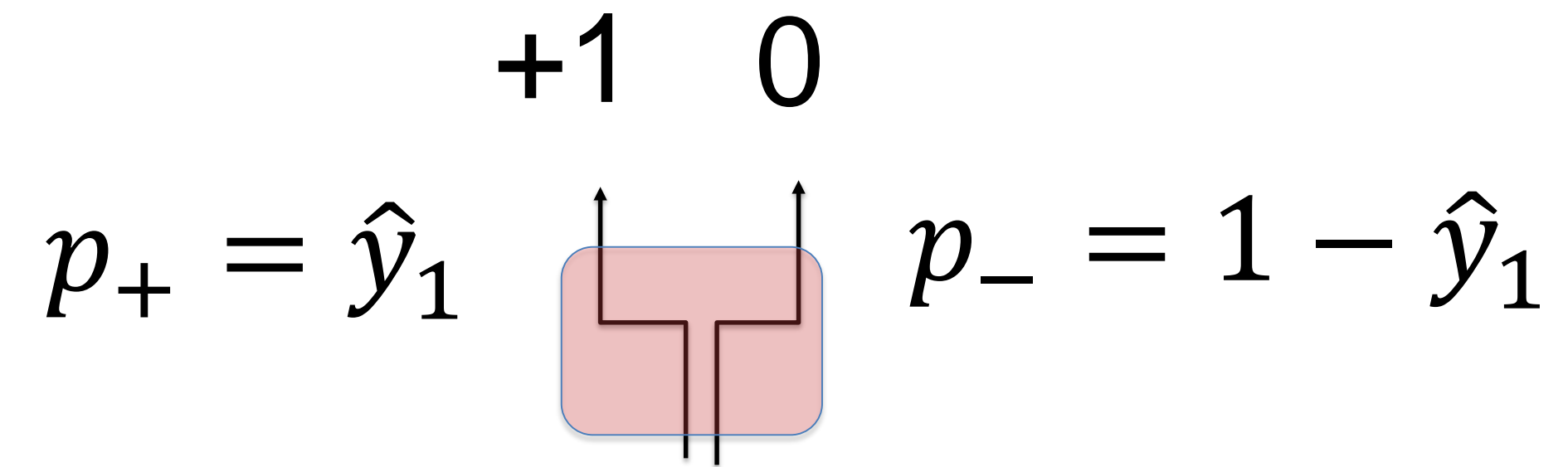
1. The statistical view: single class

Take the output \hat{y}_1 and generate predicted labels \hat{t}_1 probabilistically



→ generative model for class label
with $\hat{y}_1 = P(C_1|\mathbf{x}) = P(\hat{t}_1 = 1|\mathbf{x})$

↑
predicted label



Artificial Neural Networks: Lecture 3

Wulfram Gerstner
EPFL, Lausanne, Switzerland

Statistical Classification by Deep Networks

1. The statistical view: generative model
2. **The likelihood of data under a model**

2. The likelihood of data under a model

Overall aim:

What is the likelihood that my set of P data points

$$\{ (\mathbf{x}^\mu, t^\mu) \ , \quad 1 \leq \mu \leq P \};$$

could have been generated by my model?

2. The likelihood of data under a model

Detour:

forget about labeled data, and just think of input patterns

What is the likelihood that a set of P data points

$$\{x^k ; 1 \leq k \leq P \};$$

could have been generated by my model?

2. Example: Gaussian distribution

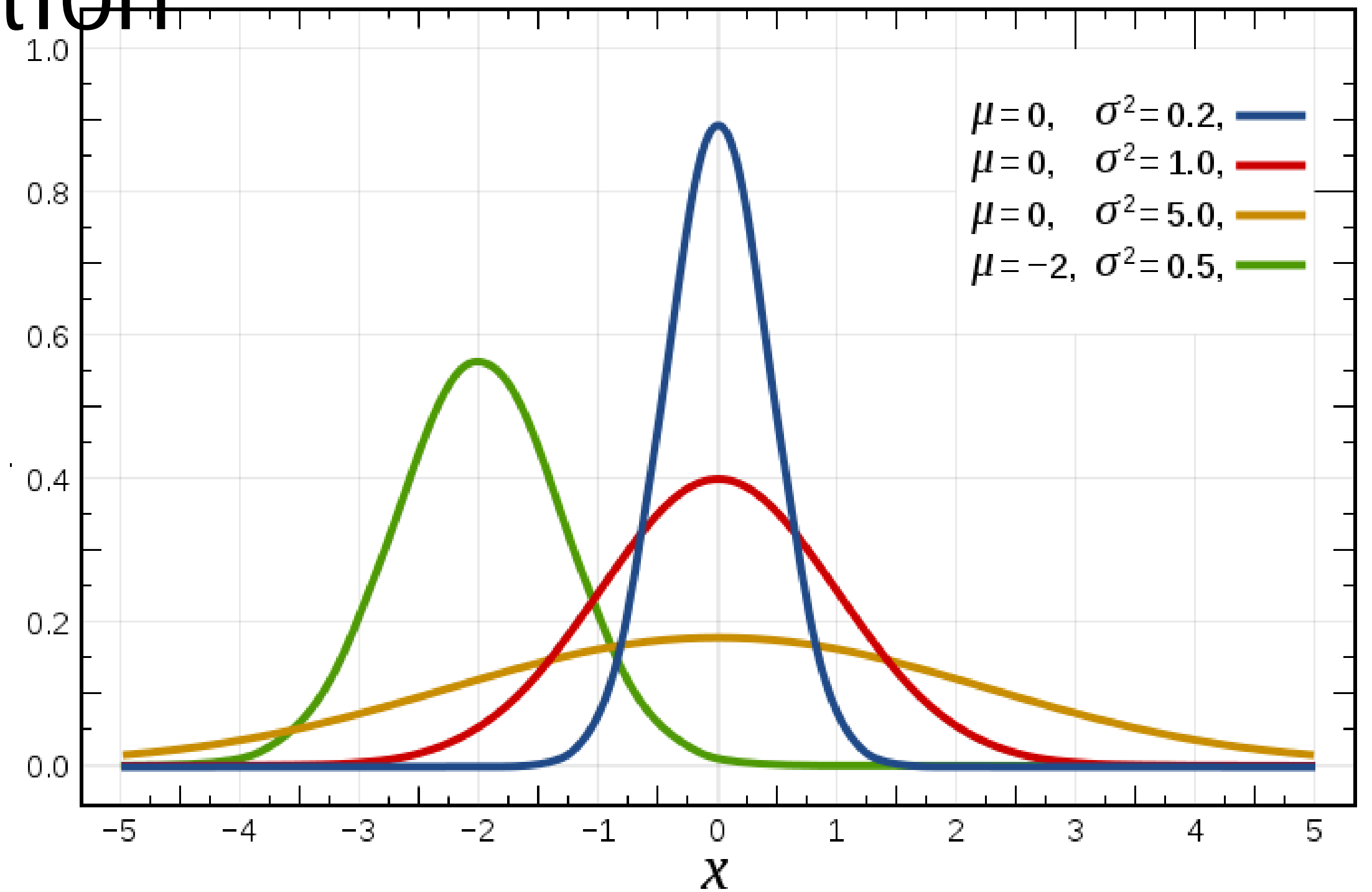
$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{\frac{-(x-\mu)^2}{2\sigma^2}\right\}$$

this depends on 2 parameters

$$\{w_1, w_2\} = \{\mu, \sigma\}$$

center

width

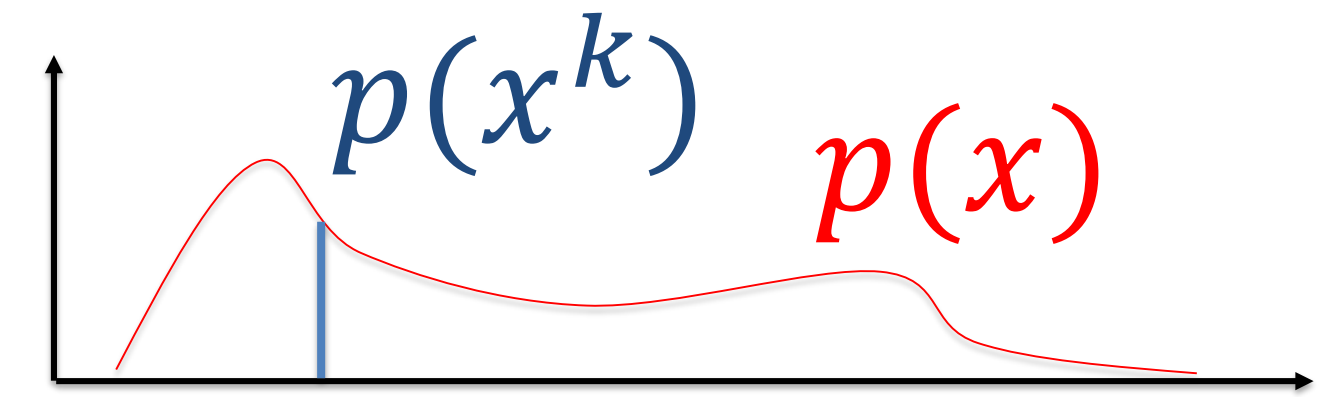


https://en.wikipedia.org/wiki/Gaussian_function#/media/

2. Random Data Generation Process

Likelihood that a random data generation process draws one sample k with value x^k is

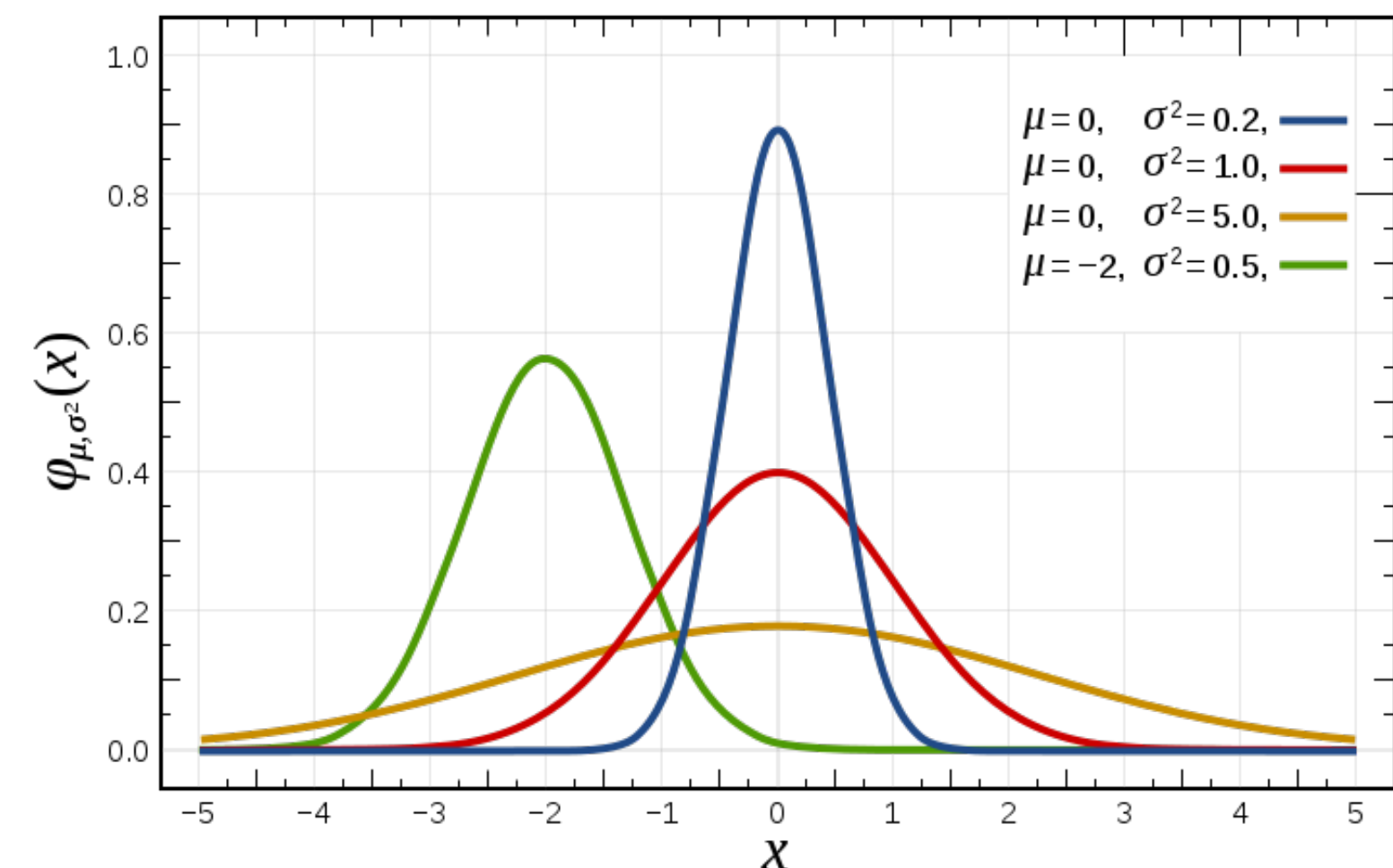
$$\sim p(x^k)$$



Example: for the specific case of the Gaussian

$$p(x^k) = \frac{1}{\sqrt{2\pi}\sigma} \exp \left\{ \frac{-(x^k - \mu)^2}{2\sigma^2} \right\}$$

Blackboard 1:
Likelihood of P data points



Blackboard 1:

derive Likelihood function

2. Likelihood function (beyond Gaussian)

Suppose the likelihood for generating a data point \mathbf{x}^k using my model is $p(\mathbf{x}^k)$

Suppose that data points are generated independently.

Then the likelihood that **my actual data set**

$$\mathbf{X} = \{\mathbf{x}^k; 1 \leq k \leq P \};$$

could have been generated by my model is

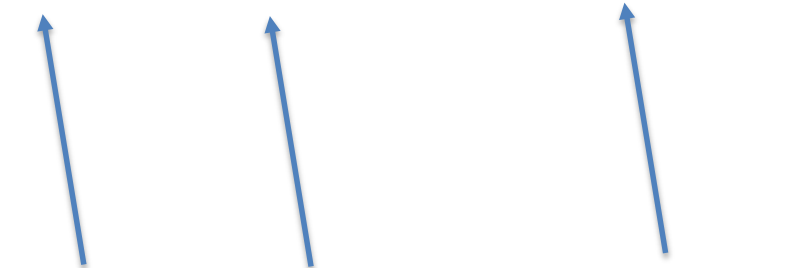
$$p_{model}(\mathbf{X}) = p(\mathbf{x}^1) p(\mathbf{x}^2) p(\mathbf{x}^3) \dots p(\mathbf{x}^P)$$

2. Maximum Likelihood (beyond Gaussian)

$$p_{model}(\mathbf{X}) = p(\mathbf{x}^1) p(\mathbf{x}^2) p(\mathbf{x}^3) \dots p(\mathbf{x}^P)$$

BUT this likelihood depends on the parameters of my model

$$p_{model}(\mathbf{X}) = p_{model}(\mathbf{X} | \{w_1, w_2, \dots, w_n\})$$



parameters

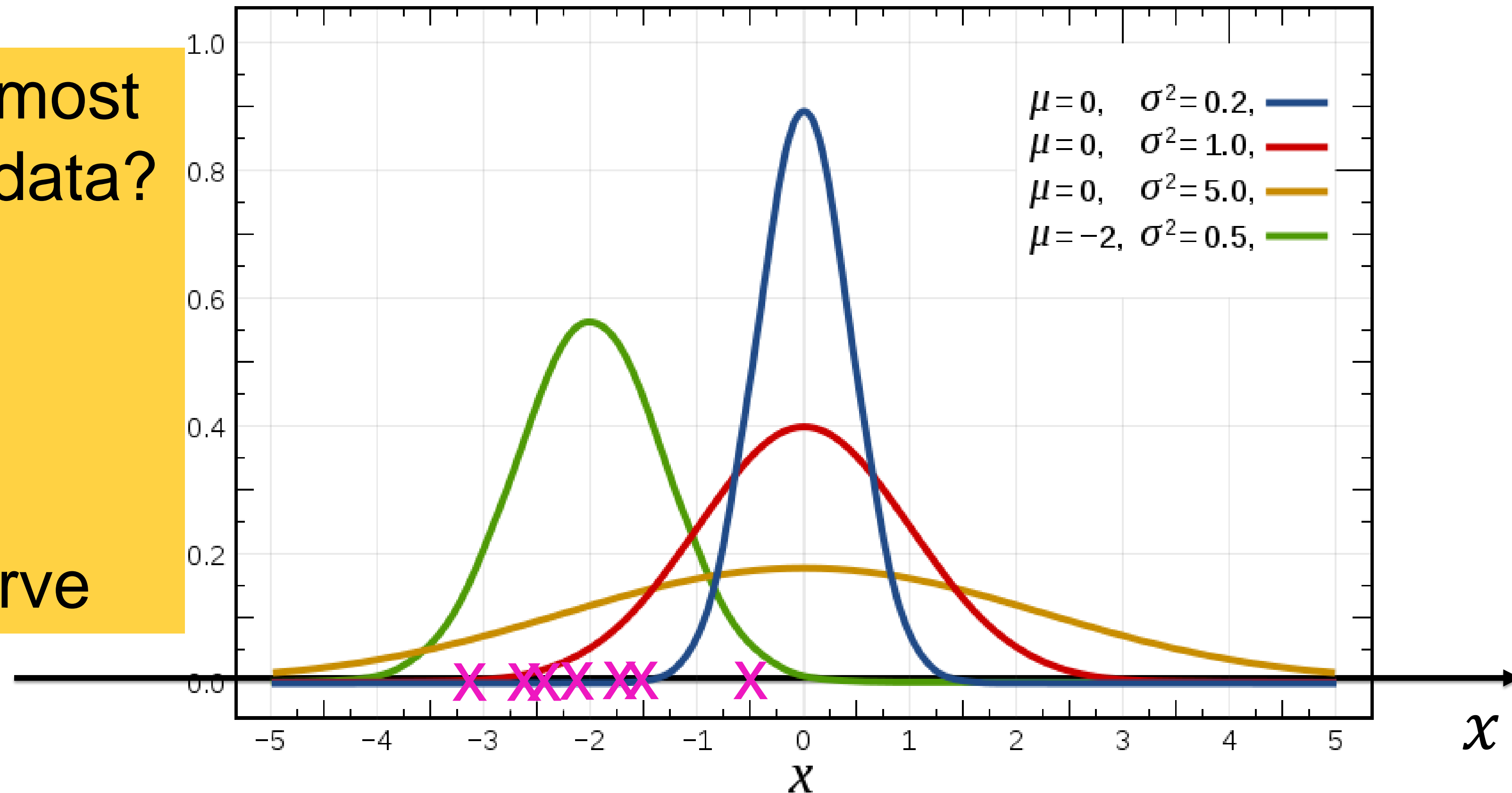
Choose the parameters such that the likelihood is maximal!

2. Example: Gaussian distribution

Likelihood of point x^k is $p(x^k) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{\frac{-(x^k - \mu)^2}{2\sigma^2}\right\}$

Which Gaussian is most consistent with the data?

- ☐ green curve
- ☐ blue curve
- ☐ red curve
- ☐ brown-orange curve



2. Example: Gaussian

$$p_{model}(\mathbf{X}) = p(\mathbf{x}^1) p(\mathbf{x}^2) p(\mathbf{x}^3) \dots p(\mathbf{x}^P)$$

The likelihood depends on the 2 parameters of my Gaussian

$$p_{model}(\mathbf{X}) = p_{model}(\mathbf{X}|\{w_1, w_2\})$$

$$p_{model}(\mathbf{X}) = p_{model}(\mathbf{X}|\{\mu, \sigma\})$$

Exercise 1 NOW! (8 minutes): you have P data points

Calculate the **optimal choice** of parameter μ :

To do so maximize $p_{model}(\mathbf{X})$ with respect to μ

Blackboard 2:

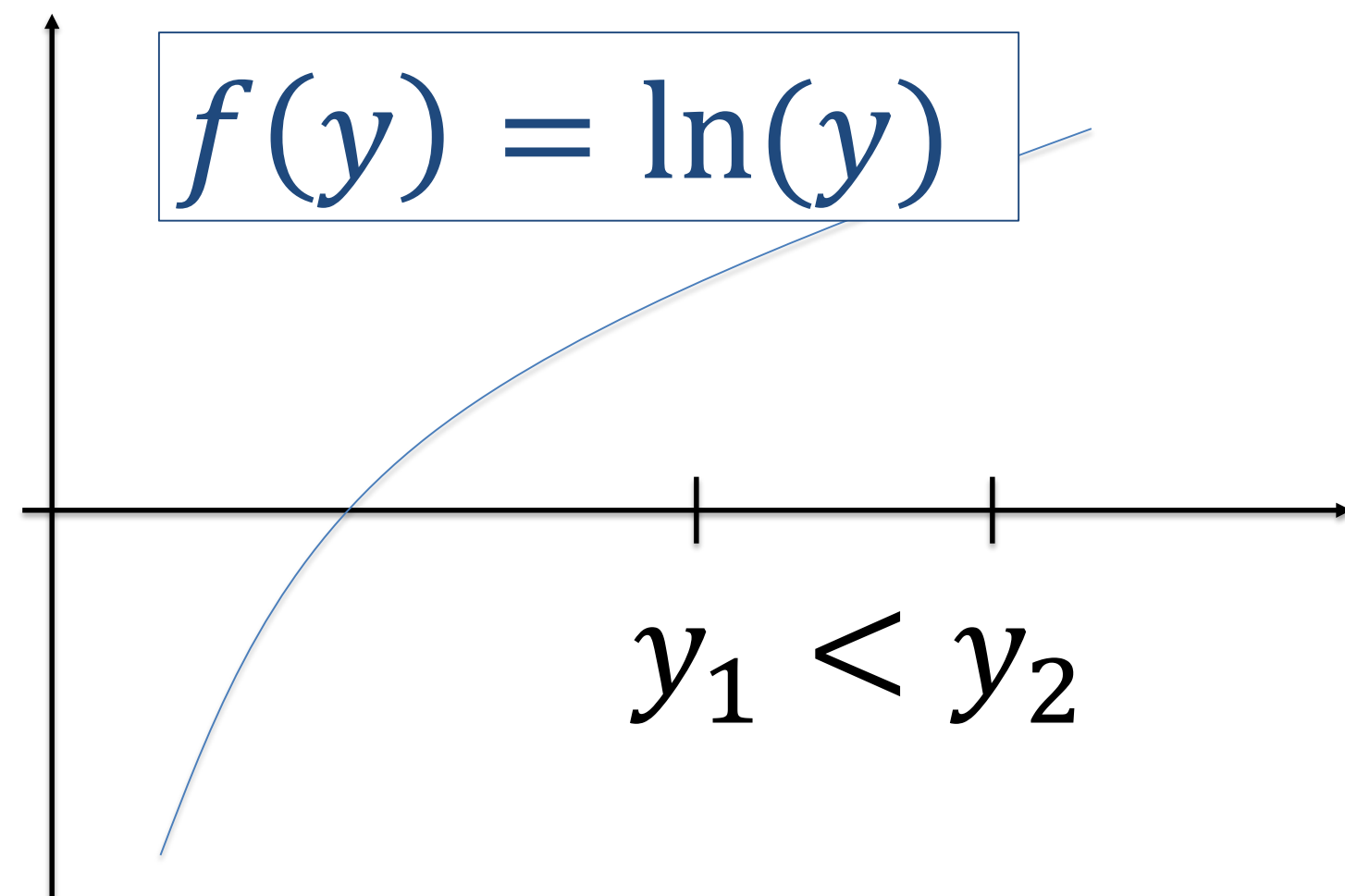
Gaussian: best parameter choice for center

2. Maximum Likelihood (general)

Choose the parameters such that the likelihood

$$p_{model}(\mathbf{X}|\{w_1, w_2, \dots w_n\}) = p(\mathbf{x}^1) p(\mathbf{x}^2) p(\mathbf{x}^3) \dots p(\mathbf{x}^P)$$

is maximal



Note:

Instead of maximizing

$$p_{model}(\mathbf{X}|param)$$

you can also maximize

$$\ln(p_{model}(\mathbf{X}|param))$$

2. Maximum Likelihood (general)

Choose the parameters such that the likelihood

$$p_{model}(\mathbf{X}|\{w_1, w_2, \dots w_n\}) = p(\mathbf{x}^1) p(\mathbf{x}^2) p(\mathbf{x}^3) \dots p(\mathbf{x}^P)$$

is maximal is equivalent to maximizing the log-likelihood

$$LL = \ln(p_{model}) = \sum_k \ln p(\mathbf{x}^k)$$

“Maximize the likelihood that the given data could have been generated by your model”

(even though you know that the data points were generated by a process in the real world that might be very different)

Artificial Neural Networks: Lecture 3

Wulfram Gerstner
EPFL, Lausanne, Switzerland

Statistical Classification by Deep Networks

1. The statistical view: generative model
2. The likelihood of data under a model
3. **Application to artificial neural networks**

3. The likelihood of data under a neural network model

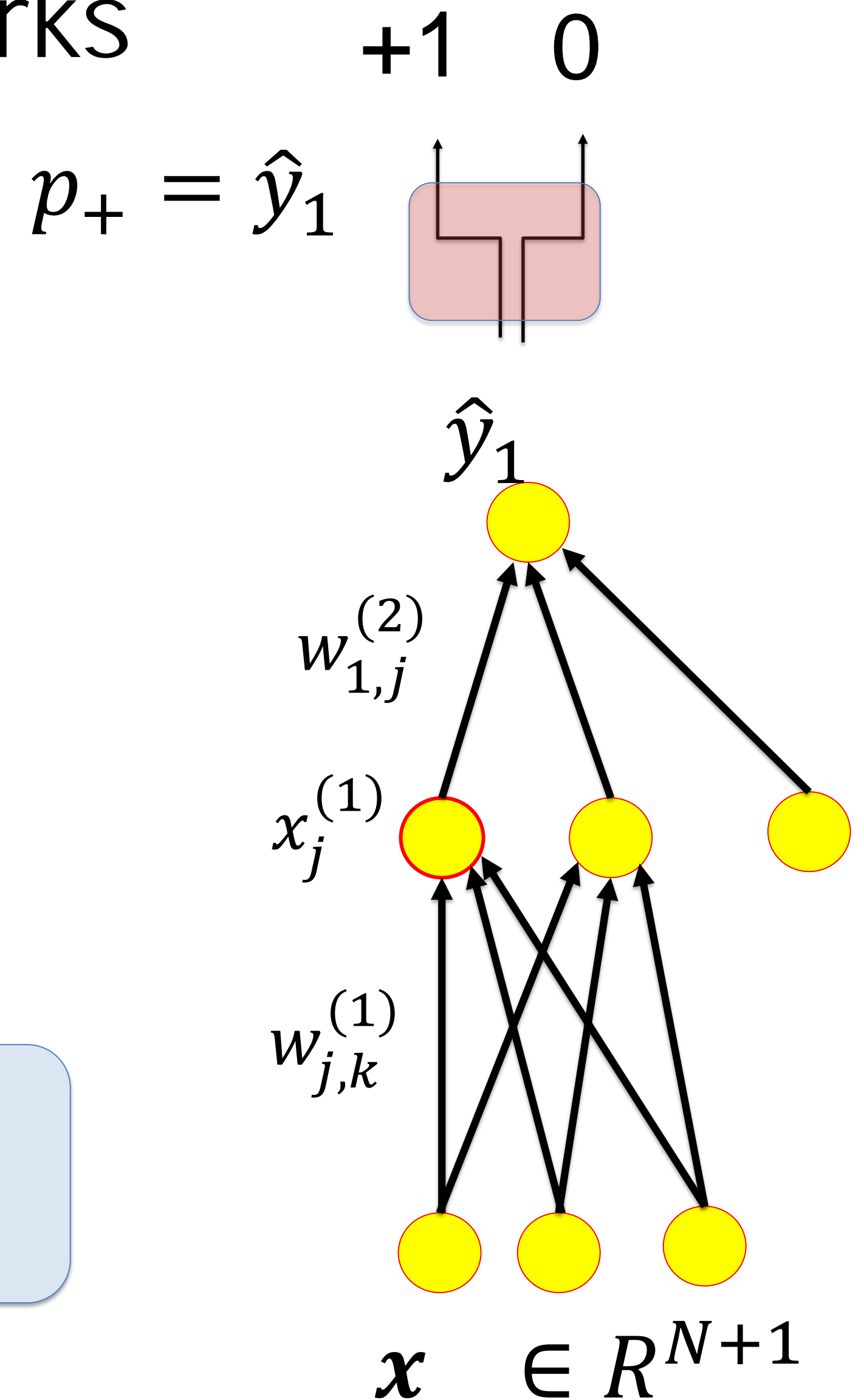
Overall aim:

What is the likelihood that my set of P data points

$$\{ (\mathbf{x}^\mu, t^\mu) \ , \quad 1 \leq \mu \leq P \};$$

could have been generated by my model?

3. Maximum Likelihood for neural networks



Blackboard 3:
Likelihood of P input-output pairs

Blackboard 3:
Likelihood of P input-output pairs

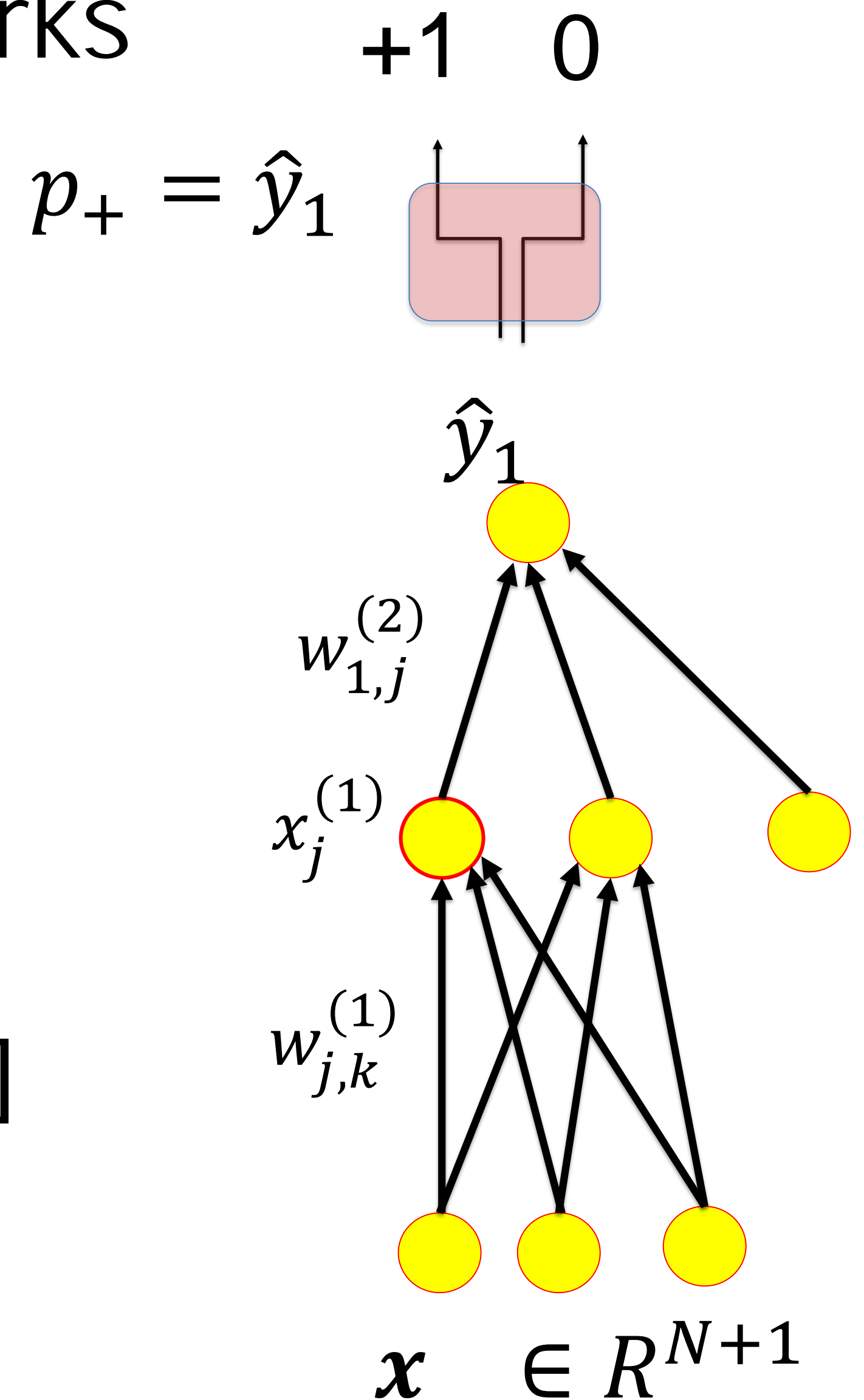
3. Maximum Likelihood for neural networks

Minimize the negative log-likelihood

$$E(\mathbf{w}) = -LL = -\ln(p_{model})$$

↑
parameters= all weights, all layers
↓

$$E(\mathbf{w}) = -\sum_{\mu} [t^{\mu} \ln \hat{y}^{\mu} + (1 - t^{\mu}) \ln (1 - \hat{y}^{\mu})]$$



3. Cross-entropy error function for neural networks

Suppose we minimize the cross-entropy error function

$$E(\mathbf{w}) = -\sum_{\mu} [t^{\mu} \ln \hat{y}^{\mu} + (1 - t^{\mu}) \ln(1 - \hat{y}^{\mu})]$$

Can we be sure that the output \hat{y}^{μ} will represent the probability?

Intuitive answer: **No, because**

A We will need enough data for training

(not just 10 data points for a complex task)

B We need a sufficiently flexible network

(not a simple perceptron for XOR task)

3. Output = probability ?

Suppose we minimize the cross-entropy error function

$$E(\mathbf{w}) = -\sum_{\mu} [t^{\mu} \ln \hat{y}^{\mu} + (1 - t^{\mu}) \ln (1 - \hat{y}^{\mu})]$$

Assume

A We have enough data for training

B We have a sufficiently flexible network

Blackboard 4:

From Cross-entropy to output probabilities

Blackboard 4:

From Cross-entropy to output probabilities

QUIZ: Maximum likelihood solution means

- ☐ find the unique set of parameters that generated the data
- ☐ find the unique set of parameters that best explains the data
- ☐ find the best set of parameters such that your model could have generated the data

A cross-entropy error function for single class output

- ☐ is consistent with the idea that the output \hat{y}_1 of your network can be interpreted as

$$\hat{y}_1 = P(C_1|\mathbf{x})$$

- ☐ guarantees that the output \hat{y}_1 of your network can be interpreted as $\hat{y}_1 = P(C_1|\mathbf{x})$

Artificial Neural Networks: Lecture 3

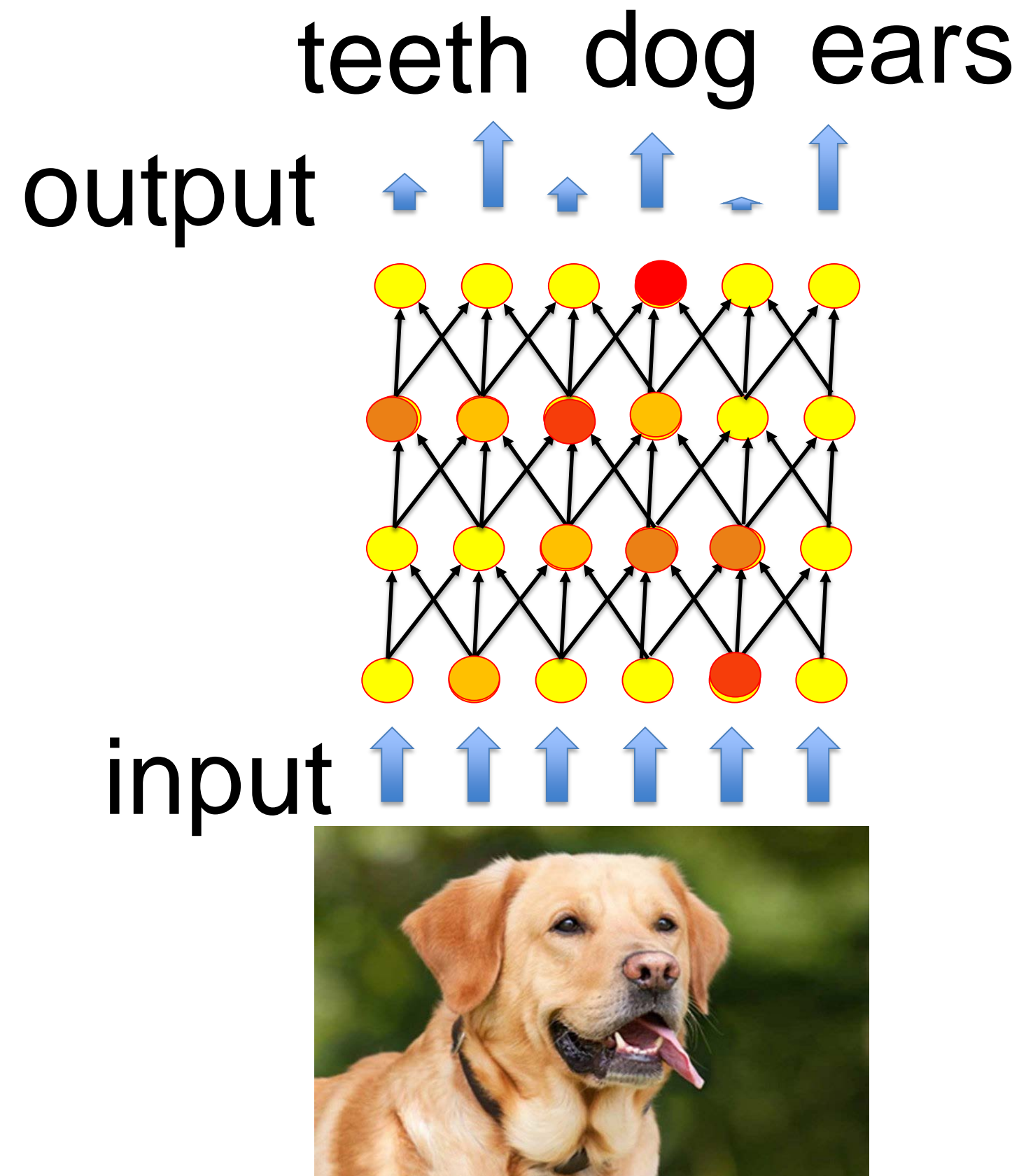
Wulfram Gerstner
EPFL, Lausanne, Switzerland

Statistical Classification by Deep Networks

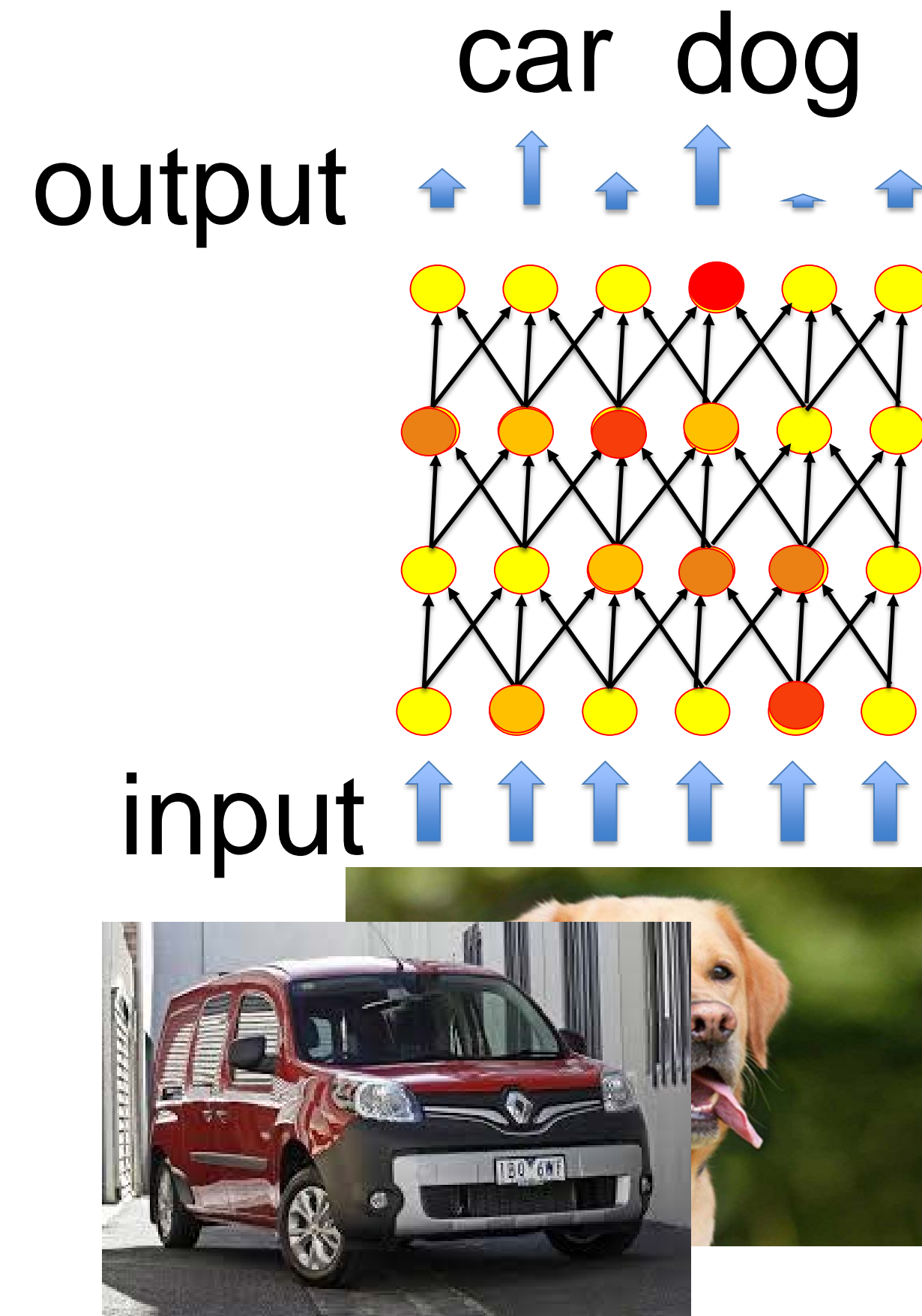
1. The statistical view: generative model
2. The likelihood of data under a model
3. Application to artificial neural networks
4. Multi-class and 1-hot coding

4. Multiple Classes

multiple attributes



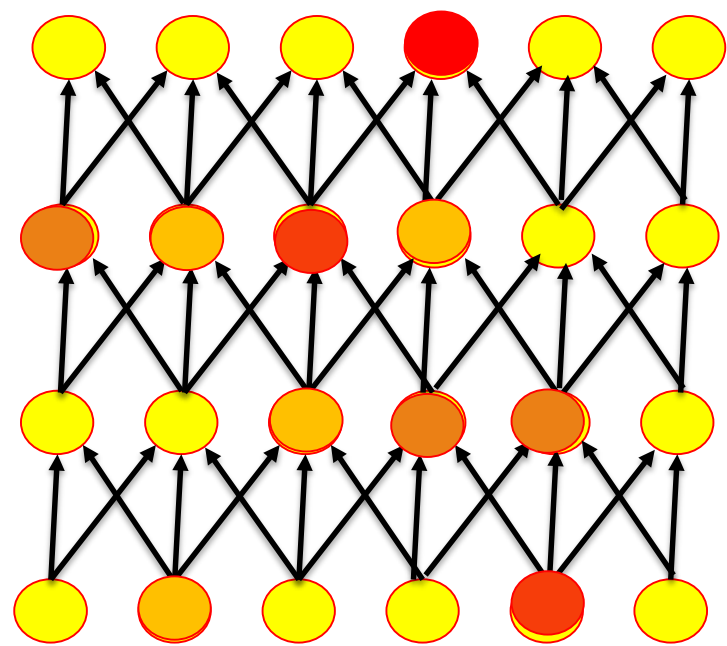
mutually exclusive classes



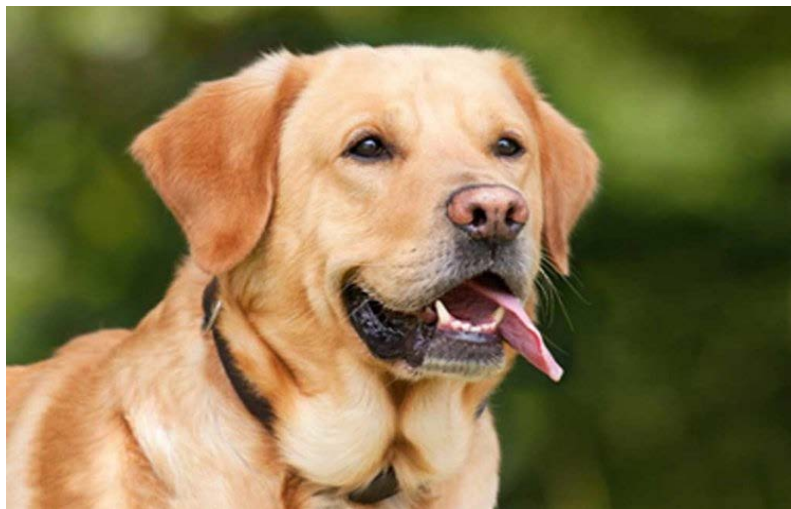
4. Multiple Classes: Multiple attributes

Multiple attributes:

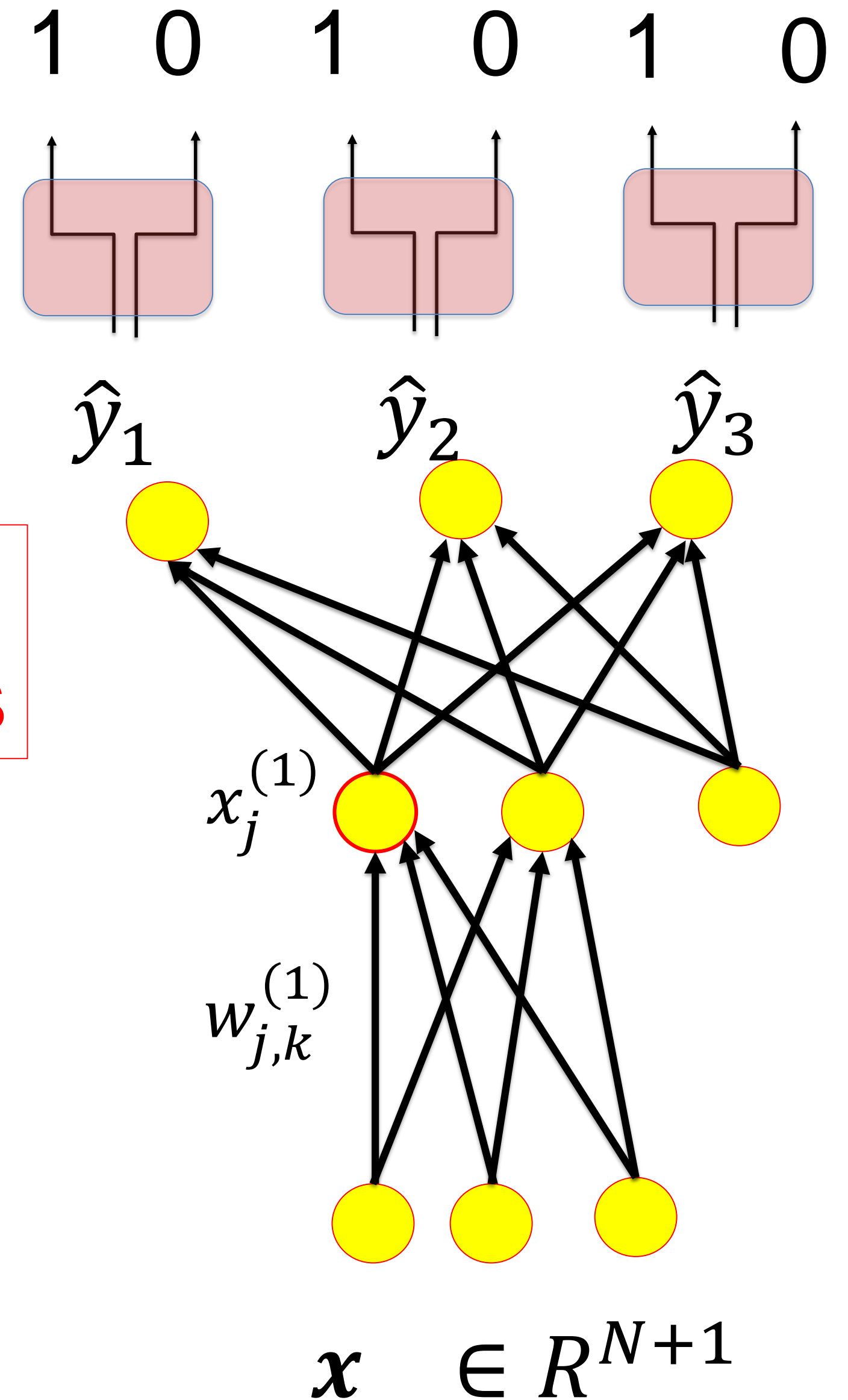
teeth dog ears
output



input



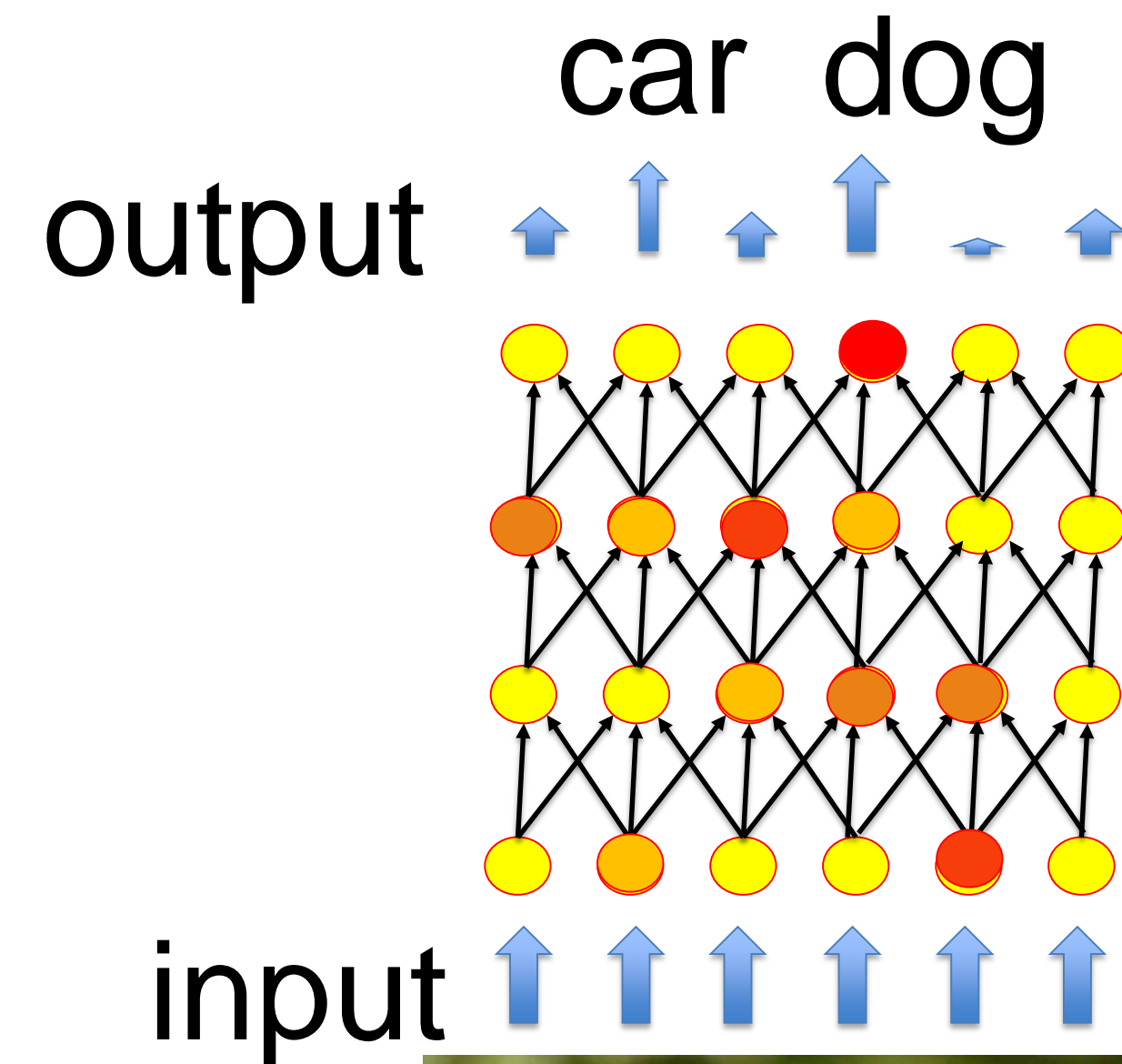
equivalent to several
single-class decisions



4. Multiple Classes: Mutually exclusive classes

mutually exclusive classes

either car or dog:
only one can be true
→
outputs interact

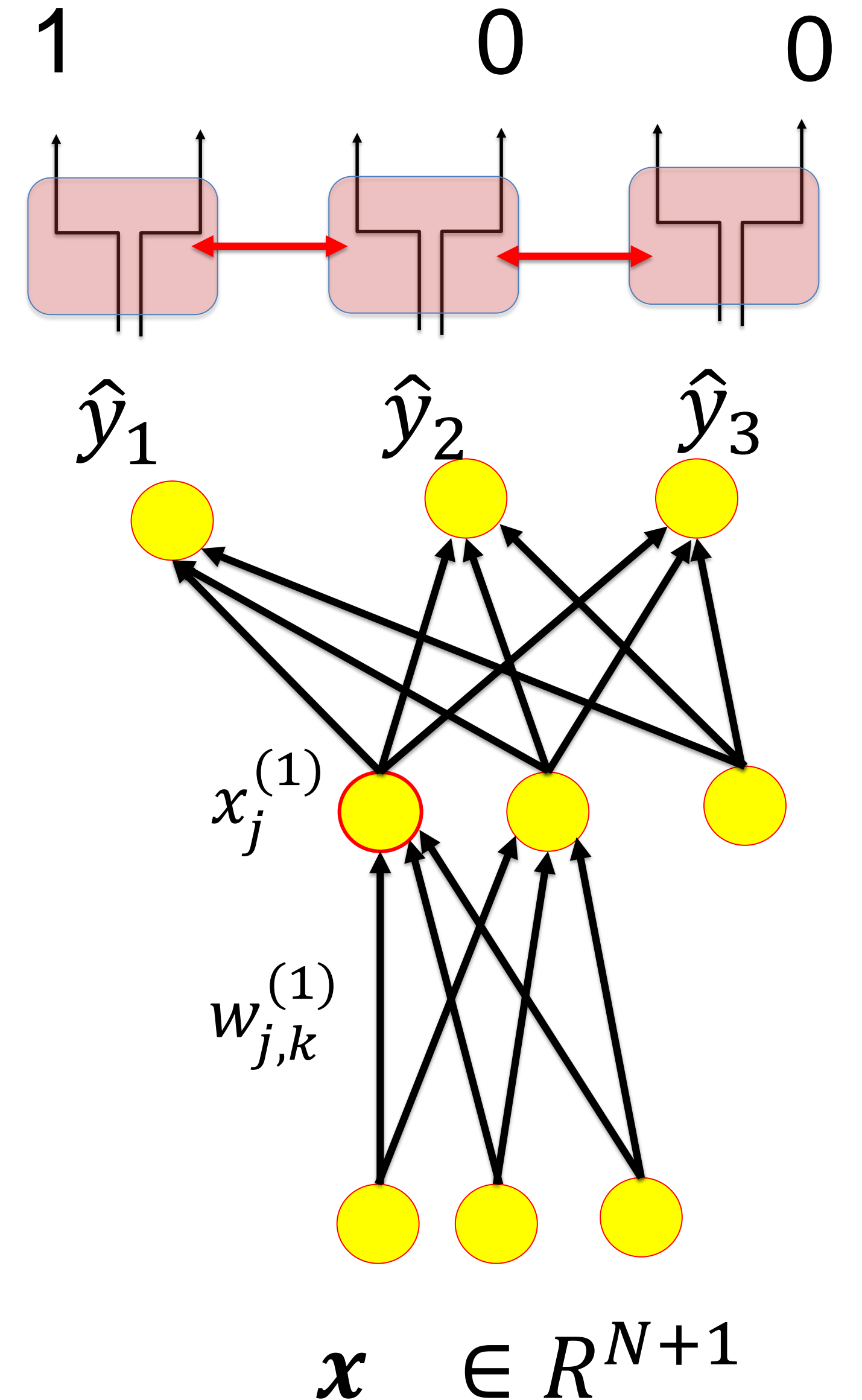


4. Exclusive Multiple Classes

$$\hat{y}_1 = P(C_1|\mathbf{x}) = \mathbf{P}(\hat{t}_1 = 1|\mathbf{x})$$

1-hot-coding:

$$\hat{t}_k^\mu = 1 \rightarrow \hat{t}_j^\mu = 0 \text{ for } j \neq k$$



4. Exclusive Multiple Classes

$$\hat{y}_1 = P(C_1|\mathbf{x}) = \mathbf{P}(\hat{t}_1 = 1|\mathbf{x})$$

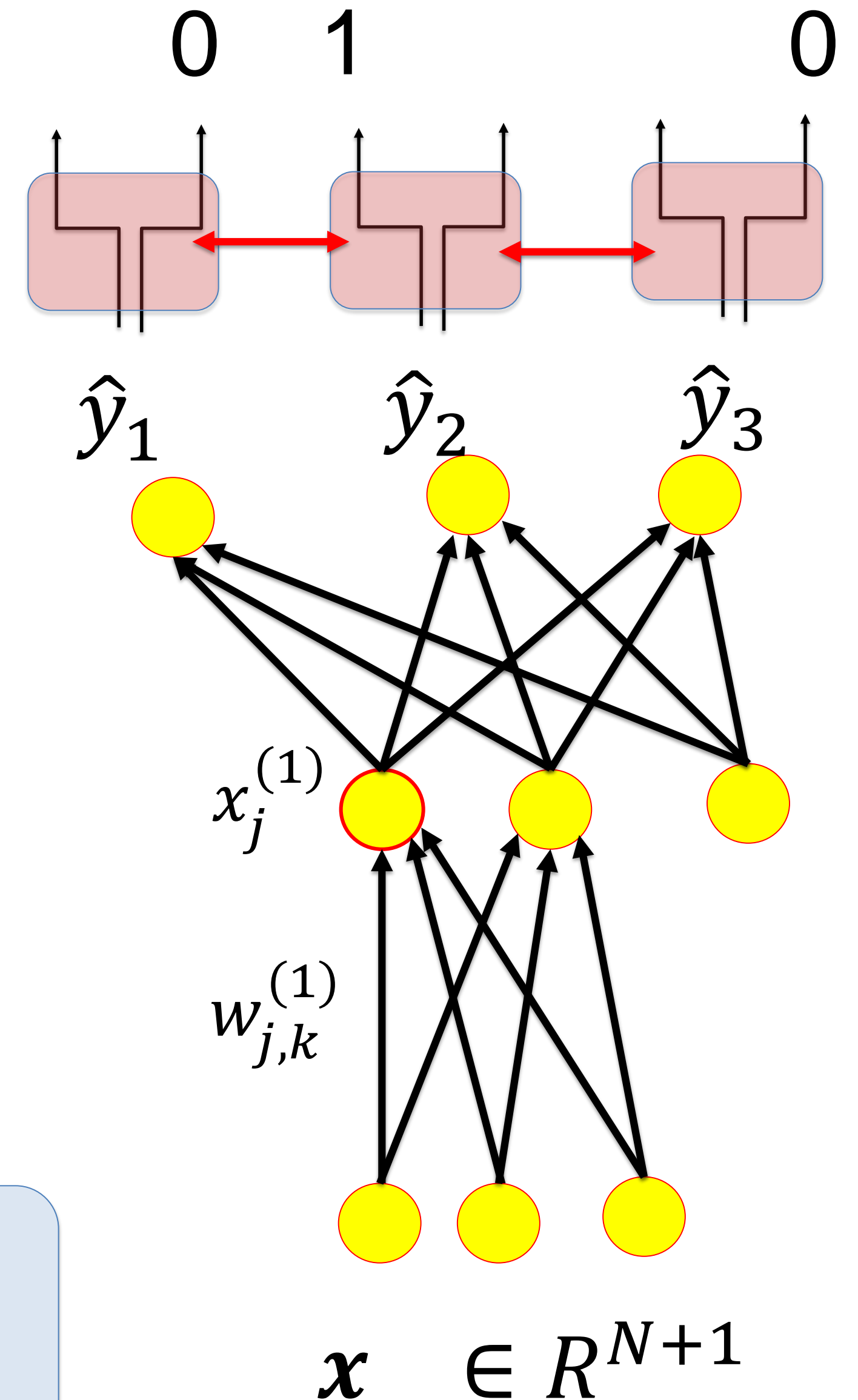
1-hot-coding:

$$\hat{t}_k^\mu = 1 \rightarrow \hat{t}_j^\mu = 0 \text{ for } j \neq k$$

Outputs are NOT independent:

$$\sum_{k=1}^K t_k^\mu = 1 \quad \text{exactly one output is 1}$$

Blackboard 5:
derive likelihood function



Blackboard 5:
Likelihood of P input-output pairs

4. Cross entropy error for neural networks: Multiclass

We have a total of K classes (mutually exclusive: either dog or car)

Minimize* the **cross-entropy**

$$E(\mathbf{w}) = - \sum_{k=1}^K \sum_{\mu} [t_k^{\mu} \ln \hat{y}_k^{\mu}]$$

parameters= all weights, all layers

*Minimization under the constraint:

$$\sum_{k=1}^K \hat{y}_k^{\mu} = 1$$

Compare: **KL divergence between outputs and targets**

$$\text{KL}(\mathbf{w}) = -\{\sum_{k=1}^K \sum_{\mu} [t_k^{\mu} \ln \hat{y}_k^{\mu}] - \sum_{\mu} [t_k^{\mu} \ln t_k^{\mu}]\}$$

$$\text{KL}(\mathbf{w}) = E(\mathbf{w}) + \text{constant}$$

Artificial Neural Networks: Lecture 3

Wulfram Gerstner
EPFL, Lausanne, Switzerland

Statistical Classification by Deep Networks

1. The statistical view: generative model
2. The likelihood of data under a model
3. Application to artificial neural networks
4. Multi-class problems
5. **Sigmoidal as a natural output function**

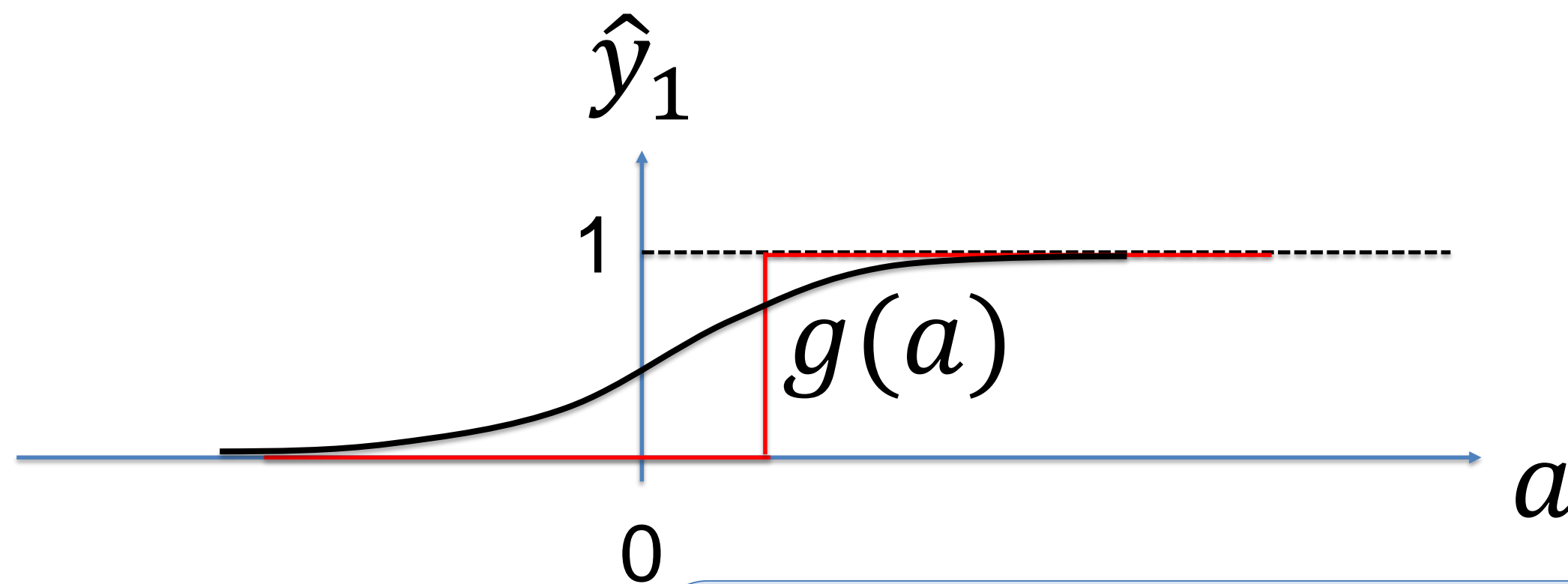
5. Why sigmoidal output ? – single class

$$\hat{y}_1 = P(C_1|\mathbf{x}) = P(\hat{t}_1 = 1|\mathbf{x})$$

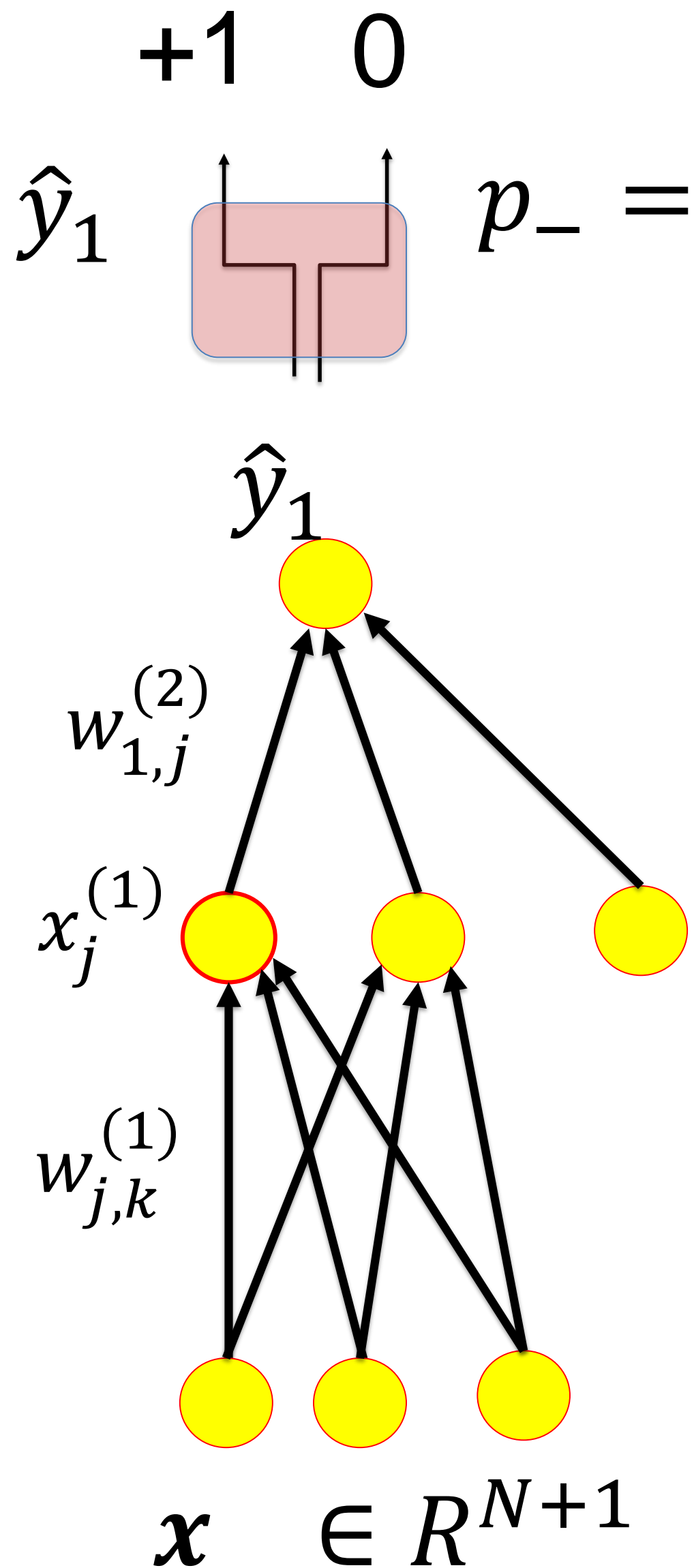
$$p_+ = \hat{y}_1 \quad p_- = 1 - \hat{y}_1$$

Observations (single-class):

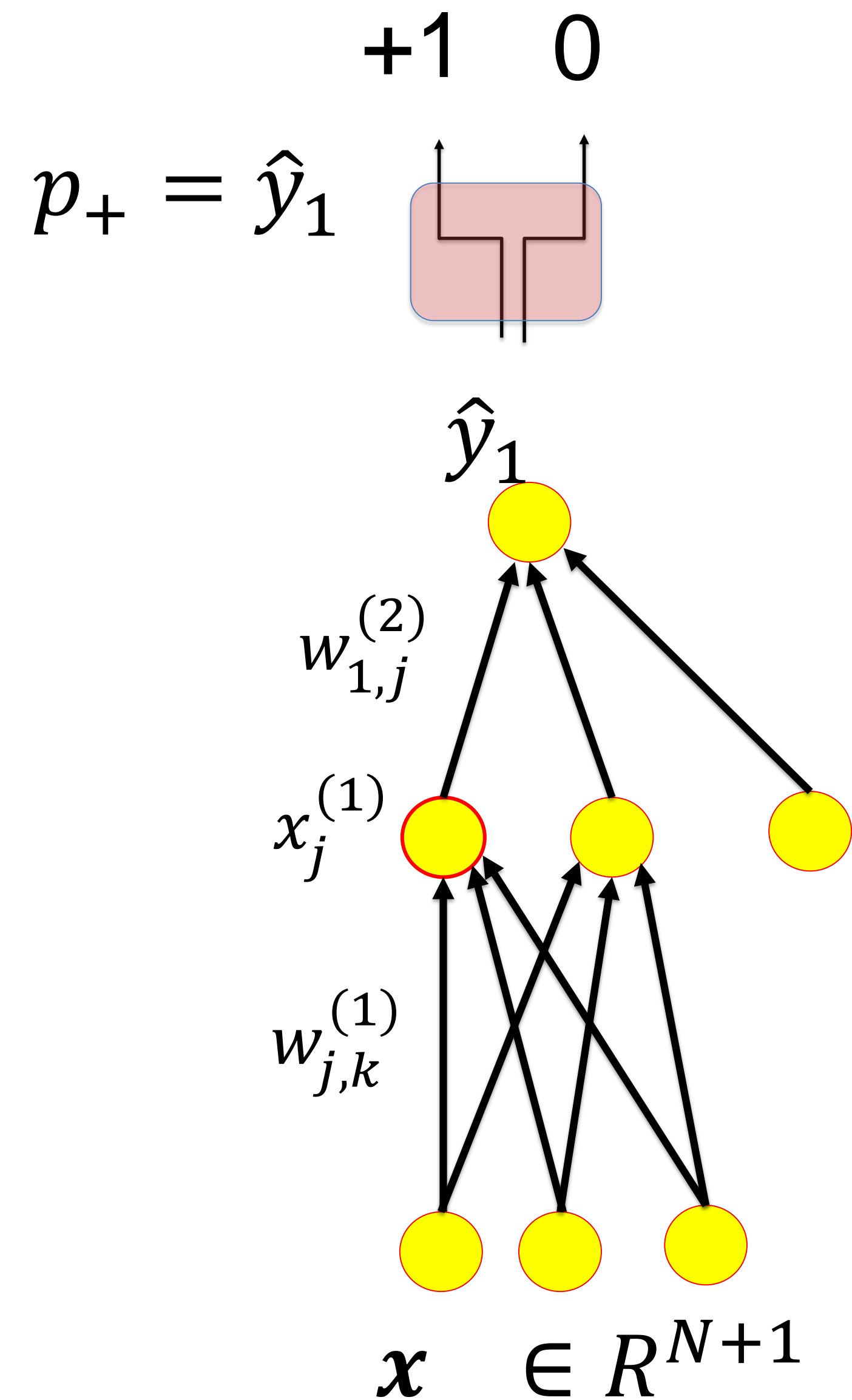
- Probability must be between 0 and 1
- Intuitively: smooth is better



Blackboard 6:
derive optimal sigmoidal



Blackboard 6:
derive optimal sigmoidal

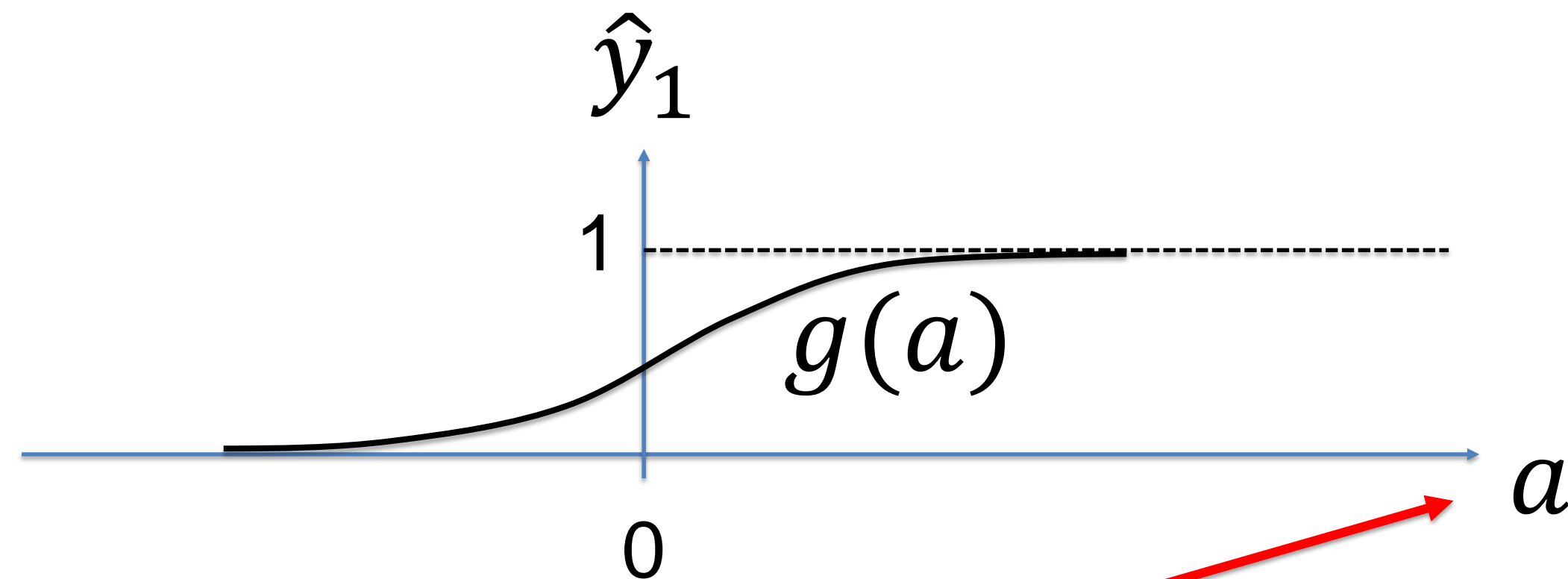


5. Why sigmoidal output ? – single class

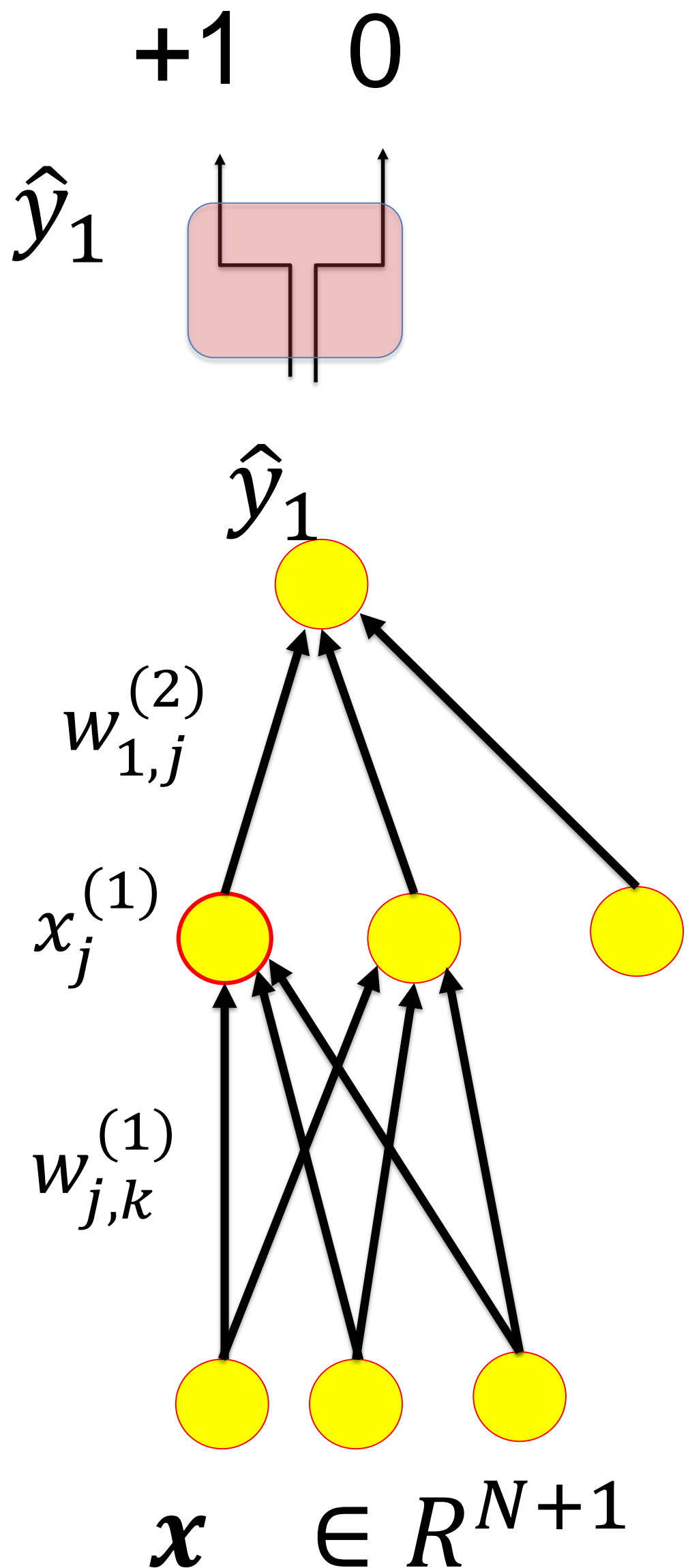
$$\hat{y}_1 = P(C_1|\mathbf{x}) = P(\hat{t}_1 = 1|\mathbf{x})$$

$$p_+ = \hat{y}_1$$

$$\hat{y}_1 = g(a) = \frac{1}{1 + e^{-a}}$$



total input a into output neuron can be interpreted as log-prob. ratio



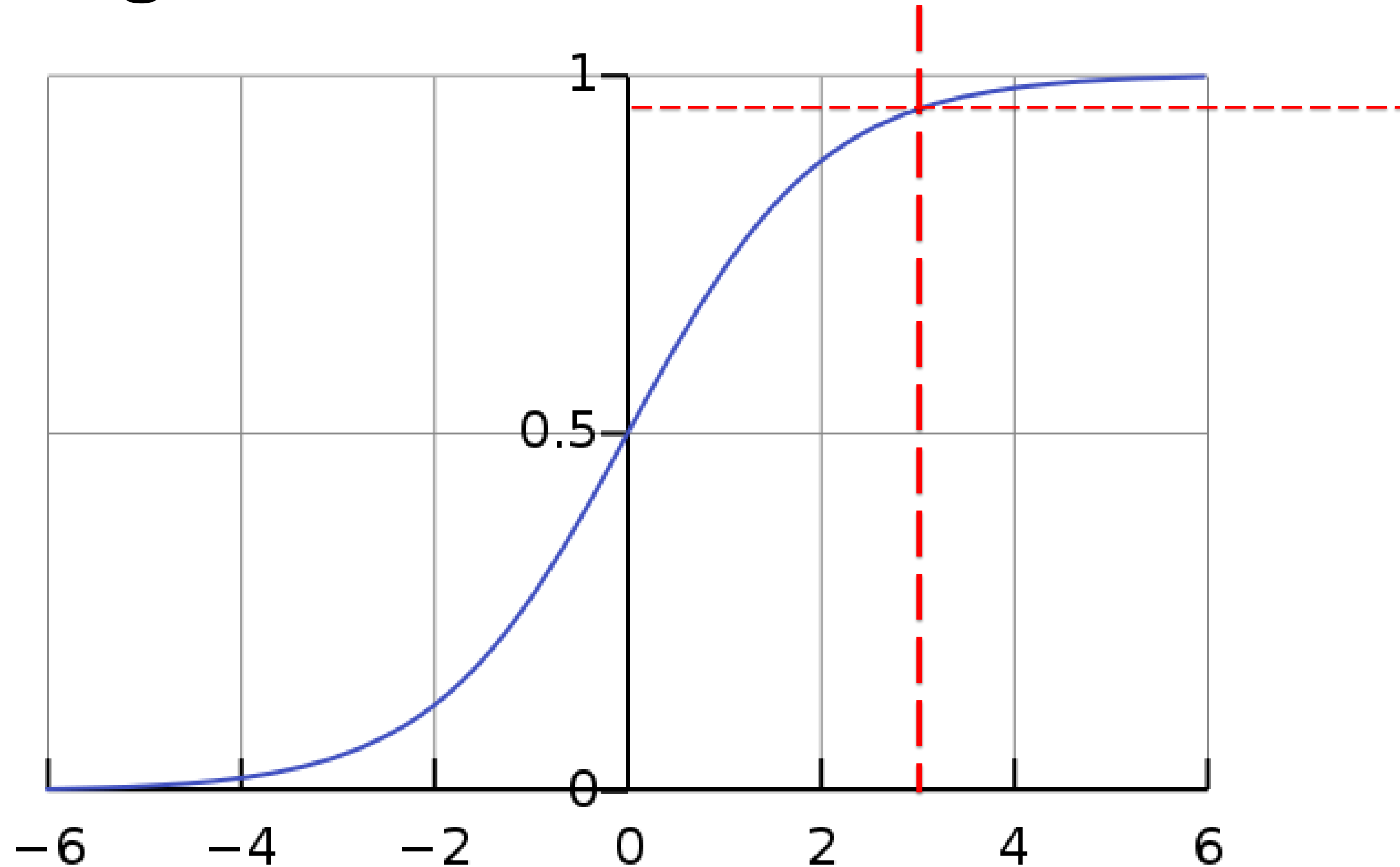
5. sigmoidal output = **logistic function**

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

Rule of thumb:

for $a = 3$: $g(3) = 0.95$

for $a = -3$: $g(-3) = 0.05$



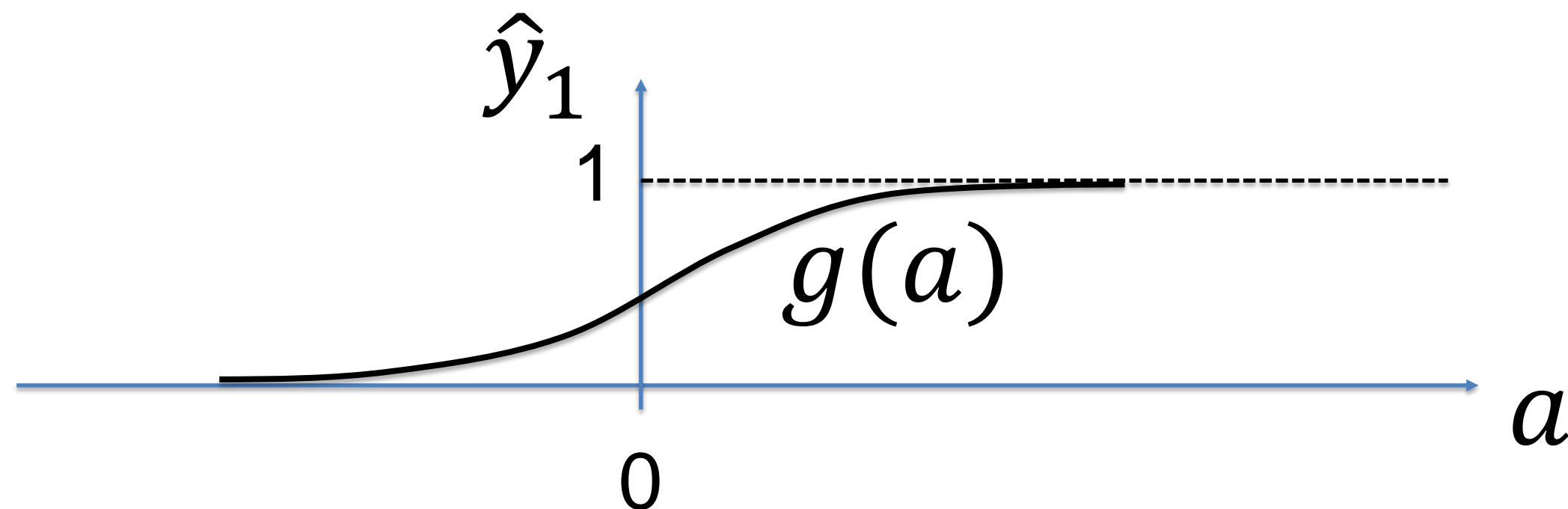
https://en.wikipedia.org/wiki/Logistic_function

5. Why sigmoidal output ?

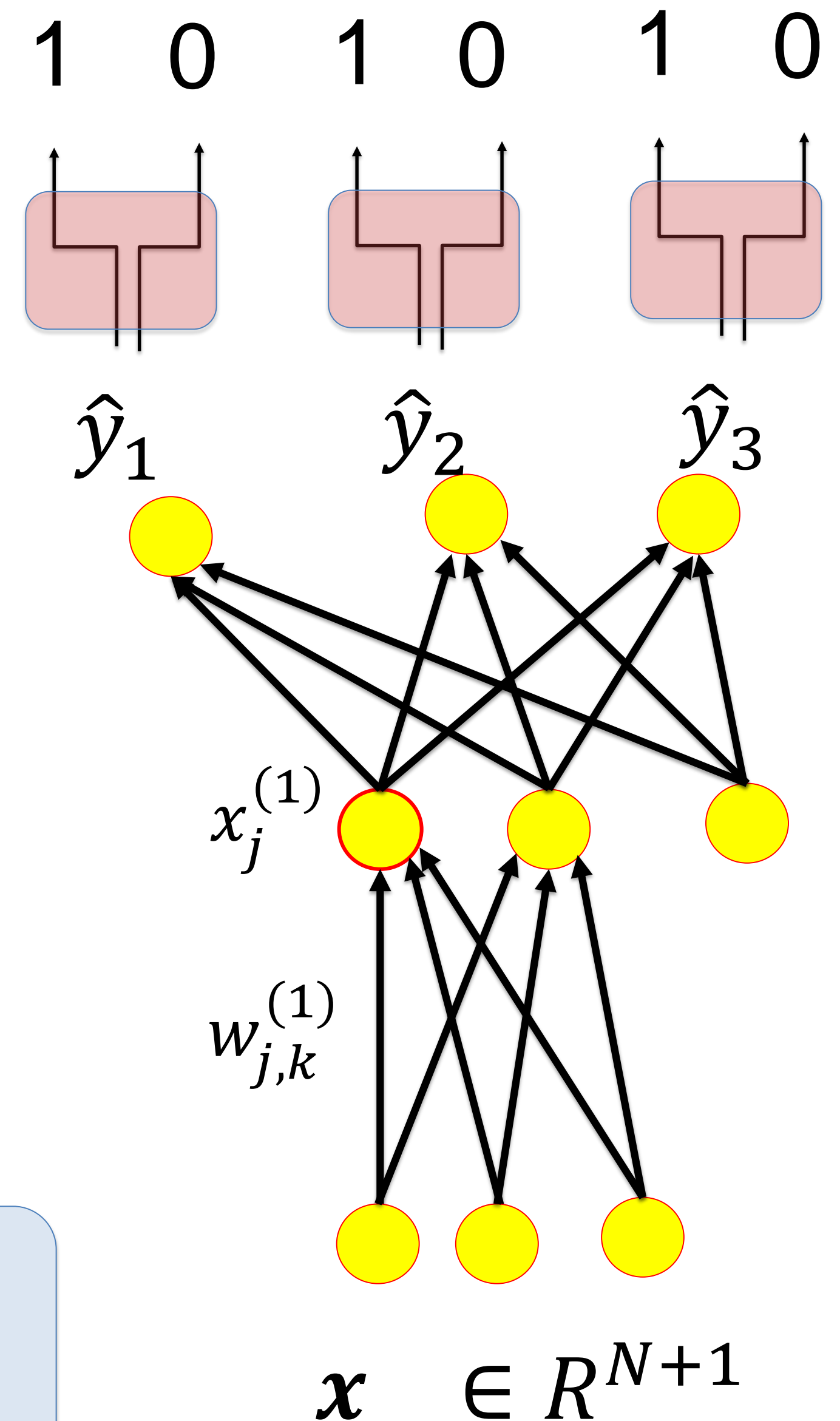
$$\hat{y}_1 = P(C_1|\mathbf{x}) = P(\hat{t}_1 = 1|\mathbf{x})$$

Observations (multiple-classes):

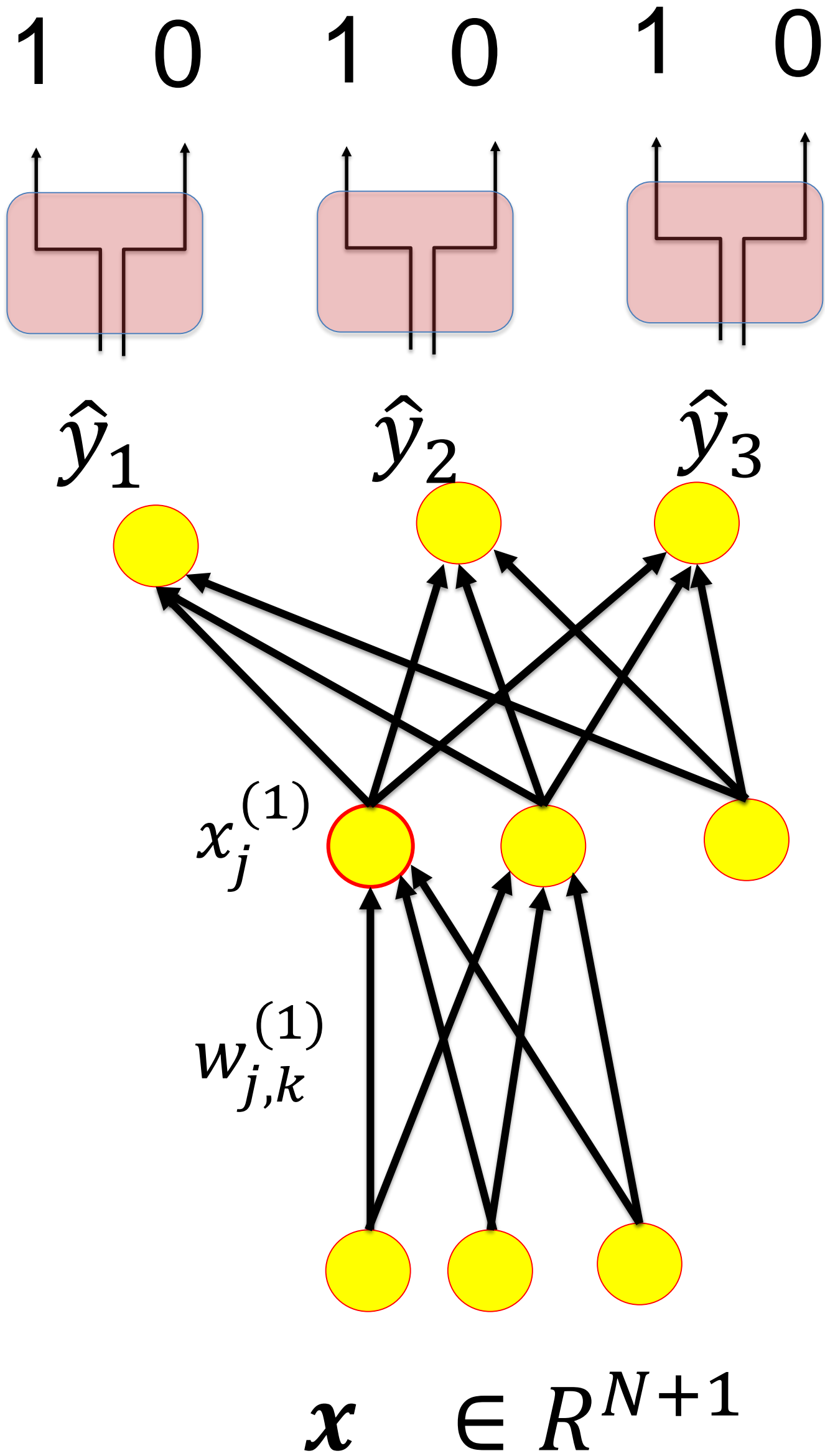
- Probabilities must sum to one!



Blackboard 7:
derive optimal output



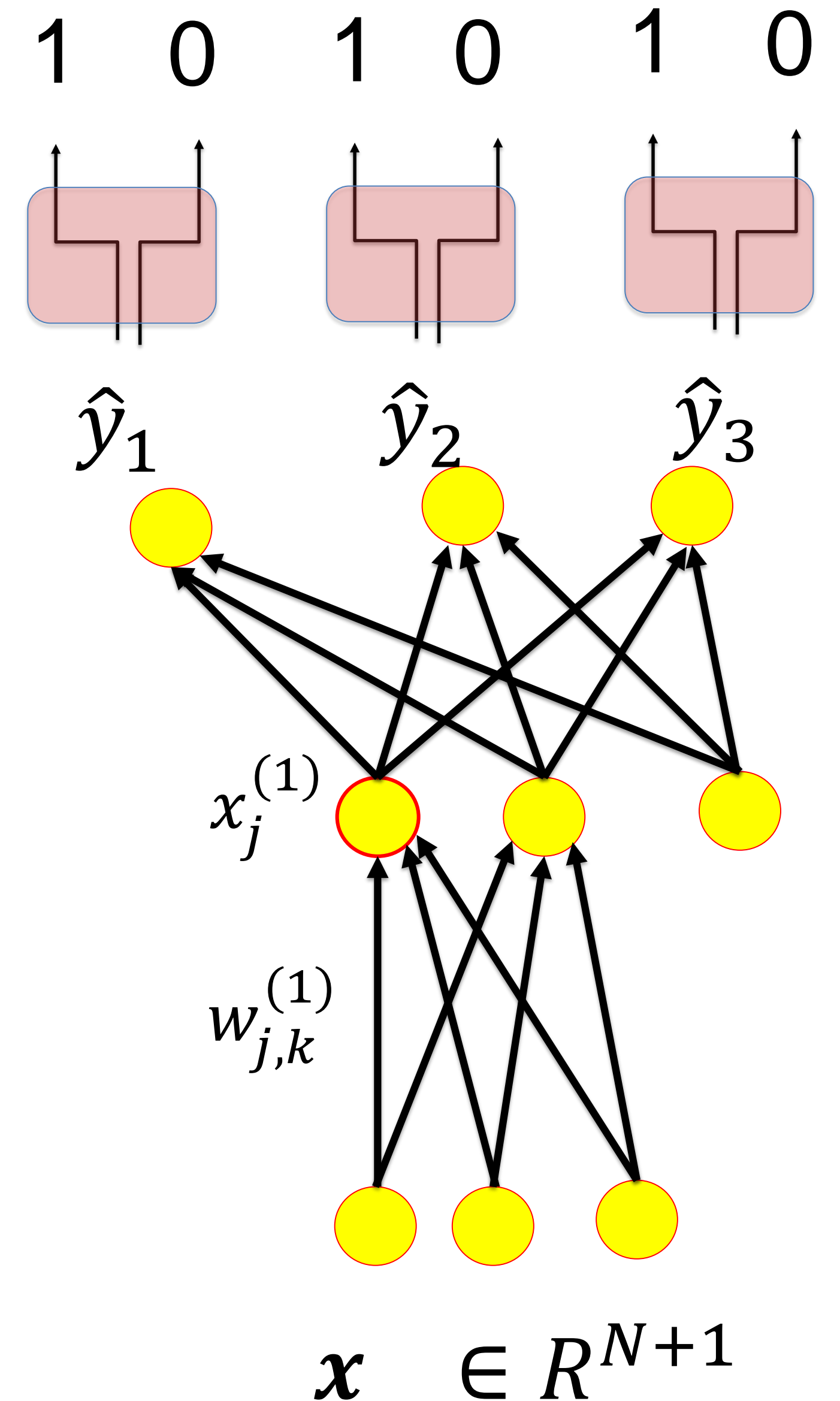
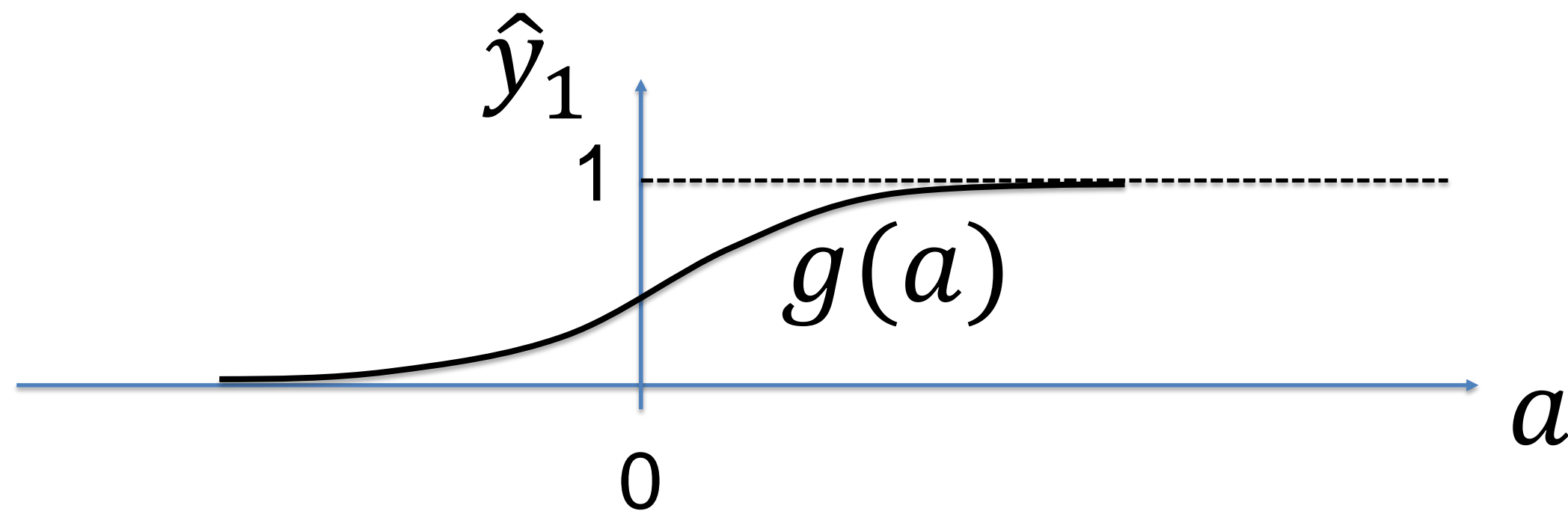
Blackboard 7:
derive optimal output



5. Softmax output

$$\hat{y}_k = P(C_k|\mathbf{x}) = P(\hat{t}_k = 1|\mathbf{x})$$

$$\hat{y}_k = P(C_k|\mathbf{x}) = \frac{\exp(a_k)}{\sum_j \exp(a_j)}$$



Artificial Neural Networks: Lecture 3

Statistical Classification by Deep Networks

1. The statistical view: generative model
2. The likelihood of data under a model
3. Application to artificial neural networks
4. Multi-class problems
5. Sigmoidal as a natural output function
6. **Rectified linear for hidden units**

6. Modern Neural Networks

output layer

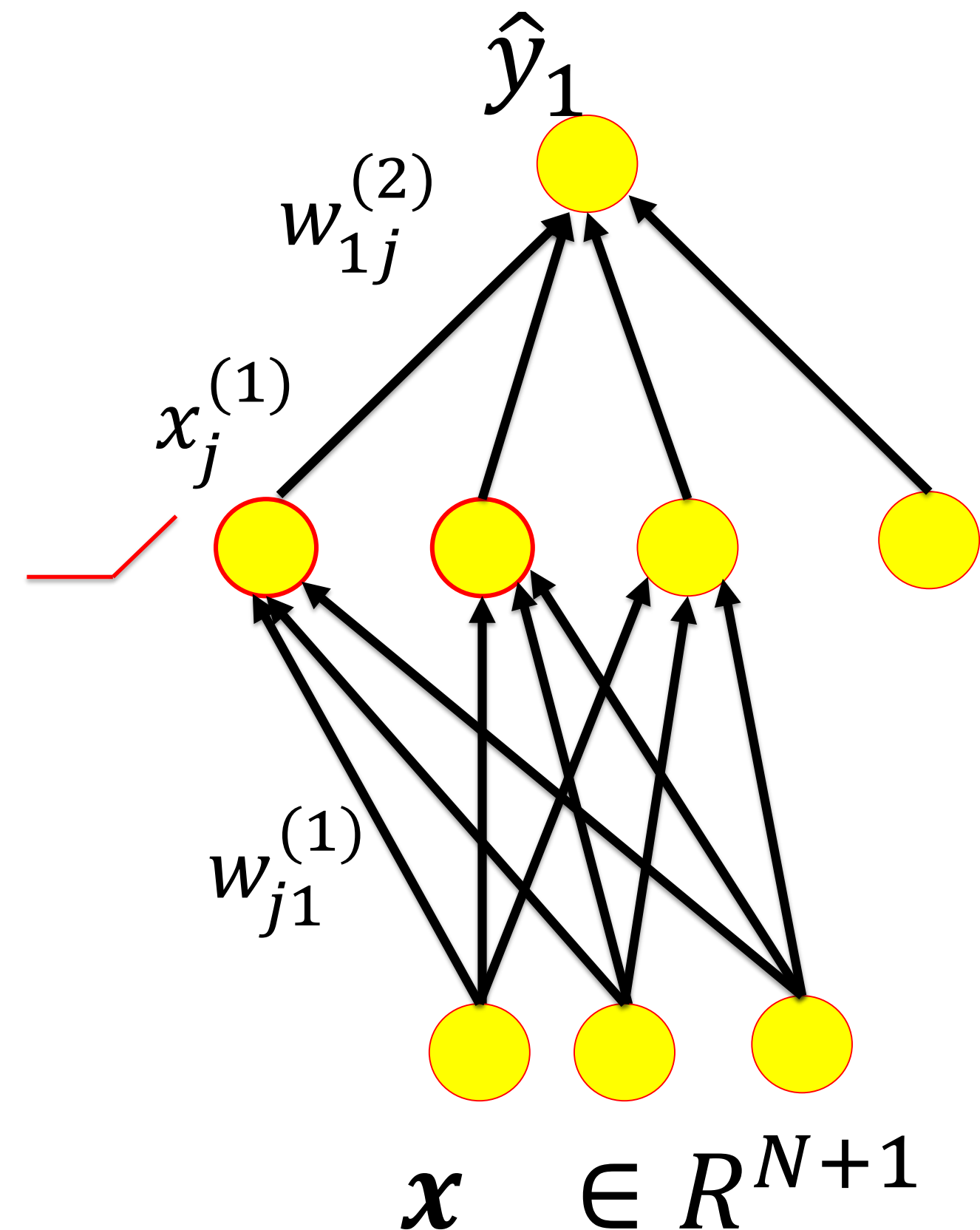
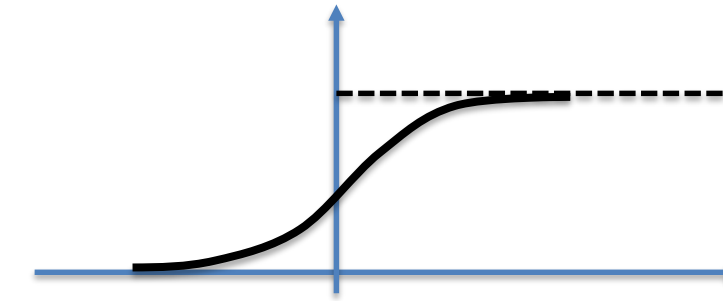
use sigmoidal unit (single-class)
or softmax (exclusive multiclass)

hidden layer

use rectified linear unit in $N+1$ dim.

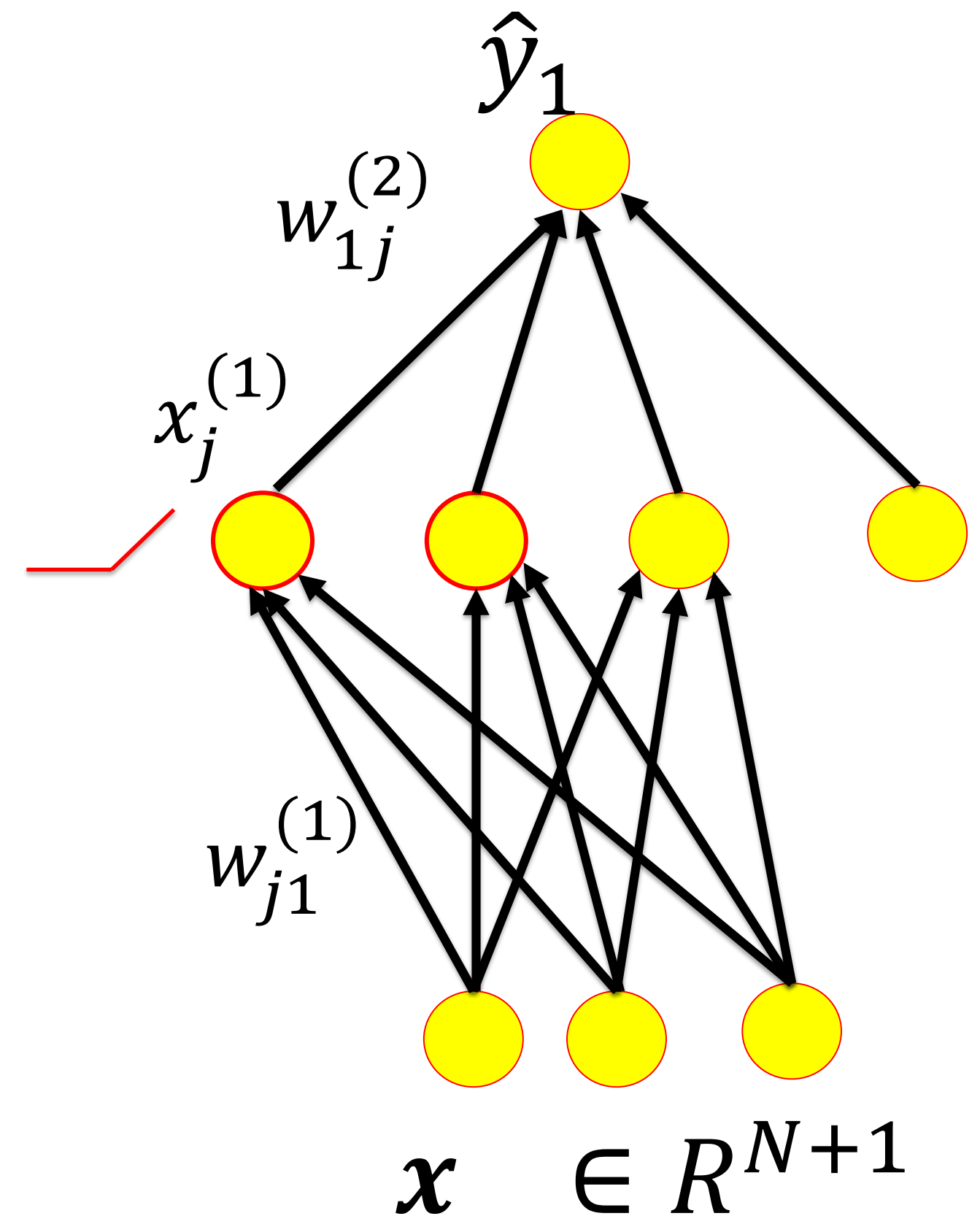
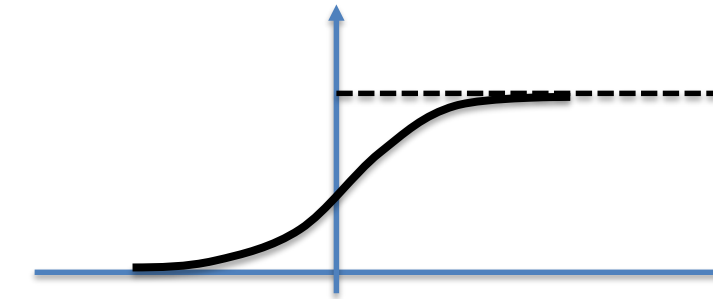
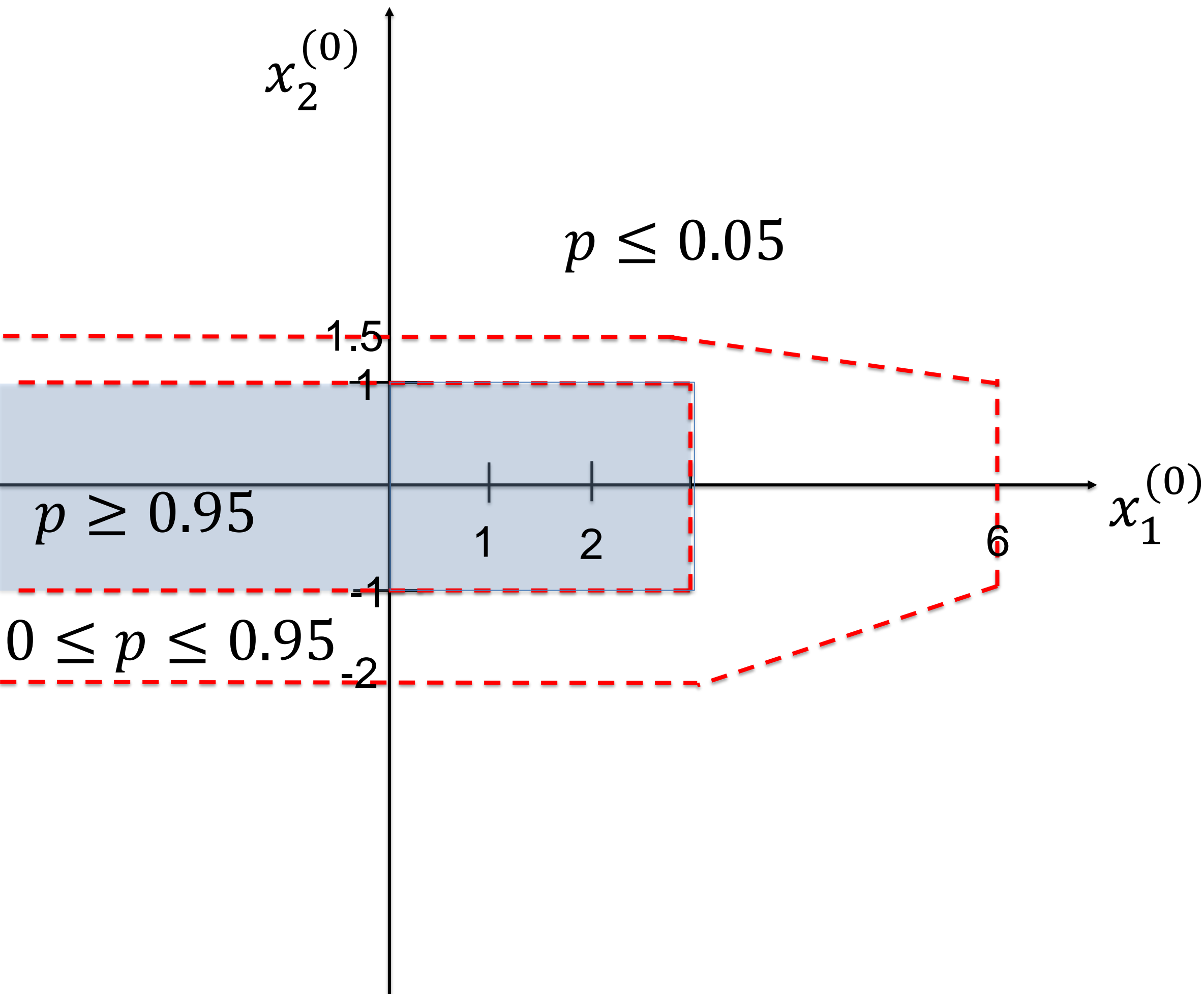

$$f(x) = x \text{ for } x > 0$$

$$f(x) = 0 \text{ for } x < 0 \text{ or } x = 0$$

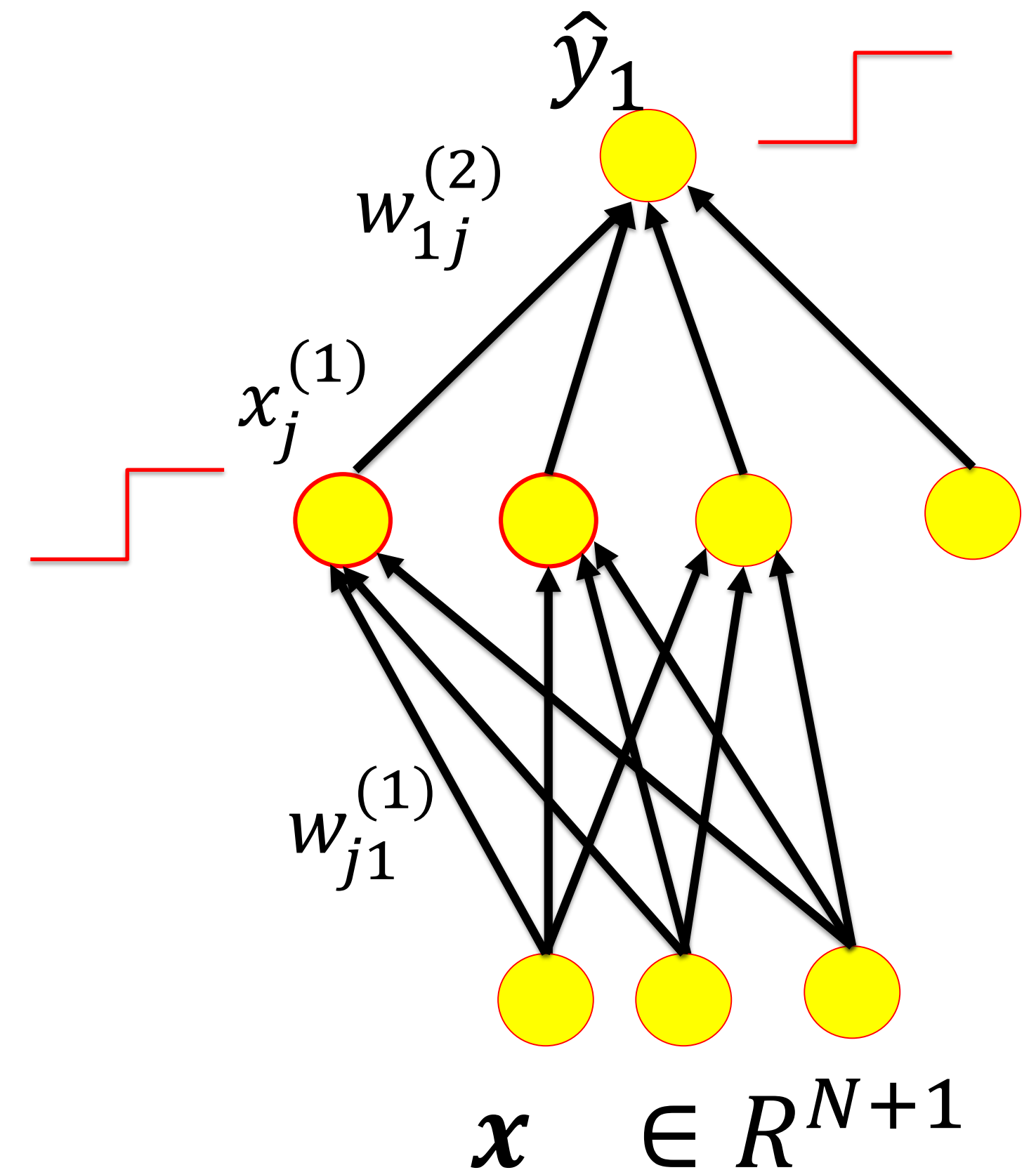
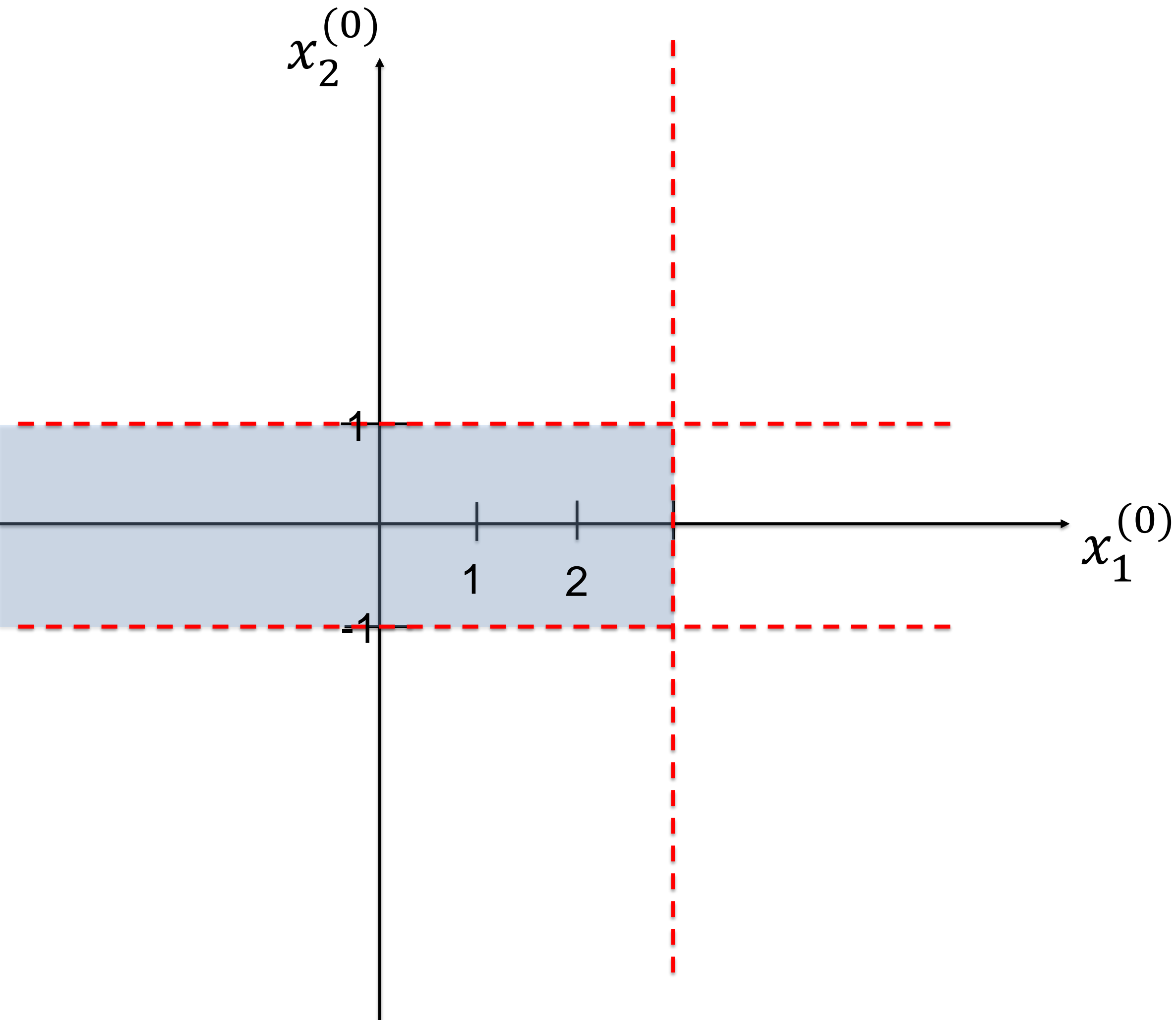


Preparation for Exercises:

Link multilayer networks to probabilities



Preparation for Exercises: there are many solutions!!!!



QUIZ: Modern Neural Networks

- [] piecewise linear units should be used in all layers
- [] piecewise linear units should be used in the hidden layers
- [] softmax unit should be used for exclusive multi-class in an output layer with 1-hot coding
- [] sigmoidal unit should be used for single-class problems
- [] two-class problems are the same as single-class problems
- [] multiple-attribute-class problems are the same as single-class
- [] it's great we can interpret the output as a probability, because

$$\hat{y}_1 = P(C_1|\mathbf{x})$$

- [] if we are careful in the model design, we may interpret the output as a probability that the data belongs to the class

Artificial Neural Networks: Lecture 3

Statistical classification by deep networks

Objectives for today:

- The cross-entropy error is the optimal loss function for classification tasks
- The sigmoidal (softmax) is the optimal output unit for classification tasks
- Exclusive Multi-class by '1-hot coding'
- Under certain conditions we may interpret the output as a probability
- Piecewise linear units are preferable for hidden layers

Reading for this lecture:

Bishop 2006, Ch. 4.2 and 4.3

Pattern recognition and Machine Learning

or

Bishop 1995, Ch. 6.7 – 6.9

Neural networks for pattern recognition

or

Goodfellow et al., 2016 Ch. 5.5 and 3.13 of

Deep Learning