Multilayer Perceptron

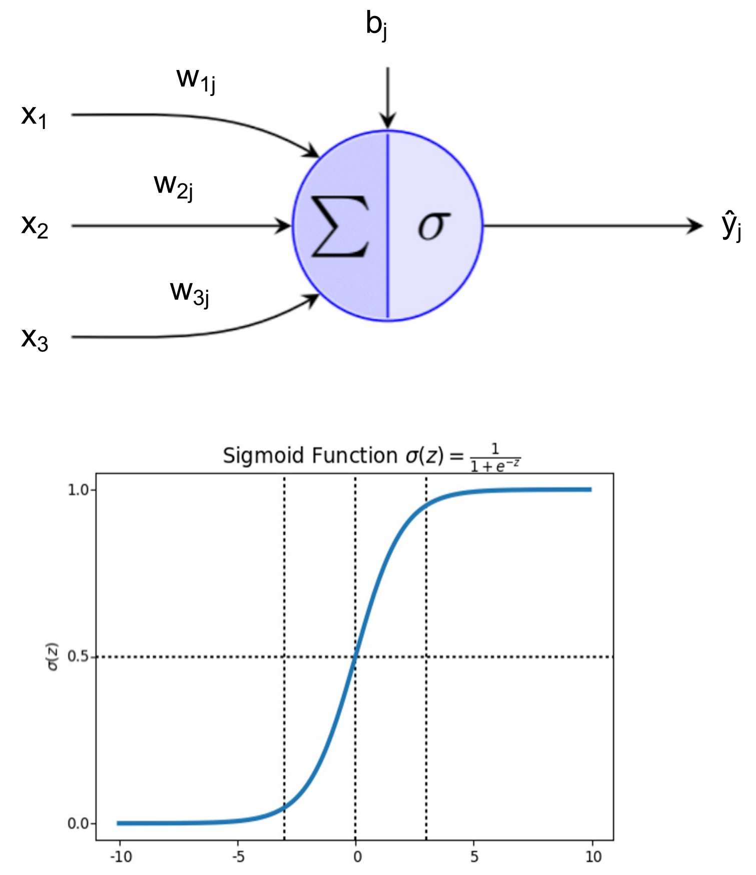
AI+X
INSTITUTE FOR ARTIFICIAL INTELLIGENCE

Perceptron

- Input may have multiple values
- Output may have multiple values
  - Multiple parallel Perceptrons
  - Input is the same for all Perceptrons
- Activation function – sigmoid
  - $\sigma' = \sigma(1 - \sigma)$

$$\hat{\boldsymbol{y}} = \sigma\,(W\mathbf{x} + \mathbf{b})$$

$$\mathcal{L} = ({}^{1}/{}_{2})\,\Sigma_k\,(\hat{y}_k - y_k)^2$$



[1] Rosenblatt, F. (1957). The Perceptron – a perceiving and recognizing automaton. Cornell Aeronautical Laboratory, Inc. Report

$$\frac{\partial \mathcal{L}}{\partial w_{ij}} = \frac{\partial}{\partial w_{ij}} (½)\sum(\hat{y}_k - y_k)^2 = \frac{\partial}{\partial w_{ij}} (½)(\hat{y}_j - y_j)^2 =$$

$$(\hat{y}_j - y_j)\frac{\partial}{\partial w_{ij}}(\hat{y}_j - y_j) = (\hat{y}_j - y_j)\frac{\partial}{\partial w_{ij}}\sigma(\mathbf{xw}_{*j} + b_j) - y_j =$$

$$(\hat{y}_j - y_j)\sigma(\hat{y}_j)(1-\sigma(\hat{y}_j))\frac{\partial}{\partial w_{ij}}\mathbf{xw}_{*j} = (\hat{y}_j - y_j)\sigma(\hat{y}_j)(1-\sigma(\hat{y}_j))x_i$$

$$\frac{\partial \mathcal{L}}{\partial b_j} = (\hat{y}_j - y_j)\sigma(\hat{y}_j)(1-\sigma(\hat{y}_j))$$

Multilayer Perceptron (MLP)

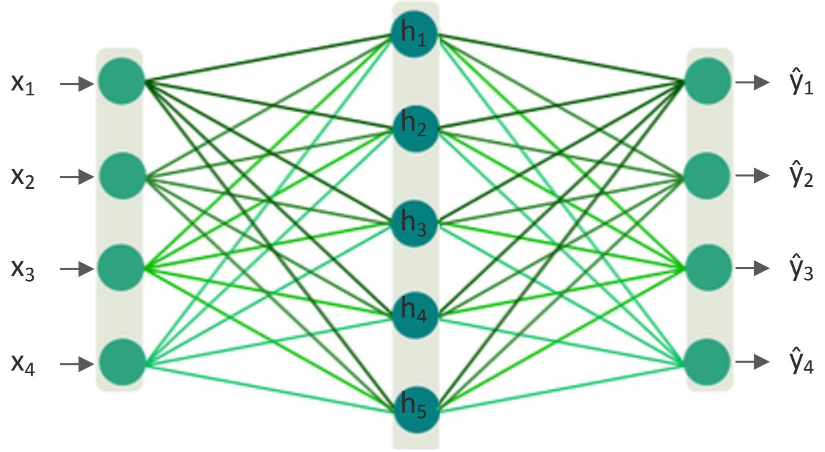
- Consecutive layers of parallel Perceptrons
- Introduces the concept of hidden layer

$$\mathbf{h} = \sigma(W_1\mathbf{x} + \mathbf{b_1})$$
$$\hat{\mathbf{y}} = \sigma(W_2\mathbf{h} + \mathbf{b_2})$$

$$\mathcal{L} = (^1/_2)\,\Sigma_k\,(\hat{y}_k - y_k)^2$$

Number of parameters:

$$|W_1| = |\mathbf{x}| * |\mathbf{h}|$$
$$|b_1| = |\mathbf{h}|$$
$$|W_2| = |\mathbf{h}| * |\mathbf{y}|$$
$$|b_2| = |\mathbf{y}|$$



[1] Rosenblatt, F. (1961). Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms. Spartan Books, Washington DC
[2] Rumelhart, D., Hinton, G. & Williams, R. (1986). Learning representations by back-propagating errors. Nature 323, 533-536

Knowledge Check 1

What happens if we do not use an activation function in between layers of parallel Perceptrons?

- A. Nothing changes
- B. The network loses representation capacity, so we will need to add many more layers to achieve similar results
- C. No matter how many layers we use, the network will have the same capacity of a single-layer network

Why do we need activation functions?

- Last layer: regularize output
- Intermediate layers: handle nonlinearities
  - Consecutive linear layer are equivalent to a single layer

$$\mathbf{h} = W_1\mathbf{x} + \mathbf{b_1}$$
$$\hat{\mathbf{y}} = W_2\mathbf{h} + \mathbf{b_2}$$

$$\hat{\mathbf{y}} = W_2(\mathbf{W_1}\mathbf{x} + \mathbf{b_1}) + \mathbf{b_2}$$

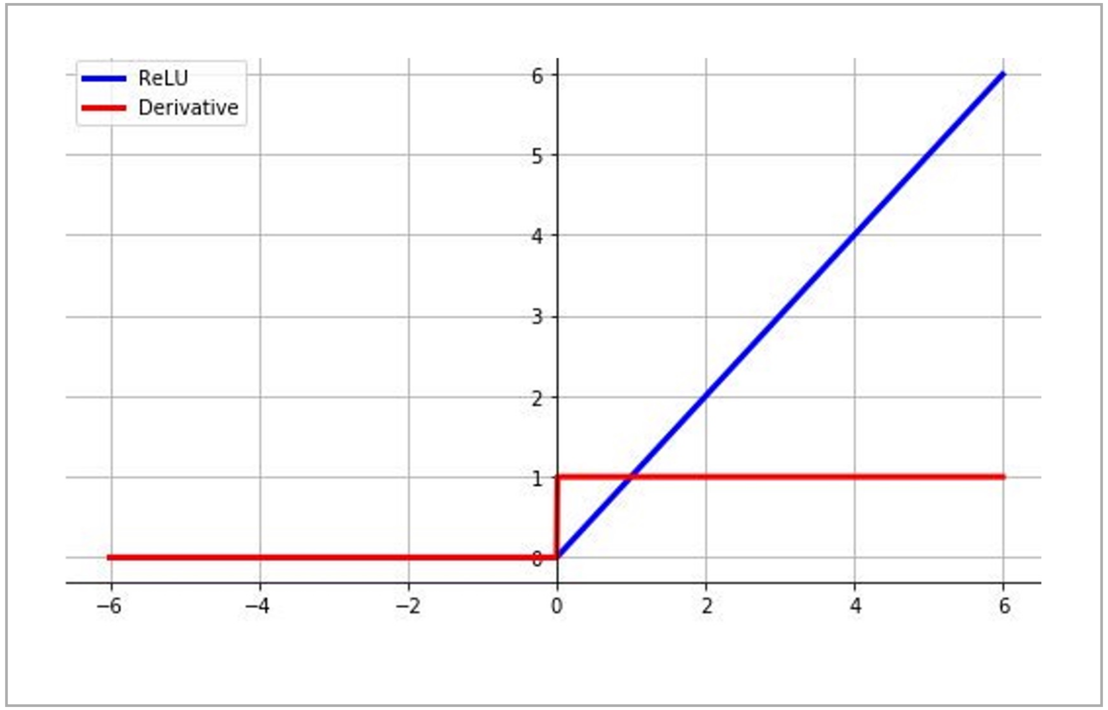
$$\hat{\mathbf{y}} = W_2W_1\mathbf{x} + W_2\mathbf{b_{1+}}\ \mathbf{b_2}$$

$$W = W_2W_1$$

$$\mathbf{b} = W_2\mathbf{b_1} + \mathbf{b_2}$$

Activation Functions

- Rectified Linear Unit (ReLU)
- ReLU(z) = max(0,z)
    - ReLU'(z) = 1 if z ≥ 0
    - ReLU'(z) = 0 if z < 0
- Simplicity
- Linear behavior
- Avoids saturation

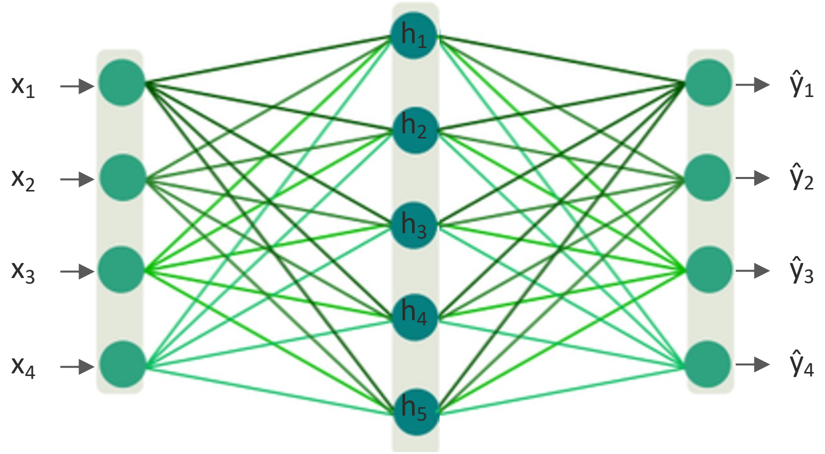


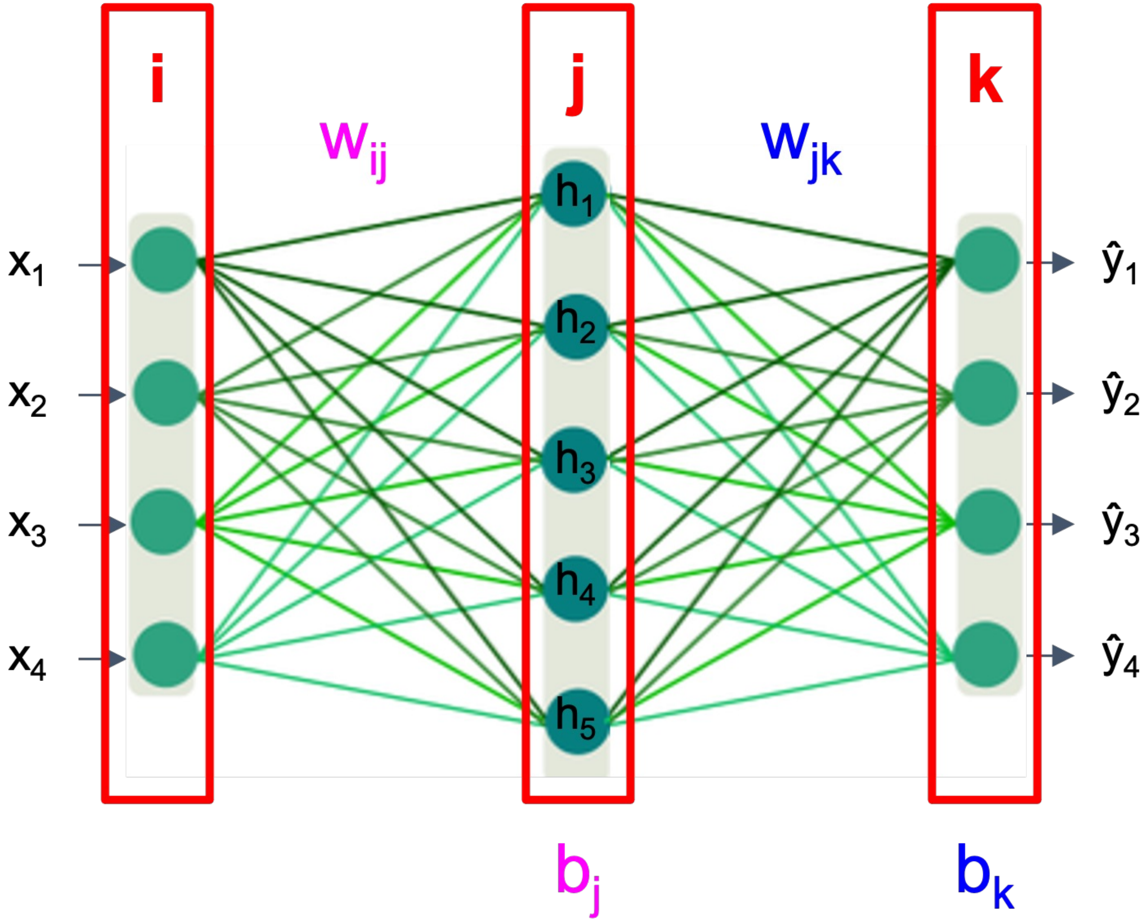
[1] Hahnloser, R. & Seung, H. (2001). Permitted and Forbidden Sets in Symmetric Threshold-Linear Networks. Neural Information Processing Systems
[2] Glorot, X., Bordes, A. & Bengio, Y. (2011). Deep Sparse Rectifier Neural Networks. International Conference on Artificial Intelligence and Statistics

Updated Multilayer Perceptron

$$\mathbf{h} = \mathrm{ReLU}(W_1\mathbf{x} + \mathbf{b_1})$$
$$\hat{\mathbf{y}} = W_2\mathbf{h} + \mathbf{b_2}$$

$$\mathcal{L} = (^1/_2)\, \Sigma_k\, (\hat{y}_k - y_k)^2$$




i
j
k
w_{ij}
w_{jk}
x_1
x_2
x_3
x_4
h_1
h_2
h_3
h_4
h_5
$\hat{y}_1$
$\hat{y}_2$
$\hat{y}_3$
$\hat{y}_4$
b_j
b_k

$$w_{jk}^{t+1} = w_{jk}^{t} - \lambda \nabla_{w[jk]} = w_{jk}^{t} - \lambda \frac{\partial \mathcal{L}}{\partial w_{jk}}$$

$$b_{k}^{t+1} = b_{k}^{t} - \lambda \nabla_{b[k]} = b_{k}^{t} - \lambda \frac{\partial \mathcal{L}}{\partial b_{k}}$$

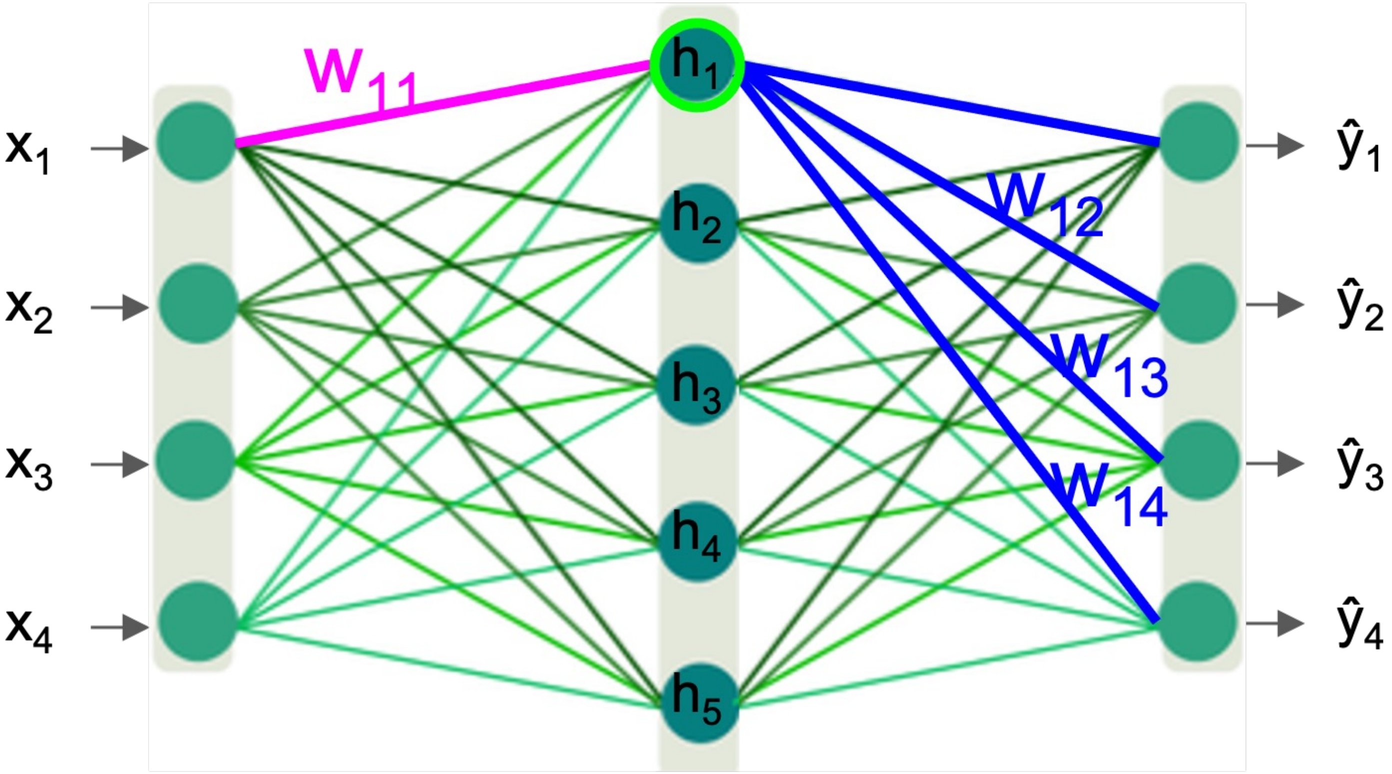
$$\delta_k = (\hat{y}_k - y_k)$$

$$\frac{\partial \mathcal{L}}{\partial w_{jk}} = (\hat{y}_k - y_k)h_j = \delta_k h_j$$

$$\frac{\partial \mathcal{L}}{\partial b_k} = (\hat{y}_k - y_k) = \delta_k$$

$$w_{ij}^{t+1} = w_{ij}^{t} - \lambda \nabla_{w[ij]} = w_{ij}^{t} - \lambda \frac{\partial \mathcal{L}}{\partial w_{ij}}$$

$$b_{j}^{t+1} = b_{j}^{t} - \lambda \nabla_{b[j]} = b_{j}^{t} - \lambda \frac{\partial \mathcal{L}}{\partial b_{j}}$$


w11
h1
h2
h3
h4
h5
x1
x2
x3
x4
w12
w13
w14
ŷ1
ŷ2
ŷ3
ŷ4

$$\frac{\partial \mathcal{L}}{\partial w_{ij}} = \frac{\partial}{\partial w_{ij}}(½)\sum_k(\hat{y}_k - y_k)^2 = (½)\sum_k \frac{\partial}{\partial w_{ij}}(\hat{y}_k - y_k)^2 =$$

$$\sum_k(\hat{y}_k - y_k)\frac{\partial}{\partial w_{ij}}\hat{y}_k = \sum_k \delta_k \frac{\partial}{\partial w_{ij}}\mathbf{h}\mathbf{w}_{*k} + b_k = \sum_k \delta_k \frac{\partial}{\partial w_{ij}} h_j w_{jk} =$$

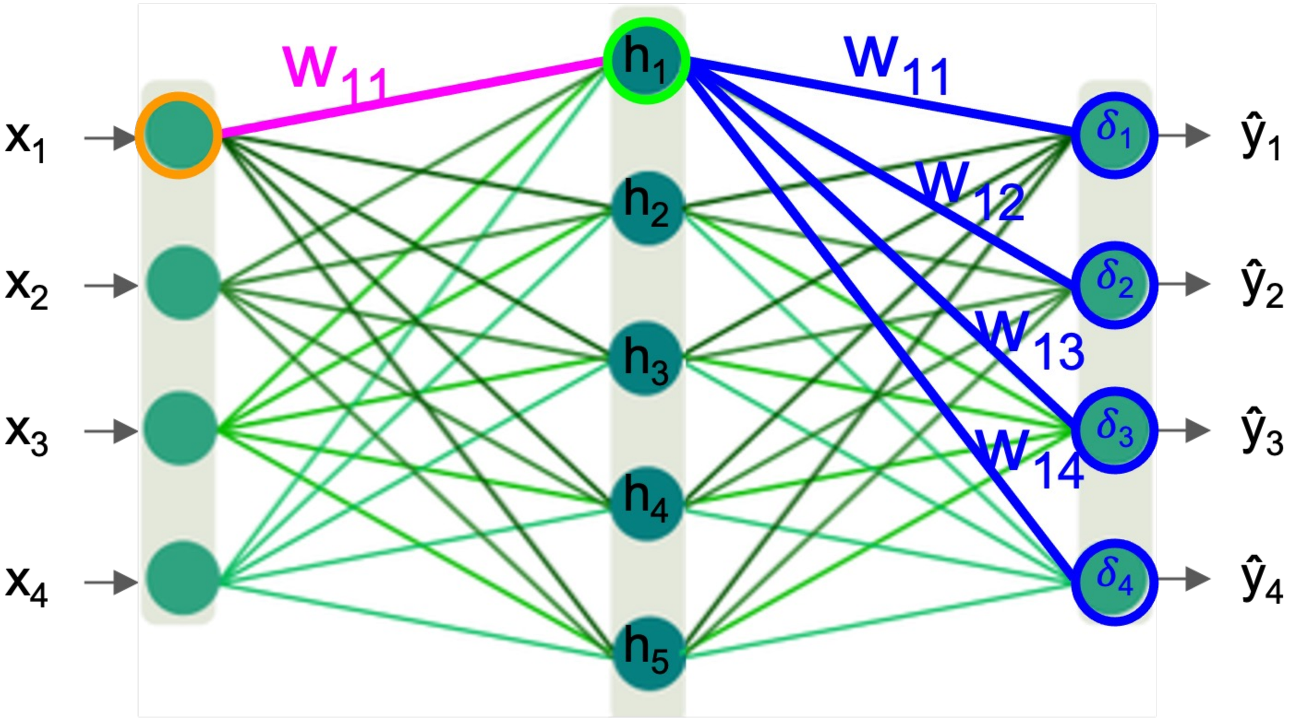
$$\sum_k \delta_k \frac{\partial}{\partial h_j} h_j w_{jk} \frac{\partial}{\partial w_{ij}} h_j = \sum_k \delta_k w_{jk} \frac{\partial}{\partial w_{ij}} \mathrm{ReLU}(\mathbf{x}\mathbf{w}_{*j} + b_j) =$$

$$\sum_k \delta_k w_{jk} \mathrm{ReLU}'(h_j) \frac{\partial}{\partial w_{ij}} \mathbf{x}\mathbf{w}_{*j} + b_j = \mathrm{ReLU}'(h_j) x_i \sum_k \delta_k w_{jk}$$

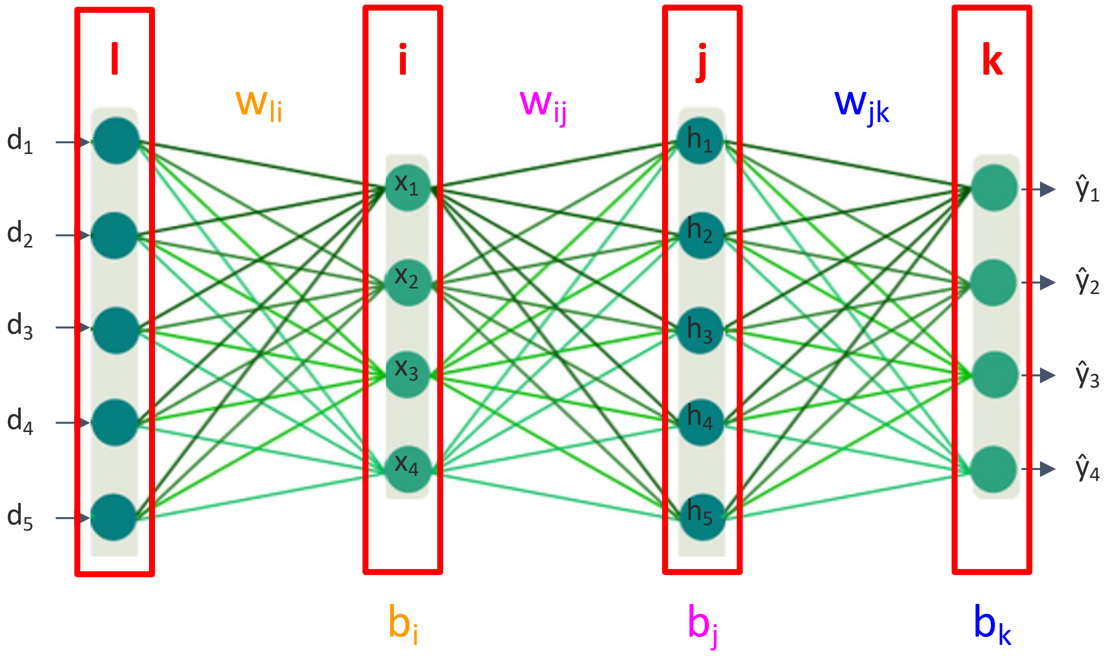
$$\delta_j = \begin{cases} \sum_k \delta_k w_{jk} & \text{if } h_j \geq 0 \\ 0 & \text{if } h_j < 0 \end{cases}$$

$$\frac{\partial \mathcal{L}}{\partial w_{ij}} = \delta_j x_i$$

$$\frac{\partial \mathcal{L}}{\partial b_j} = \delta_j$$



x_1
x_2
x_3
x_4
w_{11}
h_1
h_2
h_3
h_4
h_5
w_{11}
w_{12}
w_{13}
w_{14}
δ_1
δ_2
δ_3
δ_4
$\hat{y}_1$
$\hat{y}_2$
$\hat{y}_3$
$\hat{y}_4$



l
i
j
k
w_{li}
w_{ij}
w_{jk}
d_1
d_2
d_3
d_4
d_5
x_1
x_2
x_3
x_4
h_1
h_2
h_3
h_4
h_5
$\hat{y}_1$
$\hat{y}_2$
$\hat{y}_3$
$\hat{y}_4$
b_i
b_j
b_k

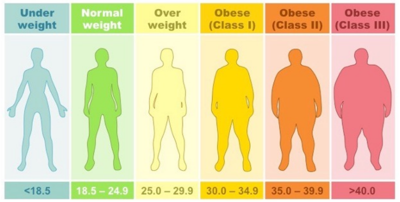
$$\delta_i = \begin{cases} \sum_j \delta_j w_{ij} & \text{if } x_i \geq 0 \\ 0 & \text{if } x_i < 0 \end{cases}$$

$$\frac{\partial \mathcal{L}}{\partial w_{li}} = \delta_i d_l$$

$$\frac{\partial \mathcal{L}}{\partial b_i} = \delta_i$$

Non-Linear Regression – Multivariate Data

- x = (height, weight) = (x_0, x_1)
- y = BMI



WEIGHT

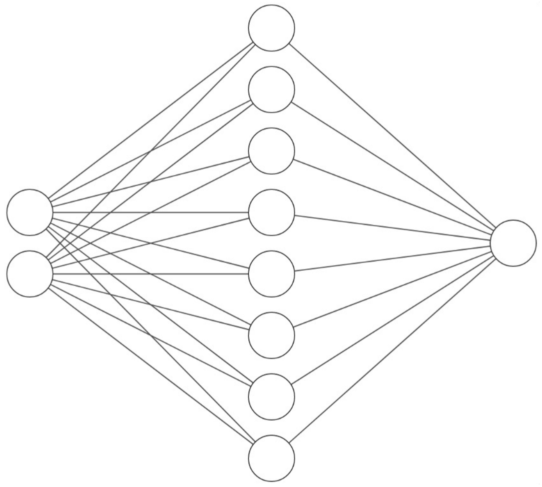
HEIGHT

	lbs	90	100	110	120	130	140	150	160	170	180	190	200	210	220	230	240	250	260	270	280	290
	kgs	41	45	50	54	59	64	68	73	77	82	86	91	95	100	104	109	113	118	122	127	132
ft/in	cm																					
4'8"	142.2	20	22	25	27	29	31	34	36	38	40	43	45	47	49	52	54	56	58	61	63	65
4'9"	144.7	19	22	24	16	28	30	32	35	37	39	41	43	45	48	50	52	54	56	58	61	63
4'10"	147.3	19	21	19	25	27	29	31	33	36	38	40	42	44	46	48	50	52	54	56	59	61
4'11"	149.8	18	20	22	24	26	28	30	32	34	26	38	40	42	44	46	48	51	53	55	57	59
5'0"	152.4	18	20	21	23	25	27	29	31	33	35	37	39	41	43	45	47	49	51	53	55	57
5'1"	154.9	17	19	21	22	25	26	28	30	32	34	36	39	40	42	43	45	47	49	51	53	55
5'2"	157.4	16	18	20	22	24	26	27	29	31	33	35	37	38	20	42	44	46	48	49	51	53
5'3"	160.0	16	18	19	21	23	25	27	28	30	32	34	35	37	39	41	43	44	46	48	50	51
5'4"	162.5	15	17	19	21	22	24	26	27	29	31	33	34	36	27	39	41	43	45	46	48	50
5'5"	165.1	15	17	18	20	22	23	25	27	28	30	32	33	35	27	38	40	42	43	45	47	49
5'6"	167.6	15	16	18	19	21	23	24	26	27	29	31	32	34	26	37	39	40	42	44	45	47
5'7"	170.1	14	16	17	19	20	22	24	25	27	29	30	31	33	25	36	38	39	41	42	44	45
5'8"	172.7	14	15	17	18	20	21	23	24	26	27	29	30	32	24	35	37	38	40	41	43	44
5'9"	175.2	13	15	16	18	19	21	22	24	25	27	28	30	31	24	34	35	37	39	40	41	43
5'10"	177.8	13	14	16	17	19	20	22	23	24	26	27	29	30	23	33	34	36	37	39	40	42
5'11"	180.3	13	14	15	14	18	20	21	22	24	25	27	28	29	22	32	33	35	36	38	39	40
6'0"	182.8	12	14	15	16	18	16	20	22	23	24	26	27	28	22	31	33	34	35	37	38	39
6'1"	185.4	12	13	15	16	17	18	20	21	22	24	25	26	28	21	30	32	33	34	36	37	39
6'2"	187.9	12	13	14	15	17	18	19	21	22	23	24	26	27	21	30	31	32	33	35	36	37
6'3"	190.5	11	13	14	15	16	17	19	20	21	23	24	25	26	20	29	30	31	32	34	35	36
6'4"	193.0	11	12	13	15	16	17	18	19	21	22	23	24	26	19	28	29	30	32	33	34	35
6'5"	195.5	11	12	13	14	15	17	18	19	20	21	23	24	25	19	27	28	30	31	32	33	34
6'6"	198.1	10	12	13	14	15	16	17	18	20	21	22	23	24	18	27	28	29	30	31	32	34
6'7"	200.6	10	11	12	14	15	16	17	18	19	20	21	23	24	18	26	27	28	29	30	32	33
6'8"	203.2	10	11	12	13	14	15	16	18	19	20	21	22	23	18	25	26	27	28	30	31	32
6'9"	205.7	10	11	12	13	14	15	16	17	18	19	20	21	23	17	25	25	27	28	29	30	31
6'10"	208.2	9	10	12	13	14	14	16	17	18	19	20	21	22	17	24	25	26	27	28	29	30
6'11"	210.8	9	10	11	12	13	14	15	16	17	18	19	20	21	16	23	25	25	27	28	29	30

Figure 9

Non-Linear Regression – Multivariate Data

- Let's use a 2-layer MLP with 8 hidden units and ReLU activation in the first layer
- $h_j = \text{ReLU}(w_{0j} * x_0 + w_{1j} * x_i + b_j)$
- $\hat{y} = w_0 h_0 + w_1 h_1 + \ldots + w_7 h_7 + b$



- One gradient per weight:

$\delta = (\hat{y} - y)$

$$\frac{\partial \mathcal{L}}{\partial b} = \nabla_b = \delta$$

$$\frac{\partial \mathcal{L}}{\partial w_j} = \nabla_{w[j] =} \delta h_j$$

$$\delta_j = \begin{cases} \delta wj \text{ if } hj \geqslant 0 \\ 0 \text{ if } hj < 0 \end{cases}$$

$$\frac{\partial \mathcal{L}}{\partial b_j} = \nabla_{b[j] =} \delta_j$$

$$\frac{\partial \mathcal{L}}{\partial w_{ij}} = \nabla_{w[i,j] =} \delta_j x_i$$

MLP - Gradient Descent

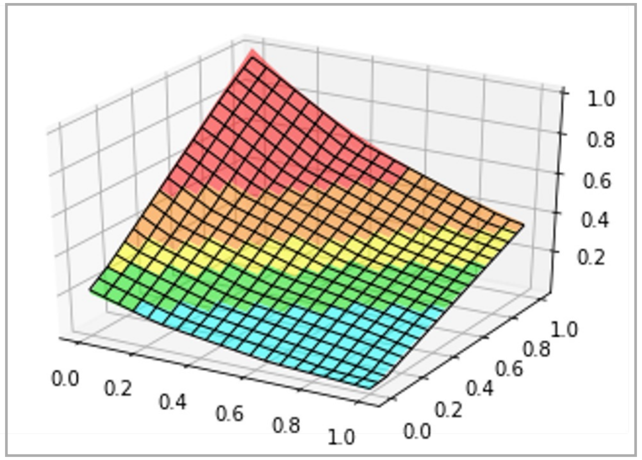
```
def gradient_descent_step(X, Y, W0, b0, W1, b1, learning_rate):
  H_ = np.matmul(X, np.transpose(W0)) + b0
  H = np.maximum(H_, 0)
  Y_ = np.matmul(H, np.transpose(W1)) + b1

  delta_Y = Y_ - Y
  W1_grad = np.matmul(np.transpose(delta_Y), H) / H.shape[0]
  b1_grad = np.mean(delta_Y, axis=0)

  delta_H = np.multiply(np.matmul(delta_Y, W1), np.where(H_>=0.0, 1.0, 0.0))
  W0_grad = np.matmul(np.transpose(delta_H), X) / X.shape[0]
  b0_grad = np.mean(delta_H, axis=0)

  W0_ = W0 - learning_rate * W0_grad
  b0_ = b0 - learning_rate * b0_grad
  W1_ = W1 - learning_rate * W1_grad
  b1_ = b1 - learning_rate * b1_grad

  return W0_, b0_, W1_, b1_
```



Common Practice #3: Weight Initialization

- Randomly initialize weights, otherwise learning is impossible
- Common approaches:
  - Normal distribution → $\mathcal{N}(0, \sigma)$, where σ = sqrt(2 / (#in + #out))
  - Uniform distribution → U(-a, a), where a = sqrt(6 / (#in + #out))
  - #in and #out are the number of inputs and outputs in a weight tensor
- Transparent to the user in modern APIs

Knowledge Check 2

When using a 2-layer MLP for solving a problem, how many hidden units should we use?

A. As many as possible, so we can maximize the performance of the network

B. As little as possible, so we can minimize the computational cost

C. This number must be defined randomly, as there is no logical way to define it

D. This number must be at least as large as the input size

You have reached the end of the lecture.