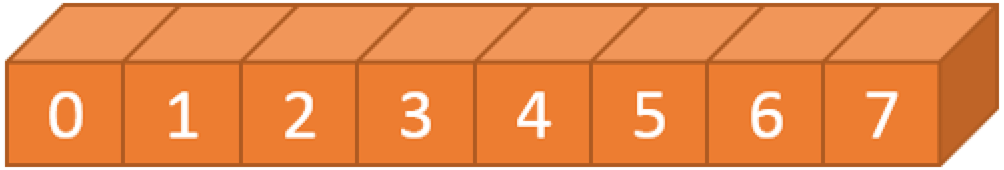
秋学期　深層学習ゼミ

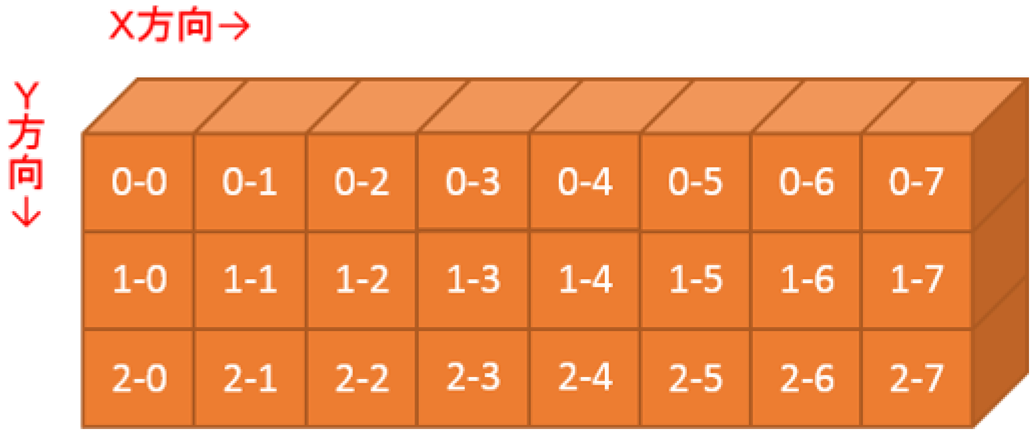
3.3 多次元配列の計算

3.3.1 多次元配列

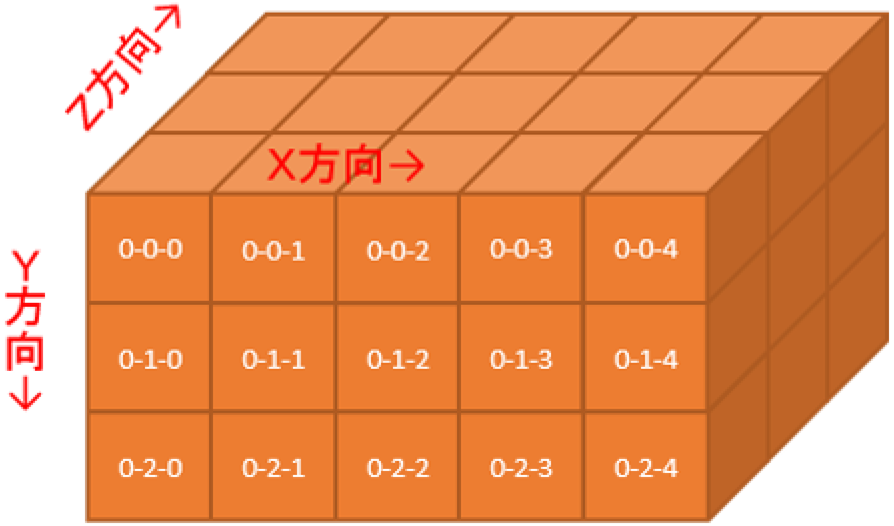
多次元配列は数字の集合



配列



2次元配列



3次元配列

多次元配列の作り方

ゼロイチゼミ

```python
import numpy as np

# (1次元)配列
A = np.array([1, 2, 3, 4])
print(A)              # 配列の表示
print(np.ndim(A))     # 配列の次元
print(A.shape)        # 配列の形状
print(A.shape[0])     # 配列の形状（タプル）の1つ目の要素

# 2次元配列
B = np.array([[1, 2], [3, 4], [5, 6]])
print(B)
print(np.ndim(B))
print(B.shape)
```

行列 = 2 次元配列

$$B = \begin{bmatrix} [1, & 2] \\ [3, & 4] \\ [5, & 6] \end{bmatrix} (2\text{次元配列}) = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix} (\text{行列})$$

$$1 \quad 2 \rightarrow \text{行} \qquad \begin{matrix} 1 \\ 3 \\ 5 \end{matrix} \rightarrow \text{列}$$

$$B\text{は}3 \times 2\text{の配列}, 3 \times 2\text{の行列}$$

3.3.2 行列の積

2×2 の場合

$$\begin{pmatrix} a_1 & a_2 \\ a_3 & a_4 \end{pmatrix} \times \begin{pmatrix} b_1 & b_2 \\ b_3 & b_4 \end{pmatrix} = \begin{pmatrix} a_1b_1 + a_2b_3 & a_1b_2 + a_2b_4 \\ a_3b_1 + a_4b_3 & a_3b_2 + a_4b_4 \end{pmatrix}$$

【例】

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \times \begin{pmatrix} 5 & 6 \\ 7 & 8 \end{pmatrix} = \begin{pmatrix} 1\times5+2\times7 & 1\times6+2\times8 \\ 3\times5+4\times7 & 3\times6+4\times8 \end{pmatrix}$$

$$= \begin{pmatrix} 19 & 22 \\ 43 & 50 \end{pmatrix}$$

行列の積の性質

行列の積には、以下の性質がある

- 行列の積は、「左の行列の列数」と「右の行列の行数」が等しくないと計算できない
- 「l×m の行列」と「m×n の行列」の積が、「l×n の行列」となる

$$\underset{l \times m}{A} \quad \underset{m \times n}{B} = \underset{l \times n}{C}$$

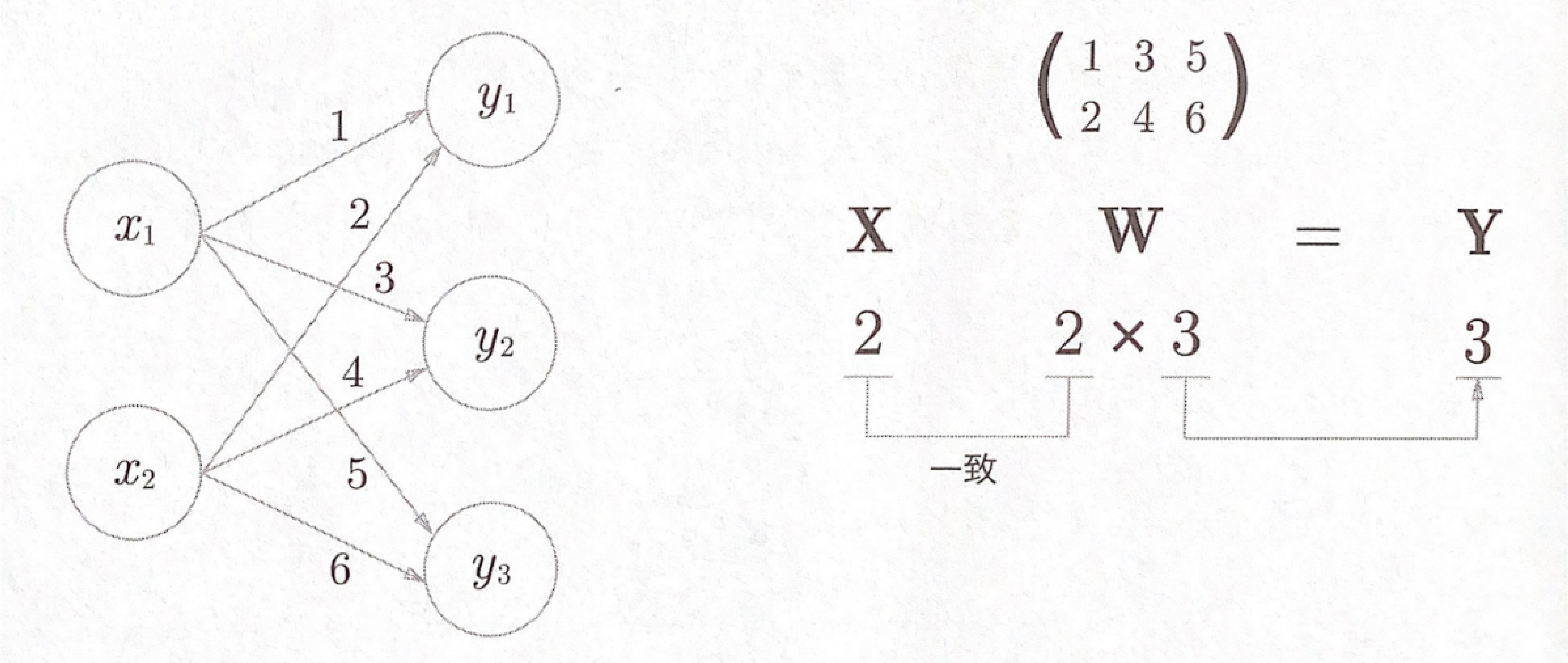
- AB と BA では結果は異なる

行列の積の計算方法

```python
# 行列の積（2×2）
A = np.array([[1, 2], [3, 4]])
B = np.array([[5, 6], [7, 8]])
print(A.shape)      # -> (2, 2)
print(B.shape)      # -> (2, 2)
print(np.dot(A, B)) # AとBの積を計算

# 行列の積（2×3と3×2）
A = np.array([[1, 2, 3], [4, 5, 6]])
B = np.array([[1, 2], [3, 4], [5, 6]])
print(A.shape)      # -> (2, 3)
print(B.shape)      # -> (3, 2)
print(np.dot(A, B))
print(np.dot(B, A)) # この2つは結果が異なる
```

ニューラルネットワークの行列の積



入力と重みを行列として積を取ると出力になる

ニューラルネットワークの行列の積

ゼロイチゼミ

```
X = np.array([1, 2])  # 入力
W = np.array([[1, 3, 5], [2, 4, 6]]) # 重み
Y = np.dot(X, W) # 出力
print(Y)
```

- 関数を一回実行するだけで出力が求まる
- ニューラルネットワークを実装する上で行列の積はとても重要

まとめ

- 多次元配列は数字の集合、2 次元配列は行列と呼ばれる
- 横の要素の並びを行、縦の要素の並びを列という
- 「$l \times m$ の行列」と「$m \times n$ の行列」の積は、「$l \times n$ の行列」となる
- ニューラルネットワークの実装において、行列の積はとても重要