aa-5. 深層学習（ディープラーニング）

（人工知能）

URL: https://www.kkaneko.jp/ai/mi/index.html

金子邦彦

CC BY NC SA

① **ニューラルネットワークの基本的仕組み、から深層学習（ディープラーニング）の概要**、分類への応用、学習の原理まで、**必要な事項**を体系的に説明。

② **ニューロンの働き**、**層構造**、**結合の重み**の役割を図を交えてビジュアルに分かりやすく説明。イメージしやすい。

③ 原理の説明と、例での説明の両方により、ニューラルネットワークの実際の応用と本質を理解。

まとめや確認問題で**理解を確実**に。**易しい内容から難しい内容まで幅広く**説明。

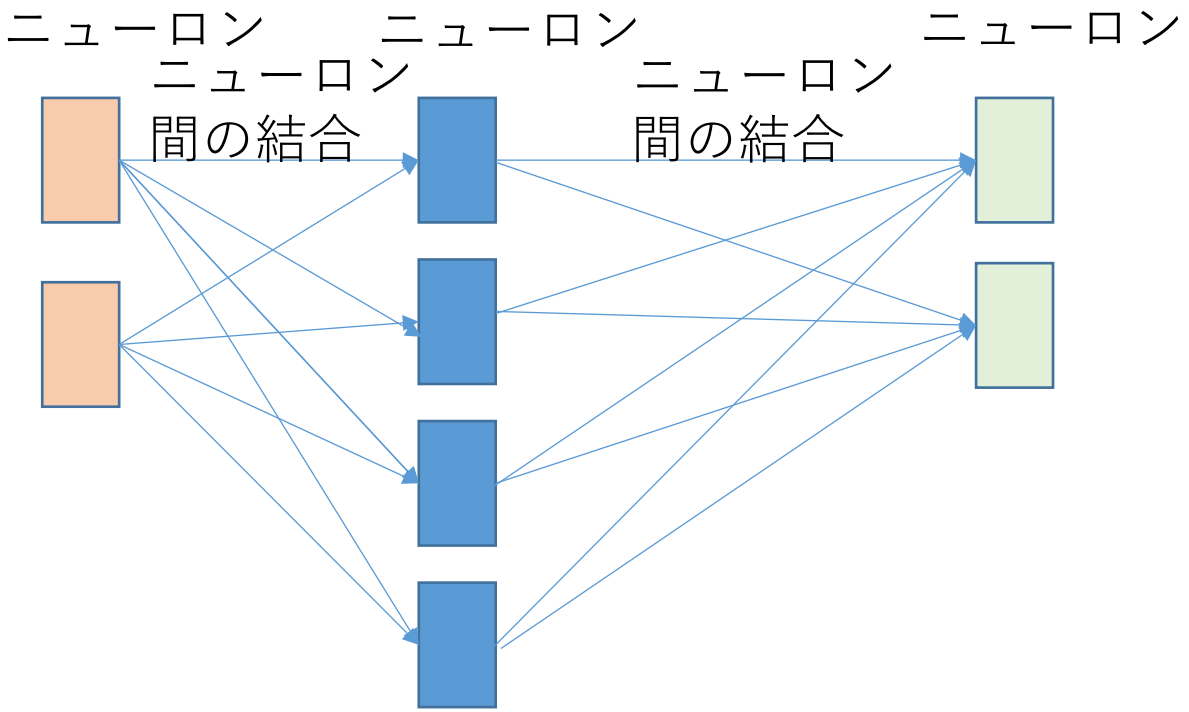
アウトライン

1. ニューラルネットワークの基本構造
2. 活性化関数とその役割
3. フォワードプロパゲーション
4. ディープラーニングの特徴と応用
5. ニューラルネットワークを用いた分類

5-1. ニューラルネットワークの基本構造

ニューラルネットワークの仕組み

入力の重みづけ，合計とバイアス，活性化関数の適用を行う**ニューロン**が**ネットワーク**を形成



入力の重みづけ，合計とバイアス，活性化関数の適用

1. **入力の重みづけ**: **各入力値に対して**，それぞれの**対応する重み**を掛ける．例： **0.3**×**0.1**　**-0.5**×**0.8**　**0.2**×**-0.5**
2. **合計とバイアス**: **合計を計算**し，**バイアス**（数値）を加える．例 **0.03　-0.4　-0.1 の合計は -0.47．これに 0.2 を加える**
3. **活性化関数の適用**: 結果に，**活性化関数**を適用．**ニューロンの活性度**を得る（→次のニューロンの入力になる）

バイアス 0.2

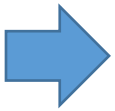
0.3　× 0.1　⇒ 0.03

-0.5　× 0.8　⇒ -0.4

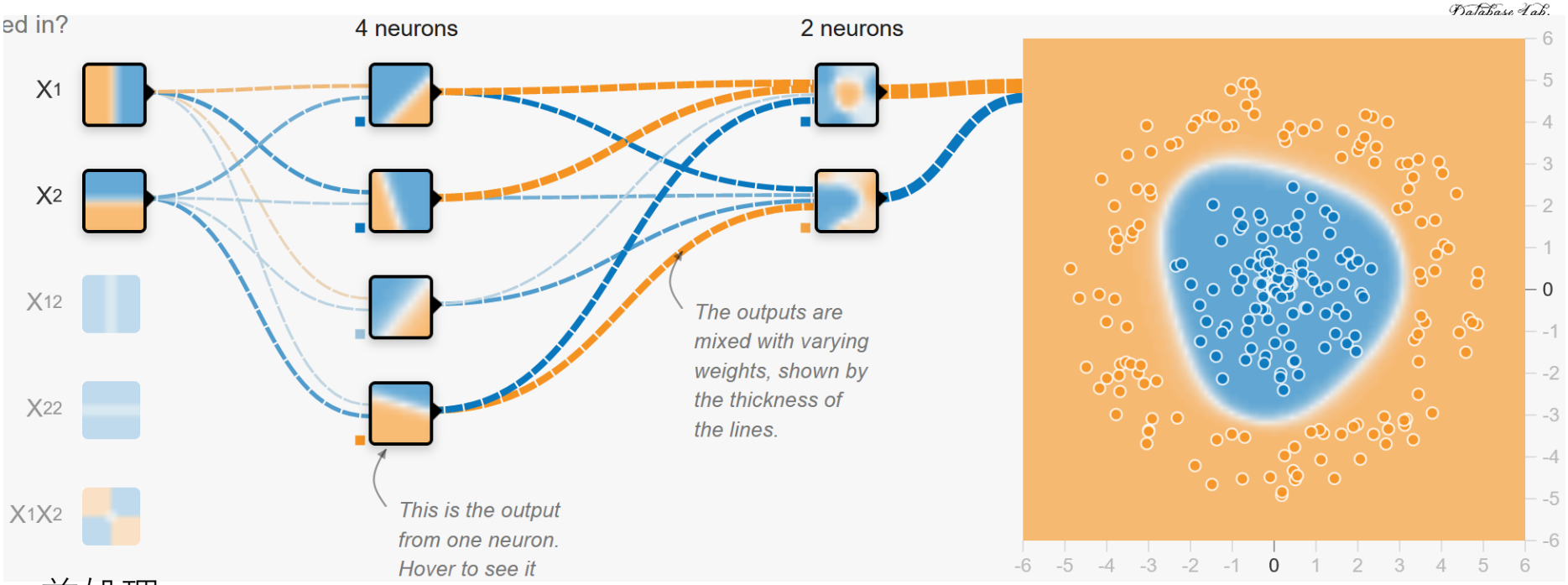
0.2　× -0.5　⇒ -0.1

入力　重み

合計 -0.47



-0.27 に活性化関数を適用



前処理
（データが青い部分にあれば活性化）

データが中央にあれば活性化

→結合→ 1層目 →結合→ 2層目

ニューラルネットワーク

URL: https://playground.tensorflow.org/

ニューロンの活性化と非活性化

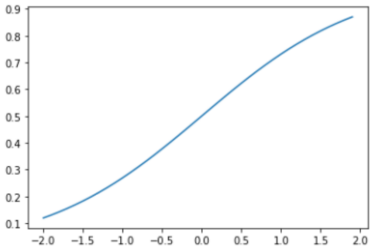
ニューロンの出力が，プラスの大きな値
→　**活性化**している

ニューロンの出力が，０に近い値
→　**活性化**していない

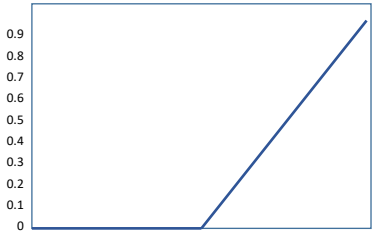
5-2. 活性化関数とその役割

活性化関数

- **ニューロン**の**活性度を決定**する関数
- **入力の重みづけ**，**合計とバイアス**の**結果**に，**活性化関数を適用**し，ニューロンの活性度を決定
- **ReLU**や**シグモイド**など，さまざまな種類



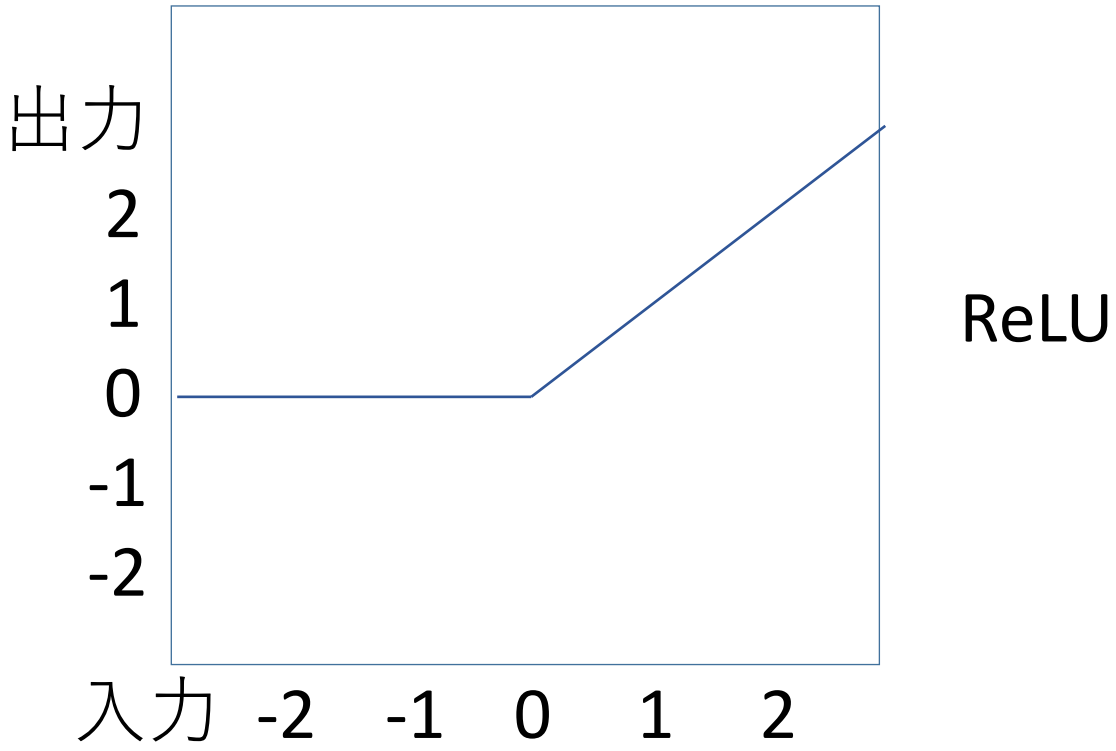
シグモイド



ReLU

ReLU

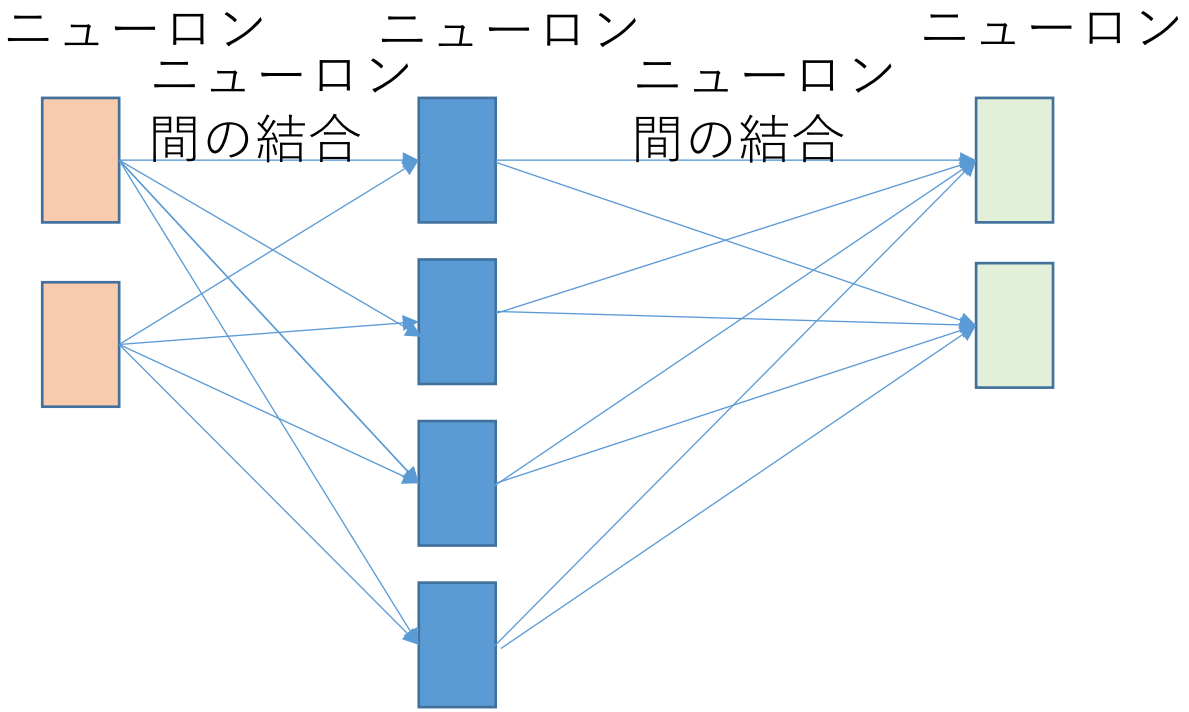
- **2011年に発表**された**活性化関数**
- 入力が**0以下の場合は0を返し，0より大きい場合はその値をそのまま返す**特性を持つ
- シンプルで，性能が良く，実績が豊富で，扱いやすい



5-3. フォワードプロパゲーション

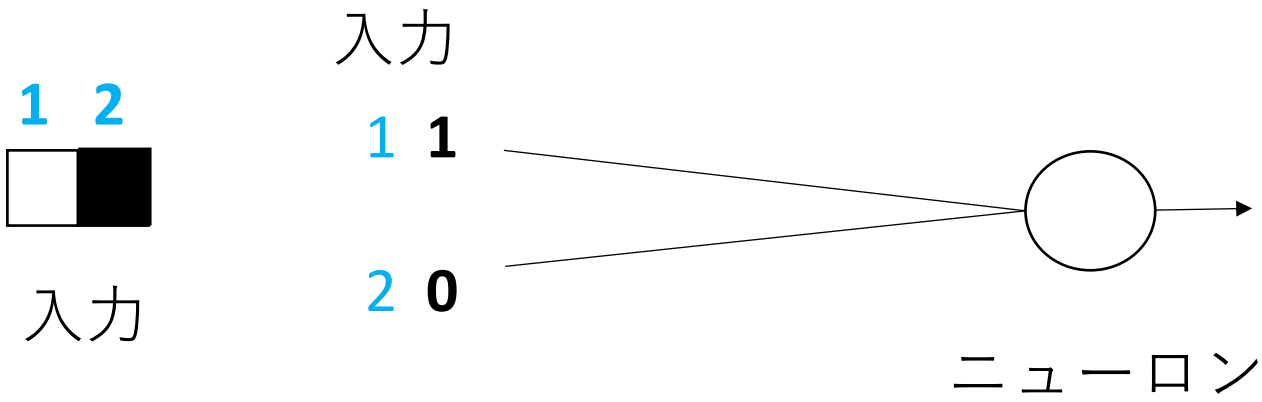
ニューラルネットワークの仕組み

入力の重みづけ，合計とバイアス，活性化関数の適用を行う**ニューロン**が**ネットワーク**を形成



①入力

数値，複数可　　例：１ ０



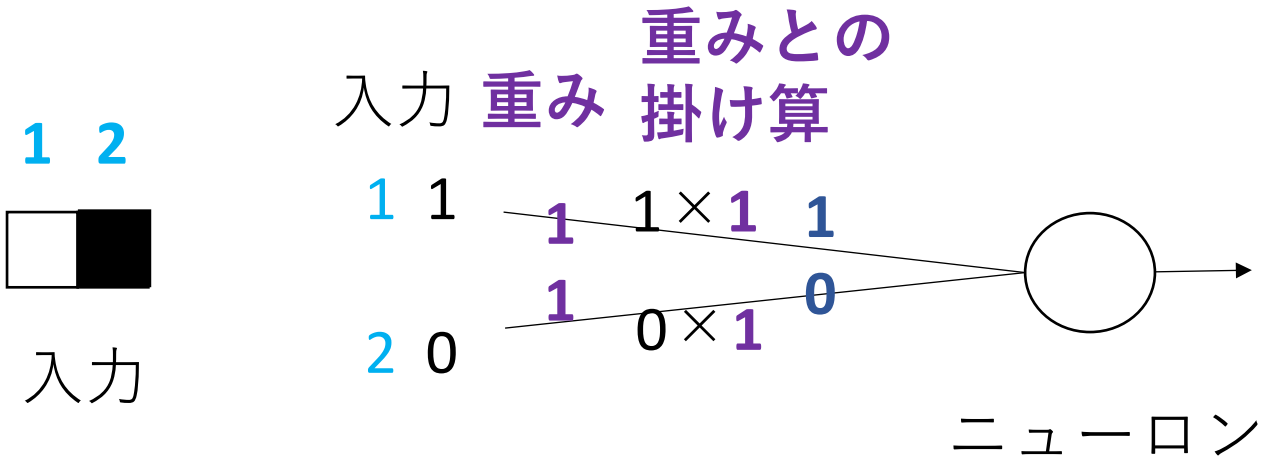
白を１
黒を０とする

②入力の重みづけ

- 各入力値に対して，それぞれの対応する**重みを掛ける**

  例：**1**×**1**　**0**×**1**

- **重み**は，**入力データの重要度**を決定（**学習過程**で**自動的に調整**され，望ましい出力が得られるように最適化される）



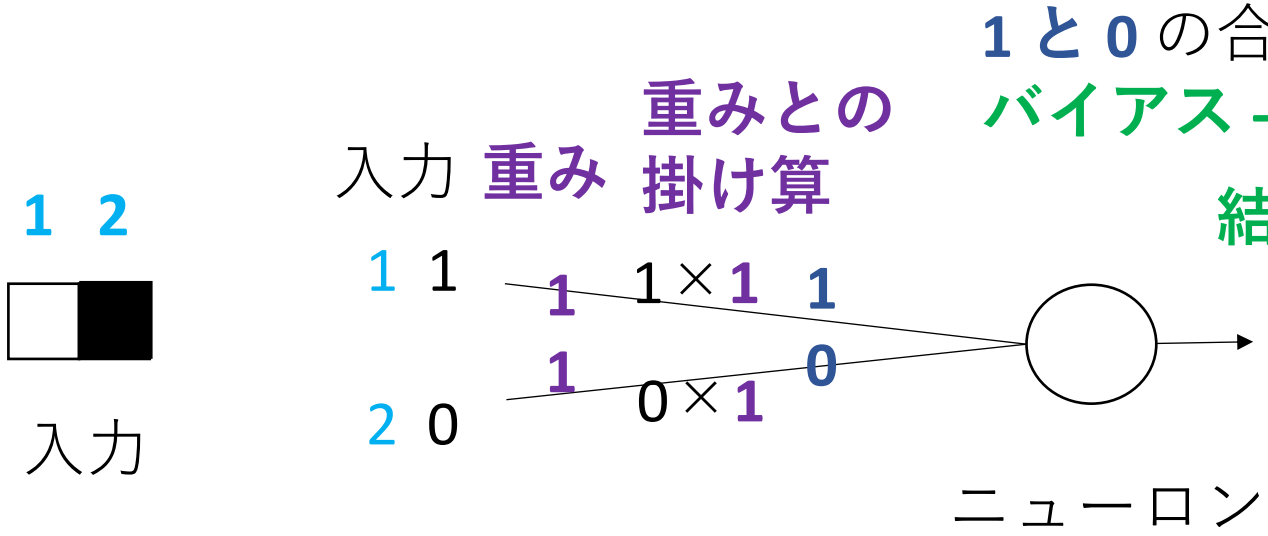
白を１
黒を０とする

③バイアス

- **合計**に**バイアス**（数値）を加える.

  例 **1 0 の合計は 1. これに -0.5 を加える**

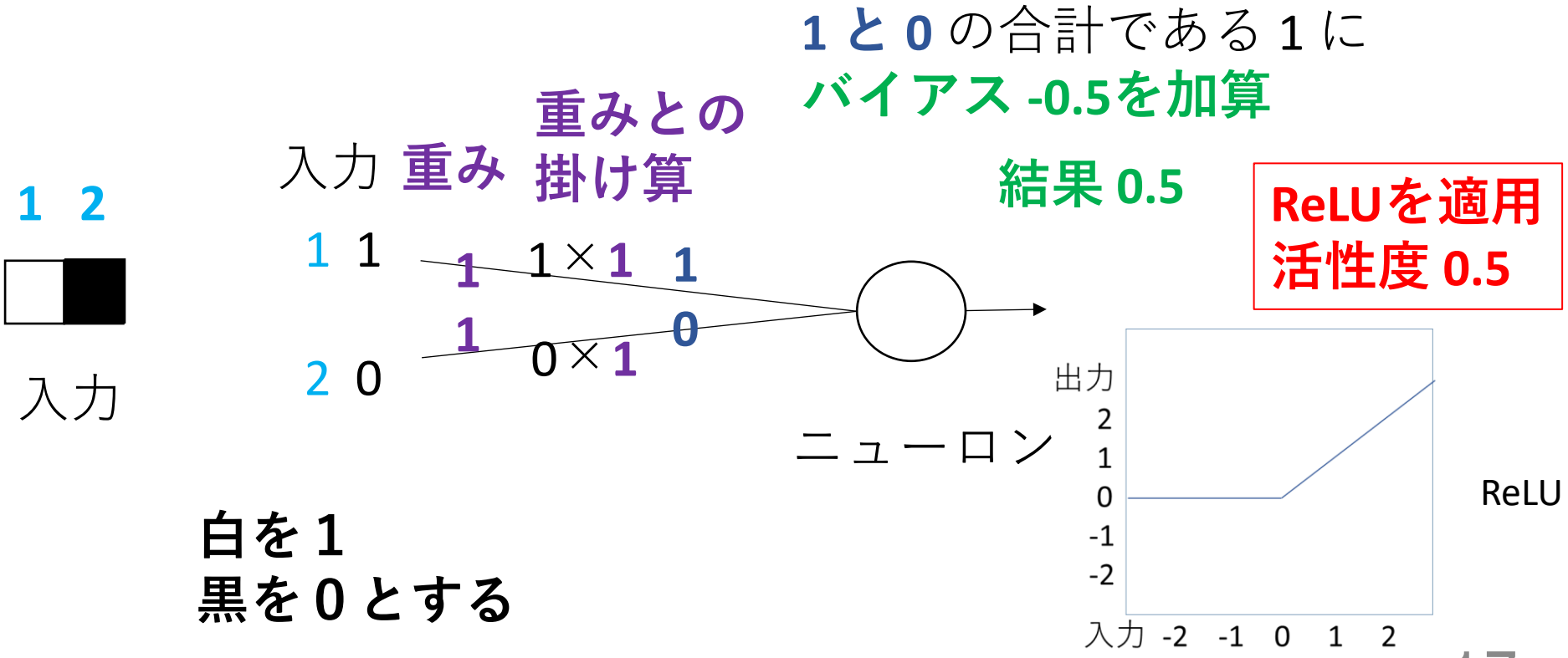
- **バイアス**は，**ニューロンの活性化しやすさ**を決定（**学習過程**で**自動的に調整**され，望ましい出力が得られるように最適化される）



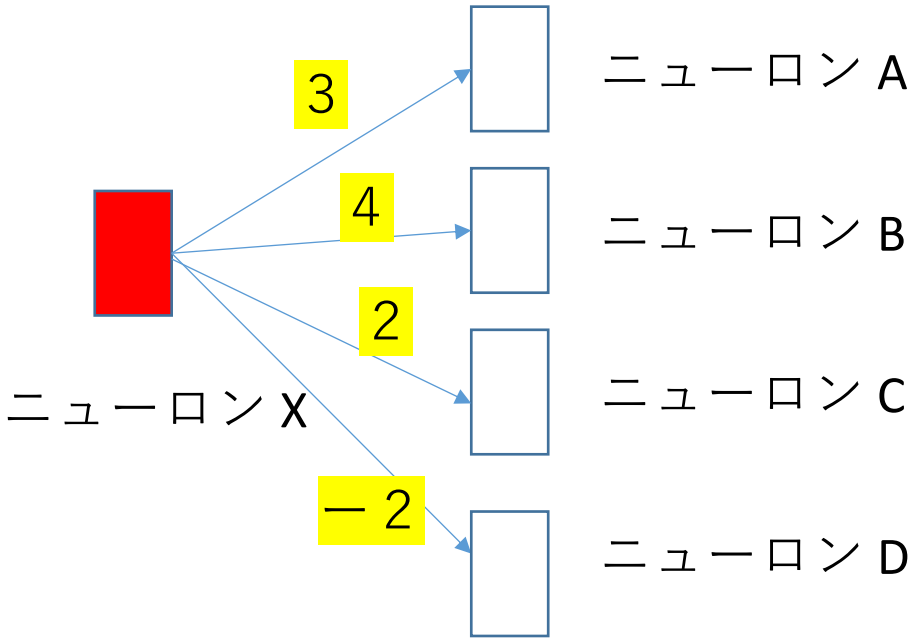
白を 1
黒を 0 とする

④活性化関数の適用

- 結果に，**活性化関数**を適用．**ニューロンの活性度**を得る
  （→次のニューロンの入力になる）
- **値が大きいほどニューロンが活性化している**ことを示す



活性度（数値）は，次のニューロンの入力になる



次の二ューロンの**入力**
＝　**前**の二ューロンの**出力**
×　**結合の重み**

結合の重みは，
結合ごとに**違う値**

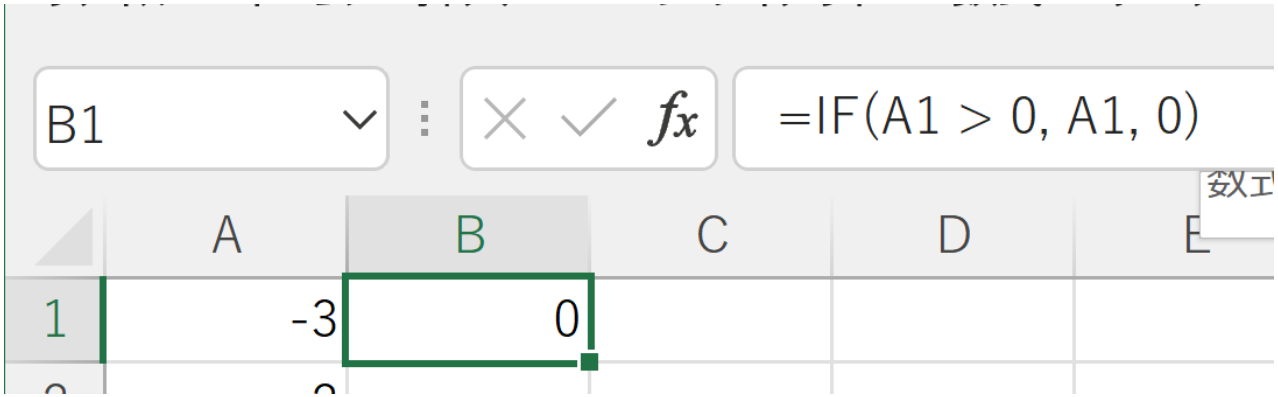
活性化関数 ReLU

0 **以下**の場合　→　結果は 0

0 **以上**の場合　→　結果は，そのままの値

Excel では，次のような式になる

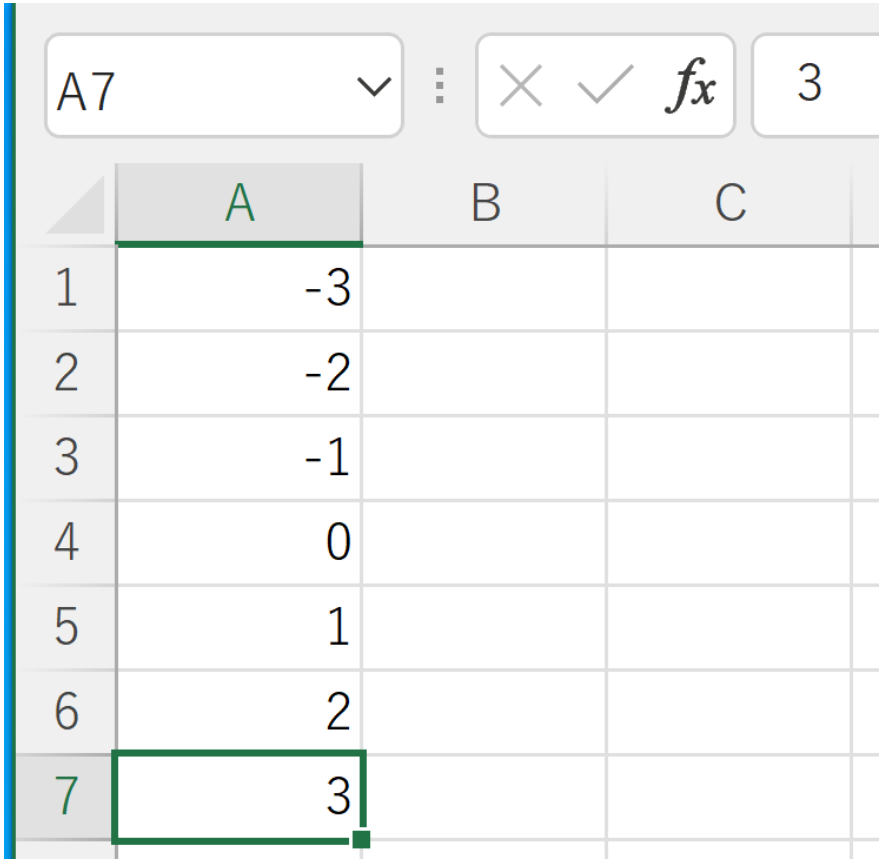
式「=IF(A1 > 0, A1, 0)」



演習

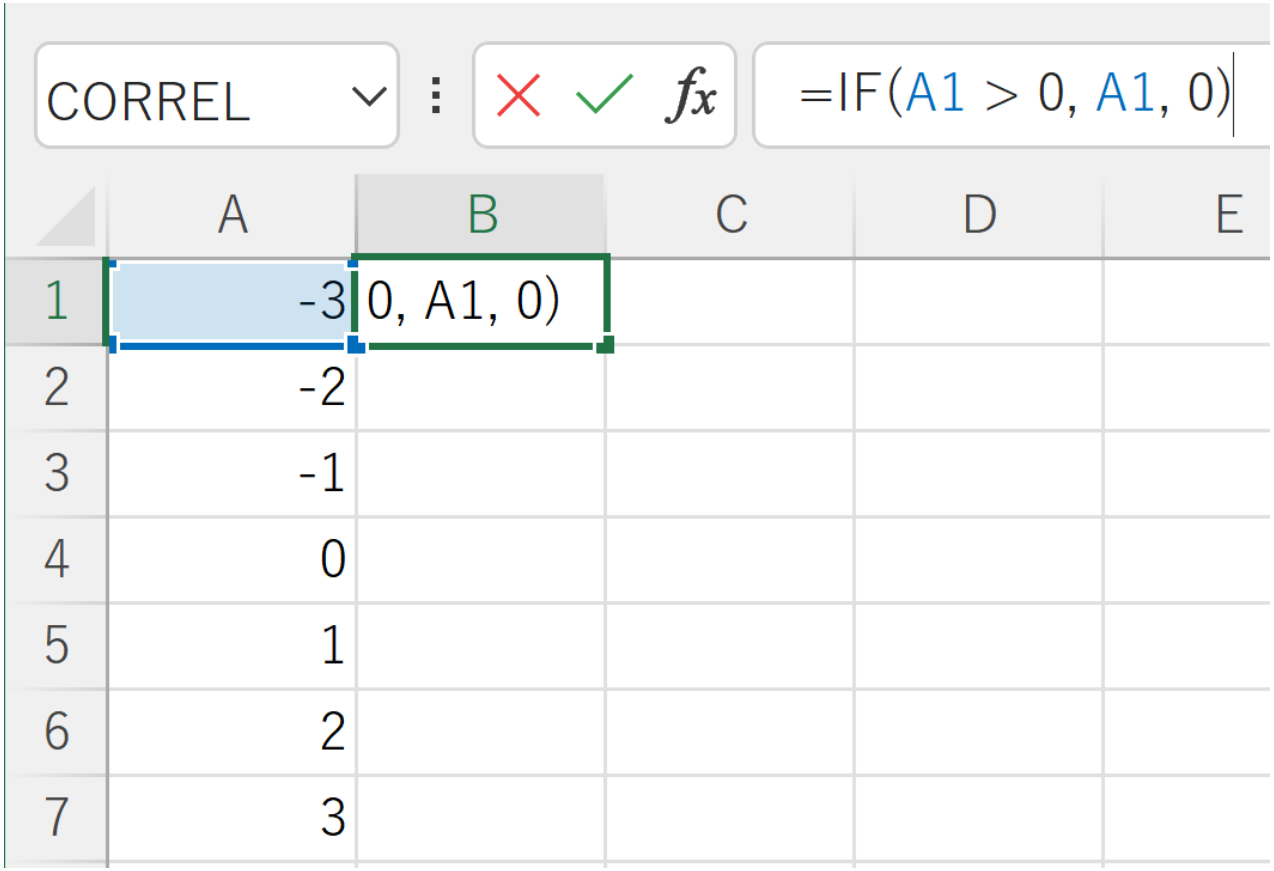
Excel を起動．起動したら「空白のブック」を選ぶ

① セル **A1 から A7 に，値 -3, -2, -1, 0, 1, 2, 3（半角で）**

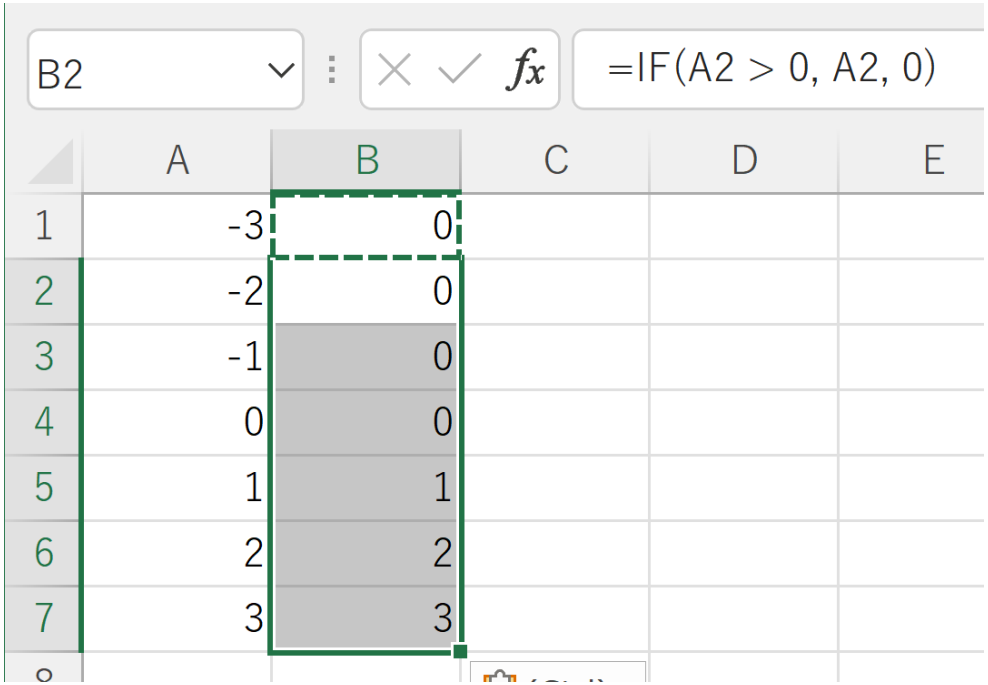


② ReLU の式

セル **B1** に**式**「 **=IF(A1 > 0, A1, 0)** 」



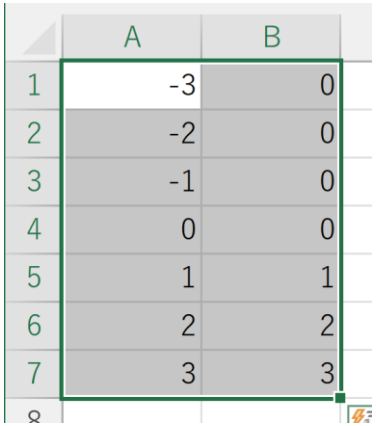
③ ReLU の式
セル **B1** の式を,
B2 から B7 に「**コピー&貼り付け**」
右クリックメニューが便利



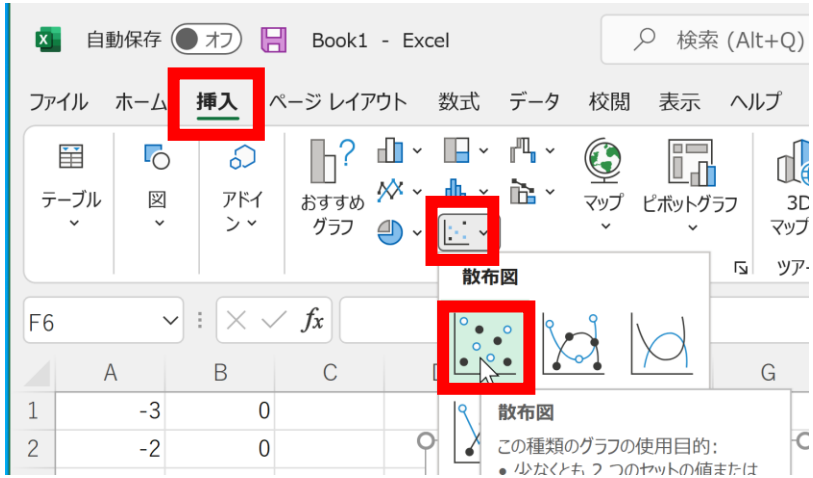
④ 散布図の作成

	A	B
1	-3	0
2	-2	0
3	-1	0
4	0	0
5	1	1
6	2	2
7	3	3

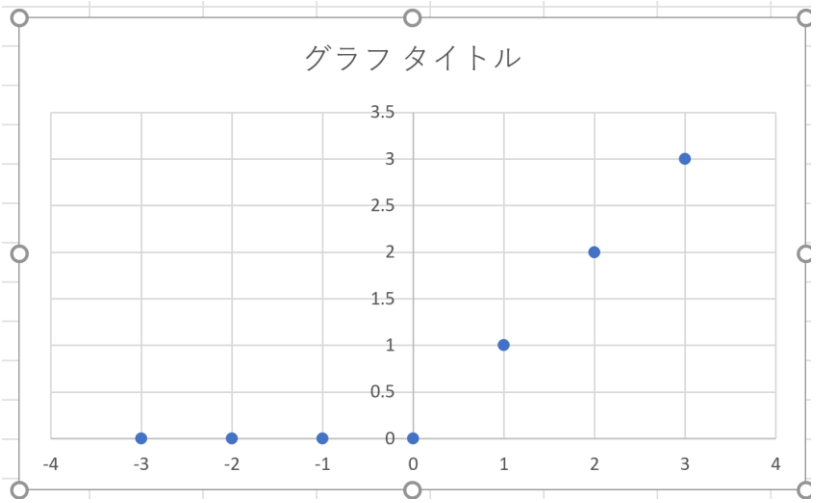
元データ



① グラフ化したい部分を**範囲選択**



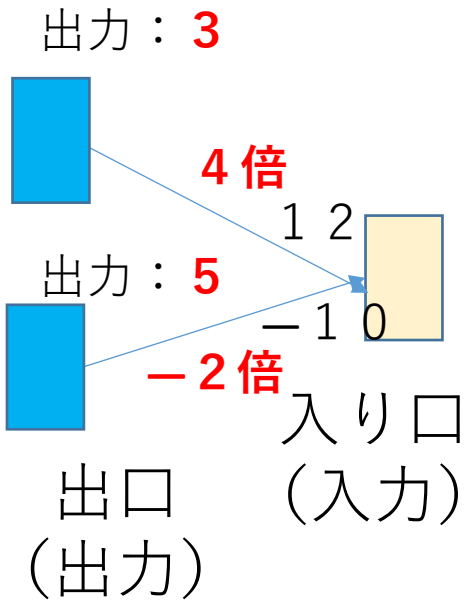
② リボンで「**挿入**」→散布図の選択



散布図が得られる

⑤ 前のニューロンの出力と，結合の重み

セル A8, A9, B8, B9 に次の値を入れる



8	3	4
9	5	-2

⑥ （入力）×（結合の重み）の**合計**

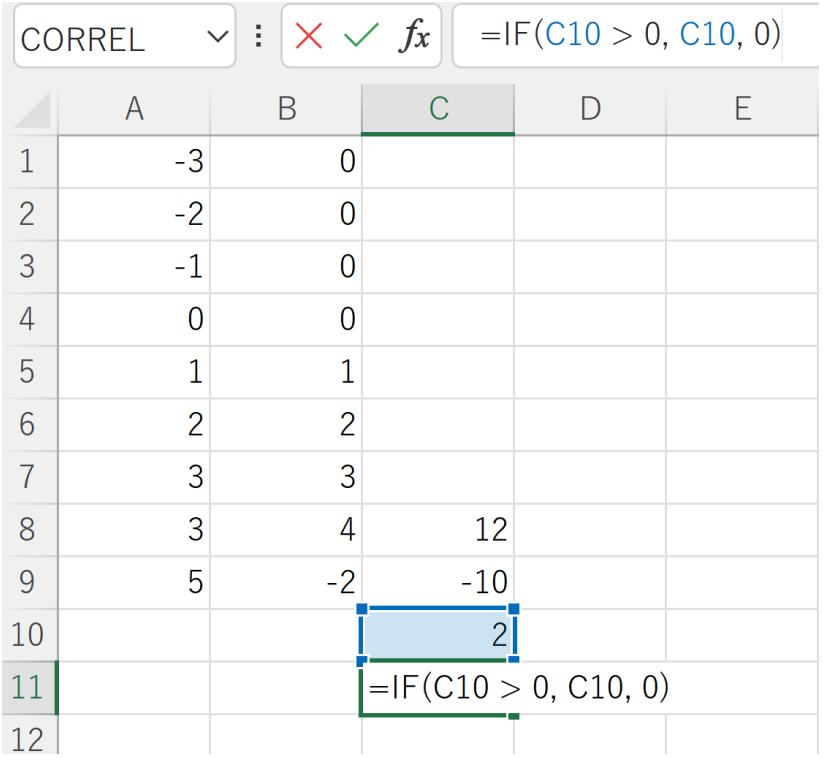
- **セル C8 に，式「=A8*B8」**
- **セル C9 に，式「=A9*B9」**
- **セル C10 に，式「=SUM(C8:C9)」**

8	3	4	12
9	5	-2	-10
10			2

ここではバイアスは 0 と考える

⑦求まった合計に**活性化関数 ReLU** を適用
→　ニューロンの出力になる

・セル C11 に，式「 =IF(C10 > 0, C10, 0) 」



演習　ReLU（Pythonで実現）

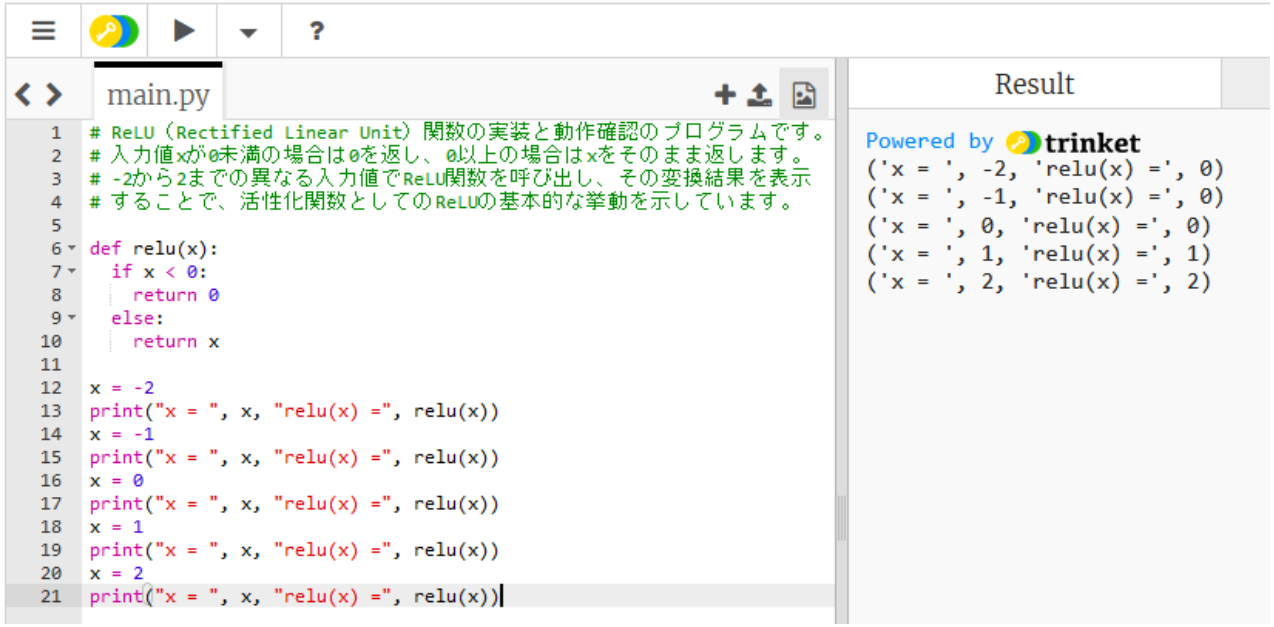
演習の狙い

- **ReLU関数の特徴**（負の入力を0に変換、正の入力はそのまま出力）を**直感的に理解**
- **異なる入力値（負数、0、正数）**で実験して**結果を確認**する体験
- if と else を使うプログラミング（条件分岐）

① trinket の次のページを開く

https://trinket.io/python/61c6503fcada

② 実行結果が，次のように表示されることを確認



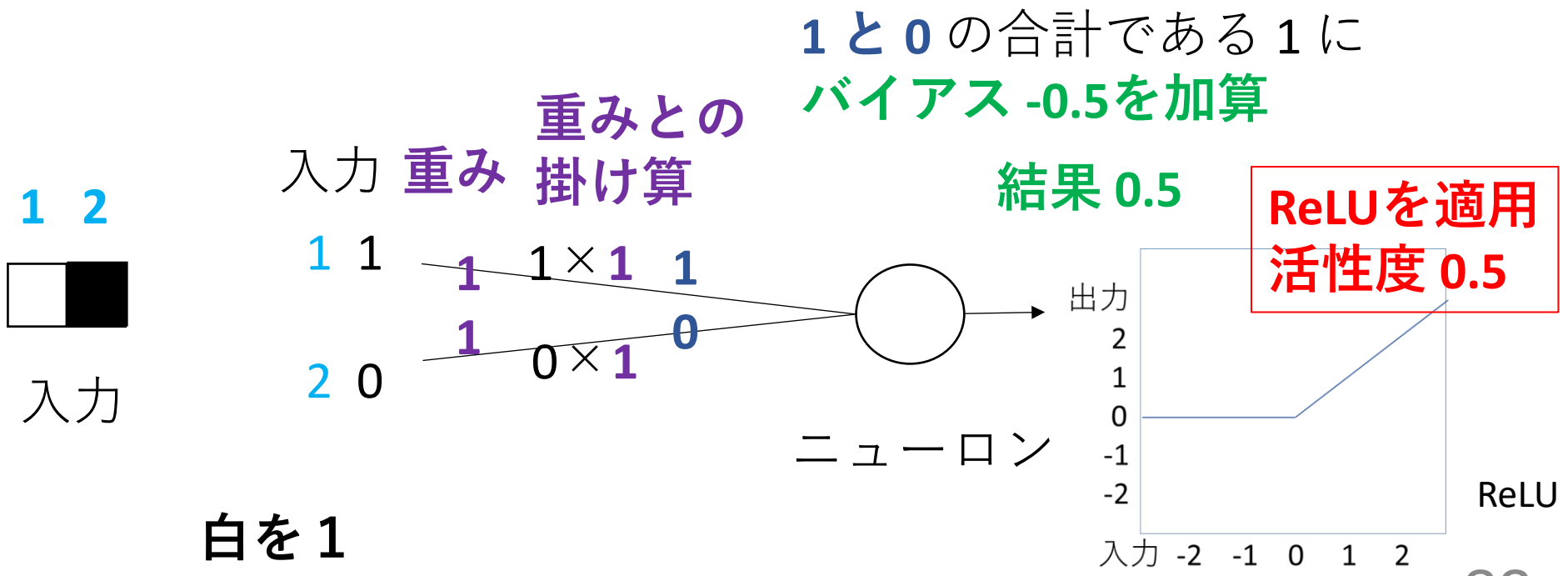
- 実行が開始しないときは、「**実行ボタン**」で**実行**
- ソースコードを**書き替えて再度実行**することも可能

演習　入力の重みづけ，合計とバイアス，活性化関数の適用

演習の狙い

- **重み，バイアス，活性化関数（ReLU）**という3つの基本要素の役割と連携を具体的に理解できます.
- これは複雑なニューラルネットワークを理解するための重要な第一歩となります.

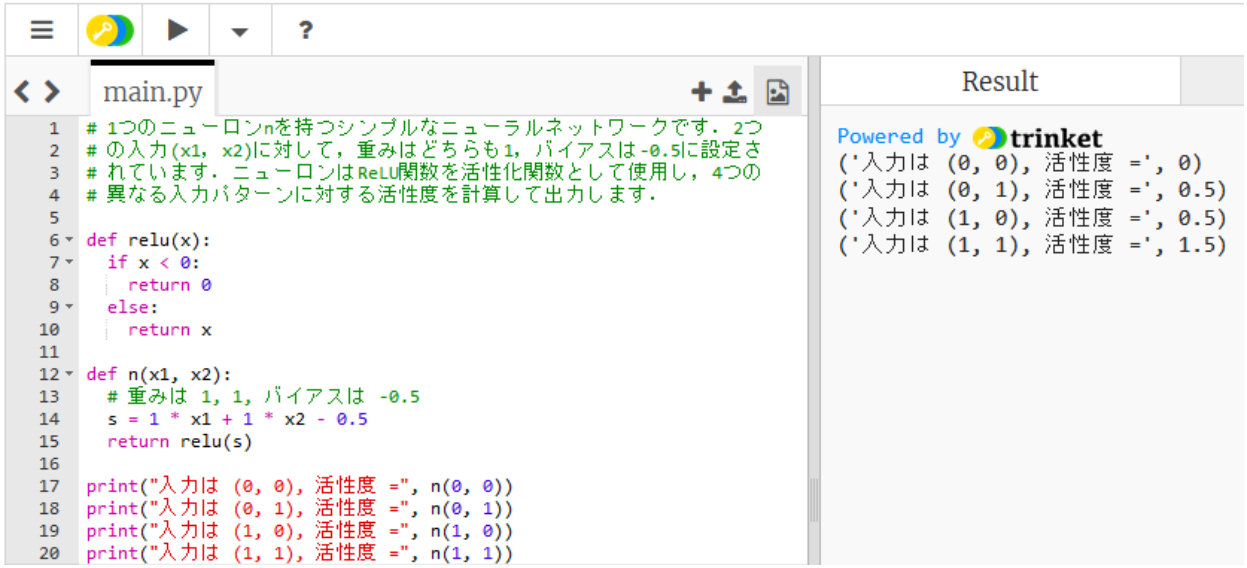
【入力が１０のとき】



① trinket の次のページを開く

https://trinket.io/python/1c339b660ee1

② 実行結果が，次のように表示されることを確認



- 実行が開始しないときは、「**実行ボタン**」で**実行**
- ソースコードを**書き替えて再度実行**することも可能

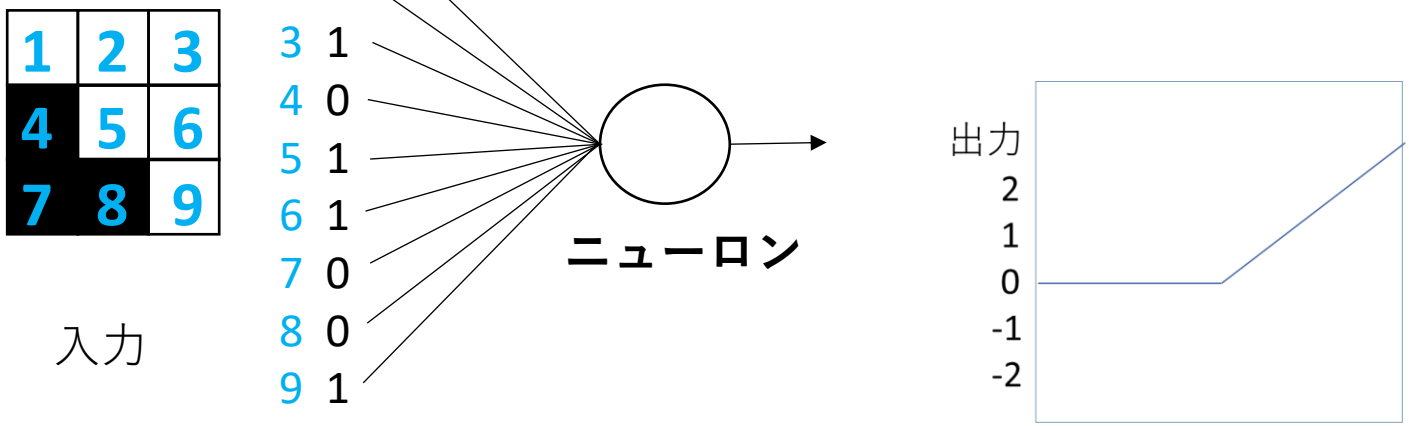
ニューラルネットワークの基本まとめ

- 各**ニューロン**は、前の層からの**入力**を受け取る
- **入力**には**重み**が関連付けられており、**各データがニューロンの活性化に与える影響**を調整する
- **ニューロン**は**バイアス**を持ち、**入力**に**重み**をかけた**総和**に**バイアス**を**加算**する
- **活性化関数**による**処理**を行い、その結果を**次の層のニューロン**に伝える
- ReLUなどの**活性化関数**がある。
- **ニューロン**は**特定のパターン**の**入力**に対して**活性化**し、その**活性度**は**入力**、**重み**、**バイアス**、**活性化関数**によって決まる

ニューラルネットワークによるパターン認識の例

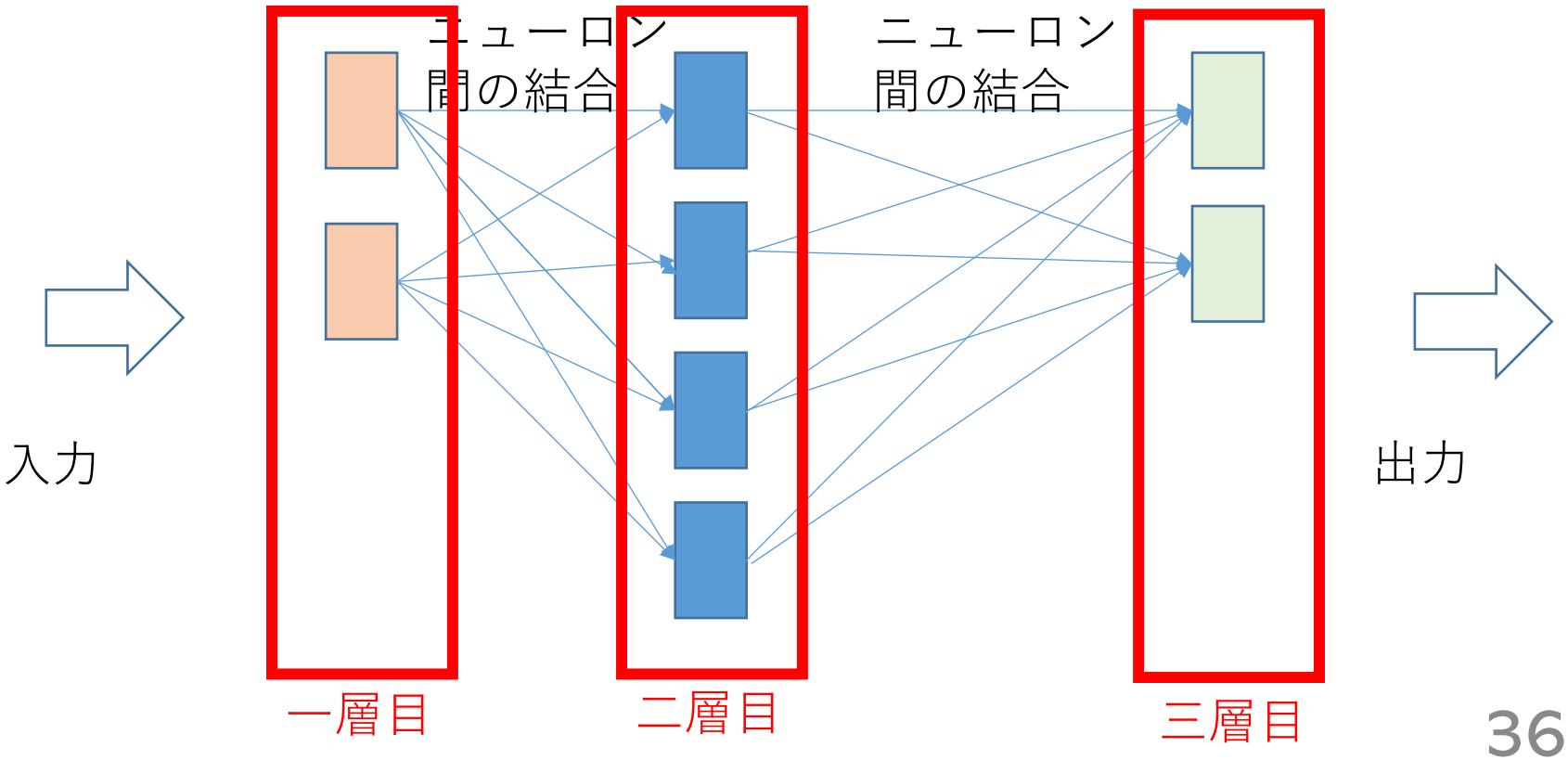
1. **入力の重みづけ**:
   1 × 1, 1 × 1, 1 × 1, 0 × 0, 1 × 1, 1 × 1, 0 × 0, 0 × 0, 1 × 1
2. **合計とバイアス**:
   合計 **6** にバイアス **-5** を足す．結果 **1**
3. **活性化関数の適用**

   結果 **1** に **ReLU** を適用した場合は，活性度 **1**



ニューラルネットワークの層構造

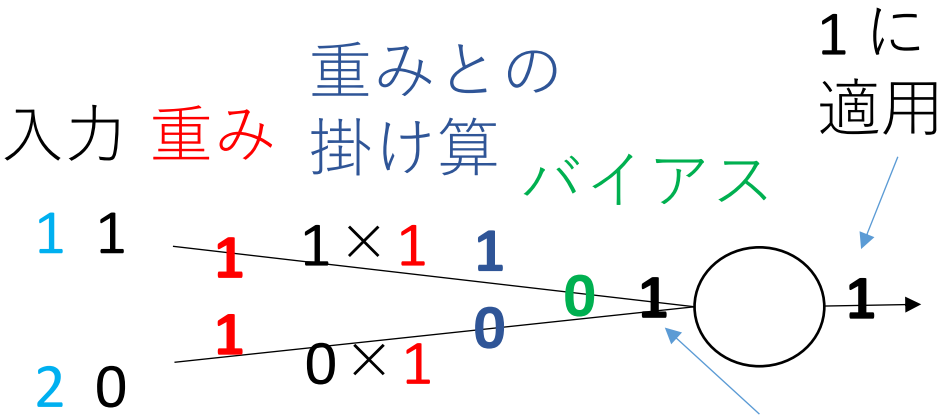
ニューラルネットワークの基本構造であり，**複数の層**から成る．**データは入力から出力への一方向**に流れ，**各層**は**同じ種類のニューロンで構成**される．



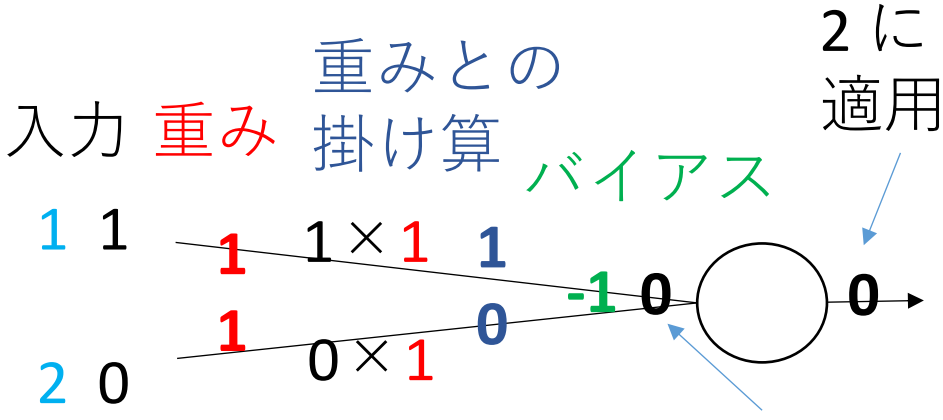
フォワードプロパゲーション

- **フォワードプロパゲーション**は，**ニューラルネットワーク**において，**データが入力層から出力層へ順方向に伝播**する仕組み.
- **各ニューロン**は**前の層から受け取ったデータを処理**し，その結果を**次の層に伝達**する.

バラメータ（バイアス）の変化による出力の変化

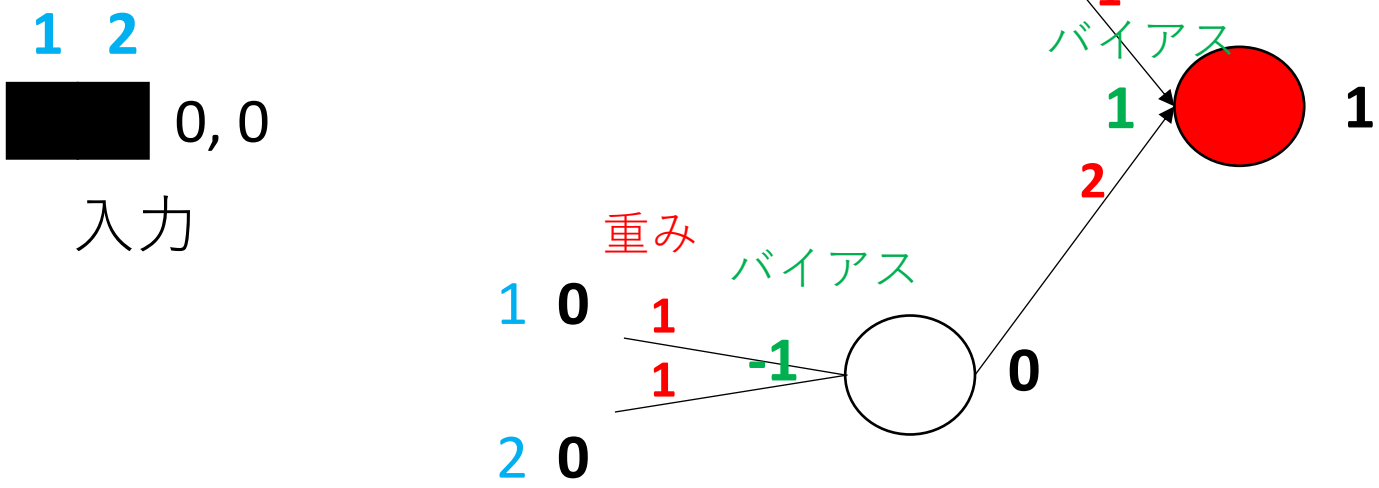


重み 1, 1　バイアス 0

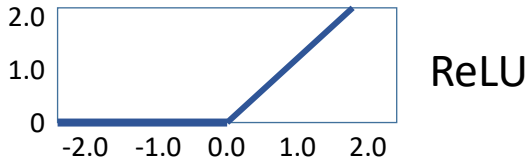


重み 1, 1　**バイアス -1**

大きなプラスの値を
出力している
ニューロン＝活性化

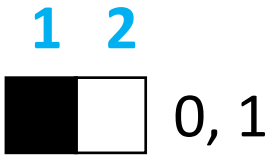


3つのニューロンの活性化関数はすべて ReLU

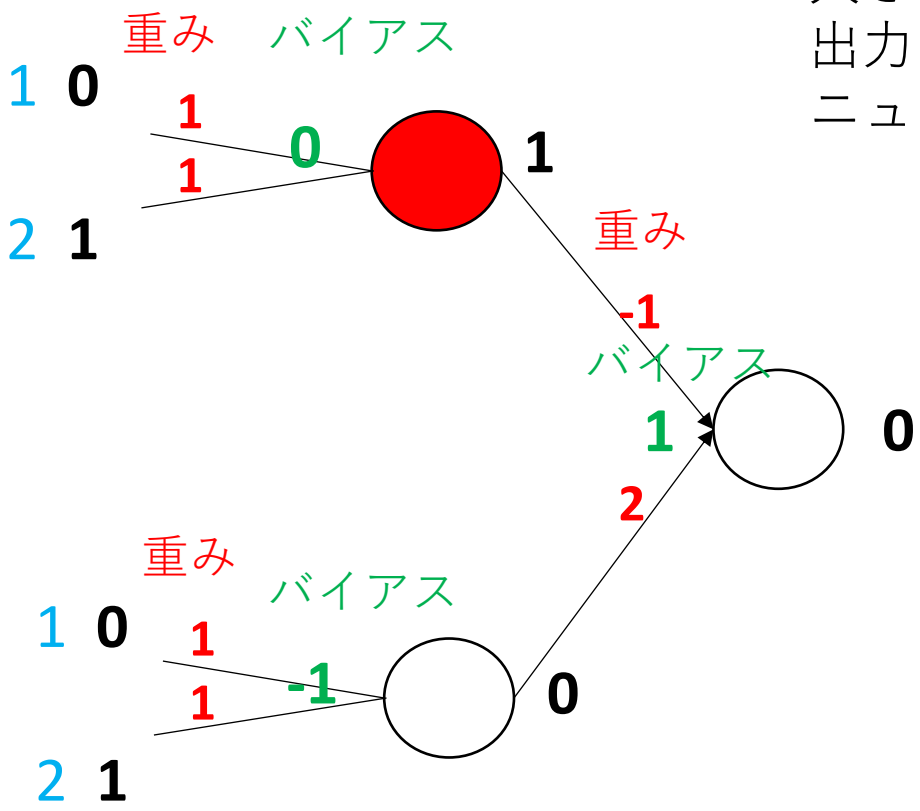


ReLU

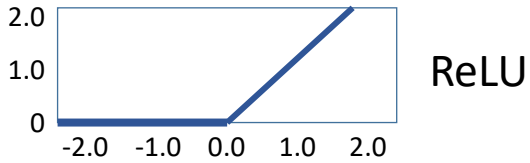
大きなプラスの値を
出力している
ニューロン＝活性化



入力

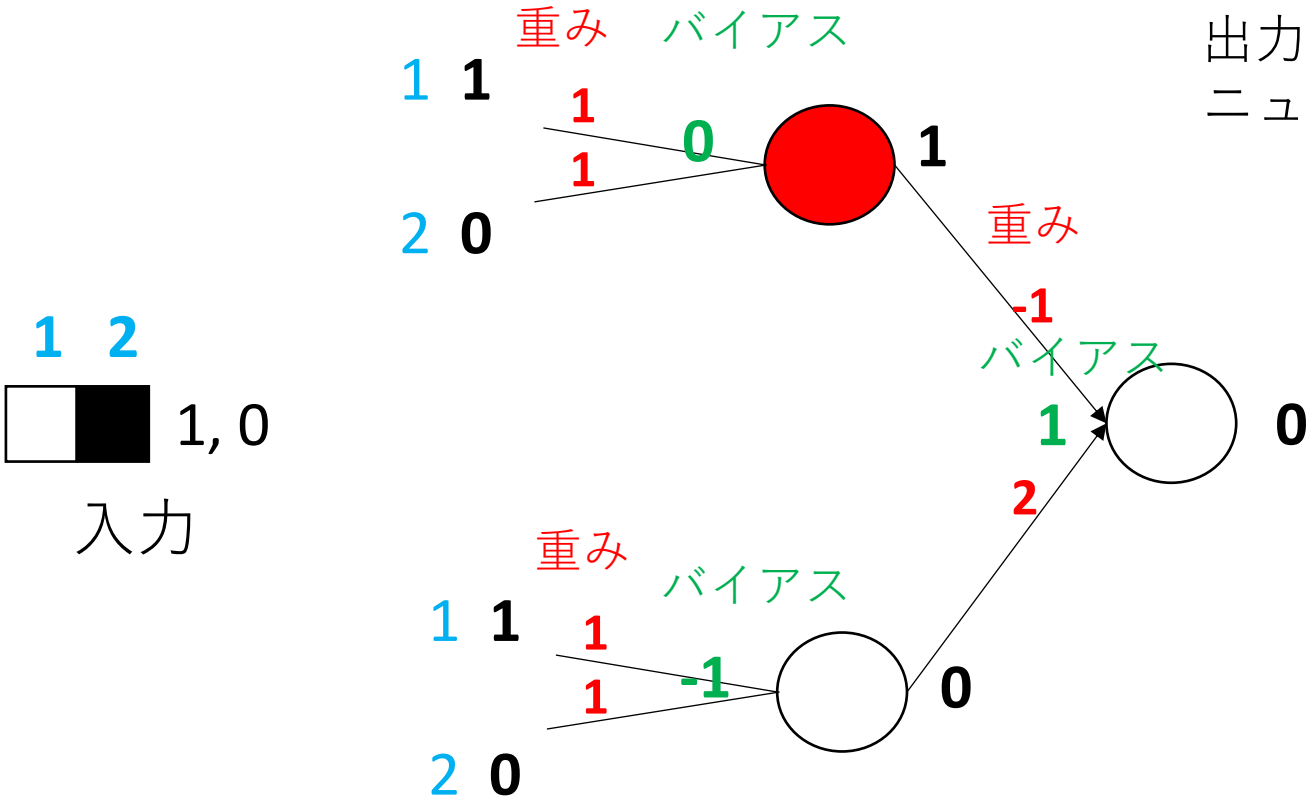


3つのニューロンの活性化関数はすべて ReLU

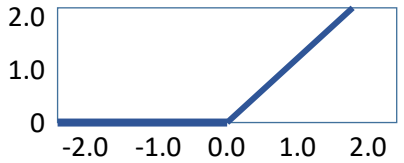


ReLU

大きなプラスの値を
出力している
ニューロン＝活性化

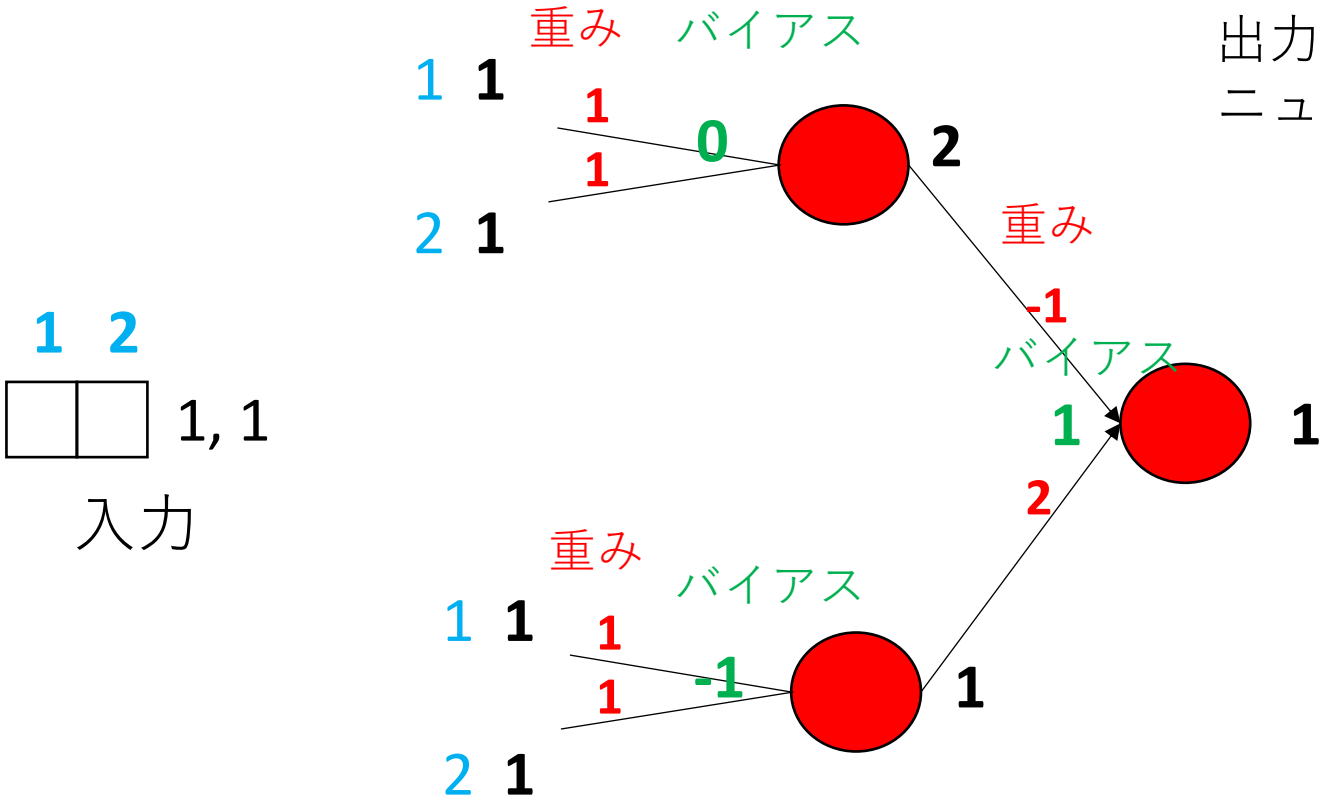


3つのニューロンの活性化関数はすべて ReLU

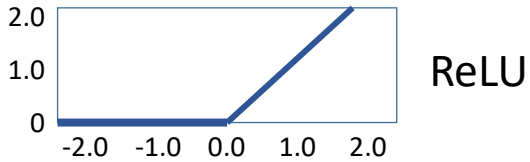


ReLU

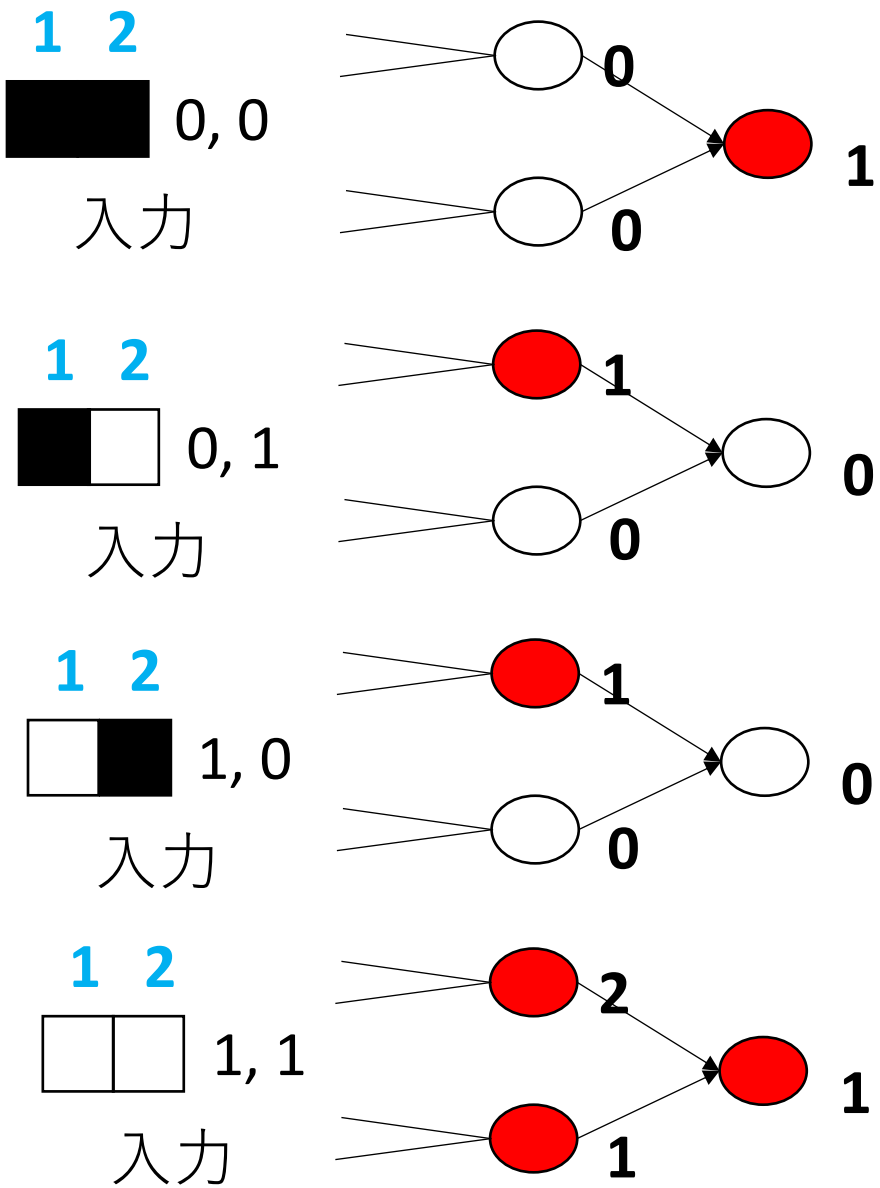
大きなプラスの値を
出力している
ニューロン＝活性化



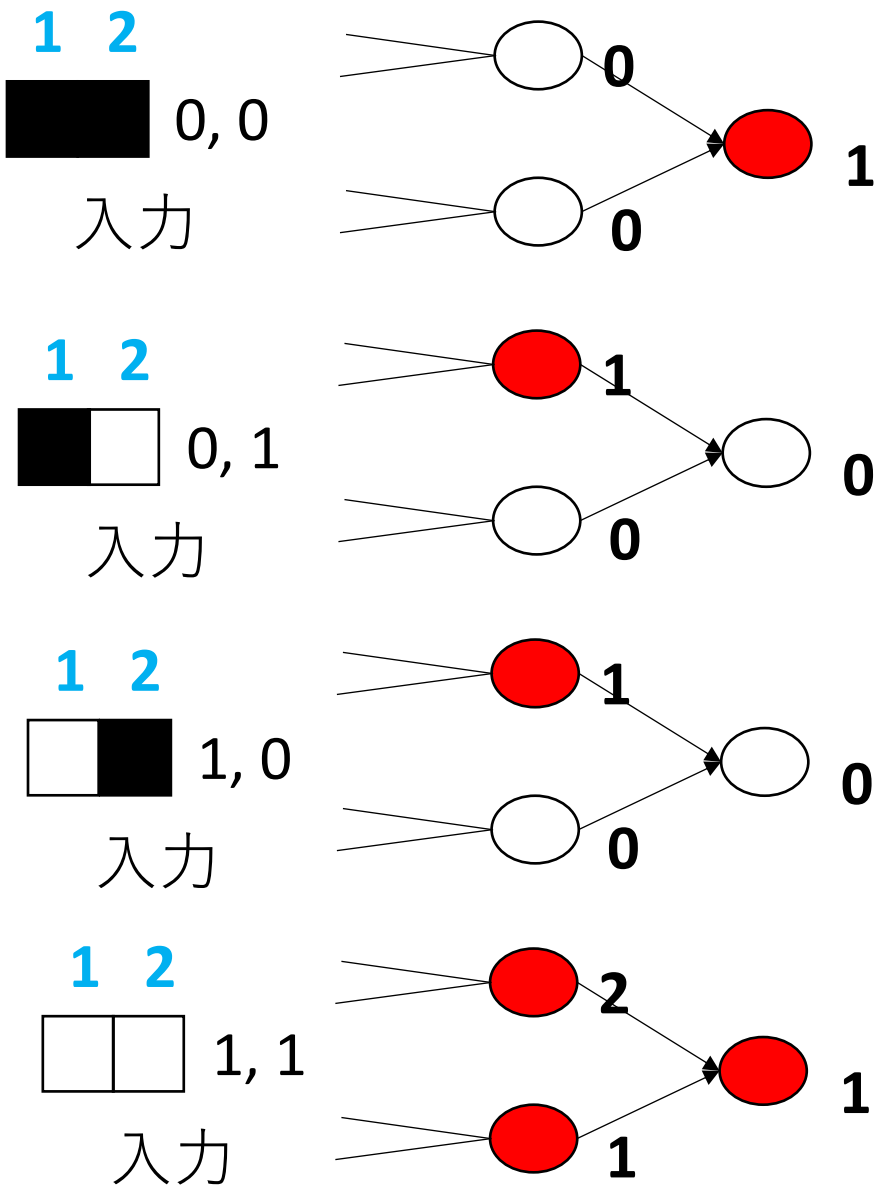
3つのニューロンの活性化関数はすべて ReLU



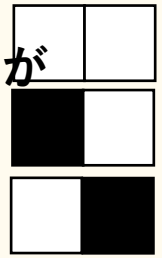
入力の変化による活性度の変化

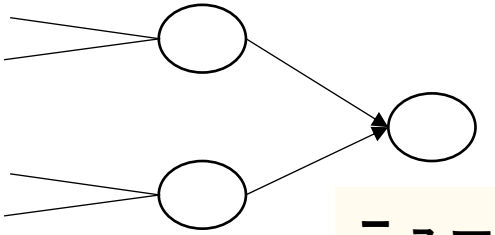


ニューロンは特定のパターンに応じて活性化する

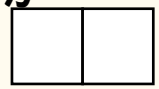


ニューロンが識別するパターン

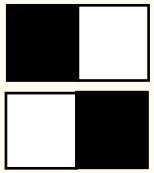




ニューロンが識別するパターン



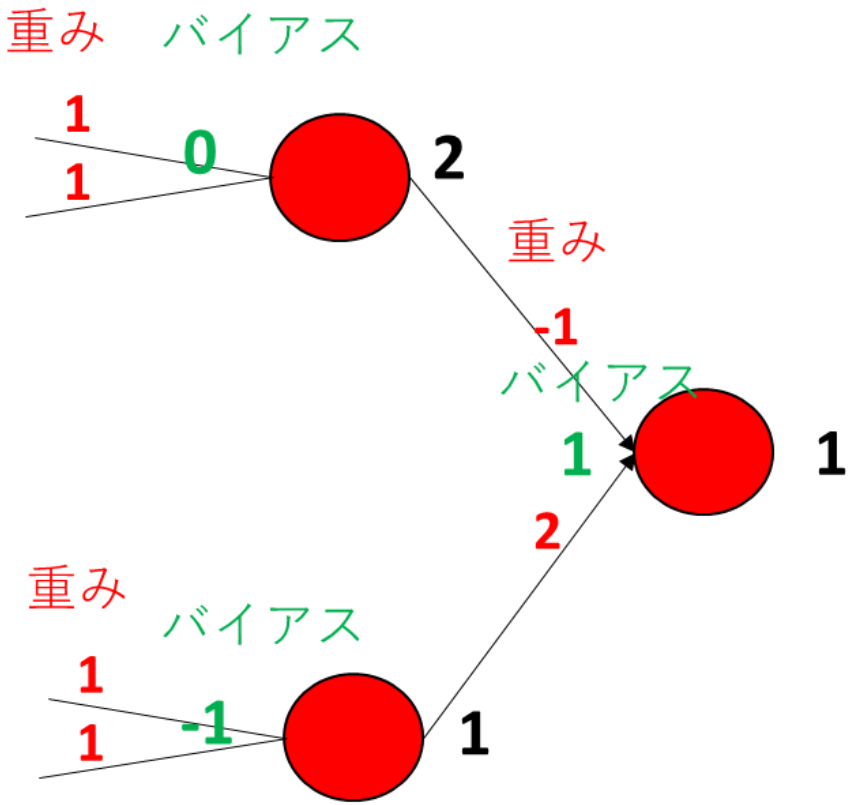
ニューロンが識別するパターン



演習 ニューラルネットワーク

演習の狙い

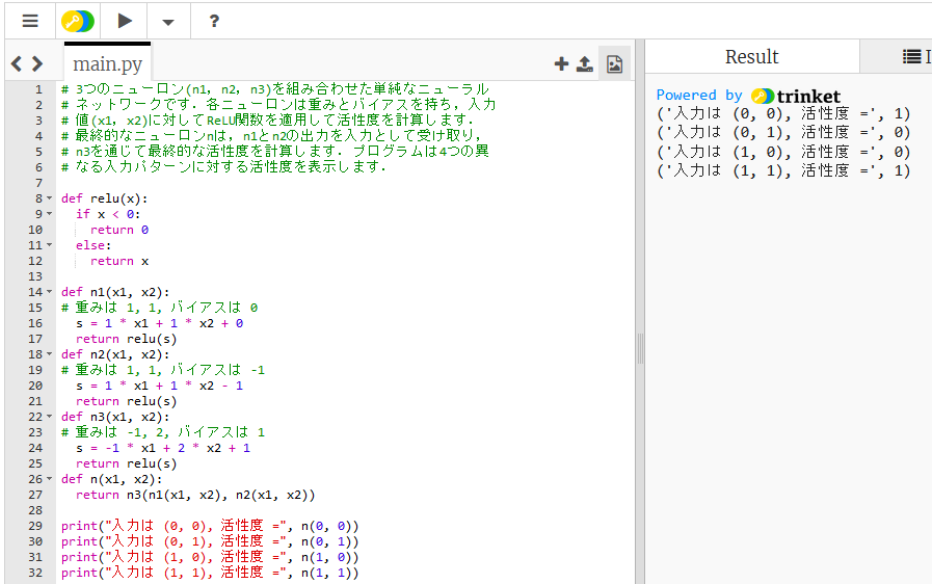
- 複数のニューロンが層を形成
- １層の場合より複雑な判断が可能になる
- ニューラルネットワークによる高度な情報処理の基礎を体験



① trinket の次のページを開く

https://trinket.io/python/3cbc3f3ed057

② 実行結果が，次のように表示されることを確認




```
# 3つのニューロン(n1, n2, n3)を組み合わせた単純なニューラル
# ネットワークです．各ニューロンは重みとバイアスを持ち，入力
# 値(x1, x2)に対してReLU関数を適用して活性度を計算します．
# 最終的なニューロンn3は，n1とn2の出力を入力として受け取り，
# n3を通じて最終的な活性度を計算します．プログラムは4つの異
# なる入力パターンに対する活性度を表示します．

def relu(x):
  if x < 0:
    return 0
  else:
    return x

def n1(x1, x2):
# 重みは 1, 1, バイアスは 0
  s = 1 * x1 + 1 * x2 + 0
  return relu(s)
def n2(x1, x2):
# 重みは 1, 1, バイアスは -1
  s = 1 * x1 + 1 * x2 - 1
  return relu(s)
def n3(x1, x2):
# 重みは -1, 2, バイアスは 1
  s = -1 * x1 + 2 * x2 + 1
  return relu(s)
def n(x1, x2):
  return n3(n1(x1, x2), n2(x1, x2))

print("入力は (0, 0), 活性度 =", n(0, 0))
print("入力は (0, 1), 活性度 =", n(0, 1))
print("入力は (1, 0), 活性度 =", n(1, 0))
print("入力は (1, 1), 活性度 =", n(1, 1))
```

- 実行が開始しないときは、「**実行ボタン**」で**実行**
- ソースコードを**書き替えて再度実行**することも可能

ニューラルネットワークのまとめ

- **ニューラルネットワーク**は，単純な原理で動く
- **ニューロン**は**複数の入力**を受け取る．中では，**活性化関数**を用いた**値の変換**を行う．
- **層構造**と**全結合**のニューラルネットワークが最もシンプルである．**入力から出力の方向へ一方向にデータが流れる**．**２つの層の間のニューロン**は**すべて結合**．
- **ニューラルネットワーク**は，**結合の重みとバイアスを調整**して，**望みの出力を得る**．

ニューラルネットワークの**学習**では，**データを利用**して，**望みの出力が得られるよう**に，**結合の重みとバイアスの調整**が行われる．画像認識や自然言語処理などに広く利用されている．

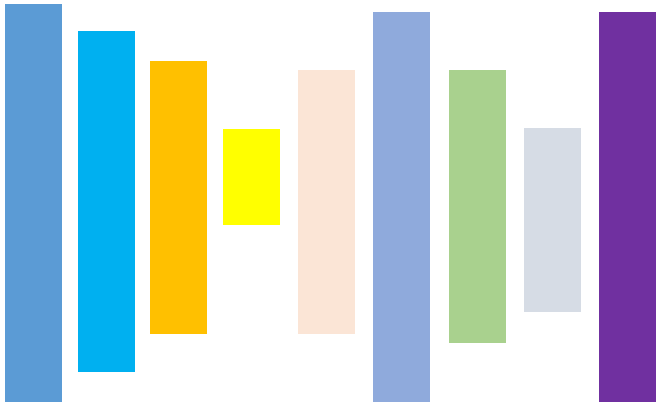
5-4. ディープラーニングの特徴と応用

ディープニューラルネットワーク

・**ディープニューラルネットワーク**は，

層が深い（**層の数が多い**）**ニューラルネットワーク**

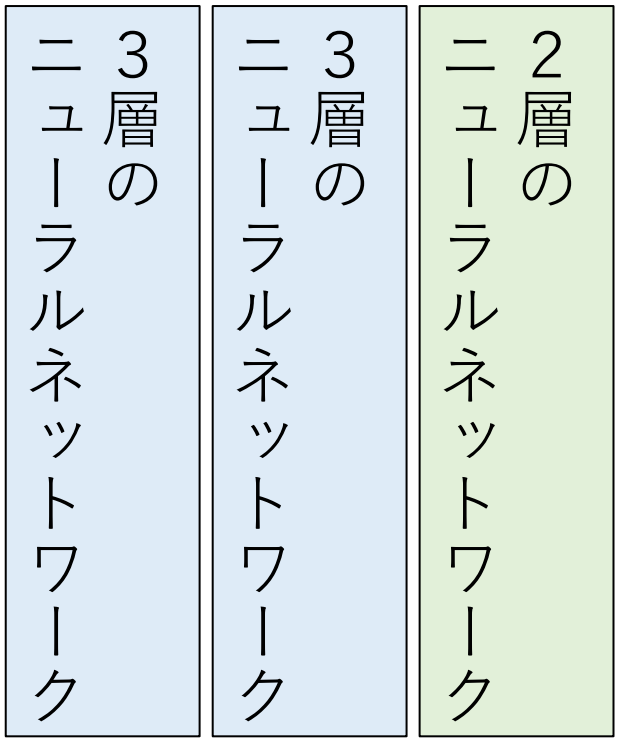
層の数が少ない（浅い）



層の数が多い（深い）

ディープニューラルネットワーク

- 層が浅い（**層の数が少ない**）**ニューラルネットワーク**を**組み合わせる**こともある



合計で８層

（さまざまな組み合わせがありえる）

ディープラーニングが広く利用されている理由

- **多様なデータに対応**

例：画像、テキスト、音声、動画など

- **応用の広さ**

例：画像認識、自然言語処理、音声認識など

- **パターン抽出能力**

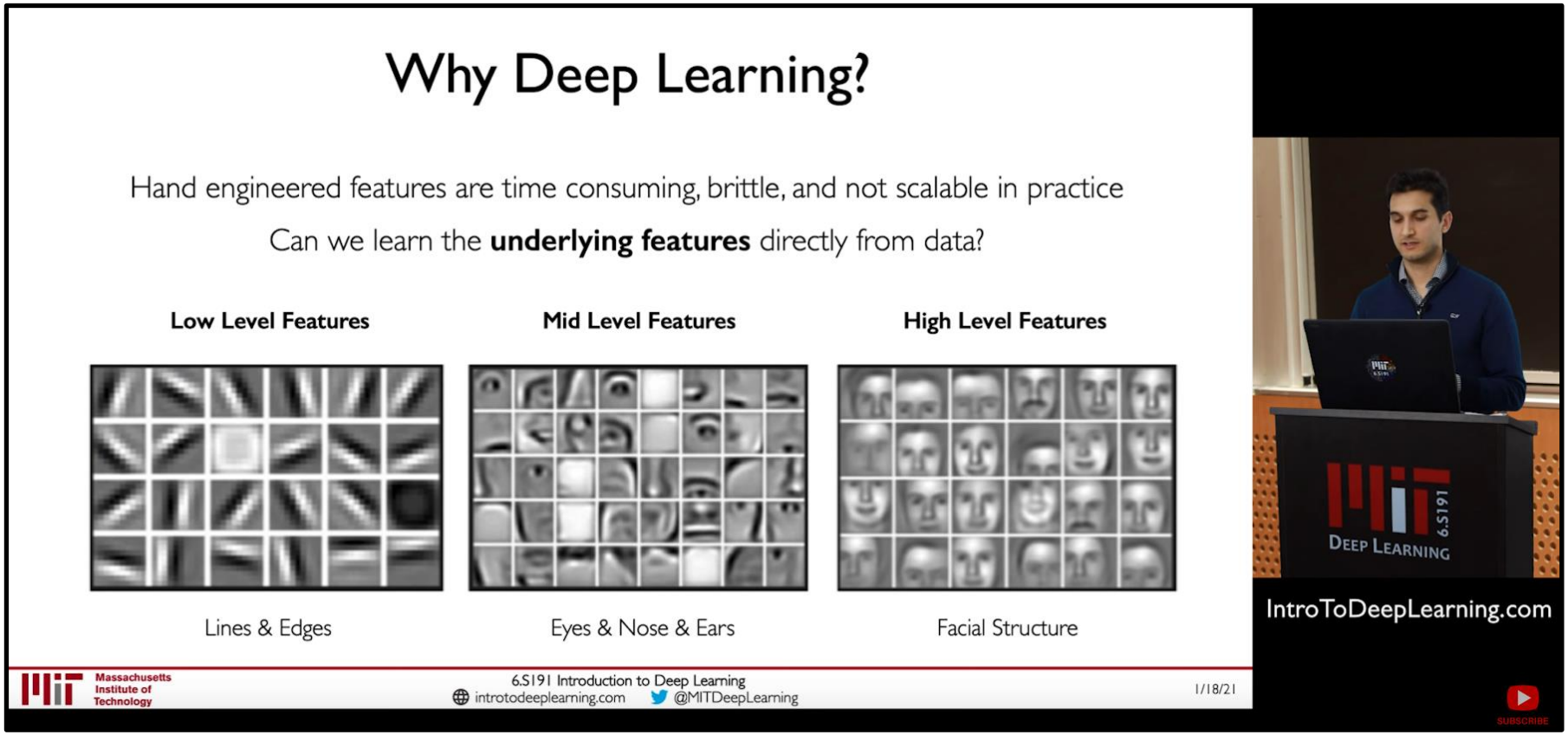
複雑なパターンを抽出する能力に優れる。

- **自動でのタスク実行**

パターン抽出ならびにタスク実行は自動で行われる。

ディープラーニングへの期待

さまざまなレベルのパターン



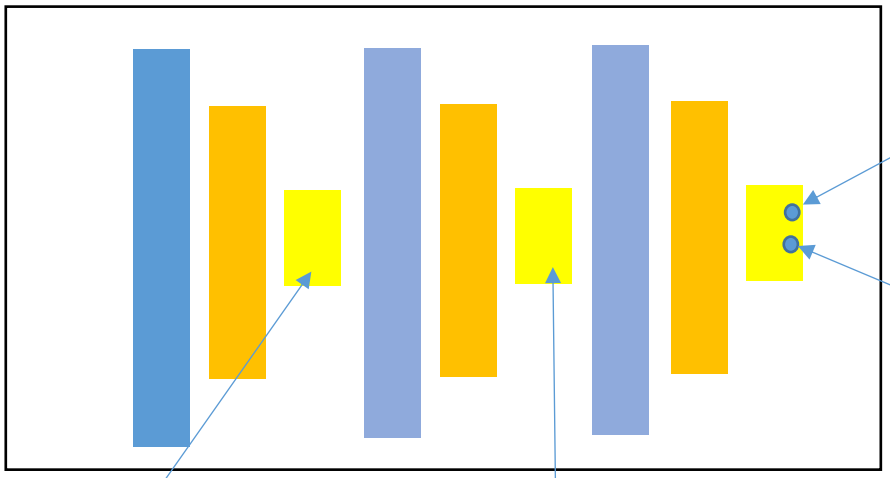
線や点のレベル　目，鼻，耳のレベル　顔の構造のレベル

MIT Introduction to Deep Learning | 6.S191,
https://www.youtube.com/watch?v=5tvmMX8r_OM
の「Why Deep Learning」のページ

教師なし学習のニュース (2011年)

- **教師なし学習**（この画像が「人の画像である」，「猫である」という**正解がない**）
- 訓練データ: YouTube から**ランダムに選ばれた画像 1000万枚**
- **1000台のマシンで，3日間の学習**
- **9層のニューラルネットワーク**を使用

高次のパターンを認識できる能力を獲得

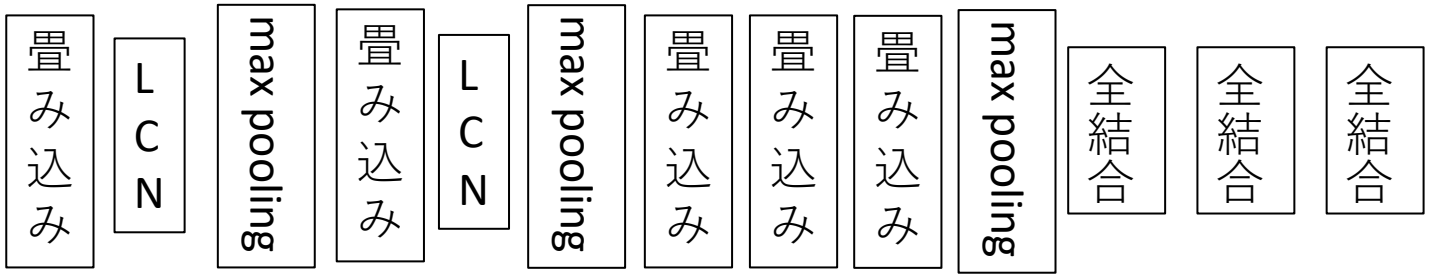


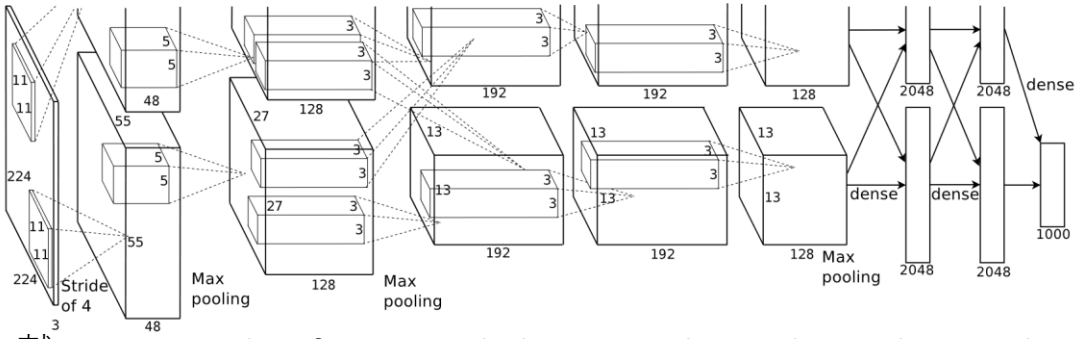
文献: Building high-level features using large scale unsupervised learning
Quoc V. Le, Marc'Aurelio Ranzato, Rajat Monga, Matthieu Devin, Kai Chen, Greg S. Corrado, Jeff Dean, Andrew Y. Ng, arXiv 1112.6209, 2011, last revised 2012.

教師あり学習のニュース (2012年)

- **教師あり学習**の AlexNet で**画像分類**を行う
- **訓練データ: 画像約 100万枚以上**（ImageNet データセット, 22000種類に分類済み）
- ILSVRCコンペティション: 画像を **1000 種類**に**分類**
- **ディープニューラルネットワーク**を使用

畳み込み, max pooling, 正規化(LCN), softmax, ReLU, ドロップアウト

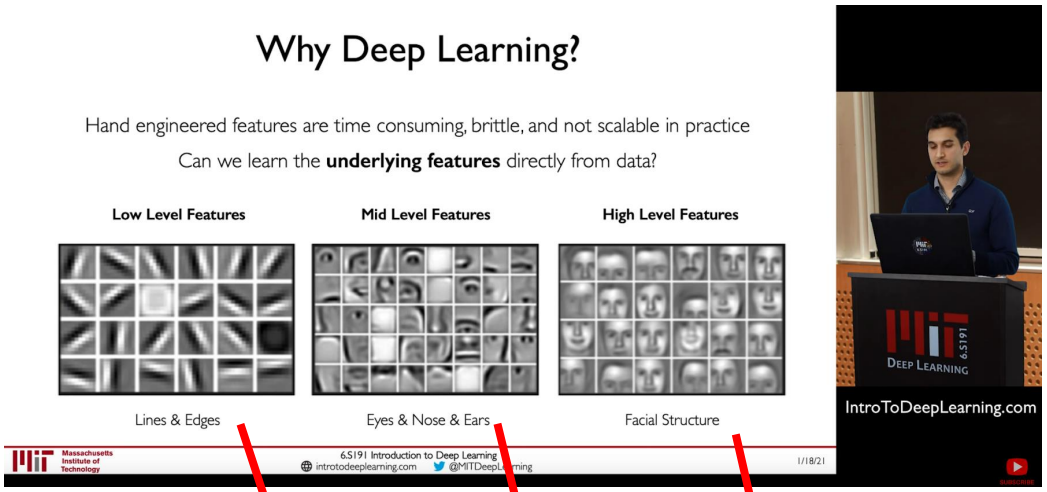


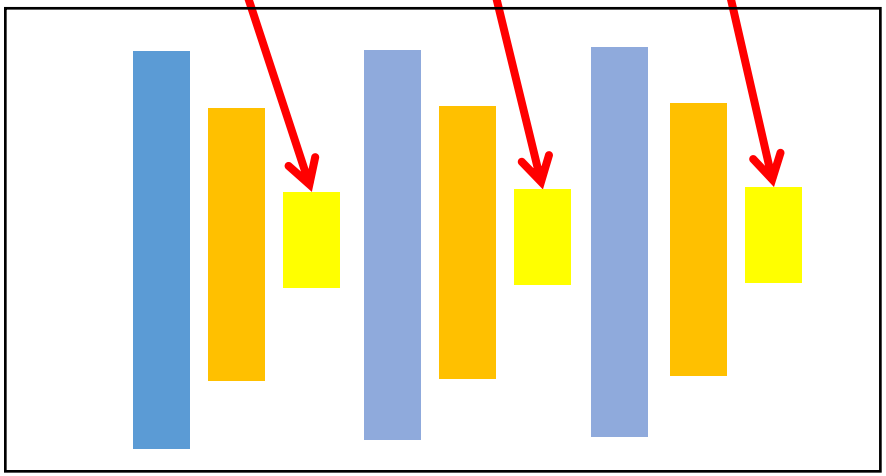


文献: ImageNet classification with deep convolutional neural networks, Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, NIPS'12, 2012.

ディープラーニングへの期待

さまざまなレベルのパターンを**抽出・認識**できるようになる」という考える場合も





「より出力に近い層では，**より高次のパターン**を認識したい」という考え方も

画像分類の精度の向上

ImageNet データセットの画像分類の結果

- **ディープラーニング**の進展
- **画像分類は，場合によっては，AIが人間と同等の精度**　とも考えられるように

画像分類の誤り率 (top 5 error)

人間: 5.1 %

PReLU による画像分類: 4.9 %

（2015年発表）

文献: Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun,
Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification
arXiv:1502.01852, 2015.

ディープラーニングの研究の進展

多層でも学習が**成功するため**のさまざまな工夫

- 活性化関数: ReLU など
- ドロップアウト
- He の初期化
- 大量の**データ**の準備
- **繰り返し学習**による**学習不足**の**解消**
- **データ拡張**による**データ**の増量
- **ニューラルネットワーク**を**高速にシミュレーション**できる高性能のコンピュータ

ディープラーニングの応用と特徴

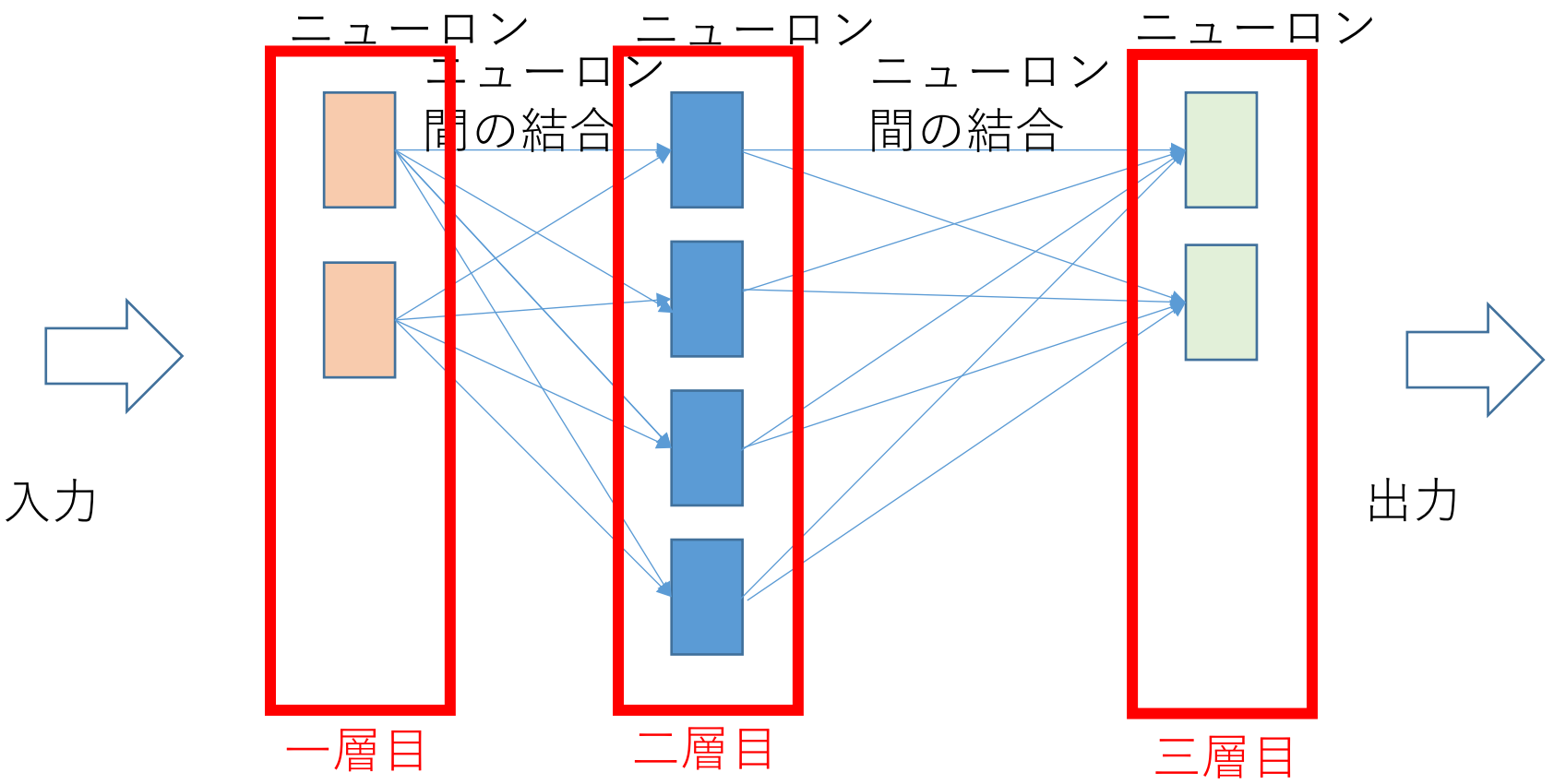
- **ディープラーニング**は**機械学習**の一種であり、人工**ニューラルネットワーク**を使用して**データから学習**し、複雑な**タスクを実行**する技術

- 「ディープ」の名前は、**多層のニューラルネットワーク**を使用することに由来

- ディープラーニングが広く利用される理由は、**多様なデータに適用**でき、**さまざまなタスク**で高性能を発揮するため。例えば、**画像認識**、**自然言語処理**、**音声認識**など。

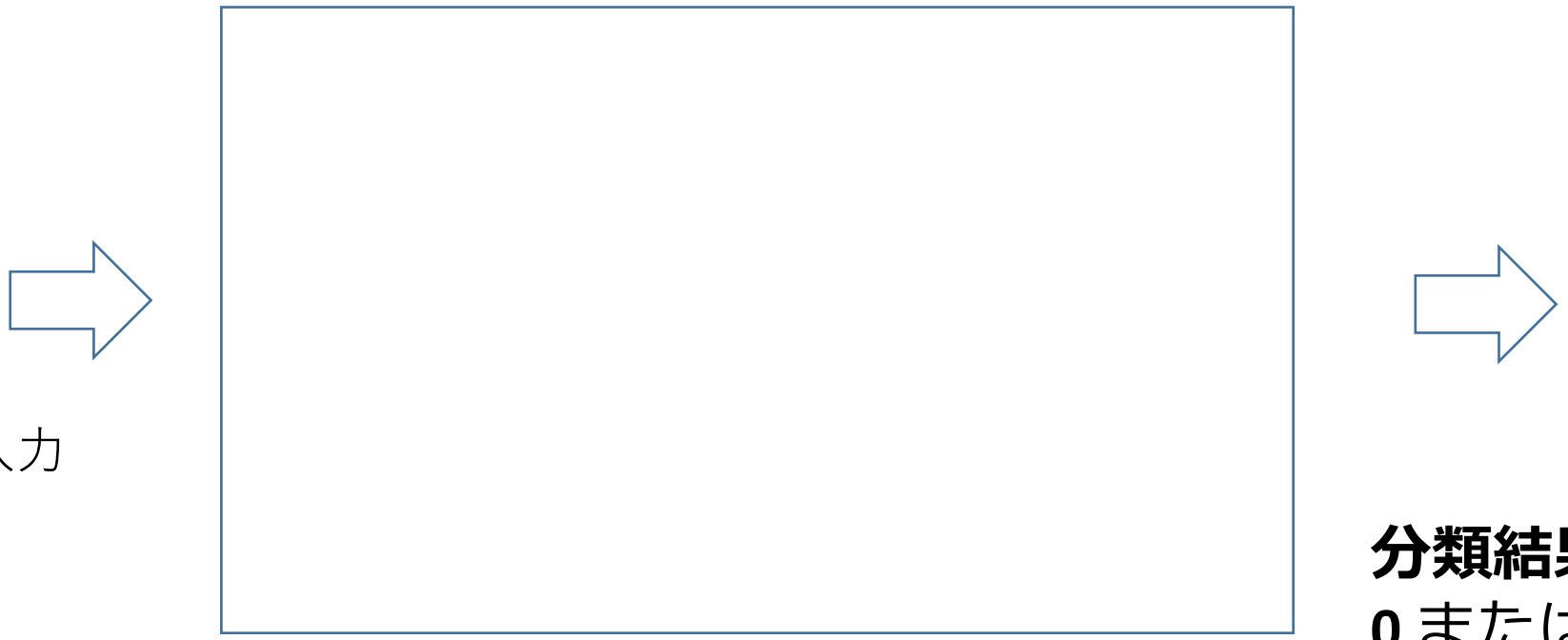
5-5. ニューラルネットワークを用いた分類

層構造とデータの流れ



- **層構造**では，**データ**は，**入力から出力への一方向**に流れる
- **各層**は、**同じ種類のニューロン**で構成

３種類の中から１つに分類する場合

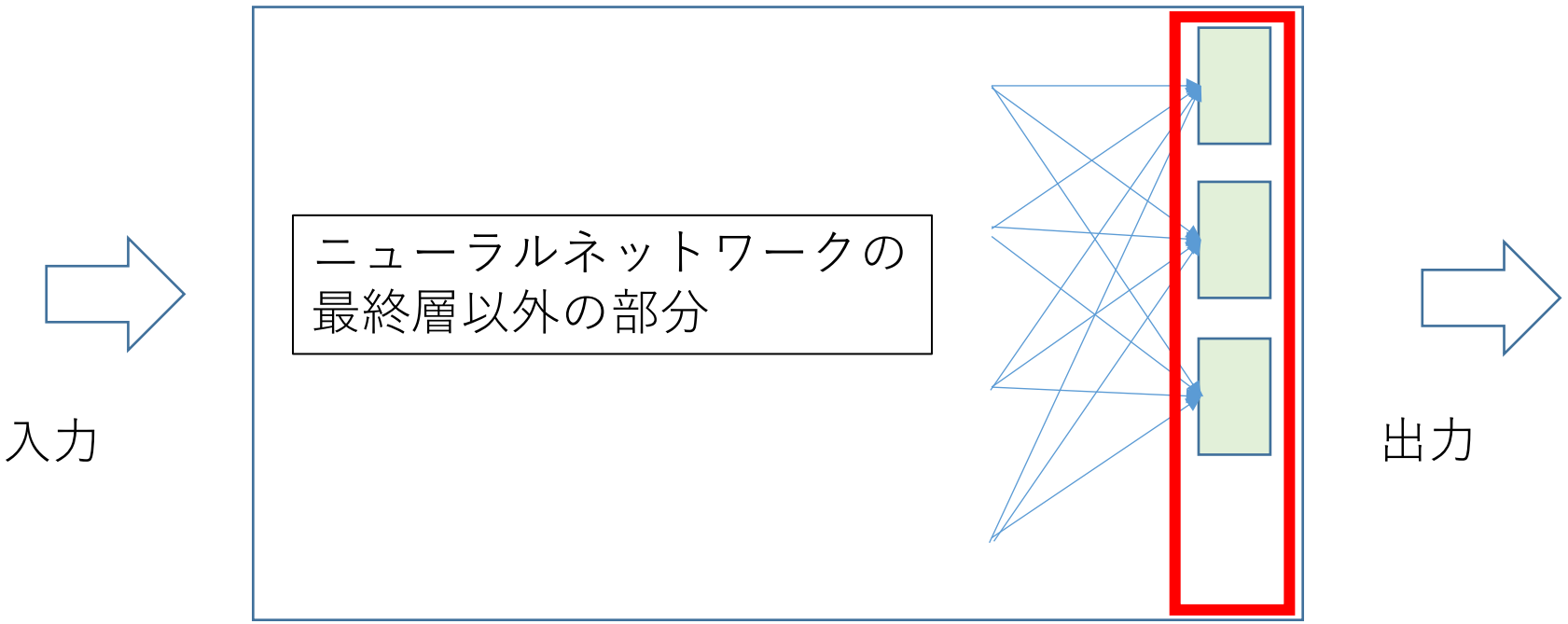


入力

分類結果
0 または **1**
または **2**

３種類の中から１つに分類する場合

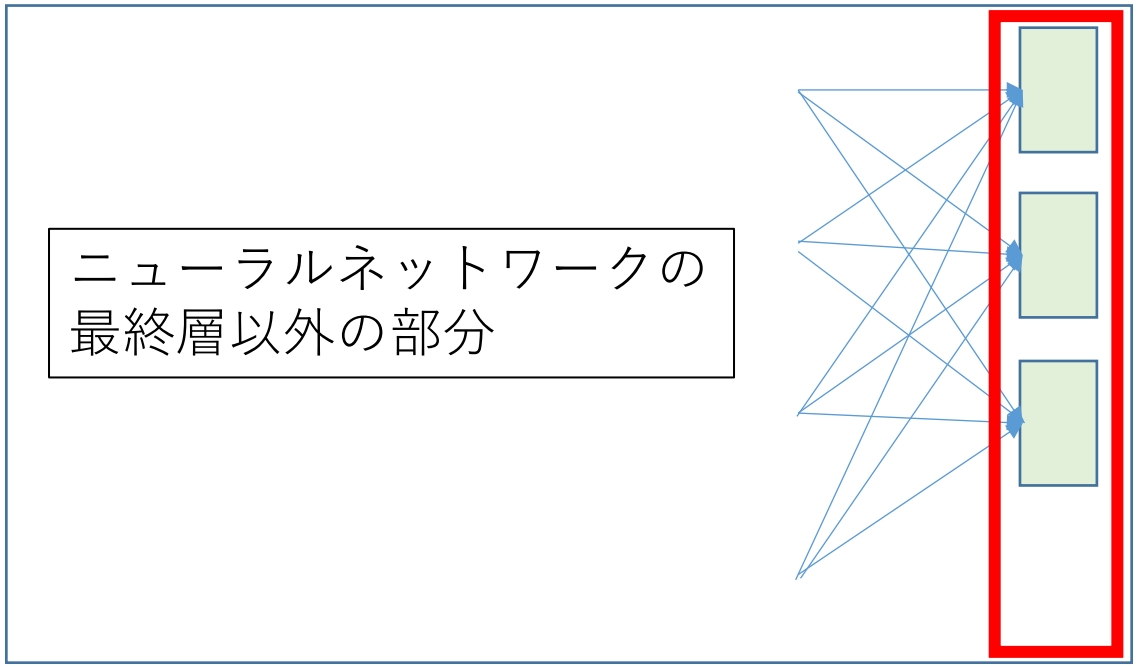
最終層のニューロン数：**3**　にする　　ニューロン



データは入力から出力の方向へ

最終層について，１つが強く
活性化するように調整

ニューロン



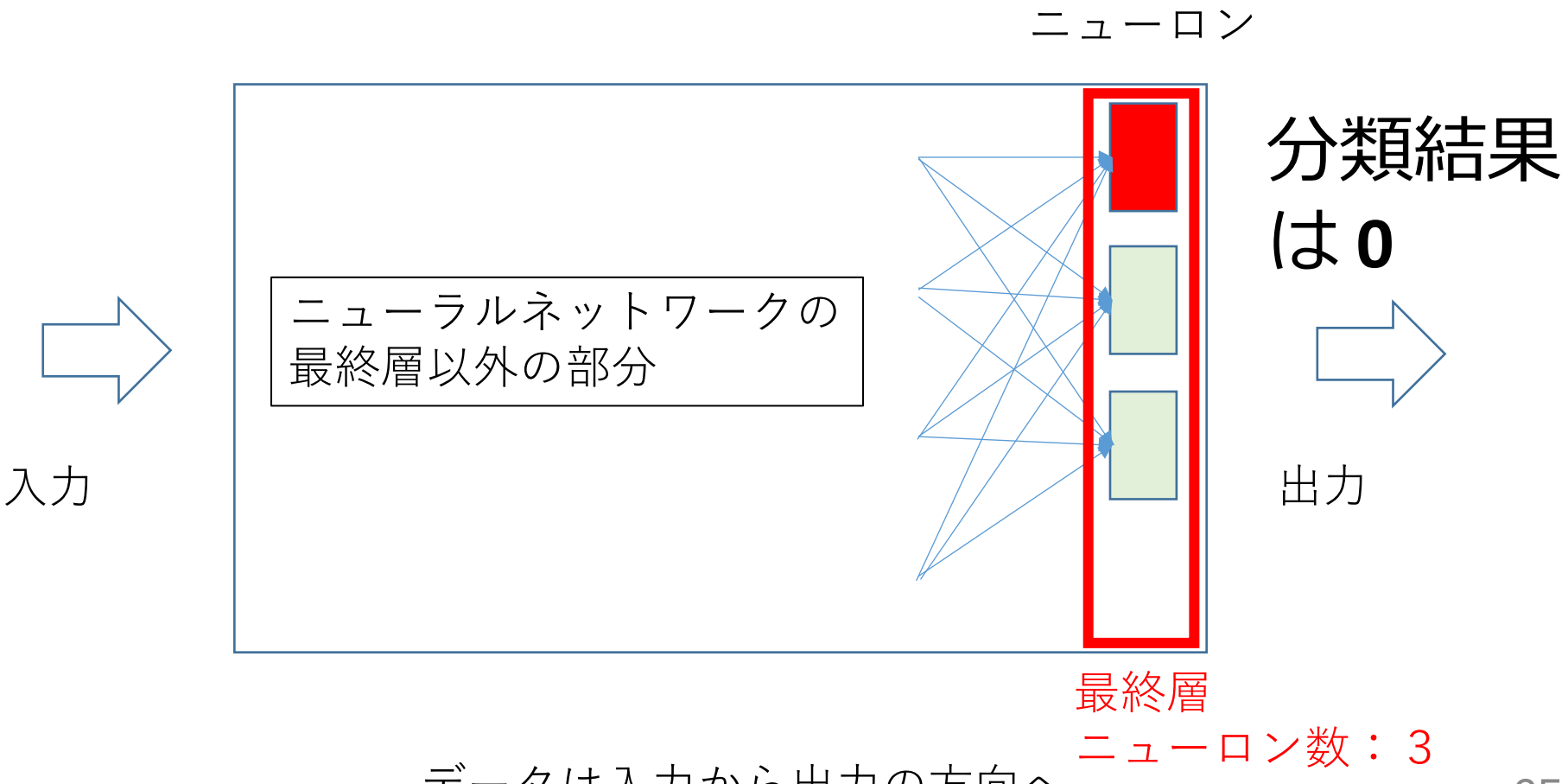
入力

出力

最終層
ニューロン数：３

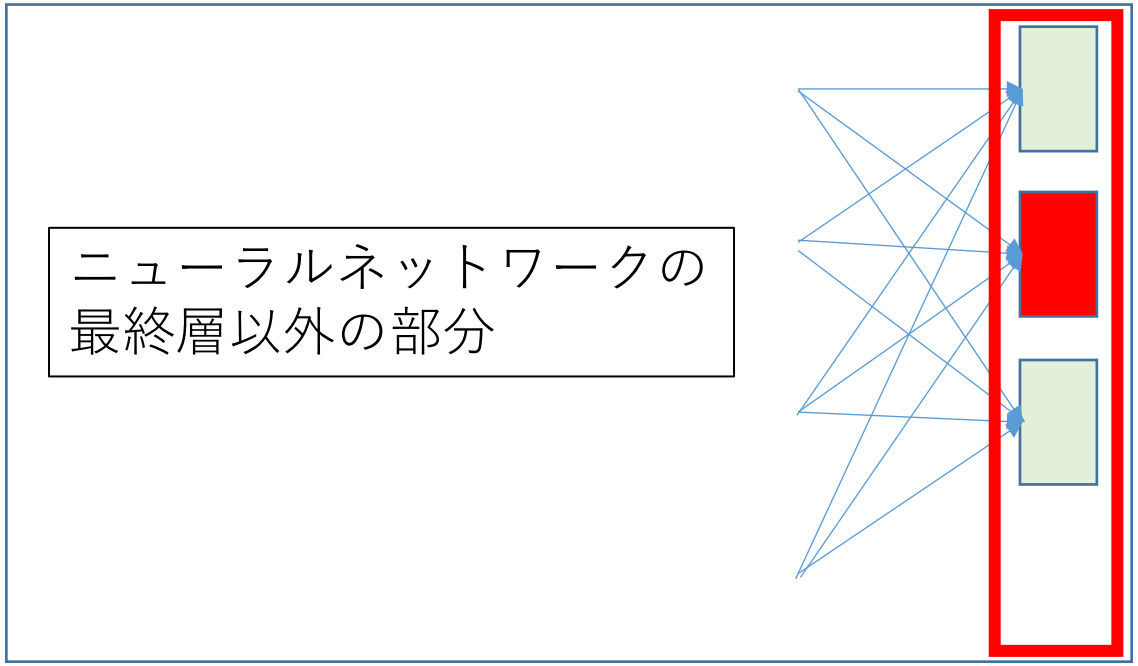
データは入力から出力の方向へ

最終層について，１つが強く
活性化するように調整



最終層について，１つが強く
活性化するように調整

ニューロン



分類結果は1

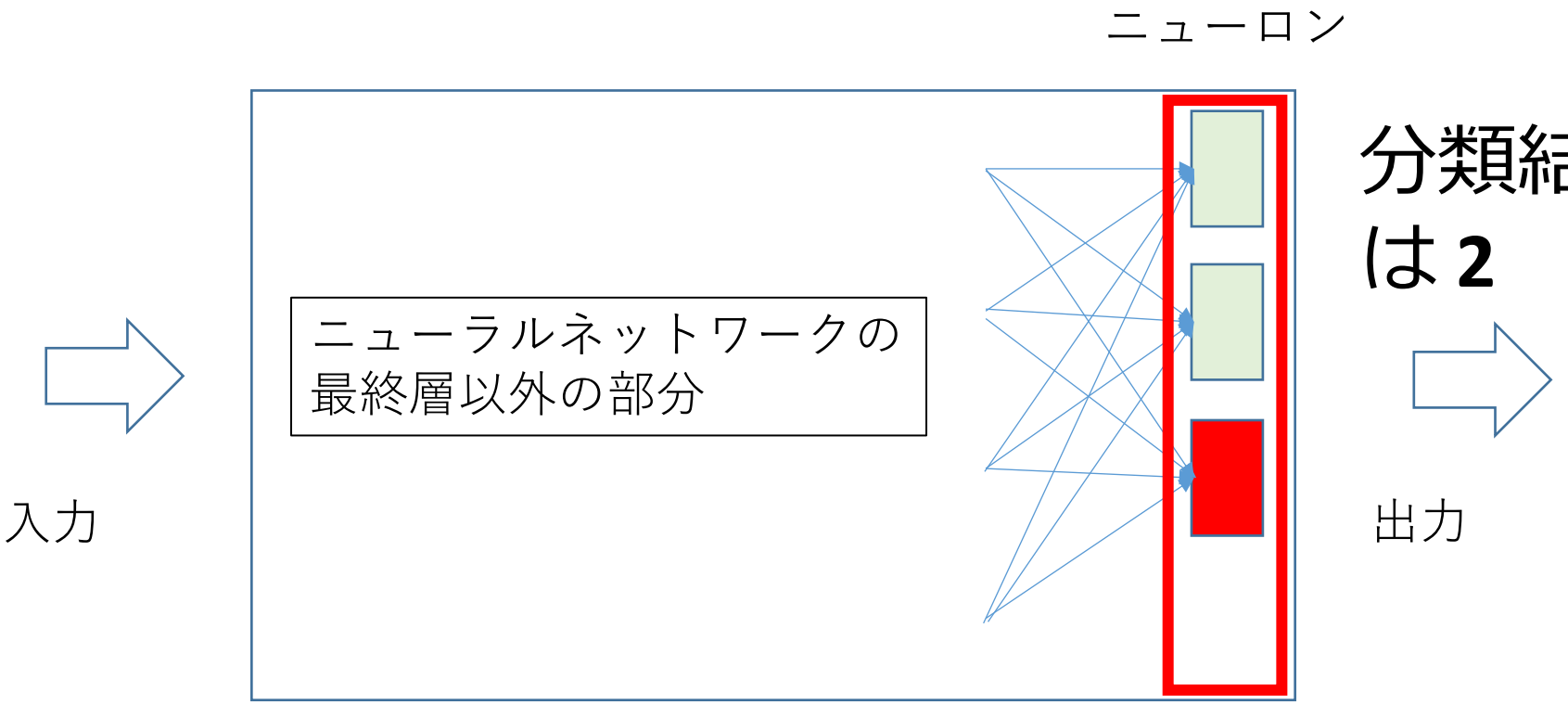
入力

出力

最終層
ニューロン数：３

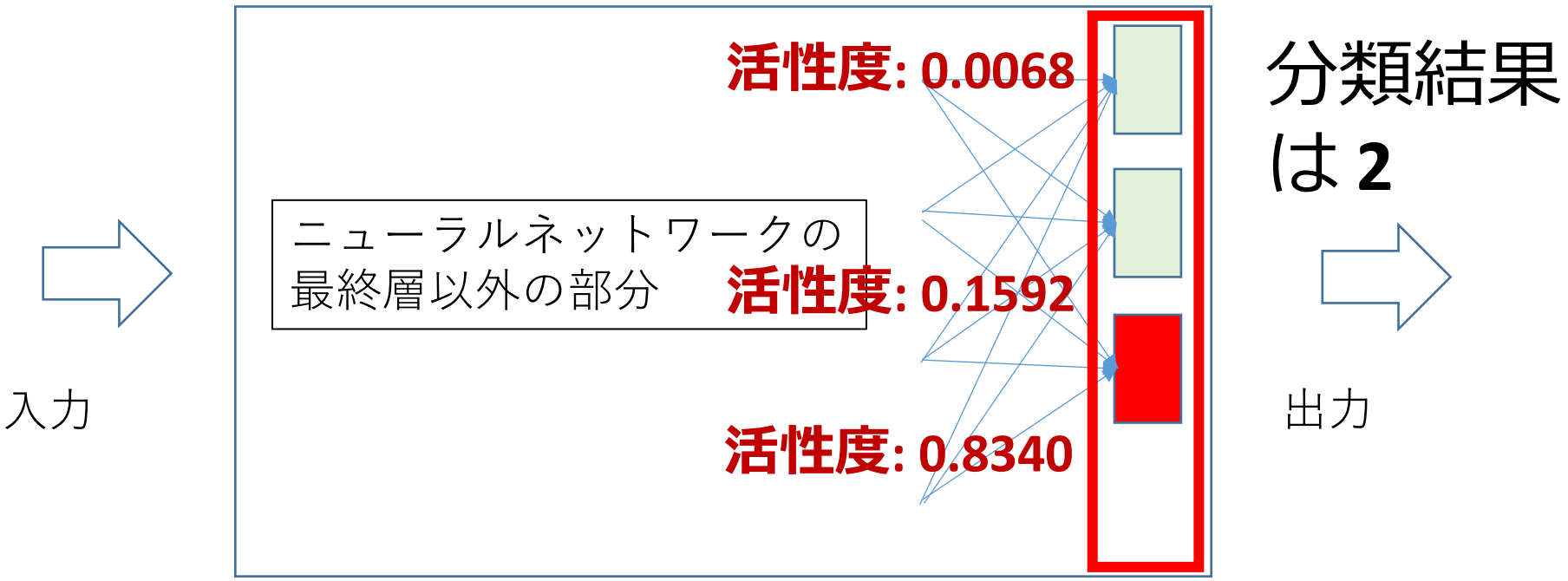
データは入力から出力の方向へ

最終層について，１つが強く
活性化するように調整



データは入力から出力の方向へ

最終層について，１つが強く
活性化するように調整



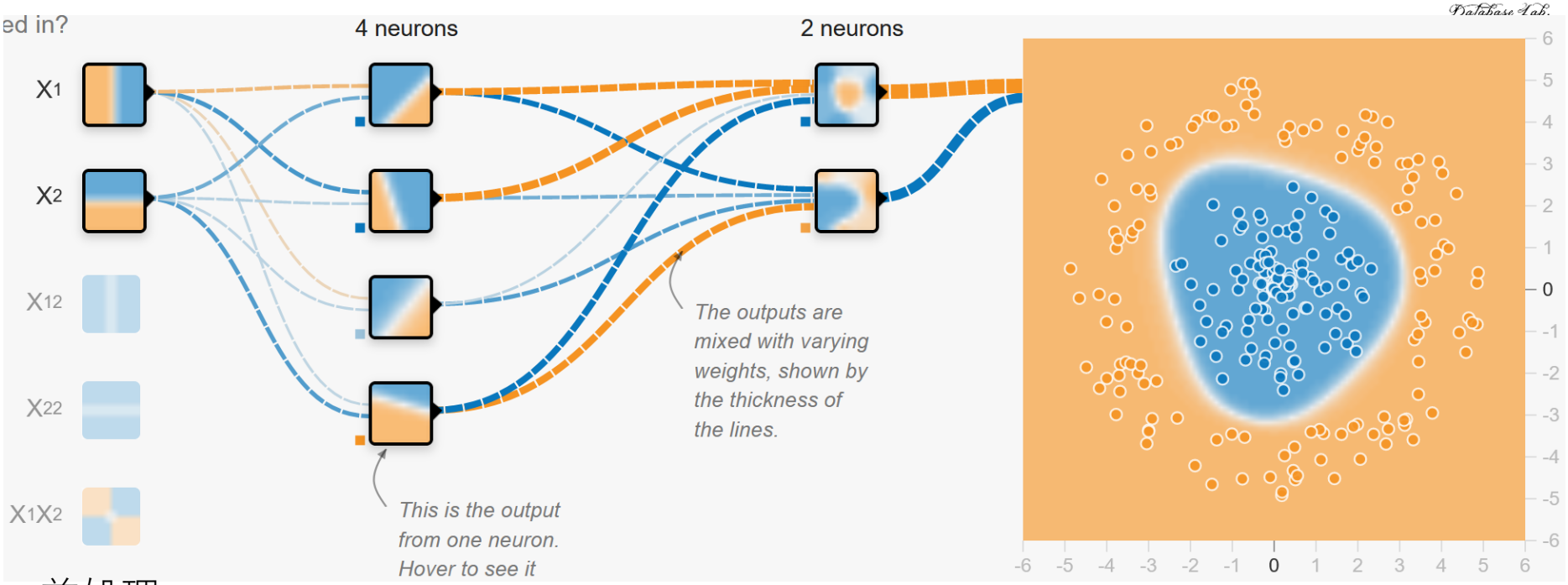
実際には，活性度は**0 から 1** のような数値である.
最も**活性度**の**値が高いもの**が選ばれて，分類結果となる

ニューラルネットワークを用いた分類

ニューラルネットワークを**分類**に使うとき

- **最終層**の**ニューロン**で、**最も活性度の値の高いもの**が選ばれて、分類結果となる

- **誤差**がある



前処理
（データが青い部分にあれば活性化）

データが中央にあれば活性化

→結合→ 1層目 →結合→ 2層目

ニューラルネットワーク

URL: https://playground.tensorflow.org/

ニューラルネットワークのまとめ

- **ニューラルネットワーク**は，単純な原理で動く
- **ニューロン**は**複数の入力**を受け取る．中では，**活性化関数**を用いた**値の変換**を行う．
- **層構造**と**全結合**のニューラルネットワークが最もシンプルである．**入力から出力の方向へ一方向にデータが流れる**．**２つの層の間のニューロン**は**すべて結合**．
- **ニューラルネットワーク**は，**結合の重みとバイアスを調整**して，**望みの出力を得る**．

ニューラルネットワークの**学習**では，**データを利用**して，**望みの出力が得られるよう**に，**結合の重みとバイアスの調整**が行われる．画像認識や自然言語処理などに広く利用されている．

授業の学ぶ意義と満足感

- ニューラルネットワークの仕組みを基礎から学び、人工知能（AI）の技術への理解の深まり
- 工学を学ぶことのへの意欲の高まり
- ディープラーニング（深層学習）の実践的活用に役立つ知識とスキルの向上
- 人工知能（AI）の応用可能性の広さを認識し、様々な分野での課題解決に挑戦する意欲が湧く。

図解と例示を通じて学び、学習意欲を高めましょう。技術の面白さを実感しましょう。