1216583c
谷　紗耶加

ニューラル
ネットワーク

ニューラルネットワークとは

- 人間の脳には１４０億個のニューロン⇒規則に従って結合⇒神経回路
- 神経細胞が結ばれた神経回路をモデル化

⇒　**ニューラルネットワーク**
(neural networks)

- 非線形回帰分析、非線形判別分析の有力な機械学習の方法

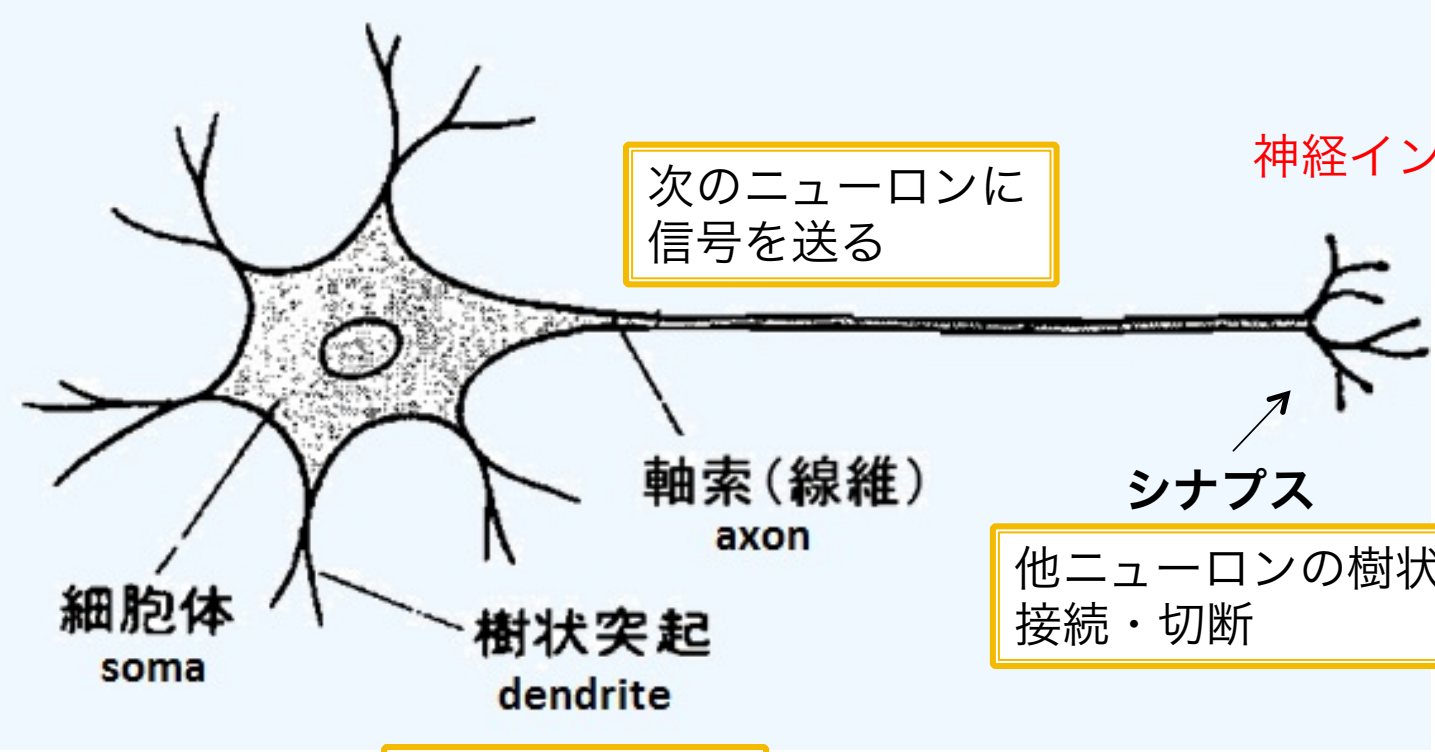
ニューラルネットワークの基礎

- 1943年　マカロックとピッツ

- 神経細胞（ニューロン）
  ⇒神経回路を構成する最小単位
  ⇒細胞体、軸索、樹状突起、シナプス

- 信号処理、信号伝達

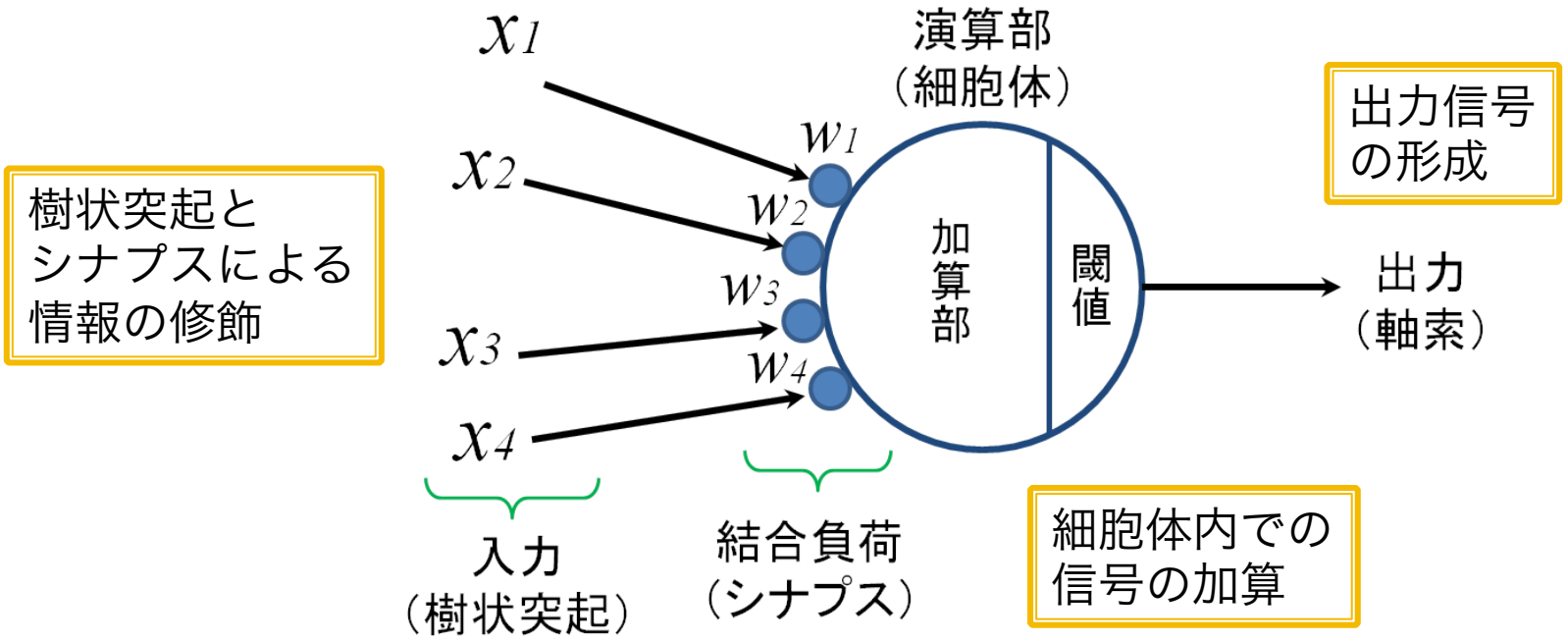
神経細胞（ニューロン）の図



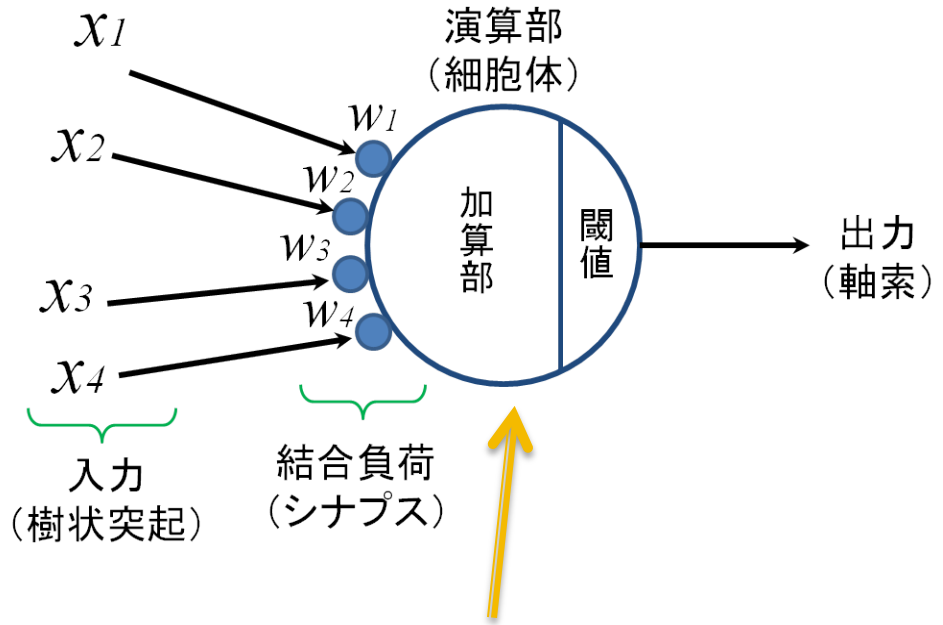
形式ニューロンの構造

- ニューロンが多数並列に接続されたシステムを数理的にモデル化
  ⇒　ニューラルネットワーク

- 構成要素　⇒　形式ニューロン（ユニット）



入力と出力の関係



$y=f(u)=\{\blacksquare 1 \quad u\geq\theta 0 \quad u<\theta$

あるいは

$y=f(u)=1/1+e\uparrow -u$

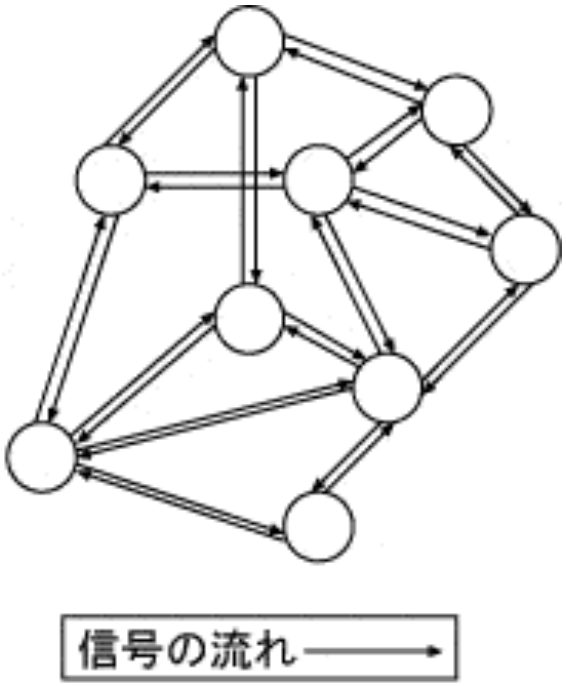
（ただし $u=\sum i=1\uparrow p\blacksquare w\downarrow i \, x\downarrow i$ ）

入力値$x\downarrow i$　に　重み$w\downarrow i$
を掛けた代数和　$u=\sum i=1\uparrow p\blacksquare w\downarrow i \, x\downarrow i$

モデルの分類（ネットワークモデル）

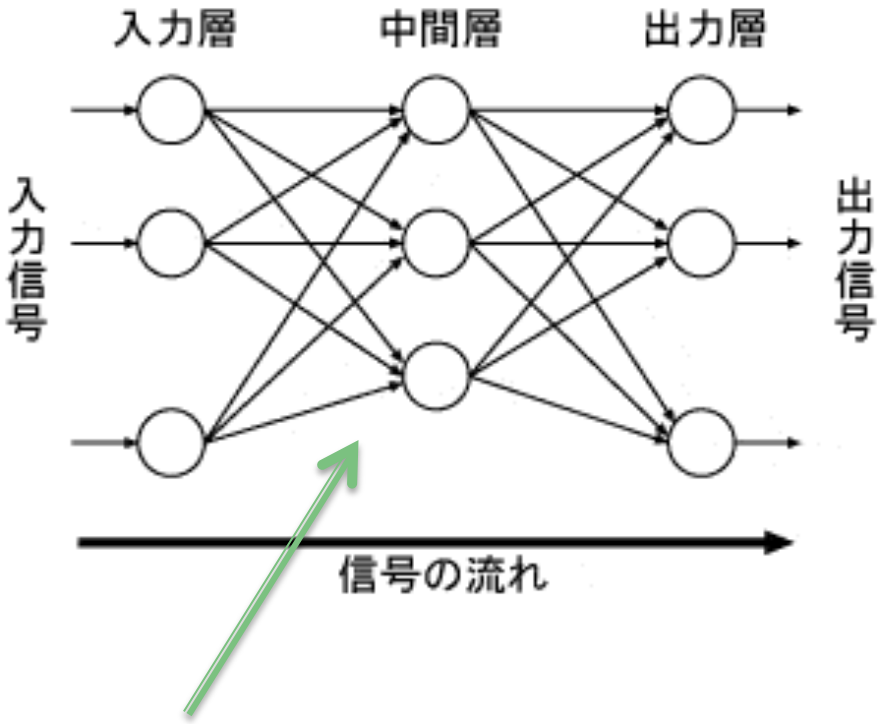
- **階層型ネットワーク**
  - 最も多く使われる
  - 信号は一方向

- **非階層型ネットワーク**
  - 階層型ネットワーク以外
  - 信号が任意の方向に流れる
    （逆方向、ループ状もある）



相互結合型ネットワーク
（非階層型）

階層型ネットワーク



ニューロンが階層的に結合

入力層と中間層（隠れ層）の各ユニット出力は次の層のすべてのユニットにリンク　　⇒　**階層的完全結合型**

フィードフォワード型（前向きのみ）

中間層および中間層のニューロンの数が学習の回数および収束に直接影響

計算の負担を考えると少な目がよい
一般的には３層

単一中間層ニューラルネットワーク

$$y_k = \phi_0 \left(a_k + \sum_h w_{hk} \phi_h \left(a_h + \sum_i w_{ih} x_i \right) \right)$$

（1）入力データ　$x_1, x_2, \ldots, x_p$　をニューラルネットワークに与え、最初の段階では結合の重みとして　$w_1, w_2, \ldots, w_p$　に小さいランダム値を与えて出力を求める

（2）出力結果と学習用のデータを比較し、新しい結合の重み

$w(j+1) = w(j) + n \times \delta \times O_{net}$

を計算する。

（3）最適な解が得られるまで
（2）を繰り返す。

$w(j+1)$　$j+1$回目の結合の重み
$w(j)$　j回目の結合の重み
n　定数
δ　ニューロンの出力結果と学習用のデータとの差に関する関数
中間層、出力層によって異なる
O_{net}　ニューロンの出力結果

パッケージと関数

- nnet
  - 中間層ありのニューラルネットワークのパッケージ、Rに標準装備
  - 教師ありの学習法アルゴリズム

```
nnet(formula, data, weights, size,...)
nnet(x, y, weights, size,...)
```

- predict
  - nnetを学習モデルとし、当てはめて値を返す

```
predict(学習したモデル, データ, type="")
```

ケーススタディ（1）

- Irisデータを用いて、品種の識別
- 学習用データ（iris.train）とテストデータ(iris.test)を作成

```
> even.n <- 2*(1:75)
> iris.train <- iris[even.n,]
> iris.test <- iris[-even.n,]
```

ケーススタディ（2）

- 作製した教師（学習、訓練）データを用い、関数nnetによるニューラルネットワークモデルの作製

```
> library(nnet)
> iris.nnet <- nnet(Species~.,size=3,decay=0.1,data=iris.train)
# weights:  27
initial  value 85.657162
iter  10 value 42.310822
iter  20 value 36.658685
iter  30 value 19.230029
iter  40 value 18.460511
iter  50 value 17.455241
final  value 17.432930
converged
```

decay:
衰退重みのパラメータ

size:
隠れ層（入力層と中間層）のユニット数

data:
データセットの名前

ケーススタディ（3）

- 関数predictを用いてテストデータについて予測・判別

predict(学習したモデル，　データ，　type="")

```
> iris.nnetp <- predict(iris.nnet,iris.test[,-5],type="class")
> table(iris.test[,5], iris.nnetp)
            iris.nnetp
             setosa versicolor virginica
  setosa         25          0         0
  versicolor      0         23         2
  virginica       0          0        25
> 
```

７５の個体の中２つが誤識別

その他

- 自己組織化マップ（第６章）
  - 学習データなしのニューラルネットワークの１つ
- class（パッケージ）
  - Ｒ実装、学習ベクトル量子化(LVQ: Learning Vector Quantization)アルゴリズム

```
> library(class)
> cl <- iris.train[,5]
> cd <- lvqinit(iris.train[,1:4],cl)
> cd1 <- lvq1(iris.train[,1:4],cl,cd)
> test.re <- lvqtest(cd1, iris.test[,1:4])
> table(iris.test[,5],test.re)
            test.re
             setosa versicolor virginica
  setosa         25          0         0
  versicolor      0         23         2
  virginica       0          1        24
```

（実行例）

LVQは変数の数が多い場合に強み