[連載]フリーソフトによるデータ解析・マイニング第 23 回

RとWEKAによるニューラルネットワーク

1. ニューラルネットワークとは

我々人間の脳には約 140 億個のニューロン(神経細胞)があり、それぞれのニューロンはある規則に従い結合され神経回路を形成していると言われている。

大脳皮質の薄い切片を肉眼で見えるように処理したものを図 1 に示す。こうして見えるのは神経細胞の 2%に過ぎないそうである。

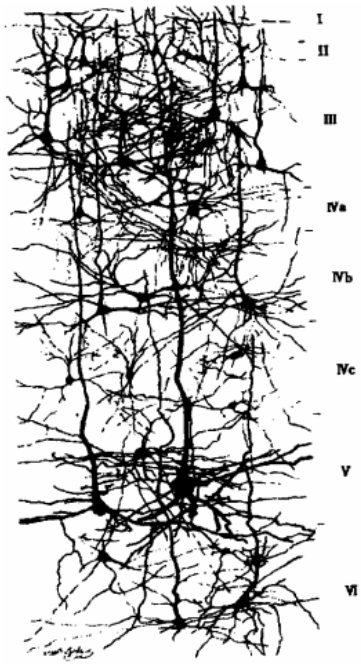


図 1　神経細胞の結合状況の標本

(J. Leroy Conel, The Postnatal Development of The Human Cerebral cortex, Vol. VI. Harvard Univ. Press, 1959)

神経細胞が結ばれた神経回路をモデル化したものを人工ニューラルネットワーク、通常略してニューラルネットワークと呼んでいる。ニューラルネットワークに関する研究の発端は、1943 年のマカロックとピッツ(W. S. McCulloch and W. H. Pitts)の研究にさかのぼる。その後数回の研究ブームの起伏を経て研究が進められてきた。日本では 1990 年頃から画像認識、音声認識、株予測などに応用された。

図 1 の神経回路を構成する最小単位である神経細胞(ニューロン)は細胞体、軸索、樹状突起、シナプス（Synapse）により構成されている。そのイメージを図 2 に示す。

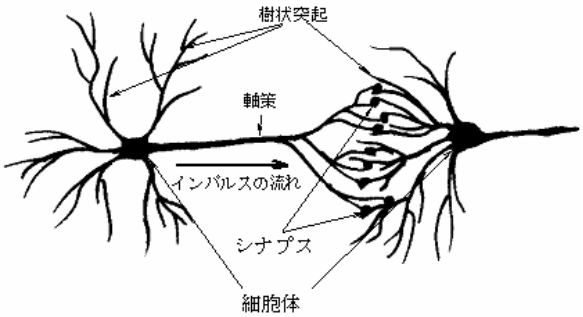


図 2　神経細胞の型態

細胞体は、他の細胞から送られた信号を処理し、ある条件を満たすと次のニューロンに信号を送る。軸策はニューロンの出力信号を次のニューロンに送る経路であり、樹状突起は信号を受け取り細胞体に転送する経路である。軸策、樹状突起は細胞体から多数分布した樹状の線維の集まりであるが、軸策は樹状突起より長く伸びている特徴がある。シナプスは軸策先に付いた粒状のもので、他のニューロンの樹状突起と接続・切断する機能を果たす．ニューロンとニューロンの間ではシナプスという接触点を通じて結び付けられ信号の転送を行なう。軸策から流れる電気信号を神経インパルスと呼び、略してインパルスと呼ぶ。

このようなニューロンが多数並列に接続された場合の全体の機能を、数理的にモデル化したものがニューラルネットワークである。ニューラルネットワークの構成要素は形式ニューロンである。形式ニューロンはニューロンの数理的モデルで、ニューロンの

- ✧　樹状突起とシナプスによる情報の修飾
- ✧　細胞体内での信号の加算
- ✧　出力信号の生成

に着目しモデル化している。形式ニューロンの構造を図 3 に示す。

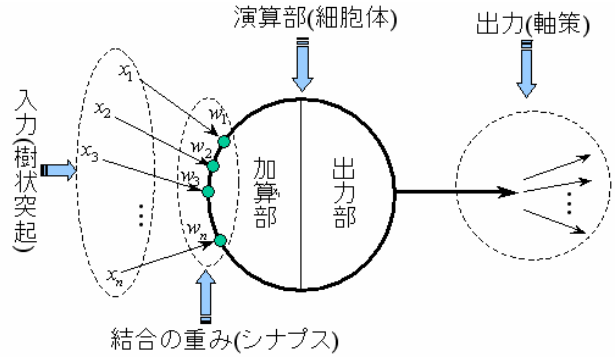


図 3　形式ニューロンの構造

図 3 の $x_1, x_2, x_3, \cdots, x_n$ は樹状突起による入力信号で、$w_1, w_2, w_3, \cdots, w_n$ はそれぞれ入力信号に対応するシナプスの結合の重みである。加算部ではそれぞれの入力値 x_i に重み w_i を掛けた代数和

$u=\sum_{i=1}^{n} w_i x_i$ を求め、出力部は出力信号 $f(u)$ を生成する。ニューラルネットワークはこのような形式ニューロンの相互結合により情報の転送・処理を行なう。形式ニューロンでの入力と出力との関係は、通常次のような関数

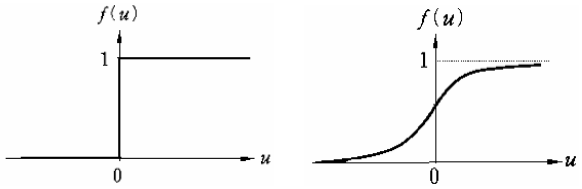
$$y=f(u)=\begin{cases}1 & u\geq\theta\\ 0 & u<\theta\end{cases}$$

あるいは

$$y=f(u)=\frac{1}{1+e^{-u}}\text{、}$$

（ただし、$u=\sum_{i=1}^{n} w_i x_i$ である。）

で表わされる。それぞれの関数のグラフを図 4 に示す。



（a）　階段型関数　　（b）　S 字型関数

図 4　出力関数

出力関数 $y=f(u)$ は u の関数であり、u は入力変数 x_i と重み w_i の線形結合である。ニューラルネットワークにおける結合の重み w_i としては、初めの段階ではランダムで小さい値が与えられ、学習を通じて w_i は徐々に最適な値に置き換えられる。

２．ニューラルネットワークモデル

複数の形式ニューロンがネットワーク状に結合したものをニューラルネットワークと呼ぶが、その結合の仕方によりニューラルネットワークの構造が決まり、構造が異なるとネットワークの機能と特徴もまた異なることになる。このネットワーク構造をネットワークモデルと言う。現時点で多く使われているネットワークモデルは、階層型ネットワークと非階層型ネットワークに分けられる。

（1）階層型ネットワーク

階層型ネットワークは、最も多く使われているニューラルネットワークのモデルである。階層型ネットワークは複数の形式ニューロンが図 5 のように階層的に結合した構造を取っている。ただし、中間層は 1 層とは限らない。図 5 では形式ニューロン（ユニットとも言う）が 3 層に並んでおり、入力層と中間層(隠れ層とも呼ぶ)の各ユニットの出力は次の層のすべてのユニッ

トにリンクされている。このようなモデルを階層的完全結合型と呼ぶ。階層型は、すべて完全結合とは限らない。また、入力層のユニットが中間層を跳び越え直接出力層のユニットと結合することも可能である。

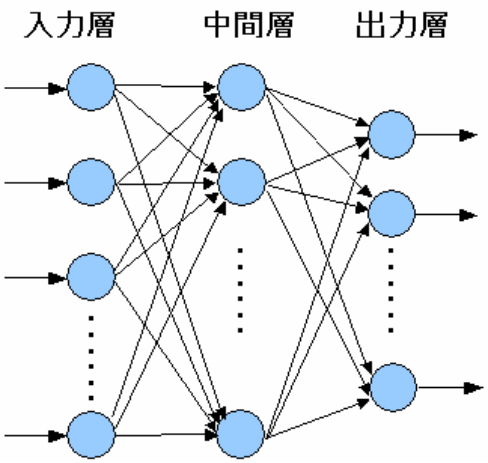


図 5　3 層完全結合をもつ階層型ニューラルネットワーク

図 5 のようなニューラルネットワークモデルは、次の式で定式化することができる。

$$y_k = \phi_0(\alpha_k + \sum_h w_{hk}\phi_h(\alpha_h + \sum_i w_{ih}x_i))$$

ニューラルネットワークの特徴の一つは学習機能にある。学習に関するアルゴリズムは大きく教師ありの学習と、教師なしの学習に分けられる。階層型ニューラルネットワークでは教師ありの学習に適している。

階層型ネットワークは信号の流れの立場ではフィードフォワード型（前向きのみ）と呼ぶ。階層型ネットワークでは、中間層の層数および中間層のニューロンの数が学習の回数及び収束に直接影響を与えるため、中間層、および中間層のニューロンの数をいくつにするかが問題となる。これらは、入力のデータの構造に依存するので一概には言えない。理論的には 4 層にした方が汎化能力の点で優れていると言われているが、計算の負担などを考えると層数を少な目にした方がよい。一般的には 3 層でかなりの問題が解けると言われている。中間層のニューロンの数の決定に関しては

- ✧　ニューロンの数を順次追加する方法
- ✧　ニューロンの数を順次削除する方法
- ✧　情報量理論を用いる方法(たとえば AIC を用いる方法)
- ✧　統計解析の方法

などの方法が提案されている。しかし、多くのデータでは、隠れ層のニューロンの数(ユニット)が当てはまりに与える影響は大きくないことが知られている。

(2) 非階層型ネットワーク

非階層型ネットワークは、階層型ネットワークと異なり、信号が入力層から中間層、出力層の順に単一の方向に限らず、任意の方向に流れることも許すネットワークモデルである。信号が逆方向、またはループ状に流れることもある。非階層型ネットワークの学習は、教師なしの学習アルゴリズムに適する。

3. ニューラルネットワークの学習プロセス

ニューラルネットワークによるデータ解析方法と従来のデータ解析方法との大きく異なる点はニューラルネットワークが学習機能を持っている点にある。学習とは、計算した結果（出力）を学習用のデータに近づくように重みの修正などを繰り返し、最適な解を求めることである。ニューラルネットワークでは、

✧　シナプスの結合の重みの学習
✧　出力関数の学習
✧　中間層におけるニューロンの数の学習

などの学習機能が考えられる。その中、最も基本的で、かつ重要なのはシナプスの結合の重みに関する学習である。ここでは階層型ネットワークにおけるシナプスの結合の重みを定めるための学習について大まかに説明する。

階層型ネットワークにおける結合の重みを定めるための学習は次のステツプに分けられる。

✧　入力データ $x_1, x_2, \cdots, x_q$ 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 $w_1, w_2, \cdots, w_q$ に小さいランダム値を与えて出力を求める。

✧　出力結果と学習用データを比較して新しい結合の重み

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

を計算する。ここで

$w(j+1)$ ： $j+1$回目の結合の重み

$w(j)$ ： j回目の結合の重み

η ：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で、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

である。

✧　新しい結合の重みに基づいて新たな出力値を求める。

✧　満足する結果が得られるまで重み $w(j+1)$ の計算を繰り返す。

ニューラルネットワークは、従来のデータ解析方法より柔軟性を持っており、潜在力を持っている。すでに多くのアルゴリズムが実用され、商業用データ解析のソフトウエアパッケージ

に実装されている。

ニューラルネットワークが最も得意とするのはパターン認識・分類、ノイズが混在しているデータ処理、関数の近似である。ニューラルネットワークが最も早く応用されたのは、手書き文字の識別を含む画像のパターン認識、音声認識などの情報工学に関連する分野であった。その後医学の病気の診断、財務分析、経済分析、市場分析などにも応用されている。最近では、人文科学の領域の研究への応用も見られるようになっている。

ニューラルネットワークに関する書物は数多く市販されている。短時間で入門するのには、次のようなホームページも大いに参考になる。次のサイトでは簡単なシミュレーションもできる。

http://mars.elcom.nitech.ac.jp/java-cai/neuro/menu.html

ニューラルネットワークの詳細に関しては、村田久氏による本誌の連載(No.128～134)を参照したい。

4. Rによるニューラルネットワーク

Rには、次の3種類のニューラルネットワークパッケージがある。

- ✧ nnet(単一中間層の階層型ニューラルネットパッケージ)
- ✧ AMORE（柔軟なニューラルネットワークパッケージ）
- ✧ SOM(自己組織化マップパッケージ)

本稿では、単一中間層のニューラルネットワークパッケージnnetを紹介する。パッケージnnetでは、関数 **nnet** を用いて学習を行い、学習モデルによる当てはめは関数 **predict** を用いる。

関数 **nnet** の書き式は次の2種類がある。その主な引数を表1に示す。

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

表1　関数nnetの主な引数

引数	説明	デフォルト
formula	y~x1+x2の形式	
data	データセットの名前	
x	説明変数のデータ	
y	目的変数のデータ	
subset	dataの中の一部分	
weights	個体の重み	1
rang	初期の重みのランダム値の範囲[-rang,rang]	0.7
Wts	結合の重みの初期値、指定しないときにはrang範囲の一様乱数を用いる	乱数
size	隠れ層のユニット数	
lineout	線形出力のユニットの数	FALSE
entropy	エントロピー法を使用(最大条件付の尤度)	FALSE
softmax	対数確率モデルを使用	FALSE

skip	入力と出力を直接結合	FALSE
decay	衰退重みのパラメータ	0
maxit	最適化計算の繰り返し回数	100
Hess	重みに関する最適な Hessian 値を返す	FALSE
trace	最適化計算の過程を出力	TRUE
MaxNWts	最大許される重みの数	1000
abstol	当てはめ値による計算を止める制御値	1.0e-4
reltol	最適値による計算を止める制御値	1.0e-8

パッケージ nnet における関数 **predict** の書き式を次に示す。

predict(学習したモデル, データ,type="")

引数 type を"raw"に指定すると当てはめの値を返し、"class"に指定すると対応するグループ(判別分析のグループ)に関する値を返す。

実際のデータを用いて関数 **nnet** の使用方法について説明する。ここでも、iris データを用い、まず判別分析の場合と同じく次のように学習用のデータ(train.data)とテスト用のデータ(test.data) を作成する。

```
>even.n<-2*(1:75)
>train.data<-iris[even.n,]
>test.data<-iris[-even.n,]
>Iris.lab<-factor(c(rep("S",25),rep("C",25),rep("V",25)))
>train.data[,5]<-Iris.lab
>test.data[,5]<-Iris.lab
```

作成した学習データを用いて次のように関数 **nnet** によるニューラルネットワークモデルを作成し、関数 **predict** を用いてテストデータについて予測・判別を行なう。

```
>library(nnet)
>iris.nnet<- nnet(Species~ .,size=3,decay=0.1,data=train.data)
>Y<-predict(iris.nnet,test.data,type="class")
>table(test.data[,5],Y)
   Y
     C  S  V
  C 23  0  2
  S  0 25  0
  V  0  0 25
```

返された結果からわかるように、上記のニューラルネットワークモデルによる iris の品種の識別問題では、75 個体の中 2 つが誤識別されている。

ニューラルネットワークのモデルは用いる引数のパラメータに依存するため、上記の結果が最善の結果とは言い切れない。

関数 **nnet** による回帰分析の当てはめ値と残差は fitted.values、residuals で返す。

分類器のパフォーマンスの検証には、通常交差確認法を用いるが、パッケージ nnet には交差確認の機能を備えていないのが残念である。

5. WEKA によるニューラルネットワーク

WEKA には、階層型ニューラルネットワークが MultilayerPerceptron と言う名前で実装されている。MultilayerPerceptron は、Weka Explorer パネルの Choose ボタンを押し、Classify ⇒ functions ⇒ MultilayerPerceptron をクリックすることで読み込まれる。図 6 に MultilayerPerceptron を読み込んだ画面コピーを示す。

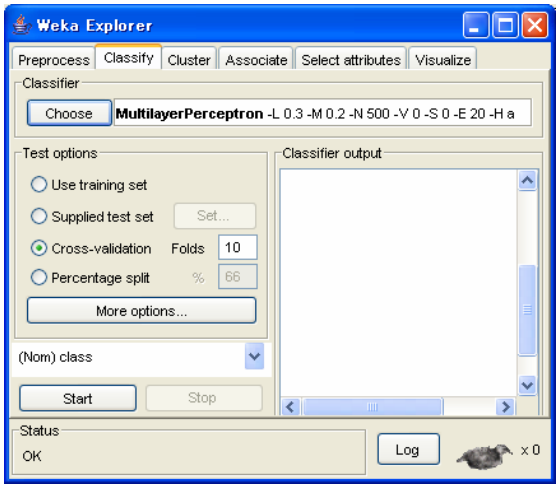


図 6　Weka Explorer パネル画面

データ iris の交差確認(n=10)の実行結果の一部分を図 7 に示す。ただし、用いたパラメータは図 8 に示すデフォルト値を用いた。

Weka Explorer パネルの左上部にある Choose ボタンの右の文字列 MultilayerPerceptron…部分をクリックすると図 8 のようなオブジェクト編集パネルが開かれる。この画面で、デフォルト値を換えることができる。編集パネルの More ボタンを押すと図 8 の項目などに関する説明のテキスト画面が開かれる。

オブジェクト編集パネルの GUI を True に指定し、実行すると図 9 のようなニューラルネットワーク構造の図示画面が開かれる。

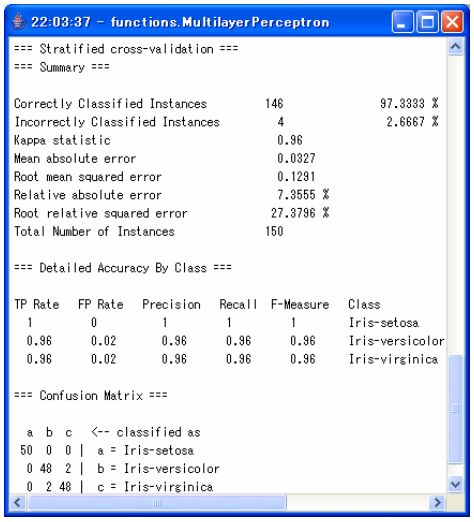


図 7　iris の交差確認の結果画面

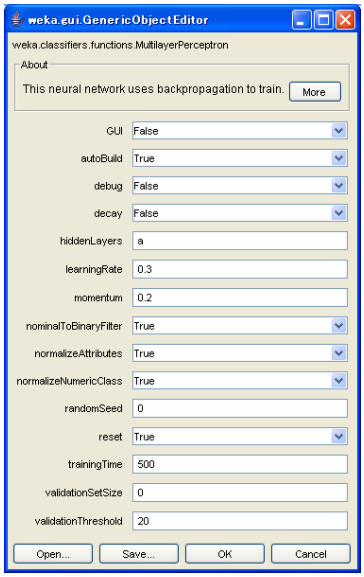


図 8　オブジェクト編集パネル

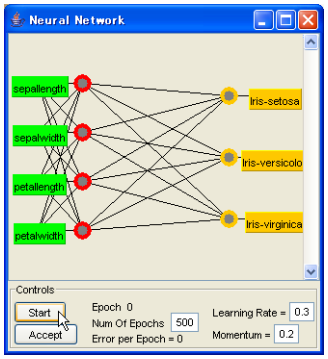


図 9　iris のニューラルネットワーク画面