06t4027h 斎藤高行

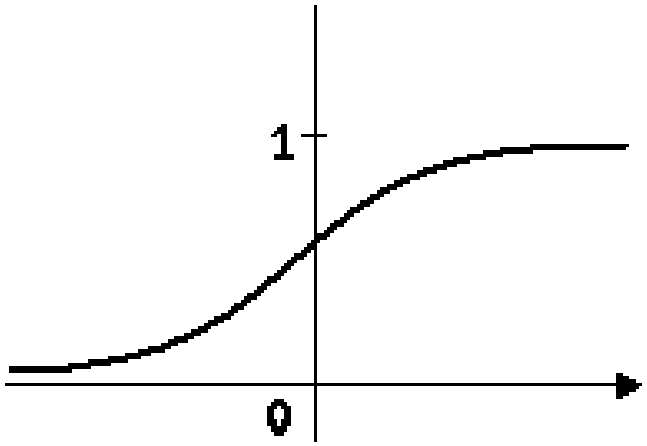
2章 ニューラルネット

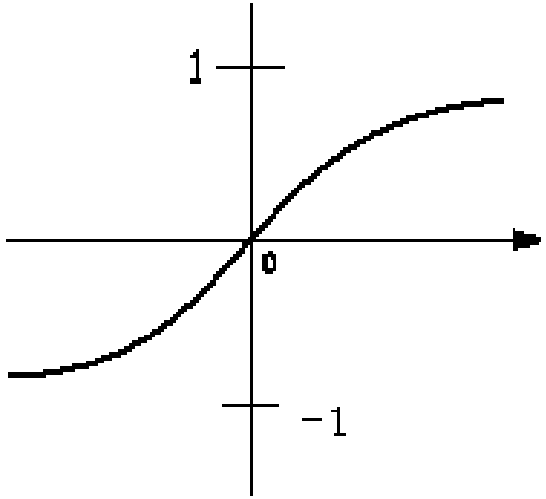
2.4 さまざまなオプション

2.5 Rによるニューラルネット

ユニット内の変換の場合

ニューラルネットのユニット内変換の場合、微分可能な関数であればよい

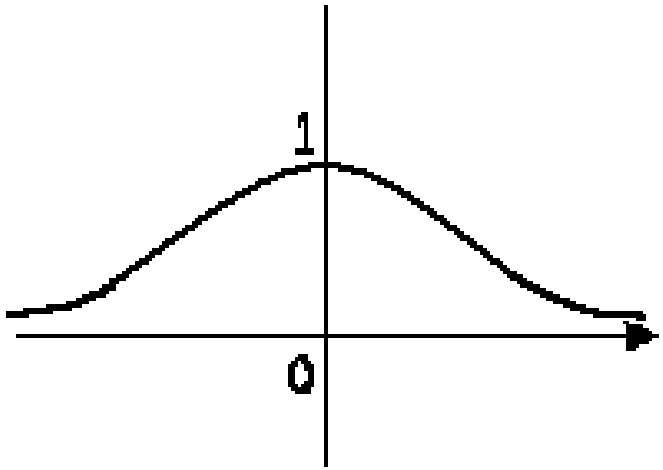




シグモイド変換

ものの数え方などの場合

ものの数え方など、高次の交互作用が存在する課題の学習には、ガウシアン関数が適している事が多い



ガウシアン関数

感度分析

ネットワークモデルにおける感度分析…

個々の予測変数の相対的な重要度を評価

モデルの予測に対してどの変数がどれだけ貢献しているかを見積もる

感度分析

＜感度分析の基本的な流儀＞

1.基準変数のレンジ(範囲)を求める。

2.予測変数以外の変数を平均値に固定し、予測変数を最小値から最大値まで変化させたときの基準変数のレンジを求める。

3.1と2のレンジの比較を、予測変数の重要度とする。

基準変数のスケーリング

モデルの何百という数に対し、ユニットの出力は0から1の範囲…おぼつかない予測

基準変数のスケーリング

基準変数のスケーリング

学習データの基準変数:

$$学習用教師信号 = \frac{基準変数 - 最小値}{最大値 - 最小値}$$

上記の通りに変換し、0から1の範囲に収める

ネットワークの出力:

$$予測値 = 出力 \times (最大値 - 最小値) + 最小値$$

予め保存しておいた推定用データを利用する

予測変数のスケーリング

予測変数の分散が大きくなる場合、それに対応する第1層の重みの絶対値を小さくする必要がある。

このときの調整で、第2層以降で局所最適解にはまり込む危険が大きい

予測変数のスケーリング

予測変数のスケーリング

基準変数と同様に設定する

$$\text{入力信号} = \frac{\text{予測変数} - \text{最小値}}{\text{最大値} - \text{最小値}}$$

最大値と最小値は定数であり、推定用データから計算される

プルーニング(枝刈り、剪定)

プルーニングとは

ニューラルネットモデルにおいて、逆伝播する誤差や重みの性質を調べ、最終層への影響の少ないユニットを訓練中に破棄するオプションのこと

・メインテナンスするモデルのサイズを小さくしてシステムを効率よく運用することができる

・同時に、学習データに対する過学習を防止する役割も果たす

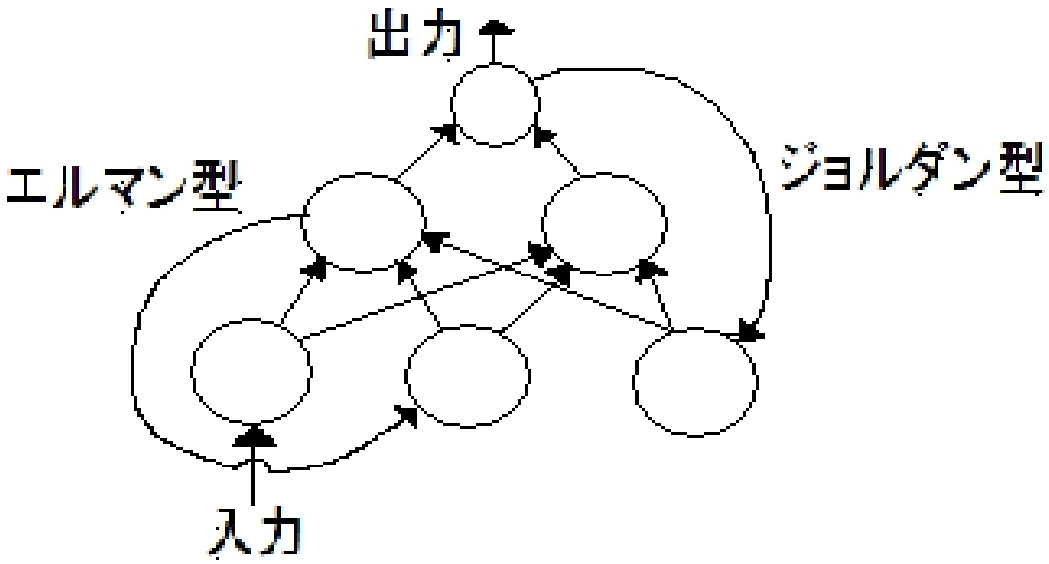
＝交差妥当化によるモデルの単純化と同じ目的で使用できる

その他のネットワークモデル

リカレント型：

上層から一部の情報が戻ってくるトポロジー

- エルマン型：隠れ層からフィードバックするモデル
- ジョルダン型：出力層からフィードバックするモデル

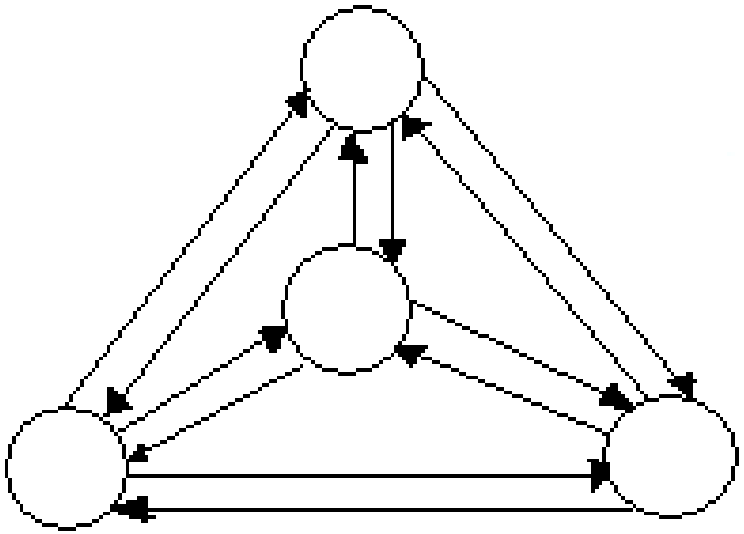


その他のネットワークモデル

相互結合型:

情報の伝達が双方向のトポロジー

原理的に入出力ユニットの区別はなく、教師信号もない



Rによるニューラルネット

Rによる分析

・偽札データの分析

・鉛筆の本数の数え方を学習するモデル

偽札データの分析

お札の判別モデルのネットワークの設定

・入力層ユニットに「下部マージン幅」と「絵の対角線の長さ」

・出力層ユニットに「お札の真偽」

・中間層にユニット3つ

偽札データの分析

パッケージnnetを使ったニューラルネットモデル学習法

・パッケージとデータ読み込み

```
library(nnet)
お札データ<-read.csv(“お札データ.csv”,header=T)
```

偽札データの分析

・下部マージンと対角線の変数をスケーリング

```
下部マージン<-(お札データ$下部マージン
    –min(お札データ$下部マージン))
    /(max(お札データ$下部マージン)
    -min(お札データ$下部マージン))

対角線<-(お札データ$対角線-min(お札データ$対角線))/
    (max(お札データ$対角線)-min(お札データ$対角線))
```

偽札データの分析

・入力層と出力層ユニットの指定

```
入力層<-cbind(下部マージン,対角線)
出力層<-class.ind(お札データ$真偽)
```

Cbind():

ベクトルや行列を列ベクトル単位で結合し、新たな行列を作る関数

Class.ind():

真偽の変数を、偽札か否かを表すダミー変数行列と、真札か否かを表すダミー変数行列に変換するもの

偽札データの分析

・変換後のダミー変数行列を確認

```
>head(適用結果,n=3)
     偽札     真札
[1,] 1       0
[2,] 1       0
[3,] 1       0
>tail(適用結果,n=3)
     偽札     真札
[213,] 0     1
[214,] 0     1
[215,] 0     1
```

偽札データの分析

・ネットワークの学習

```
set.seed(3)
ネットワーク学習<-nnet(入力層,出力層,size=3,rang=0.3,
maxit=2000)
```

nnet():階層型ネットワークの学習

size:中間層のユニット数

rang:重みの初期値の範囲

maxit:学習の最大繰り返し数

set.seed:初期値シートの固定化

偽札データの分析

・判別ルールを学習したかどうかの確認

```
ネットワーク適用<-predict(ネットワーク学習,入力層)
適用結果<-round(ネットワーク適用,digit=3)
```

predict():ネットワーク適用

round():小数点以下で数値を丸める(digitで桁数を指定)

偽札データの分析

・ネットワーク適用後の結果を確認

```
>head(適用結果,n=3)
        偽札        真札
[1,] 0.833        0.167
[2,] 0.833        0.167
[3,] 1.000        0.000
>tail(適用結果,n=3)
        偽札        真札
[213,] 0          1
[214,] 0          1
[215,] 0          1
```

偽札データの分析

出力結果は、偽札である予測確率と、真札である予測確率を返している

元のデータに近い値を返しているので、ネットワーク判断はただしいといえる

鉛筆の数え方データの分析

ネットワーク

・入力層ユニットに、1から10の「本数」

・出力層ユニットに、「ポン」「ホン」「ボン」

・中間層は一層

鉛筆の数え方データの分析

・パッケージとデータの読み込み

```
library(neural)
鉛筆データ<-read.csv(“鉛筆.csv”,header=T)
```

鉛筆の数え方データの分析

・入力層と出力層データの指定
(ここでは出力層ユニットを「ポン」に指定)
・それぞれ、行列形式に変換する

```
入力層<-as.matrix(鉛筆データ$本数)
出力層<-as.matrix(鉛筆データ$ポン)
```

as.matrix():行列形式に変換する関数

鉛筆の数え方データの分析

・GUIを通したネットワークの分析

(出力は省略…テキストP75)

```
set.seed(1)
ネットワーク学習<-rbftrain(入力層,neurons=10,出力層,it=1000)
rbftrain()
```

neurons:中間層ユニット数

it:学習の最大繰り返し数

鉛筆の数え方データの分析

・ネットワークの適用

```
ネットワーク適用<-rbf(入力層,ネットワーク学習$weight,
    ネットワーク学習$dist,ネットワーク学習$neurons,
    ネットワーク学習$sigma)
適用結果<-round(ネットワーク適用,dights=1)
```

rbf():rbttrain()の学習結果を呼び出す関数

weight:ネットワーク重み　dist:歪み

neurons:中間層のユニット数

sigma:ガウシアン関数の分布幅

鉛筆の数え方データの分析

>適用結果

```
       [1,]
[1,]   0.9
[2,]   0.1
[3,]   0.1
[4,]   0.1
[5,]   0.1
[6,]   0.9
[7,]   0.1
[8,]   0.9
[9,]   0.1
[10,]  0.9
```

鉛筆の数え方データの分析

適用結果では

1, 6, 8, 10 のときに「ポン」と数える予測確率が高くなっている。

＝学習がきちんと行われている