

統計的モデリング基礎⑤

～ロジスティック回帰とニューラルネットワーク～

鹿島久嗣
(情報学科 計算機科学コース)

マーケティング分野への応用を対象とした参考書



マーケティングの統計モデル

出版社：朝倉出版

発刊年月：2015.8

ISBN：4254128533

A5判；192ページ

マーケティングを題材としながら、基本的な統計的モデリングの方法が学べる

ロジスティック回帰

最尤推定：

データをもっともよく再現するパラメータを推定値とする

- n 個のデータ x_1, x_2, \dots, x_n から確率モデル $f(x | \theta)$ のパラメータ θ を推定したい

- n 個のデータが（互いに独立に）生成される確率（尤度）：

$$L(\theta) = \prod_{i=1}^n f(x_i | \theta)$$

実際には対数尤度で扱うことが多い

- 尤度最大になるパラメータを推定値 $\hat{\theta}$ とする

$$\hat{\theta} = \operatorname{argmax}_{\theta} \prod_{i=1}^n f(x_i | \theta) = \operatorname{argmax}_{\theta} \sum_{i=1}^n \log f(x_i | \theta)$$

- もっともデータを生成する確率が高い（「最も尤もらしい」）

最尤推定の利点： モデリングの自動化

- 最尤推定の利点：
確率モデルの形（データの生成プロセスの仮定）を決めればモデルパラメータが自動的に決まる
 - ただし、最大化問題を解く必要がある
 - 離散分布、ポアソン分布、正規分布などは解析的に解が求まる
 - 線形回帰（正規分布でノイズが載る）は連立方程式（一応、解析的な解）
 - 多くのモデルでは、最適化問題を数値的に解く必要がある

判別問題：

ダミー変数を従属変数として説明（予測）する問題

- データ（ n 組の独立変数と従属変数）
 - 独立変数： $(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(n)})$
 - (ダミー) 従属変数： $(y^{(1)}, y^{(2)}, \dots, y^{(n)}), y^{(i)} \in \{+1, -1\}$

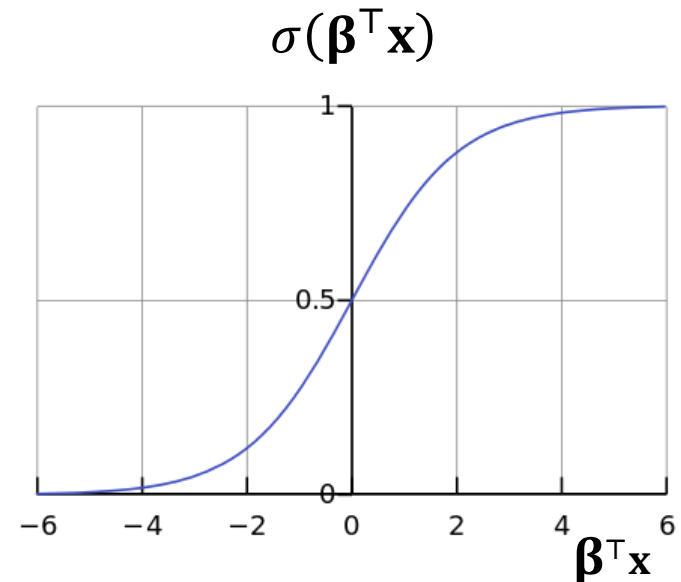
以降、表記上の利便性からダミー従属変数を
 $\{0, 1\}$ でなく $\{+1, -1\}$ と表記する
(本質的な違いはナシ)

ロジスティック回帰： ダミー変数を従属変数とするモデル

- 以前、重回帰モデルでダミー変数を従属変数とすると、厳密には少しおかしいという話だった → もっときちんと扱いたい
 - 重回帰モデル $y = \boldsymbol{\beta}^T \mathbf{x}$ の従属変数の値域は実数全体
- 従属変数の値域が $\{-1, +1\}$ もしくは $(0, 1)$ ($Y = +1$ となる確率) となるようにしたい
- ロジスティック回帰モデル：

$$P(Y = 1 | \mathbf{x}, \boldsymbol{\beta}) = \frac{1}{1 + \exp(-\boldsymbol{\beta}^T \mathbf{x})} = \sigma(\boldsymbol{\beta}^T \mathbf{x})$$

- σ : ロジスティック関数 ($\sigma: \mathbb{R} \rightarrow (0, 1)$)

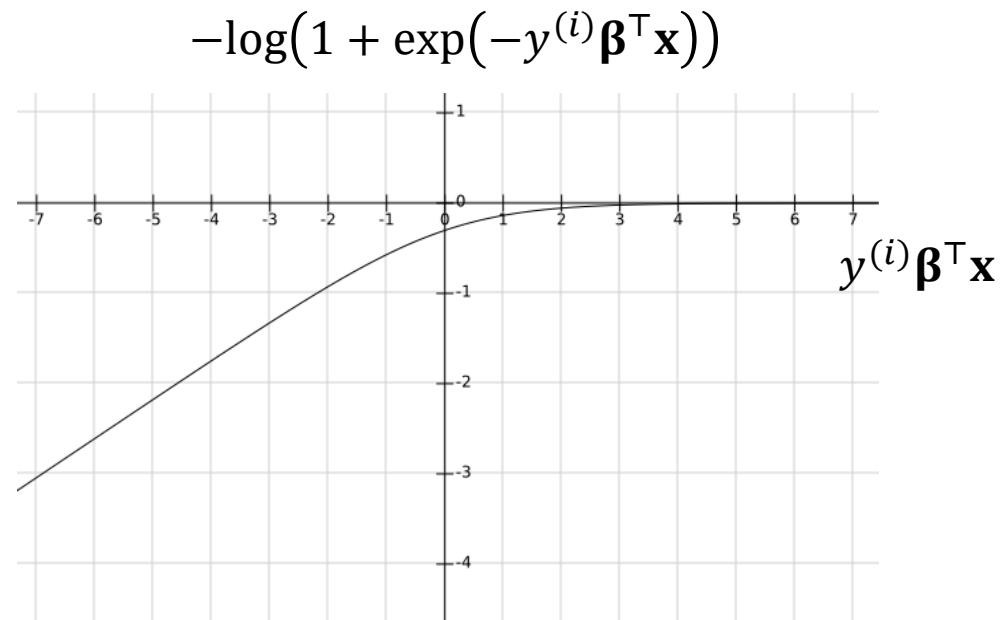


ロジスティック回帰モデルの対数尤度： 凸関数なので大局解が存在するが解析解はない

■ 対数尤度： $L(\boldsymbol{\beta}) = -\sum_{i=1}^n \log(1 + \exp(-y^{(i)} \boldsymbol{\beta}^\top \mathbf{x}^{(i)}))$
 $\left(= \sum_{i=1}^n \delta(y^{(i)} = 1) \log \frac{1}{1 + \exp(-\boldsymbol{\beta}^\top \mathbf{x}^{(i)})} + \delta(y^{(i)} = -1) \log \left(1 - \frac{1}{1 + \exp(-\boldsymbol{\beta}^\top \mathbf{x}^{(i)})} \right) \right)$

■ $L(\boldsymbol{\beta})$ は凸関数：

- 大局解がある
- 解析解はない



ロジスティック回帰のパラメータ推定： 非線形最適化法によって、パラメータ更新を繰り返す

- 最尤推定の目的関数（最大化）：

$$L(\boldsymbol{\beta}) = - \sum_{i=1}^n \log(1 + \exp(-y^{(i)} \boldsymbol{\beta}^\top \mathbf{x}^{(i)}))$$

- 解析解は得られないが、凸関数（2階微分が ≤ 0 ）
- 数値的な最適化手法を使う
 - パラメータの更新をくりかえす： $\boldsymbol{\beta}^{\text{NEW}} \leftarrow \boldsymbol{\beta} + \mathbf{d}$



パラメータの更新：

目的関数をもっとも改善するような更新を行う

- 更新 $\boldsymbol{\beta}^{\text{NEW}} \leftarrow \boldsymbol{\beta} + \mathbf{d}$ によって目的関数の値が変化する：

$$L_{\mathbf{w}}(\mathbf{d}) = - \sum_{i=1}^n \ln(1 + \exp(-y^{(i)} (\boldsymbol{\beta} + \mathbf{d})^{\top} \mathbf{x}^{(i)}))$$

- $L_{\boldsymbol{\beta}}(\mathbf{d})$ を最大化する更新差分 \mathbf{d}^* を見つけよ：

$$\mathbf{d}^* = \operatorname{argmax}_{\mathbf{d}} L_{\boldsymbol{\beta}}(\mathbf{d})$$

最良のパラメータ更新差分の決定： 目的関数をテイラー展開で2次近似

- 目的関数のテイラー展開：

$$L_{\beta}(\mathbf{d}) = L(\beta) + \mathbf{d}^{\top} \nabla L(\beta) + \frac{1}{2} \mathbf{d}^{\top} \mathbf{H}(\beta) \mathbf{d} + O(\mathbf{d}^3)$$

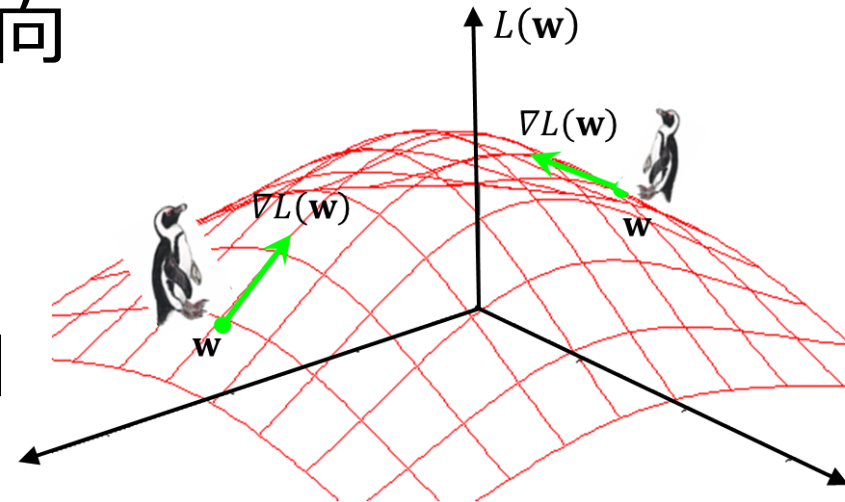
3次以上の項

- 勾配： $\nabla L(\beta) = \left(\frac{\partial L(\beta)}{\partial \beta_1}, \frac{\partial L(\beta)}{\partial \beta_2}, \dots, \frac{\partial L(\beta)}{\partial \beta_D} \right)^{\top}$

- β において目的関数が最も急な方向

- ヘッセ行列： $[H(\beta)]_{i,j} = \frac{\partial^2 L(\beta)}{\partial \beta_i \partial \beta_j}$

- β 周辺での目的関数の「曲がり方」



ニュートン法：

2次近似した目的関数を最小化する解を求める

- テイラー展開で3次以降の項を無視する：

3次以上の項

$$L_{\beta}(\mathbf{d}) \approx L(\beta) + \mathbf{d}^{\top} \nabla L(\beta) + \frac{1}{2} \mathbf{d}^{\top} \mathbf{H}(\beta) \mathbf{d} + \cancel{O(\mathbf{d}^3)}$$

- 最大化するために \mathbf{d} で微分： $\frac{\partial L_{\beta}(\mathbf{d})}{\partial \mathbf{d}} \approx \nabla L(\beta) + \mathbf{H}(\beta) \mathbf{d}$

- これを $= \mathbf{0}$ とおいて解くと： $\mathbf{d} = -\mathbf{H}(\beta)^{-1} \nabla L(\beta)$

実際には連立方程式を解く

- ニュートン法：

$$\beta^{\text{NEW}} \leftarrow \beta - \mathbf{H}(\beta)^{-1} \nabla L(\beta)$$



線形探索付きニュートン法：

近似は厳密には正しくないので線形探索と組み合わせる

- ニュートン法の更新 $\boldsymbol{\beta}^{\text{NEW}} \leftarrow \boldsymbol{\beta} - \mathbf{H}(\boldsymbol{\beta})^{-1} \nabla L(\boldsymbol{\beta})$ は2次近似が正しいことを仮定している：

$$L_{\boldsymbol{\beta}}(\mathbf{d}) \approx L(\boldsymbol{\beta}) + \mathbf{d}^{\top} \nabla L(\boldsymbol{\beta}) + \frac{1}{2} \mathbf{d}^{\top} \mathbf{H}(\boldsymbol{\beta}) \mathbf{d}$$

- 近似なので、厳密には正しくない
- そこで、更新の向きのみを採用して、更新の量 η は別途決める：
 $\boldsymbol{\beta}^{\text{NEW}} \leftarrow \boldsymbol{\beta} - \eta \mathbf{H}(\boldsymbol{\beta})^{-1} \nabla L(\boldsymbol{\beta})$
- 更新の量（学習率） $\eta > 0$ の決定法：
 - ステップ数とともに適当に減衰させる
 - あるいは、線形探索： $\eta^* = \operatorname{argmax}_{\eta} L(\boldsymbol{\beta} - \eta \mathbf{H}(\boldsymbol{\beta})^{-1} \nabla L(\boldsymbol{\beta}))$

適当な初期値から始めて、目的関数が改善しない間は η を半分にしていく

最急降下法*:

ヘッセ行列を使わずに、シンプルで軽い更新を繰り返す

- ヘッセ行列の逆行列（もしくは連立方程式を解く）は高コスト：

- ニュートン法の更新： $\beta^{\text{NEW}} \leftarrow \beta - \eta H(\beta)^{-1} \nabla L(\beta)$

- 最急降下法：

単位行列

- ヘッセ行列の逆行列 $H(\beta)^{-1}$ を $-I$ で置き換える：

$$\beta^{\text{NEW}} \leftarrow \beta + \eta \nabla L(\beta)$$

- 勾配 $\nabla L(\beta)$ は最も急な（目的関数が最も増加する）向き
- 学習率 η は線形探索で求める：



確率的最適化とミニバッチ学習： データの部分集合を用いた効率的な推定

■ 対数尤度は各データの対数尤度の和： $L(\boldsymbol{\beta}) = \sum_{i=1}^n \ell^{(i)}$

■ 勾配 $\frac{\partial L(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = \sum_{i=1}^n \frac{\partial \ell^{(i)}}{\partial \boldsymbol{\beta}}$ の計算は $O(n)$ にかかる

i 番目のデータの
対数尤度

■ 勾配をデータ1個で近似： $\frac{\partial L(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} \approx n \frac{\partial \ell^{(i)}}{\partial \boldsymbol{\beta}}$

● 確率的最適化：毎回データをランダムに選ぶ

● オンライン推定も可能（時刻 t のデータの $\ell^{(t)}$ を使う）

■ ミニバッチ学習： $1 < m < n$ 個のデータで勾配を近似：

$$\frac{\partial L(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} \approx \frac{n}{m} \sum_{j \in \text{MiniBatch}} \frac{\partial \ell^{(i)}}{\partial \boldsymbol{\beta}}$$

ロジスティック回帰の勾配計算： 比較的簡単に計算可能

- 対数尤度： $L(\boldsymbol{\beta}) = -\sum_{i=1}^n \ln(1 + \exp(-y^{(i)} \boldsymbol{\beta}^\top \mathbf{x}^{(i)}))$

- $$\begin{aligned} \frac{\partial L(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} &= -\sum_{i=1}^n \frac{1}{1 + \exp(-y^{(i)} \boldsymbol{\beta}^\top \mathbf{x}^{(i)})} \frac{\partial (1 + \exp(-y^{(i)} \boldsymbol{\beta}^\top \mathbf{x}^{(i)}))}{\partial \boldsymbol{\beta}} \\ &= \sum_{i=1}^n \frac{1}{1 + \exp(-y^{(i)} \boldsymbol{\beta}^\top \mathbf{x}^{(i)})} \exp(-y^{(i)} \boldsymbol{\beta}^\top \mathbf{x}^{(i)}) y^{(i)} \mathbf{x}^{(i)} \\ &= \sum_{i=1}^n (1 - f(y^{(i)} | \mathbf{x}^{(i)}, \boldsymbol{\beta})) y^{(i)} \mathbf{x}^{(i)} \end{aligned}$$

現在のパラメータでのモデルが与える確率

練習問題： ポアソン回帰の最尤推定

- (前回出てきた) ポアソン回帰の最尤推定

- 対数尤度：

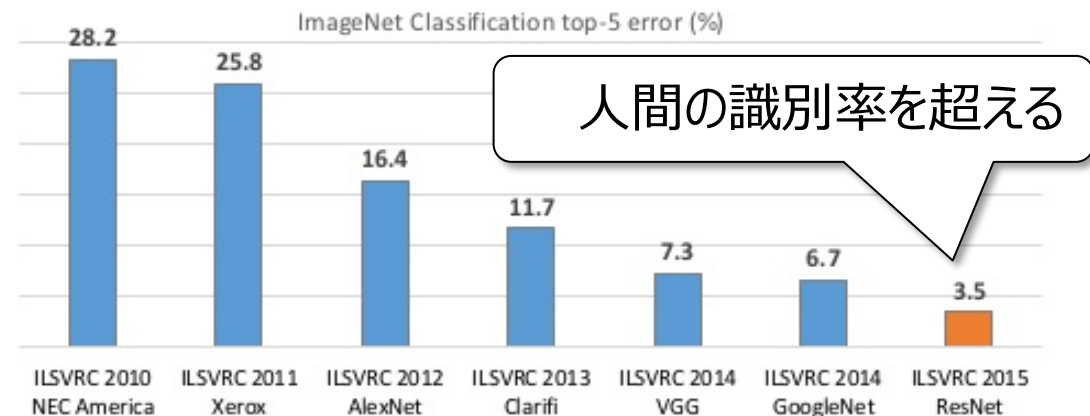
$$L(\boldsymbol{\beta}) = \sum_{i=1}^n y^{(i)} \boldsymbol{\beta}^\top \mathbf{x}^{(i)} - \sum_{i=1}^n \exp(\boldsymbol{\beta}^\top \mathbf{x}^{(i)}) + \text{const.}$$

- 解析解は求まらない
- 最急勾配法の更新式を求めてみる

ニューラルネットワーク

深層学習（ディープラーニング）の出現： 機械による認識の大幅な精度向上

- ニューラルネットワーク：
1980年代に盛んに研究がされていたがその後下火に
- 画像識別で10%以上の記録更新、一躍注目を浴びる
→ 畳み込みニューラルネットがデファクト・スタンダードに
- GAFAを筆頭に数々の企業が深層学習に大きな投資
- 研究・開発のトレンドも
深層学習が中心に



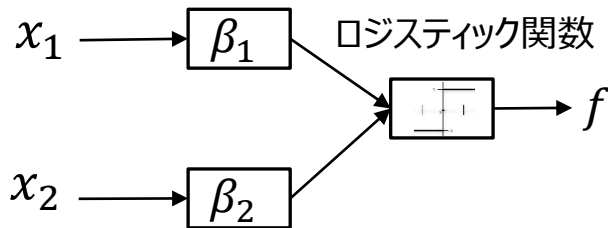
Microsoftの発表資料より抜粋

ニューラルネットワーク：

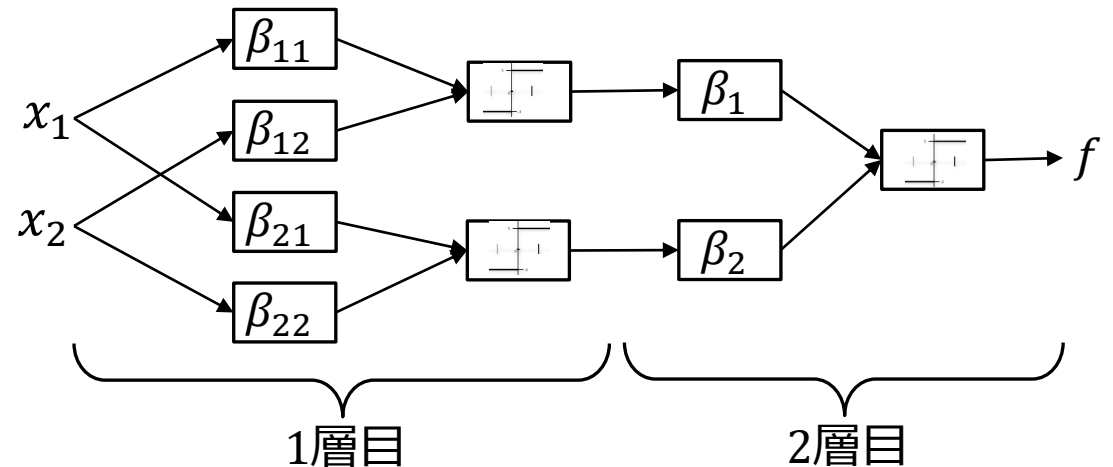
(ざっくりいえば) ロジスティック回帰モデルを連結したものの

- ニューラルネットワークはロジスティック回帰モデルを連結したものの
 - 複数のロジスティック回帰モデルの出力が、別のロジスティック回帰モデルの入力になる
 - ロジスティック関数（非線形）によりモデルに非線形性を導入
 - 両者ともに、 $y = +1$ である確率 $f(\mathbf{x}; \boldsymbol{\beta})$ を出力するモデル

ロジスティック回帰モデル



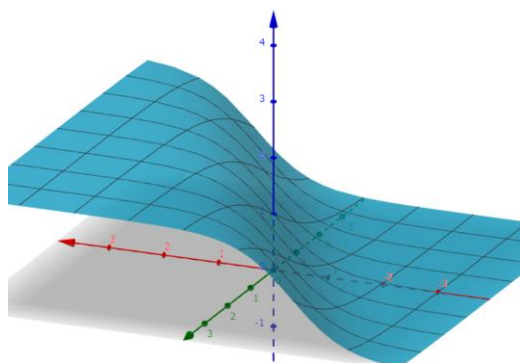
ニューラルネットワーク (2層)



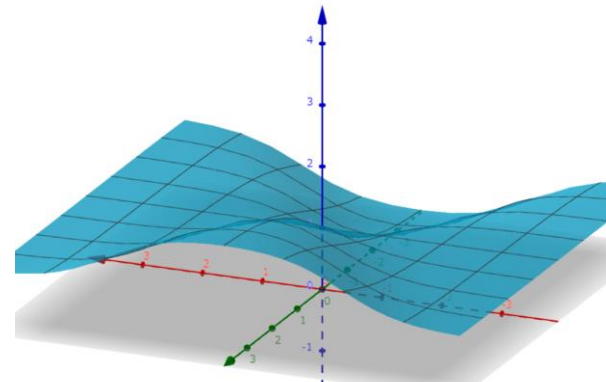
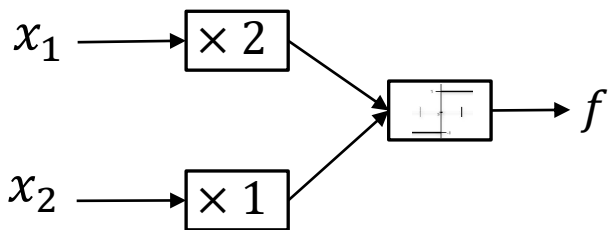
ニューラルネットワークの非線形性の例：

ロジスティック回帰を2層積むと非線形分類が可能

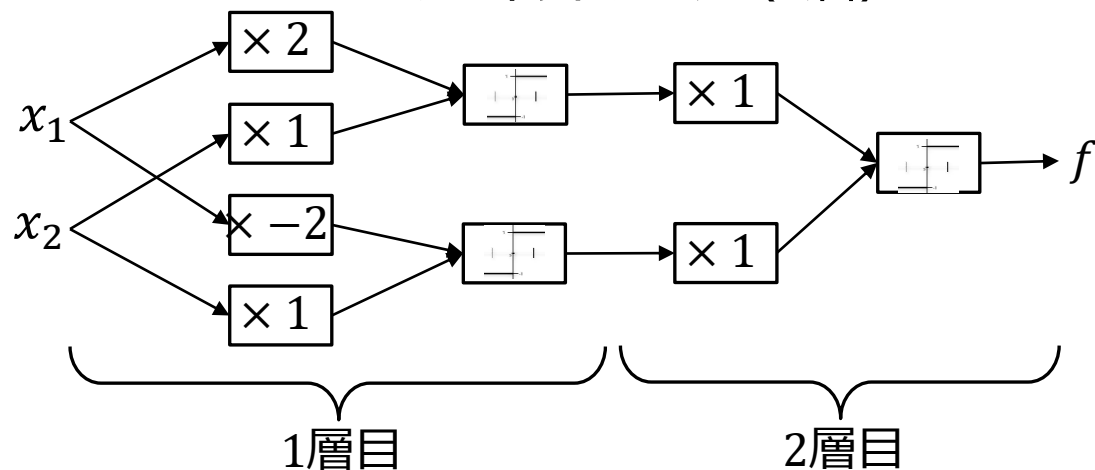
- ロジスティック回帰は1層では線形判別しかできない (AND/OR)
- 2層以上積むことで非線形の表現力を獲得 (XOR)



ロジスティック回帰モデル



ニューラルネットワーク (2層)



ニューラルネットワークのパラメータ推定：

最急降下法を適用するために勾配の計算が必要

- 対数尤度関数 $L(\boldsymbol{\beta})$ を最大化するパラメータ $\boldsymbol{\beta}$ を求める：

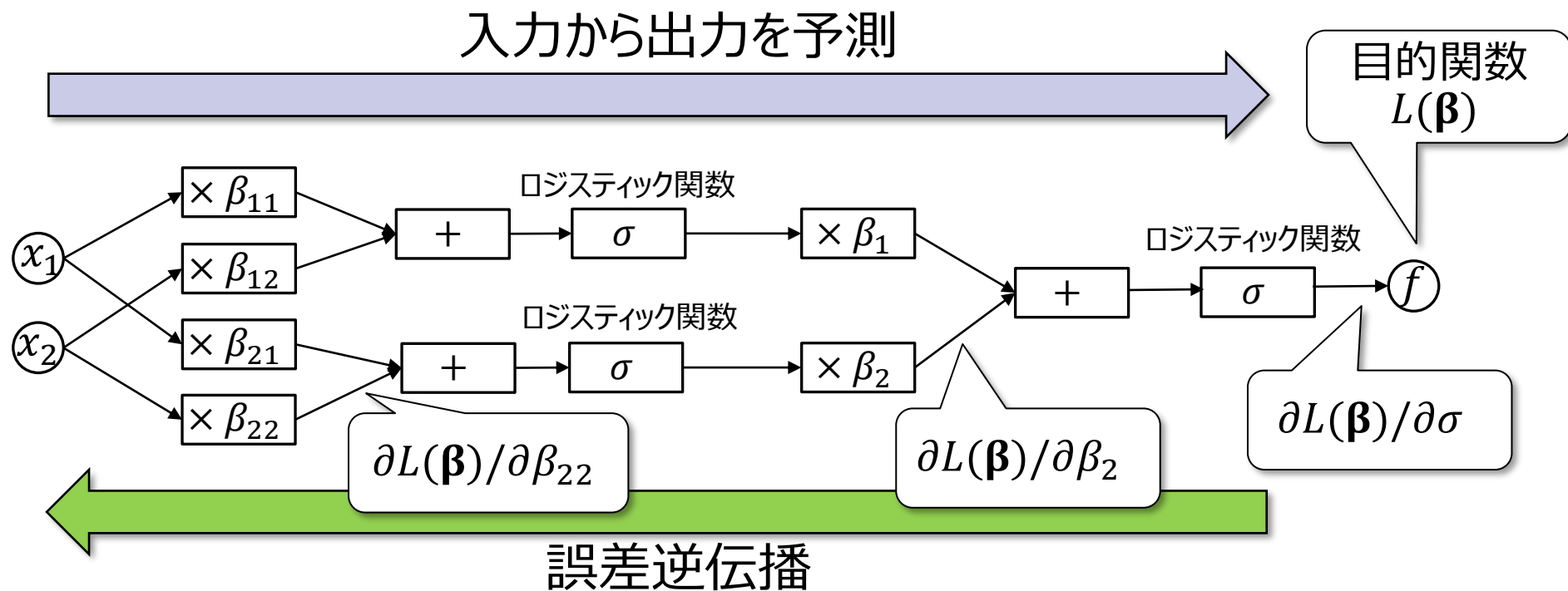
$$L(\boldsymbol{\beta}) = - \sum_{i=1}^n \left(\delta(y^{(i)} = 1) \log f(x^{(i)}) + \delta(y^{(i)} = -1) \log (1 - f(x^{(i)})) \right)$$

- $f(x^{(i)})$ は $x^{(i)}$ に対するニューラルネットの出力 ($y^{(i)} = 1$ である確率)
- 勾配 $\nabla L(\boldsymbol{\beta}) = \frac{\partial L(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}}$ が計算できれば最急降下法を適用できる：
$$\boldsymbol{\beta}^{\text{NEW}} \leftarrow \boldsymbol{\beta} + \eta \nabla L(\boldsymbol{\beta})$$
 - 実際は確率的最適化やミニバッチを用いることも多い

ニューラルネットワークのパラメータ推定法：

誤差逆伝播法（自動微分）による勾配法でパラメータ推定

- 対数尤度 $L(\boldsymbol{\beta})$ をパラメータで微分できれば勾配法で推定できる
- 誤差逆伝播法（自動微分）：層を遡って微分計算



誤差逆伝播法：

勾配を再帰的に効率的に計算できる

- 誤差逆伝播法： $L(\boldsymbol{\beta})$ の勾配 $\nabla L(\boldsymbol{\beta}) = \frac{\partial L(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}}$ を計算する方法

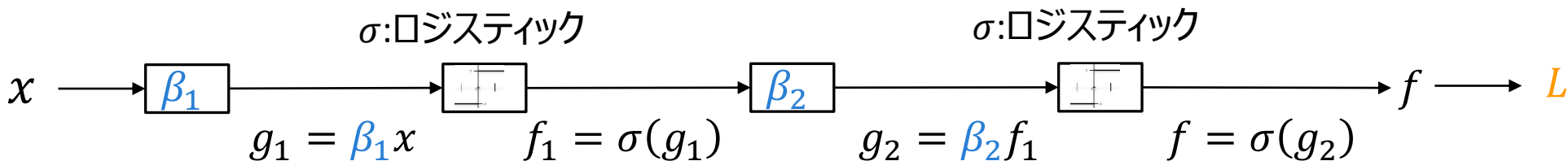
- 1次元の場合の例：

- f から「後ろ向きに」遡っていく計算（微分の連鎖率）

$$\bullet \frac{\partial L}{\partial \beta_2} = \frac{\partial L}{\partial f} \cdot \frac{\partial f}{\partial g_2} \cdot \frac{\partial g_2}{\partial \beta_2}$$

$$\bullet \frac{\partial L}{\partial \beta_1} = \frac{\partial L}{\partial f} \cdot \frac{\partial f}{\partial g_2} \cdot \frac{\partial g_2}{\partial f_1} \cdot \frac{\partial f_1}{\partial g_1} \cdot \frac{\partial g_1}{\partial \beta_1}$$

共通なので層をまたいで使いまわし可能



$$L(\boldsymbol{\beta}) = - \sum_{i=1}^n \left(\delta(y^{(i)} = 1) \log f(x^{(i)}) + \delta(y^{(i)} = -1) \log (1 - f(x^{(i)})) \right)$$

まとめ：

ロジスティック回帰とニューラルネットワーク

■ ロジスティック回帰：

- ダミー変数 $y \in \{+1, -1\}$ を従属変数とするモデル
 - $y = +1$ である確率を出力する
- 最尤推定対数尤度は、大域解をもつが、解析解をもたない
- 非線形最適化法によって、最適解を求める
 - ニュートン法、再急降下法、確率的勾配法、...

■ ニューラルネットワーク：

- (乱暴に言えば) ロジスティック回帰の多層化
- (ロジスティック回帰と異なり) 非線形の識別が可能
- 誤差逆伝播法によって、効率的に勾配が計算可能