Kyoto University

統計的モデリング基礎⑥
～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）

Department of Intelligence Science and Technology

ロジスティック回帰

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：$\left(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(n)}\right)$
  - （ダミー）従属変数：$\left(y^{(1)}, y^{(2)}, \dots, y^{(n)}\right), y^{(i)} \in \{+1, -1\}$

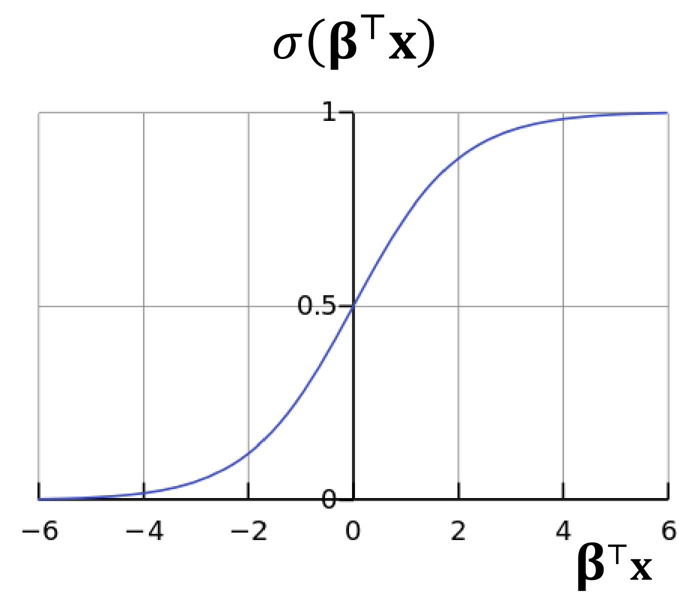
以降、表記上の利便性からダミー従属変数を$\{0,1\}$ でなく$\{+1, -1\}$と表記する
（本質的な違いはナシ）

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

- 従属変数の値域が$\{-1, +1\}$となるモデル
- ロジスティック回帰モデルは$Y = +1$となる確率を与える：

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

- σ：ロジスティック関数（$\sigma: \mathbb{R} \to (0,1)$）
  - 実数を確率に変換する



ロジスティック回帰のパラメータ推定：
パラメータの繰り返し更新によって最適解を求める

- 最尤推定の目的関数（最大化）：

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

  - = 負の対数尤度を目的関数として最小化

- 数値的な最適化手法によりパラメータを推定する

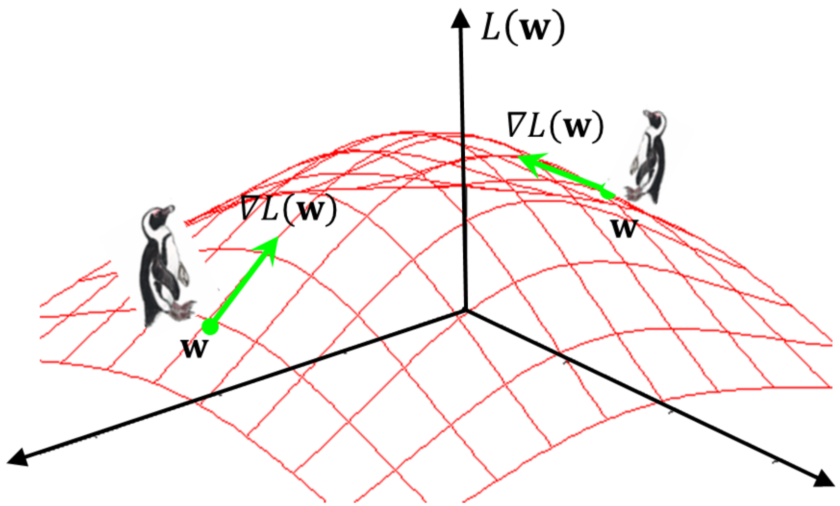
- パラメータの更新をくりかえす： $\boldsymbol{\beta}^{\mathrm{NEW}} \leftarrow \boldsymbol{\beta} + \mathbf{d}$

最急降下法※
最もシンプルな最適化手法

- 最急降下法：

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

  - 勾配$\nabla L(\boldsymbol{\beta})$ 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：



※ 最急降下法の名前は、最小化問題を想定しているが、ここでは最大化問題としているので降下じゃなくて上昇しているが、便宜上こう呼ぶことにする

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

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

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

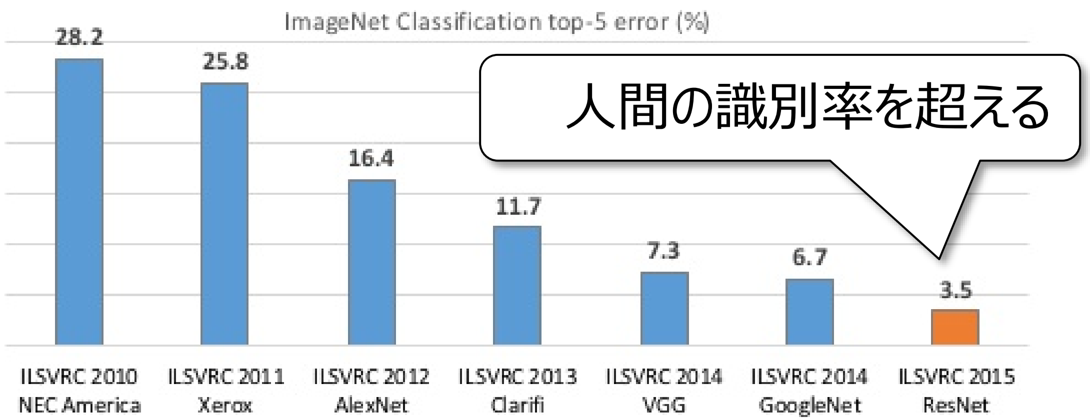
$$= \sum_{i=1}^{n} \left(1 - f(y^{(i)}|\mathbf{x}^{(i)}, \boldsymbol{\beta})\right) y^{(i)}\mathbf{x}^{(i)}$$

現在のパラメータでのモデルが与える確率

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

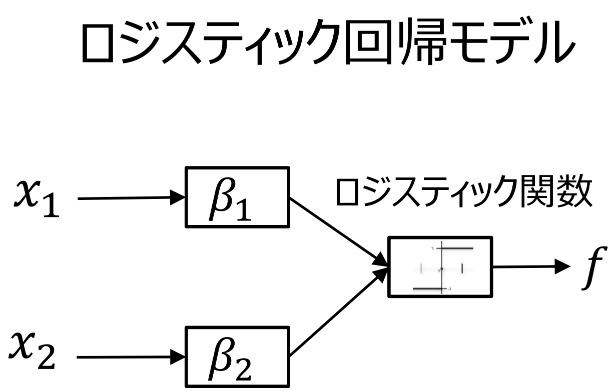
- ニューラルネットワーク：
  1980年代に盛んに研究がされていたがその後下火に
- 画像識別で10%以上の記録更新、一躍注目を浴びる
  → 畳み込みニューラルネットがデファクト・スタンダードに
- GAFAを筆頭に数々の企業が深層学習に大きな投資
- 研究・開発のトレンドも
  深層学習が中心に

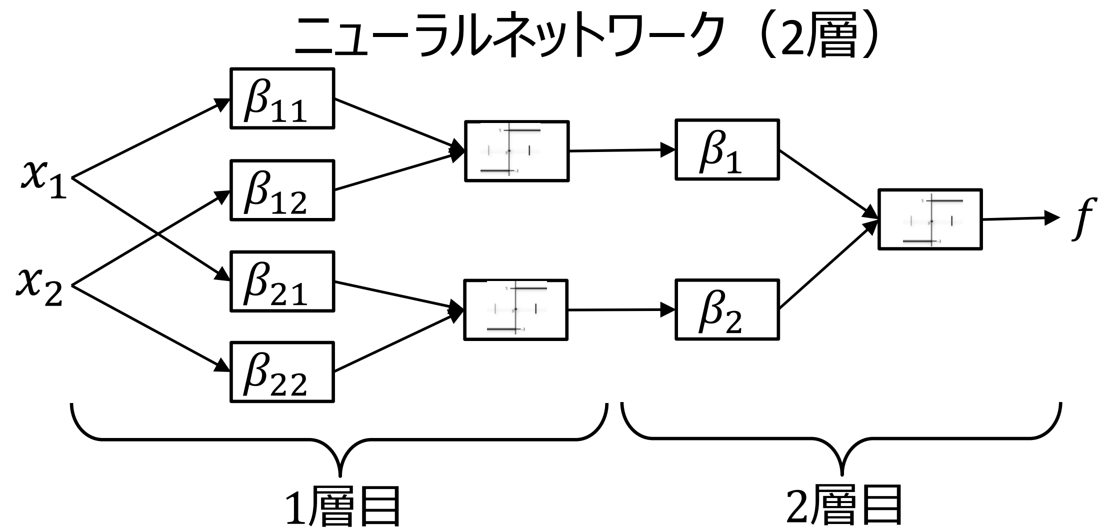


Microsoftの発表資料より抜粋

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

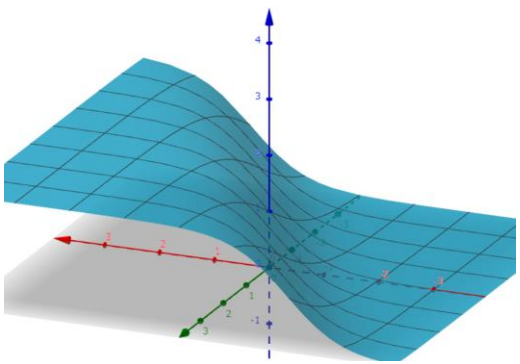
- ニューラルネットワークはロジスティック回帰モデルを連結したもの
  - 複数のロジスティック回帰モデルの出力が、別のロジスティック回帰モデルの入力になる
  - ロジスティック関数（非線形）によりモデルに非線形性を導入
  - 両者ともに、$y = +1$である確率$f(\mathbf{x}; \boldsymbol{\beta})$を出力するモデル



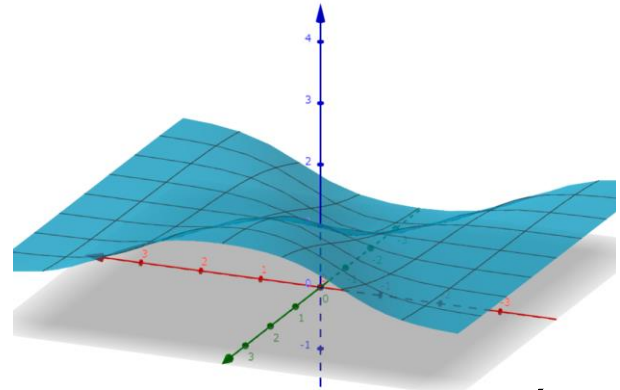


ニューラルネットワークの非線形性の例：
ロジスティック回帰を2層積むと非線形分類が可能

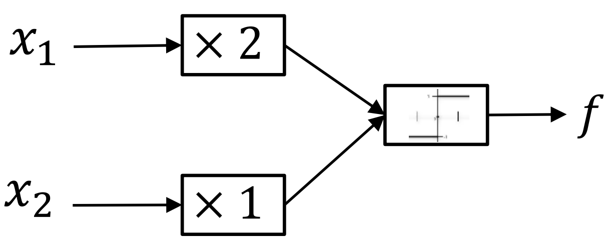
- ロジスティック回帰は1層では線形判別しかできない（AND/OR）
- 2層以上積むことで非線形の表現力を獲得（XOR）

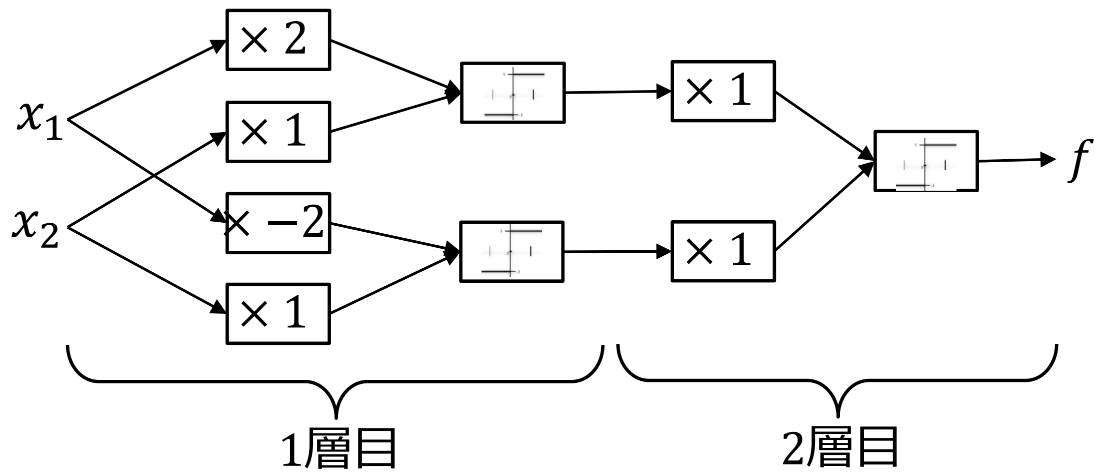


ロジスティック回帰モデル



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：

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

  - $f\left(x^{(i)}\right)$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）

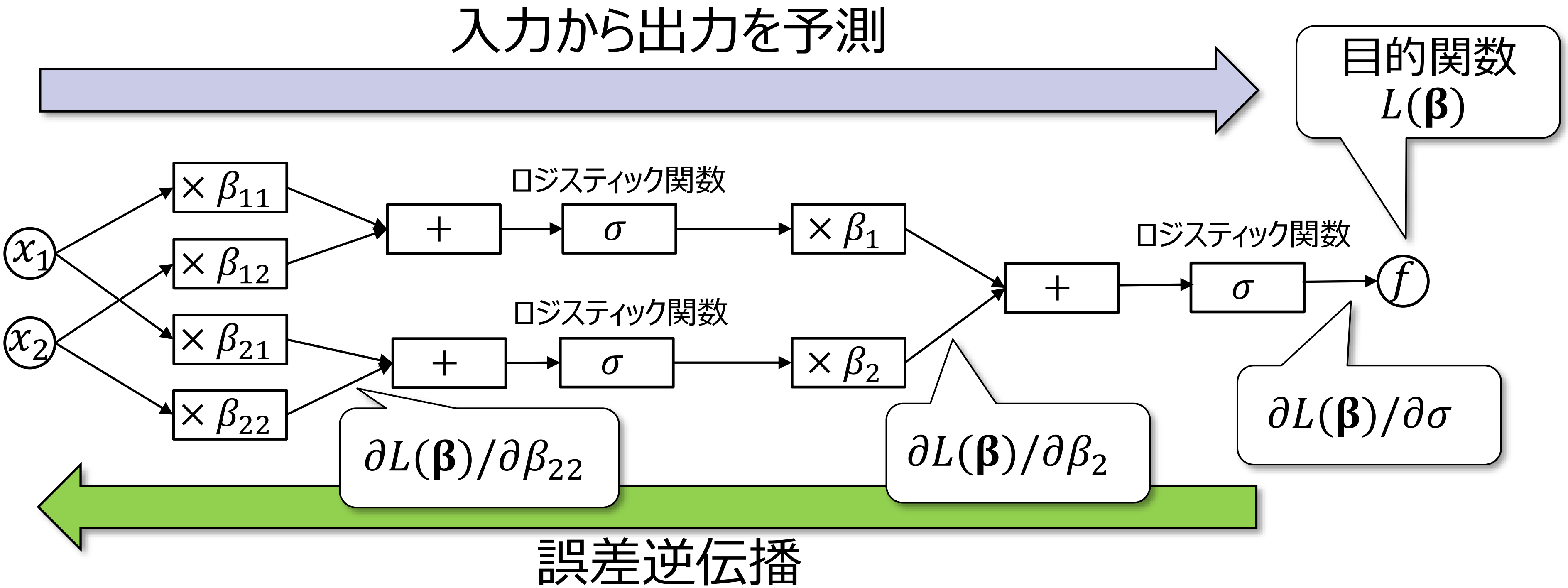
- 勾配$\nabla L(\boldsymbol{\beta}) = \dfrac{\partial L(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}}$が計算できれば最急降下法を適用できる：

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

  - 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$L(\mathbf{\beta})$ 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



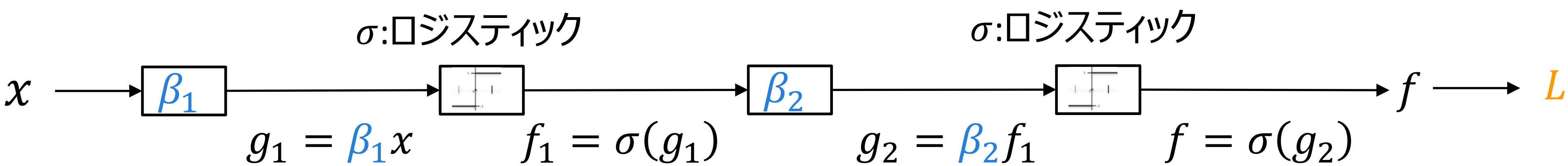
誤差逆伝播法：
勾配を再帰的に効率的に計算できる

- 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）
  - $\frac{\partial L}{\partial \beta_2} = \frac{\partial L}{\partial f} \cdot \frac{\partial f}{\partial g_2} \cdot \frac{\partial g_2}{\partial \beta_2}$
  - $\frac{\partial L}{\partial \beta_1} = \frac{\partial L}{\partial f} \cdot \frac{\partial f}{\partial g_2} \cdot \frac{\partial g_2}{\partial f_1} \cdot \frac{\partial f_1}{\partial g_1} \cdot \frac{\partial g_1}{\partial \beta_1}$

共通なので層をまたいで使いまわし可能

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



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
  - 最尤推定の対数尤度は、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：
  - （乱暴にいえば）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

計算グラフと自動微分

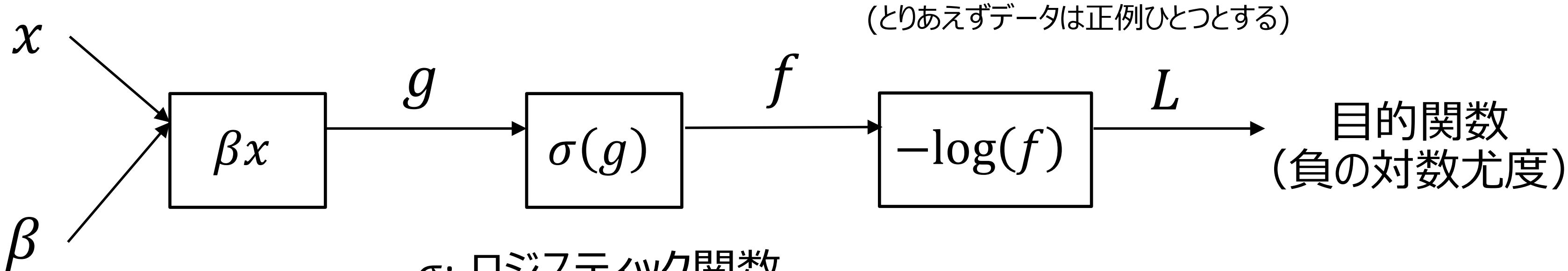
計算グラフ：
ニューラルネットの入力から出力までの計算を図示

- 計算グラフ：関数の入出力の間の関係を、単純な計算ユニットをつないで表したもの

- 計算グラフをたどりながら、入力に順番に単純な操作（重み付き和やロジスティック関数の適用など）を適用していくと、出力が得られる

- ロジスティック回帰の計算グラフ：

ロジスティック回帰の出力：$f = \sigma(\beta x)$
最適化問題の目的関数：$L = -\log f$
(とりあえずデータは正例ひとつとする)



σ: ロジスティック関数

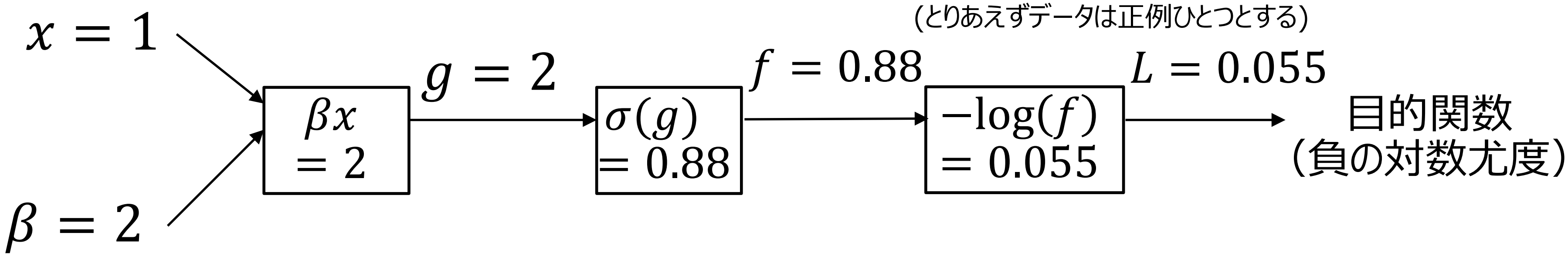
計算グラフ：
ニューラルネットの入力から出力までの計算を図示

- 計算グラフ：関数の入出力の間の関係を、単純な計算ユニットをつないで表したもの

- 計算グラフをたどりながら、入力に順番に単純な操作（重み付き和やロジスティック関数の適用など）を適用していくと、出力が得られる

- ロジスティック回帰の計算グラフ：

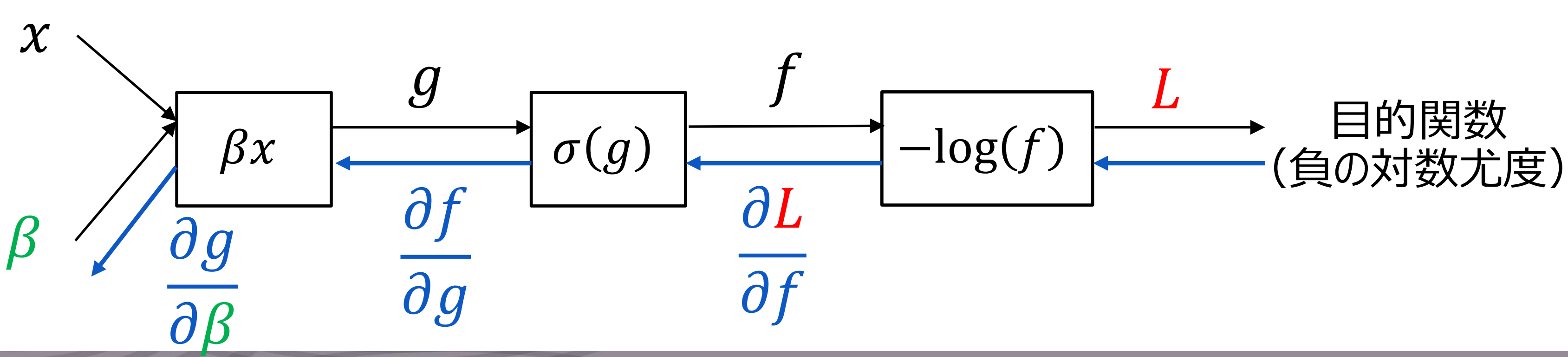
ロジスティック回帰の出力： $f = \sigma(\beta x)$
最適化問題の目的関数：$L = -\log f$
(とりあえずデータは正例ひとつとする)



σ: シグモイド関数

計算グラフ上での自動微分：
計算グラフを出力から逆向きにたどることで勾配計算

- 勾配計算：$\partial L/\partial\beta$を求めたい
- 計算グラフ上でLとβは遠い
- 計算グラフを逆向きにたどりながら、微分を計算する
  - ロジスティック回帰の場合： $\dfrac{\partial L}{\partial \beta} = \dfrac{\partial g}{\partial \beta} \cdot \dfrac{\partial f}{\partial g} \cdot \dfrac{\partial L}{\partial f}$

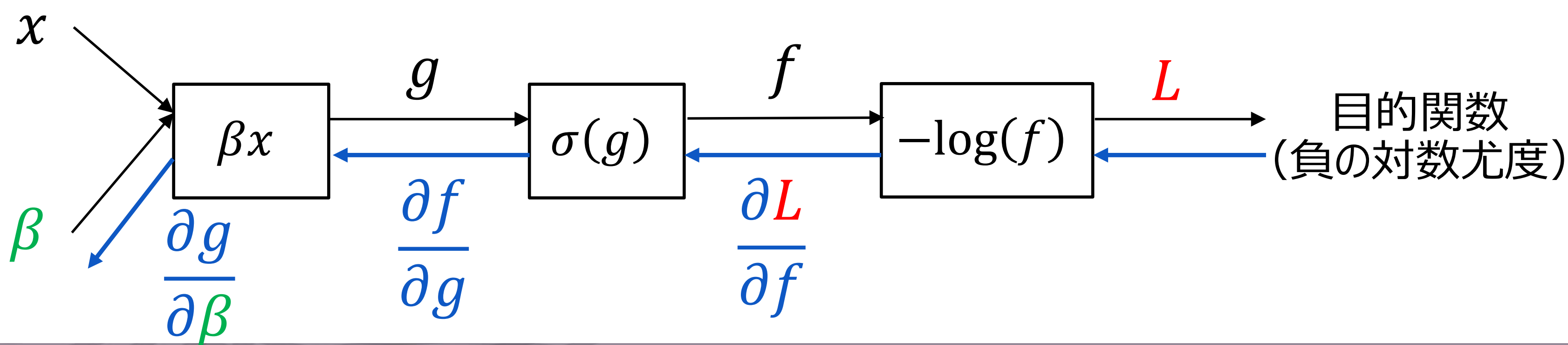


自動微分のポイント：
微分可能な計算ユニット

- 自動微分：計算グラフを逆向きにたどりながら（微分の連鎖律によって）微分を計算する

- 各ユニットは、入力について微分可能である必要がある

  - $\frac{\partial f}{\partial g} = \frac{\partial \sigma(g)}{\partial g} = \sigma(g)\big(1 - \sigma(g)\big)$ （ロジスティック関数の微分）
  - $\frac{\partial g}{\partial \beta} = \frac{\partial \beta x}{\partial \beta} = x$



ニューラルネットワーク推定のポイント：
微分可能なユニットを組みあわせて自動微分にまかせる

- ニューラルネット推定法の汎用性

1. ネットワークを「計算グラフ」で記述する
   - 各ユニットはパラメータや入力について微分可能とする
2. 誤差逆伝播で自動的に勾配が計算できる
   （自動微分）