4/15 担当：[redacted]

個人的な期待：表現学習

SVMより数%精度向上といっても応用としては弱い．
背後にあるデータの構造を自動的に抽出できないか？
職人芸 vs. ML

1章

NN：多層だと強力な表現力を持つが学習が難しい　90年代に下火
back propagation (§4)　except CNN (§6)

① DBN (Deep belief Network) §8
Hinton et al.
NNに似た多層構造
↓
RBM (restricted Boltzmann machine)に分解
+
greedy法で学習できる
DBNが
] pre training

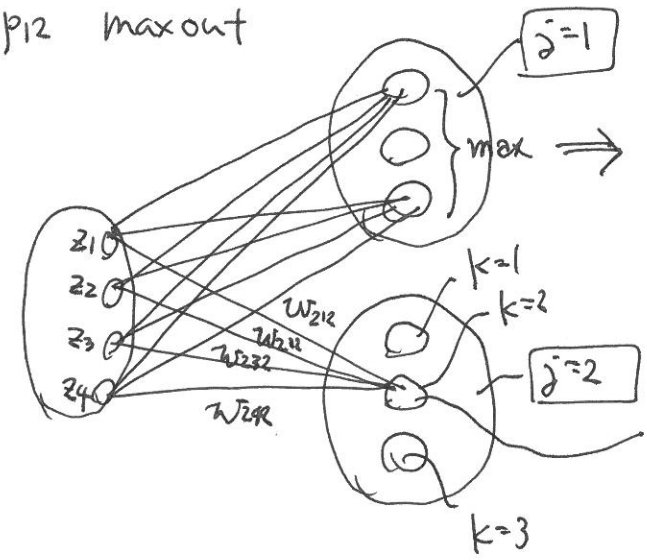
② autoencoder §5
教師なし．　入力 → 出力 = 入力 となるような学習．
↳ 多層NNの学習ができる

§1.1.3 特徴量学習．
このあたりが面白そう．

§1.1.4
問題において多数が仮定されている．
我々の場合はどれが良いか？
NLP，音声認識 → RNN §7

§2

p12 maxout




$$u_{22} = \sum_{i=1}^{4} w_{2i2}\, z_i + b_{22}$$

(25a) 式:

$$\mathbf{u}^{(l+1)} = \mathbf{W}^{(l+1)}\, \mathbf{z}^{(l)} + \mathbf{b}^{(l+1)}$$

$$\begin{bmatrix} u_1^{(l+1)} \\ \vdots \\ u_{J(l+1)}^{(l+1)} \end{bmatrix} = \begin{bmatrix} W_{11}^{(l+1)} & \cdots & W_{1I(l)}^{(l+1)} \\ \vdots & \ddots & \vdots \\ W_{J(l+1)1}^{(l+1)} & \cdots & W_{J(l+1)I(l)}^{(l+1)} \end{bmatrix} \begin{bmatrix} z_1^{(l)} \\ \vdots \\ z_{I(l)}^{(l)} \end{bmatrix} + \begin{bmatrix} b_1^{(l+1)} \\ \vdots \\ b_{J(l+1)}^{(l+1)} \end{bmatrix}$$

$J(l+1) \times 1 \quad J(l+1) \times I(l) \quad I(l) \times 1$

$$\mathbf{z}^{(l+1)} = f\left(\mathbf{u}^{(l+1)}\right)$$

$$\begin{bmatrix} z_1^{(l+1)} \\ \vdots \\ z_{J(l+1)}^{(l+1)} \end{bmatrix} = \begin{bmatrix} f(u_1^{(l+1)}) \\ \vdots \\ f(u_{J(l+1)}^{(l+1)}) \end{bmatrix}$$

$\mathbf{y} \equiv \mathbf{z}^{(L)}$ 最終出力

§2.4

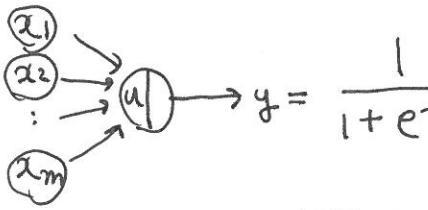
学習 { 回帰 : 二乗誤差の最小化
　　 { 分類 : MLE : 最尤推定. = 尤度最大化
　　　　同時確率.

§2.4.3 二値分類

入力 : $x$
出力 : $d \in \{0, 1\}$

事後確率 : $P(d=1|x)$ をNNでモデル化.

2層の場合




$$y = \frac{1}{1+e^{-u}} = \frac{1}{1+e^{-(w_1x_1+\cdots w_mx_m+b)}} = Pr(d=1|x) = y(x;w)$$

$$u = w_1x_1 + w_2x_2 + \cdots w_mx_m + b$$

訓練データ : $\{(x_n, d_n) \mid n=1, \cdots, N\}$ サンプル数 = N個.

$$p(d|x) = p(d=1|x)^d \cdot p(d=0|x)^{1-d}$$

$$p(d=0|x) = 1 - y(x;w)$$

$d_n$ : $x_n$ を観測したときの事後確率

$$P(d_n|x_n) = \frac{Pr(x_n|d_n) \cdot Pr(d_n)}{Pr(x_n)}$$

尤度 ← $Pr(x_n|d_n)$ ／ 事前確率 ← $Pr(d_n)$ ／ $Pr(x_n)$ — 証拠

$Pr(d_n)$, $Pr(x_n)$ は $w$ によらないので..

尤度最大 = 事後確率最大

正しくは

$$L(w) \sim \prod_{n=1}^{N} p(d_n|x_n;w)$$

$L(w)$ = 尤度

$$= \prod_{n=1}^{N} \{y(x_n;w)\}^{d_n} \cdot \{1-y(x_n;w)\}^{1-d_n}$$

$$\log L(w) \sim \sum_{n=1}^{N} \{d_n \cdot \log y(x_n;w) + (1-d_n)\log[1-y(x_n;w)]\}$$

尤度最大 = 負の尤度最小 → (2.8)式.

$$P(d=1|x) = \frac{P(d=1,x)}{P(x)} = \frac{P(x,d=1)}{P(x,d=0)+P(x,d=1)}$$

$$u \equiv \log \frac{P(x,d=1)}{P(x,d=0)}$$ 対数おいた.

$$\frac{1}{1+e^{-u}} = \frac{e^u}{e^u+1} = \frac{\frac{P(x,d=1)}{P(x,d=0)}}{\frac{P(x,d=1)}{P(x,d=0)}+1} = \frac{P(x,d=1)}{P(x,d=1)+P(x,d=0)} = P(d=1,x)$$

$(e^{\log A} = A)$

## § 2.4.4 多クラス分類

$$y_k \equiv z_k^{(L)} = \frac{\exp(u_k^{(L)})}{\sum_{j=1}^{K} \exp(u_j^{(L)})} = P(C_k|x)$$

出力層

$K=2$ とすると $y_1 + y_2 = 1$

$$\hat{C} = \underset{k}{\operatorname{argmax}}\, y_k$$

推定クラス

入力 : $x_n = \begin{bmatrix} x_1^{(n)} \\ \vdots \\ x_m^{(n)} \end{bmatrix} (n=1,\dots,N)$

目標 : $d_n = \begin{bmatrix} d_{n1} \\ d_{n2} \\ \vdots \\ d_{nK} \end{bmatrix}$ (例) $\begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$ ← 正解 $= C_3$

事後分布

$$P(d|x) = \prod_{k=1}^{K} P(C_k|x)^{d_k}$$

$$L(w) \equiv \text{尤度} \sim \prod_{n=1}^{N} P(d_n|x_n;w)$$

$w$ の関数と見る

$$= \prod_{n=1}^{N}\prod_{k=1}^{K} P(C_k|x)^{d_{nk}} = \prod_{n=1}^{N}\prod_{k=1}^{K} \{y_k(x;w)\}^{d_{nk}}$$

誤差関数

→ (2.11)式

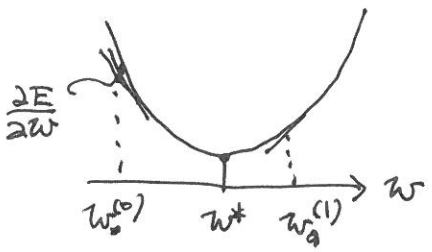
§3

学習：$w = \underset{w}{\operatorname{argmin}} E(w)$

↓

gradient descent method

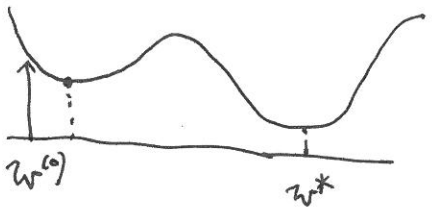
$$\nabla E \equiv \frac{\partial E}{\partial w} = \begin{bmatrix} \frac{\partial E}{\partial w_1} \\ \vdots \\ \frac{\partial E}{\partial w_M} \end{bmatrix}$$




$$w^{(t+1)} = w^{(t)} - \varepsilon \nabla E$$

$$w_0^{(1)} = w_0^{(0)} - \varepsilon \frac{\partial E}{\partial w}\Big|_{w=w^{(0)}}$$

local minimaに落ちる可能性

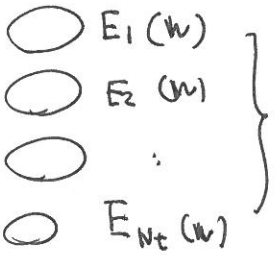



§3.2

SGD：よく使われる

- 計算量の点で上
- local minimaへのトラップを回避できる可能性
- オンライン学習

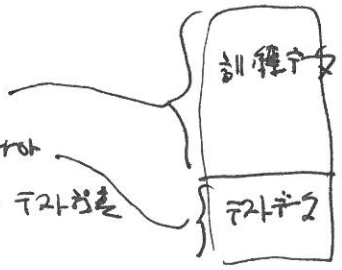
§3.3 ミニバッチ … 並列処理




$$E_t(w) = \frac{1}{N_t} \sum_{i=1}^{N_t} E_i(w)$$

§3.4

訓練誤差 training error
汎化誤差 generalization error
＝テスト誤差




NN学習のテクニック：早期終了

overfitting, overlearningを防ぐ

層数，ユニット数，ハイパーパラメタの調整
正則化，などなどetc. を使う

$$e^{-u} = \frac{-P(x, d=1)}{P(x, d=0)}$$

$$\frac{1}{1+e^{-u}} = \frac{1}{1+\frac{P(x,d=0)}{P(x,d=1)}} = \frac{P_r(x \mid d=1)}{P_r(x)} = \frac{P_r(x,u)}{(P_r(d=1)\ P_r(x)}$$

$$e^{\log x} = x$$

$$= \frac{1}{e^{\log x}} = \frac{1}{x}$$

$$u = \log \frac{P(x, d=1)}{P(x, d=0)}$$

$$\frac{1}{1+e^{-u}} = \frac{e^u}{e^u + 1} = \frac{\frac{P(x,d=1)}{P(x,d=0)}}{\frac{P(x,d=1)}{P(x,d=0)} + 1} = \frac{P(x, d=1)}{P(x \mid d=1) + P(x, d=0)} = \frac{P(x, d=1)}{P(x)}$$

$$= \frac{P(x \mid d=1)\, P(d=1)}{P(x \mid d=1) P(d=1) + P(x \mid d=0) P(d=0)}$$

$$= P.$$

§35.1 正則化

PRML本 の§1.1参照. 重み減衰.

§35.3 ドロップアウト [67] 2014年 Hinton グループ

ユニットを重み更新の度に確率的に退出

狙い: ・学習時にネットワークの自由度を強制的に小さくする → 過適合を避ける
複数ネットワークの平均に相当する効果

§36 トリック

§36.1 正規化
§36.2 拡張
§36.3 複数ネット平均

⋮ textbook

$$\mathrm{Var}[XY] = [E(X)]^2 \mathrm{Var}[Y] + [E(Y)]^2 \mathrm{Var}(X) + \mathrm{Var}(X)\mathrm{Var}[Y]$$