大規模データの線形識別

Recent Advances of Large-scale Linear Classification
Guo-Xun Yuan, Chia-Hua Ho, and Chih-Jen Lin
を中心にまとめた

データの性質

- 入力データの次元と入力データ数
  - 入力データの各次元を素性とも言う
  - 次元の高い場合はsparse（有効な素性が少ない）
  - 次元が低い場合はdense（多数の素性が有効）

	入力データの次元　小 **10^3以下**	**入力データの次元　大** **10^4以上**
入力データ数　小 Mega	新規理論をとりあえず試す toy　data	正解タグ付きテキストコーパスなど （場合によっては画像も）
入力データ数　大 Giga-Tera	計測される数値データ （センサーデータ、市場データ、など）	正解タグなしの生テキストコーパス
ストリーム 時系列到着	センサーデータ	Twitter、ソーシャルメディアなど

データの性質

- 次元が大きい（10の4乗以上）の場合は線形識別も非線形識別の精度に大きな差はない。
- 次元が低いと非線形識別のほうが精度がかなりよい。非線形化すなわち高次元化による特徴抽出能力の改善のため

- 教師データ（タグ付けデータ）と生データ
  - 教師データが大きければ、それを相手に学習
  - 教師データがなければクラスタリングなど
  - 小さな教師データと多数の生データ（実際は多い）
    - Semi-supervised learning, Active learning などチャレンジング

記法

$\left(y_i, \mathbf{x}_i\right) \in \{+1, -1\} \times \mathrm{R}^n, i = 1, .., l$

decision function : $d(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b$　　$\phi(\mathbf{x}) = \mathbf{x}$なら*linear*

$\mathbf{w} = \sum_{i=1}^{l} \alpha_i \phi(\mathbf{x}_i)$

以下では　　$\mathbf{w}^T \leftarrow \left[\mathbf{w}^T \ b\right]$　　$\mathbf{x}_i^T \leftarrow \left[\mathbf{x}_i^T \ 1\right]$　　とし、bは陽には書かない

線形識別　　linear classificationは以下の最適化問題を含む

$$\min_{\mathbf{w}} f(\mathbf{w}) \equiv r(\mathbf{w}) + C \sum_{i=1}^{l} \xi(\mathbf{w}; \mathbf{x}_i, y_i)$$

$r(\mathbf{w})$: 正則化項　　$\xi(\mathbf{w}; \mathbf{x}_i, y_i)$: 損失

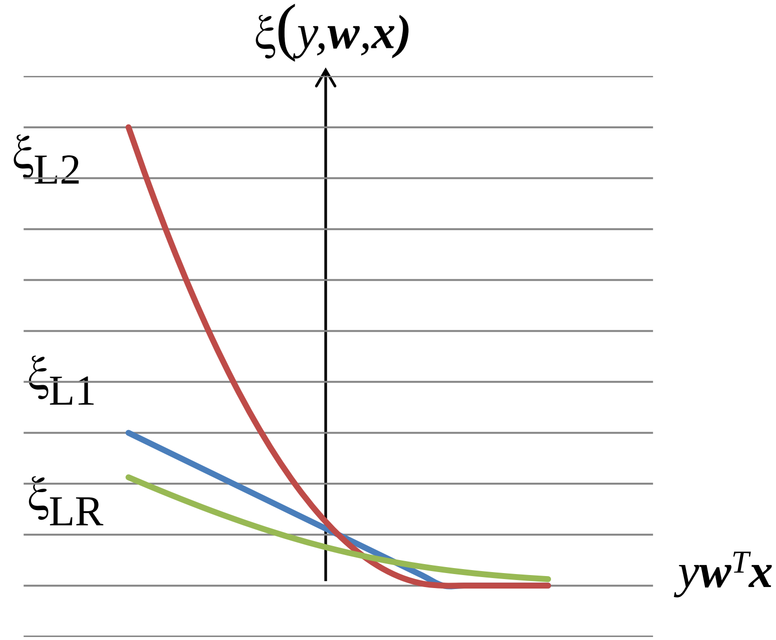
正則化項:r＋損失:ξの最小化

- 各種の損失関数(右図)

$$\xi_{L1}(\mathbf{w};\mathbf{x},y) \equiv \max\left(0, 1 - y\mathbf{w}^T\mathbf{x}\right)$$

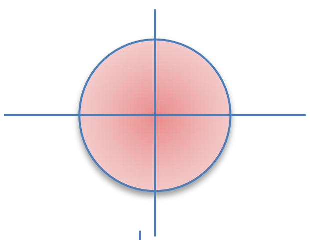
$$\xi_{L2}(\mathbf{w};\mathbf{x},y) \equiv \max\left(0, 1 - y\mathbf{w}^T\mathbf{x}\right)^2$$

$$\xi_{LR}(\mathbf{w};\mathbf{x},y) \equiv \log\left(1 + \exp(-y\mathbf{w}^T\mathbf{x})\right)$$

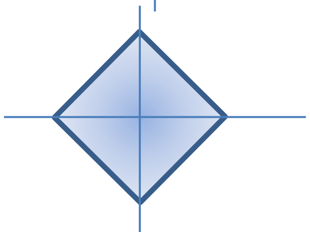


- 各種の正則化項

$$r_{L2}(\mathbf{w}) \equiv \|\mathbf{w}\|_2^2 = \frac{1}{2}\sum_{i=1}^{n} w_i^2$$



$$r_{L1}(\mathbf{w}) \equiv \|\mathbf{w}\|_1^0 = \sum_{i=1}^{n} |w_i|$$



学習アルゴリズム

- 以下では種々の学習アルゴリズムについて述べる

双対化

- 双対問題を解くほうが良いこともある。双対化とは以下の通り:

Primal problem: $\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & g_i(x) \le 0 \quad i = 1,..,n \end{array}$

Dual problem:

$$q(\mu) = \inf L(x,\mu) \quad \text{where} \quad L(x,\mu) = f(x) + \sum_{i=1}^{n} \mu_i g_i(x)$$

$$\begin{array}{ll} \text{maximize} & q(\mu) \\ \text{subject to} & \mu_i \ge 0 \quad i = 1,..,n \end{array}$$

双対化

SVM primal： $f(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|_2^2$ $g_i(\mathbf{w}) = \max\left(0, 1 - y_i \mathbf{w}^T \mathbf{x}_i\right) \le 0$

SVM dual： $\inf L(x,\mu) = \inf \frac{1}{2}\|\mathbf{w}\|_2^2 + \sum_{i=1}^n \mu_i \max\left(0, 1 - y_i \mathbf{w}^T \mathbf{x}_i\right)$

$$\Rightarrow \mathbf{w} = \sum_{i=1}^n y_i \mu_i \mathbf{x}_i$$

$$\Rightarrow \max -\frac{1}{2}\sum_{i=1}^n \sum_{j=1}^n \mu_i \boxed{y_i \mathbf{x}_i^T \mathbf{x}_j y_j} \mu_j + \sum_{i=1}^n \mu_i$$

$$\Rightarrow \min \frac{1}{2}\mu^T Q \mu - \mathbf{1}^T \mu \quad \text{subject to} \quad 0 \le \mu_i \le C$$

この制約は$\max\left(0, 1 - y_i \mathbf{w}^T \mathbf{x}_i\right) \le \eta_i$ and $\min \frac{1}{2}\|\mathbf{w}\|_2^2 + C\eta_i$

という線形分離できない場合の定式化から出る

(数理手法IV　カーネル法　サポートベクターマシン　参照)

比較

- Gradient Descentのような低次の手法はupdateは簡単でメモリも少ない。収束が遅く、解は精密ではないが、実は性能に大差ない

- ニュートン法のような高次の手法は、収束は早いが、updateのコストは高い（例えば、Hessian^{-1}の計算とかLBFGSとか）。また、解空間がスムーズ（2階微分可能）でないといけない。精密な最適化には有効。

  - exp/logの計算はコストが高いことを忘れないように
  - 並列化する場合はコア間、ノード間の通信のコストも大きいことに注意

Pegasos : Shalev-Schwalz et al. ICML 2007

- L2正則化＋L1損失のSVM

$$\min_{\mathbf{w}} f(\mathbf{w};B) \equiv \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i\in B}\max\left(0, 1-y_i\mathbf{w}^T\mathbf{x}\right) \quad B\text{は全学習データ}$$

- Pegasosの更新は次式の劣微分による

$$\mathbf{w} \leftarrow \mathbf{w} - \eta\partial f(\mathbf{w};B)$$

$$where \quad \partial f(\mathbf{w};B) = \mathbf{w} - C\sum_{i\in B^+} y_i\mathbf{x}_i \qquad B^+ \equiv \{i \mid i \in B, 1-y_i\mathbf{w}^T\mathbf{x}_i > 0\},$$

$$\eta = Cl/k, \quad l\text{はベクトル}\mathbf{w}\text{，}\ \mathbf{x}_i\text{の次元、}k\text{は繰り返し回数}$$

- 更新の後、wを半径$(Cl)^{-1/2}$の球にproject

$$\mathbf{w} \leftarrow \min\left(1, \sqrt{Cl}/\|\mathbf{w}\|_2\right)\mathbf{w}$$

- 以上を収束するまで繰り返す。

Trust Region Newton Method(TRON):導入

- データの次元:nが大きいと $\nabla^2 f(\mathbf{w})$は$n \times n$なのでメモリに置くのも逆行列の計算も大変。
  - 逆行列計算はBFGSなどによる
- まずlogistic regressionについて

$$\min_{\mathbf{w}} f(\mathbf{w}) \equiv \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{l}\log\left(1+\exp\left(-y_i\mathbf{w}^T\mathbf{x}_i\right)\right)$$

$$\nabla f(\mathbf{w}) = \mathbf{w} + C\sum_{i=1}^{l}\left(\left(1+\exp\left(-y_i\mathbf{w}^T\mathbf{x}_i\right)\right)^{-1} - 1\right)y_i\mathbf{x}_i$$

$$\nabla^2 f(\mathbf{w}) = I + CX^T DX \qquad D\text{は対角行列}$$

$$X = \left[\mathbf{x}_1^T \cdots \mathbf{x}_l^T\right]^T \qquad D_{ii} = \left(1+\exp\left(-y_i\mathbf{w}^T\mathbf{x}_i\right)\right)^{-1}\left(1-\left(1+\exp\left(-y_i\mathbf{w}^T\mathbf{x}_i\right)\right)^{-1}\right)$$

- するとNewton法は以下のようになる。

$$\mathbf{w}^{k+1} = \mathbf{w}^k + s^k$$

$$-\nabla f\left(\mathbf{w}^k\right) = \nabla^2 f\left(\mathbf{w}^k\right)s^k = \left(I + CX^T DX\right)s^k$$

$[D_{ij}]$が対角行列であることの直観的説明

$$\min_{\mathbf{w}} f(\mathbf{w}) \equiv \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{l}\log\left(1+\exp\left(-y_i\mathbf{w}^T\mathbf{x}_i\right)\right)$$

$$\nabla f(\mathbf{w}) = \mathbf{w} + C\sum_{i=1}^{l}\left(\left(1+\exp\left(-y_i\mathbf{w}^T\mathbf{x}_i\right)\right)^{-1}-1\right)y_i\mathbf{x}_i \quad \nabla^2 f(\mathbf{w}) = I + CX^TDX \qquad D\text{は対角行列}$$

$$X = \left[\mathbf{x}_1^T \cdots \mathbf{x}_l^T\right]^T \qquad D_{ii} = \left(1+\exp\left(-y_i\mathbf{w}^T\mathbf{x}_i\right)\right)^{-1}\left(1-\left(1+\exp\left(-y_i\mathbf{w}^T\mathbf{x}_i\right)\right)^{-1}\right)$$

$\boxed{2\times 2\text{で具体例}}$

$$\nabla^2\sum_i\log\left(1+\exp\left(-y_i\mathbf{w}^T\mathbf{x}_i\right)\right) = \nabla\sum_i\left(\left(1+\exp\left(-y_i\mathbf{w}^T\mathbf{x}_i\right)\right)^{-1}-1\right)y_i\mathbf{x}_i$$

$$= \sum_i\frac{\exp\left(-y_i\mathbf{w}^T\mathbf{x}_i\right)y_i\mathbf{x}_i^T y_i\mathbf{x}_i}{\left(1+\exp\left(-y_i\mathbf{w}^T\mathbf{x}_i\right)\right)^{-2}} = \sum_i\mathbf{x}_i^T D_{ii}\mathbf{x}_i$$

$$X^T = \begin{bmatrix}\mathbf{x}_1 & \mathbf{x}_2\end{bmatrix} = \begin{bmatrix}x_{11} & x_{21}\\ x_{12} & x_{22}\end{bmatrix} X = \begin{bmatrix}\mathbf{x}_1^T\\ \mathbf{x}_2^T\end{bmatrix} = \begin{bmatrix}x_{11} & x_{12}\\ x_{21} & x_{22}\end{bmatrix}$$

$$\nabla^2\sum_{i=1}^{2}\log\left(1+\exp\left(-y_i\mathbf{w}^T\mathbf{x}_i\right)\right) = \sum_{i=1}^{2}\mathbf{x}_i^T D_{ii}\mathbf{x}_i = \begin{bmatrix}x_{11}{}^2D_{11}+x_{21}{}^2D_{22} & x_{11}x_{12}D_{11}+x_{21}x_{22}D_{22}\\ x_{11}x_{12}D_{11}+x_{21}x_{22}D_{22} & x_{12}{}^2D_{11}+x_{22}{}^2D_{22}\end{bmatrix}$$

$$= \begin{bmatrix}x_{11}D_{11} & x_{21}D_{22}\\ x_{12}D_{11} & x_{22}D_{22}\end{bmatrix}\begin{bmatrix}x_{11} & x_{12}\\ x_{21} & x_{22}\end{bmatrix} = \boxed{\begin{bmatrix}x_{11} & x_{21}\\ x_{12} & x_{22}\end{bmatrix}\begin{bmatrix}D_{11} & 0\\ 0 & D_{22}\end{bmatrix}\begin{bmatrix}x_{11} & x_{12}\\ x_{21} & x_{22}\end{bmatrix}}$$

Dは対角行列

- $\mathbf{x}_i$がsparse → $$DX = \begin{pmatrix} D_{11} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & D_{11} \end{pmatrix} \begin{pmatrix} 0 & ..0x0... & 0 \\ 0 & ...0x0. & \vdots \\ \vdots & .0x0... & \end{pmatrix} \approx \begin{pmatrix} 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{pmatrix}$$

- $$\mathbf{w}^{k+1} = \mathbf{w}^k + s^k$$
$$-\nabla f\left(\mathbf{w}^k\right) = \nabla^2 f\left(\mathbf{w}^k\right) s^k = \left(I + CX^T DX\right) s^k \approx s^k$$
$$\Rightarrow \mathbf{w}^{k+1} = \mathbf{w}^k - \nabla f\left(\mathbf{w}^k\right)$$

- $(I+\varepsilon)^{-1}$はεが小さいと近似計算もできそう。
  - １次近似なら$(I+\varepsilon)^{-1} = I - \varepsilon$
$$\approx \mathbf{w}^{k+1} = \mathbf{w}^k - \nabla f\left(\mathbf{w}^k\right)\left(I - CX^T DX\right)$$

- したがって、データが高次元Sparseならlogistic regressionはHessianの計算を避けられて、効率よく計算できる。

Trust Region Newton Method(TRON):

C.-J Lin et.al. SIAM J. Optimization 9 (1999)

- 以下の問題をfの2次の展開qを用いて解く

$$\min_{\mathbf{w}} f(\mathbf{w}) \equiv r(\mathbf{w}) + C\sum_{i=1}^{l} \xi(\mathbf{w}; \mathbf{x}_i, y_i) \qquad r\text{は}L2,\ \xi\text{は微分可能}$$

$$f(\mathbf{w}+\mathbf{d}) - f(\mathbf{w}) \approx q(\mathbf{d}) = \nabla f(\mathbf{w})^T \mathbf{d} + \frac{1}{2}\mathbf{d}^T \nabla^2 f(\mathbf{w})\mathbf{d}$$

- 最適化問題

予め決めた閾値

$$\text{step}\,1 \quad \min_{\mathbf{d}} q(\mathbf{d}) \qquad \text{subject to } \|\mathbf{d}\|_2 \le \Delta$$

$$\text{step}\,2 \quad \text{if } \rho \equiv \frac{f(\mathbf{w}+\mathbf{d}) - f(\mathbf{w})}{q(\mathbf{d})} > \eta_0 \ \text{ then } \ \mathbf{w} \leftarrow \mathbf{w}+\mathbf{d}$$

$$\text{step}\,3 \ \text{ Adjust } \Delta \ \text{ according to } \rho$$

Trust Region bound Δ_kの変え方

$\Delta_k \to \Delta_{k+1}$の規則

$\eta_1 < \eta_2 < 1 \qquad \sigma_1 < \sigma_2 < 1 < \sigma_3$

$$\Delta_{k+1} \in \left[\sigma_1 \min\left\{\|\mathbf{d}_k\|, \sigma_2\Delta_k\right\}, \sigma_2\Delta_k\right] \quad \text{if} \quad \rho_k \le \eta_1$$

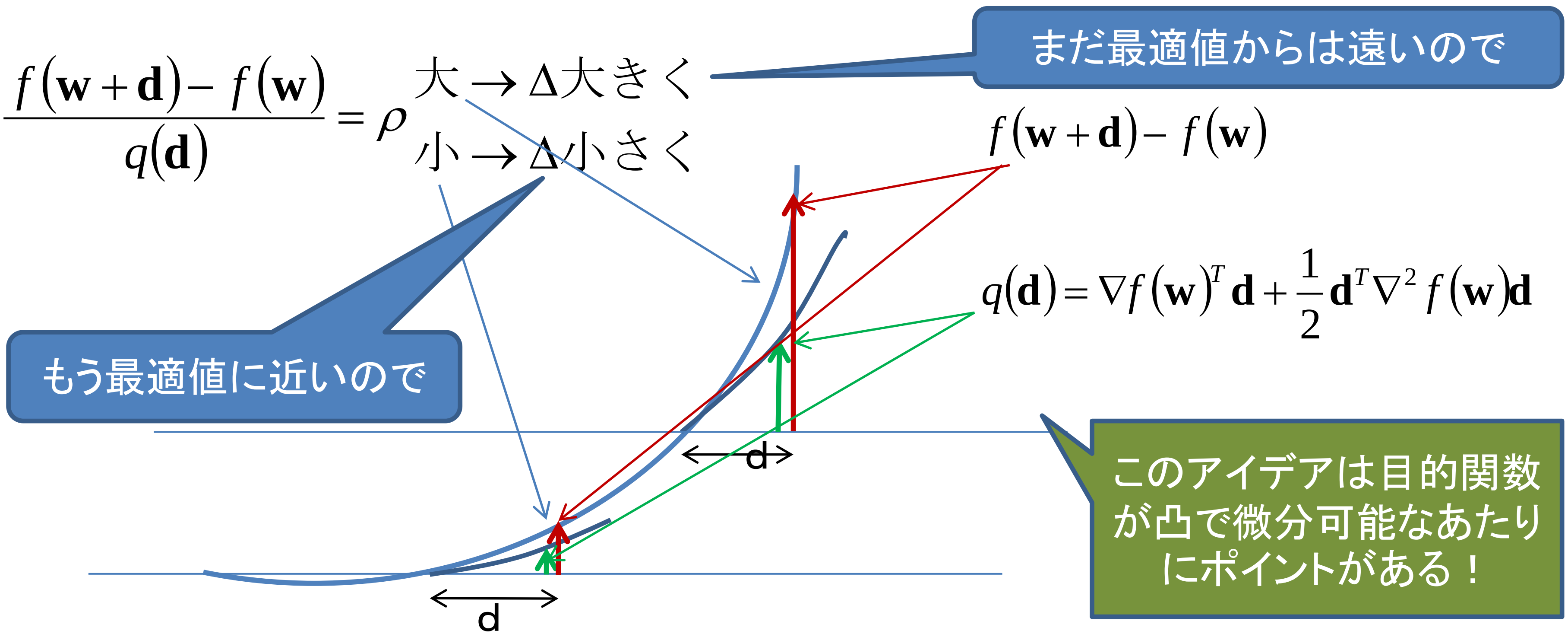
$$\Delta_{k+1} \in \left[\sigma_1\Delta_k, \sigma_3\Delta_k\right] \quad \text{if} \quad \rho_k \in (\eta_1, \eta_2)$$

$$\Delta_{k+1} \in \left[\Delta_k, \sigma_3\Delta_k\right] \quad \text{if} \quad \rho_k \ge \eta_2$$

$$\frac{f(\mathbf{w}+\mathbf{d}) - f(\mathbf{w})}{q(\mathbf{d})} = \rho$$

大→Δ大きく

小→Δ小さく



Coordinate Descent

C.-J. Hsieh, et.al. ICML2008

- Target: $L1$損失- $L2$正則化のSVM　双対化して解く。下に定義

$$\min_{\boldsymbol{\alpha}} f^{D}(\boldsymbol{\alpha}) \equiv \frac{1}{2}\boldsymbol{\alpha}^{T} Q\,\boldsymbol{\alpha} - \mathbf{e}^{T}\boldsymbol{\alpha}$$

$$\text{subject to}\quad 0 \le \alpha_i \le C \quad \forall i = 1, \ldots, l$$

$$\text{where}\quad Q_{ij} \equiv y_i y_j \mathbf{x}_i^{T} \mathbf{x}_j$$

Coordinate Descent

- Coordinate Descentは順番に1変数づつ選び、他の変数は固定して最適化。

$$\min_d f^D(\boldsymbol{\alpha}+de_i)-f^D(\boldsymbol{\alpha}) \equiv \frac{1}{2}Q_{ii}d^2+\nabla_i f^D(\boldsymbol{\alpha})d \qquad f^D\text{は}f\text{の双対}$$

$$\text{subject to} \quad 0 \le \alpha_i + d \le C \quad \text{where} \;\; e_i = \left[\underbrace{0,..0}_{i-1},1,0,...0\right]^T$$

dを計算することによるα_iの更新式は下式

$$\alpha_i \leftarrow \min\left(\max\left(\alpha_i - \frac{\nabla_i f^D(\boldsymbol{\alpha})}{Q_{ii}},0\right),C\right) \qquad (CD10)$$

Coordinate Descent　つづき

- $(CD10)$のQ_{ii}はα_iの最適化の中で1回計算すればよいが
- $\nabla_i f^D(\boldsymbol{\alpha}) = (Q\boldsymbol{\alpha})_i - 1 = \sum_{t=1}^{l} \left(y_i y_t \mathbf{x}_i^T \mathbf{x}_t\right)\alpha_t - 1$ は　　$\mathbf{x}_i^T \mathbf{x}_t \; \forall t = 1,..,l$ の計算コストが$O(nl)$でうれしくない。そこで
- $\mathbf{u} \equiv \sum_{t=1}^{l} y_t \alpha_t \mathbf{x}_t$ を保持しておけば
- $\nabla_i f^D(\boldsymbol{\alpha}) = (Q\boldsymbol{\alpha})_i - 1 = y_i \mathbf{u}^T \mathbf{x}_i - 1$　(CD20)

  となり計算コストは 以下の計算のための$O(n)$でうれしい。

  $\mathbf{u} \leftarrow \mathbf{u} + y_i\left(\alpha_i - \overline{\alpha}_i\right)\mathbf{x}_i$　　　　(CD30)

  $\overline{\alpha}_i$ (CD10)の更新前､α_i (CD10)の更新後

L1損失-L2正則化のSVMの Coordinate Descent アルゴリズム

$\boldsymbol{\alpha}$の初期化、および$\mathbf{u} = \sum_{i=1}^{l} y_i \alpha_i \mathbf{x}_i$

$Q_{ii} \ \forall i = 1, \ldots, l$の計算

while $\boldsymbol{\alpha}$ is not optimal

For $i = 1,..l$

(CD20)により$G = y_i \mathbf{u}^T \mathbf{x}_i - 1$の計算

$\overline{\alpha}_i \leftarrow \alpha_i$

$$\alpha_i \leftarrow \min\left(\max\left(\alpha_i - \frac{G}{Q_{ii}}, 0 \right), C \right)$$

$\mathbf{u} \leftarrow \mathbf{u} + y_i \left(\alpha_i - \overline{\alpha}_i \right) \mathbf{x}_i$

Newton+ Coordinate Descent

J. H. Friedman, T. Hastie, and R. Tibshirani, "Regularization paths for generalized linear models via coordinate descent," Journal of Statistical Software, vol. 33, no. 1, pp. 1–22, 2010.

- 最小化する目的関数 f は以下

L1正則化項

$$f(\mathbf{w}) = \|\mathbf{w}\|_1 + L(\mathbf{w}) \quad \text{where} \quad L(\mathbf{w}) \equiv C \sum_{i=1}^{l} \xi(\mathbf{w}; \mathbf{x}_i, y_i)$$

繰り返しごとに解くのは2次近似した以下の問題

$$\min_{\mathbf{d}} q(\mathbf{d}) \equiv \|\mathbf{w}+\mathbf{d}\|_1 - \|\mathbf{w}\|_1 + \nabla L(\mathbf{w})^T \mathbf{d} + \frac{1}{2}\mathbf{d}^T H \mathbf{d} \quad \text{(NCD10)}$$

$$\text{where} \quad H \equiv \nabla^2 L(\mathbf{w}) + \nu I$$

小さなνはHをpositive保つため導入

- ただし、L1正則化項（1-norm)のために全体を一度に解けないので、coordinate descentを導入

- (NCD10)を１変数化してみると以下のようになる

$$
\begin{aligned}
&q(\mathbf{d}+ze_j)-q(\mathbf{d})\\
&=\|\mathbf{w}+\mathbf{d}+ze_j\|_1-\cancel{\|\mathbf{w}\|_1}+\nabla L(\mathbf{w})^T(\cancel{\mathbf{d}}+ze_j)+\frac{1}{2}(\cancel{\mathbf{d}}+ze_j)^T H(\mathbf{d}+ze_j)\\
&\quad-\|\mathbf{w}+\mathbf{d}\|_1 \qquad +\cancel{\|\mathbf{w}\|_1}+\cancel{\nabla L(\mathbf{w})^T}\mathbf{d} \qquad +\frac{1}{2}\cancel{\mathbf{d}^T H\mathbf{d}}\\
&=|w_j+d_j+z|-|w_j+d_j|+G_j z+\frac{1}{2}H_{jj}z^2 \quad \text{where} \quad G\equiv\nabla L(\mathbf{w})+H\mathbf{d}
\end{aligned}
$$

- L1正則化における次元ごとの最適化の手法を利用すると

$$
z=\begin{cases}
-\dfrac{G_j+1}{H_{jj}} & \text{if } \ G_j+1\le H_{jj}(w_j+d_j)\\
-\dfrac{G_j-1}{H_{jj}} & \text{if } \ G_j-1\ge H_{jj}(w_j+d_j)\\
-(w_j+d_j) & \text{otherwise}
\end{cases}
\qquad (NCD20)
$$

(NCD20)の直観的説明

$\min_z |w+d+z| - |w+d| + Gz + \frac{1}{2}Hz^2$ を求める。$|w+d|$は定数なので無視。

$$|w+d+z| + Gz + \frac{1}{2}Hz^2 = \begin{cases} g1(z) \text{ if } z \geq -(w+d) \\ g2(z) \text{ if } z \leq -(w+d) \end{cases}$$

$$g1(z) = w+d+z+Gz+\frac{1}{2}Hz^2,\ g2(z) = -w-d-z+Gz+\frac{1}{2}Hz^2$$

$g1(z)$の2次関数としての最小値が$-(w+d)$より大きいのは

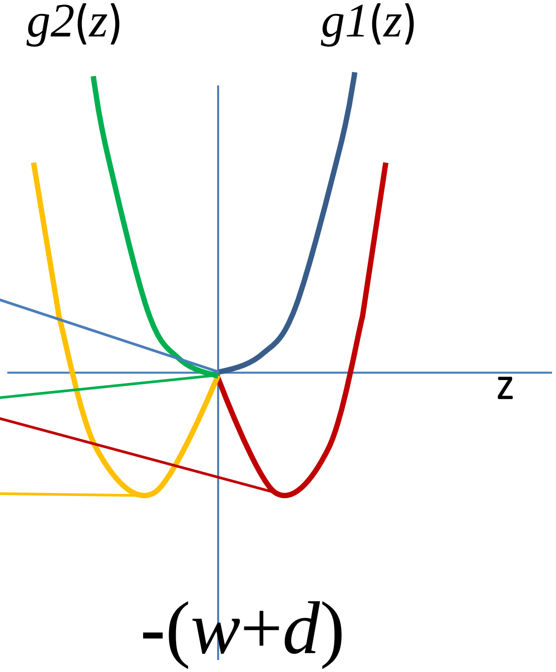
$$\frac{\partial g1(z)}{\partial z} = 0 \Rightarrow z = -\frac{G+1}{H} \geq -(w+d) \quad \text{すなわち} G+1 \leq H(w+d)$$

$$\text{then } z = -\frac{G+1}{H} \quad \text{otherwise } z = -(w+d)$$

$g2(z)$の2次関数としての最小値が$-(w+d)$より小さいのは

$$\frac{\partial g2(z)}{\partial z} = 0 \Rightarrow z = -\frac{G-1}{H} \leq -(w+d) \quad \text{すなわち} G-1 \geq H(w+d)$$

$$\text{then } z = -\frac{G-1}{H} \quad \text{otherwise } z = -(w+d)$$



- (NCD20)を各次元に対して行うと最適化問題(NCD10)の1回の繰り返しができる。
- さて、(NCD10)を直線探索で繰り返して近似精度を上げる
- 直線探索による近似アルゴリズム

$\mathbf{w}, 0<\beta, 0<\sigma$が与えられた

for k $=1,2,\ldots$

$(NCD10)$の近似解$\mathbf{d}$をcoordinate descent法$(NCD20)$で得る

$f(\mathbf{w}+\lambda\mathbf{d})-f(\mathbf{w})\le\sigma\lambda\left(\|\mathbf{w}+\mathbf{d}\|_1+\|\mathbf{w}\|_1+\nabla L(\mathbf{w})^T\mathbf{d}\right)$を満たす

ような$\lambda=\max\{1,\beta,\beta^2,\ldots\}$を見つける

$\mathbf{w}\leftarrow\mathbf{w}+\lambda\mathbf{d}$

メモリに乗り切らないビッグデータの処理に関して

- ディスクとメモリのデータ転送
  - 学習中にディスクとメモリのデータ転送回数が増えるのはもちろん問題だが
  - ビッグデータを（分割するにしても）メモリに1回転送する時間も膨大
- 分散環境（地理的な場合も含む）にデータがある場合
  - 分散環境での学習はもっと厄介
  - あまり公表された成果がない。Googleがちらっとブログで言った程度

オンライン学習による方法

- 元来が1データ毎の学習なので自然な方法
- L1-正則化&L2-損失のSVMは以下の更新式

$$\mathbf{w} \leftarrow \mathbf{w} - \eta \nabla^S \left(\frac{1}{2} \|\mathbf{w}\|_2^2 + C \max\left(0, 1 - y_i \mathbf{w}^T x_i\right) \right) \qquad \nabla^S \text{は sub-gradient}$$

$$\Rightarrow \text{if } \ 1 - y_i \mathbf{w}^T x_i > 0 \ \text{ then } \mathbf{w} \leftarrow (1 - \eta)\mathbf{w} + \eta C y_i x_i$$

収束速度が遅い

理論的決め方なし

- Coordinate Descentのように双対化して解く

$$\mathbf{u} \leftarrow \mathbf{u} + y_i \left(\alpha_i - \overline{\alpha}_i \right) \mathbf{x}_i \qquad \text{(CD30)}$$

ηを決める必要なし

- 収束の遅さを改善するために高次情報を使う

$$\mathbf{w} \leftarrow \mathbf{w} - \eta H^{-1} \nabla^{S} \left(\frac{1}{2} \|\mathbf{w}\|_2^2 + C \max\left(0, 1 - y_i \mathbf{w}^T x_i\right) \right)$$

$$H = \nabla^2 \left(\frac{1}{2} \|\mathbf{w}\|_2^2 + C \max\left(0, 1 - y_i \mathbf{w}^T x_i\right) \right) \Rightarrow$$ 対角行列になることが多い

$\Rightarrow H^{-1}$の計算が重くない

バッチ学習による方法

- ディスクアクセスを減らすために一度のあるまとまりのデータ(＝ブロック)を読み込み学習
- ブロックBだけではなく、メモリにcacheデータも残す

$$\min_{\boldsymbol{\alpha}} f^D(\boldsymbol{\alpha}) \equiv \frac{1}{2}\boldsymbol{\alpha}^T Q\,\boldsymbol{\alpha} - \mathbf{e}^T\boldsymbol{\alpha} \quad \text{subject to} \quad 0 \le \alpha_i \le C \quad \forall i = 1,\ldots,l$$

where $Q_{ij} \equiv y_i y_j \mathbf{x}_i^T \mathbf{x}_j$ を解くために

ブロックBは全データ$=\{B_1,\cdots B_m\}$から1個づつ選んで最適化

条件：$0 \le \alpha_i + d_i \le C$ for $\forall i \in B$ and $d_i = 0$ for $\forall i \notin B$

$$\min_{\mathbf{d}} f^D(\boldsymbol{\alpha}+\mathbf{d}) - f^D(\boldsymbol{\alpha}) \equiv \frac{1}{2}\mathbf{d}_B^T Q_{BB}\mathbf{d}_B + \mathbf{d}_B^T(Q\boldsymbol{\alpha}-\mathbf{e})_B$$

$$= \frac{1}{2}\mathbf{d}_B^T Q_{BB}\mathbf{d}_B + \sum_{i\in B} y_i d_i\left(\mathbf{u}^T x_i\right) - \mathbf{d}_B^T\mathbf{e}_B \quad \text{where } \mathbf{u} = \sum_{i=1}^{l} y_i \alpha_i \mathbf{x}_i$$

Q_{BB}はQのうちBに関する部分行列