1 Confidence-Weighted Learning

The main idea of Confidence-Weighted algorithms is to assume a Guassian distribution of the linear classifier $\omega \sim N(\boldsymbol{\mu}, \Sigma)$. Each update tries to stay close to the previous distrbution and ensure the probability of the current precition for $\boldsymbol{x}_i$ is larger than η.

$$\begin{aligned} Pr[y_i(\boldsymbol{\omega} \cdot \boldsymbol{x}_i) \geq 0] \geq \eta \quad or \\ y_i(\boldsymbol{\mu} \cdot \boldsymbol{x}_i \geq \phi\sqrt{\boldsymbol{x}_i^T \Sigma \boldsymbol{x}_i} \end{aligned} \tag{1}$$

This constraint can be formulated with a loss function so that:

$$l(N(\boldsymbol{\mu}, \Sigma); (\boldsymbol{x}_t, y_t)) = max(0, \phi\sqrt{\boldsymbol{x}_t^T \Sigma \boldsymbol{x}_t}) = 0 \tag{2}$$

Typical Confidence-Weighted learning algorithms (CW, AROW, SCW) take different approaches to the above constraint. CW release the above non-convex constraint (in Σ) to be

$$y_i(\boldsymbol{\mu} \cdot \boldsymbol{x}_i) \geq \phi\sqrt{\boldsymbol{x}_i^T \Sigma \boldsymbol{x}_i}$$

. Exact solves this problem directly by replacing Σ with $\Sigma = \Upsilon^2$, as Σ is positive definite. The problem of these two approaches is that it is sensitive to label noise. To tackle such limitation, AROW relaxes the above constraint by setting a regularizer with the form

$$\lambda_1 l_h^2(y_t, \boldsymbol{\mu} \cdot \boldsymbol{x}) + \lambda_2 \boldsymbol{x}_t^T \Sigma \boldsymbol{x}_t$$

. SCW relaxes Equation 2 with the same way to PA-I and PA-II. Anyway, they follow the same loss function.

Despite the different approaches to tackle the constraint, these algorithms share the same updating rule and only differ in the calculation of α_t and β_t as follows:

$$\begin{aligned} \boldsymbol{\mu}_t = \boldsymbol{\mu}_{t-1} + \alpha_t \Sigma_{t-1} y_t \boldsymbol{x}_t \\ \Sigma_t = \Sigma_{t-1} - \beta_t \Sigma_{t-1} \boldsymbol{x}_t \boldsymbol{x}_t^T \Sigma_{t-1} \end{aligned} \tag{3}$$

In the classical optimization problem

$$\min_{\omega} f(\omega)$$

, the search direction p_k is obtained by:

- Gradient Descent:
$$p_k = -\nabla_k$$
- Quasi-Newton Methods: $p_k = -B_k^{-1} \nabla_k$

In online setting, the gradient descent algorithm takes the same form. The difference is that, the step size is replaced by learning rate. First order online learning algorithms treat all the coordinates the same with constant or decaying learning rates. On the contrary, second order online learning algorithms yield a dedicated step size for each coordinate. Confidence-Weighted learning algorithms approach this by the Confidence matrix Σ (we only consider the diagonal confidence matrix), as we see in Equation 3.

For sparse online, previous learning algorithms treat all the coordinates the same by adding an regularization term $\lambda|\omega|$. In this work, we extend the regularization term to be **coordinate dependent**. We take the form of FOBOS.

$$\begin{aligned} \boldsymbol{u}_{t+\frac{1}{2}} &= \boldsymbol{u}_t - \boldsymbol{\eta}_t \cdot \boldsymbol{g}_t^f \\ \boldsymbol{u}_{t+1} = \arg\min_w \frac{1}{2}\|\boldsymbol{u} - \boldsymbol{u}_{t+\frac{1}{2}}\|^2 &+ \boldsymbol{\eta}_t \cdot \lambda \boldsymbol{u} \\ \boldsymbol{\eta}_t &= \eta_t \cdot \Sigma_t \end{aligned} \tag{4}$$

The final updating rule goes to:

$$u_{t+1,j} = sign(u_{t,j} - \eta_t \Sigma_{jj} g_{t,j}^f)[|u_{t,j} - \eta_t \Sigma_{jj} g_{t,j}^f)| - \eta_t \lambda \Sigma_{jj}] \tag{5}$$