09_Logistic_Regression

9.1 Binary Classification

9.1.1 Problem Setup

Given

- Two classes $C = \{c_1, c_2\}$
- A particular vector $\mathbf{x} \in \mathbb{R}^d$

  Do
- Assign the vector $\mathbf{x}$ to one of the two classes.
  - Namely, to calculate the probability of $\mathbf{x} \in c_i$ for all $c_i \in C$.

From the Bayesian Rule, we derive that

$$P(c_1|\mathbf{x}) = \frac{P(\mathbf{x}|c_1)P(c_1)}{P(\mathbf{x})}$$

- $= \frac{P(\mathbf{x}|c_1)P(c_1)}{P(\mathbf{x}|c_1)P(c_1) + P(\mathbf{x}|c_2)P(c_2)}$
- $= \frac{1}{1 + \frac{P(\mathbf{x}|c_2)P(c_2)}{P(\mathbf{x}|c_1)P(c_1)}}$
- $= \frac{1}{1 + e^{-\ln\left[\frac{P(\mathbf{x}|c_1)}{P(\mathbf{x}|c_2)}\right] - \ln\left[\frac{P(c_1)}{P(c_2)}\right]}}$

which can be written in the form of a logistic function:

$$P(c_1|\mathbf{x}) = \frac{1}{1 + e^{-\xi}}$$

where,

$$\xi = \ln\left[\frac{P(\mathbf{x}|c_1)}{P(\mathbf{x}|c_2)}\right] + \ln\left[\frac{P(c_1)}{P(c_2)}\right]$$

- Likelihood Ratio: $\ln\left[\frac{P(\mathbf{x}|c_1)}{P(\mathbf{x}|c_2)}\right]$
- Prior Ratio: $\ln\left[\frac{P(c_1)}{P(c_2)}\right]$

9.1.2 Multivariate Gaussian Distribution

Assumed that, within each class, the multi-variate input vector $\mathbf{x}$ follows a Gaussian Distribution with a common covariate Σ.

$$P(\mathbf{x}|c_i) = \frac{1}{(2\pi)^{\frac{d}{2}}|\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(\mathbf{x}-\mu_i)^\top \Sigma^{-1}(\mathbf{x}-\mu_i)}$$

From which we derive that

- $ln\left[\frac{P(\mathbf{x}|c_1)}{P(\mathbf{x}|c_2)}\right]$
- $= \ln\left[\frac{e^{-\frac{1}{2}(\mathbf{x}-\mu_1)^\top\Sigma^{-1}(\mathbf{x}-\mu_1)}}{e^{-\frac{1}{2}(\mathbf{x}-\mu_2)^\top\Sigma^{-1}(\mathbf{x}-\mu_2)}}\right]$
- $= \frac{1}{2}(\mathbf{x}-\mu_2)^\top\Sigma^{-1}(\mathbf{x}-\mu_2) - \frac{1}{2}(\mathbf{x}-\mu_1)^\top\Sigma^{-1}(\mathbf{x}-\mu_1)$
- $= \frac{1}{2}(\mathbf{x}^\top\Sigma^{-1} - \mu_2^\top\Sigma^{-1})(\mathbf{x}-\mu_2) - \frac{1}{2}(\mathbf{x}^\top\Sigma^{-1} - \mu_1^\top\Sigma^{-1})(\mathbf{x}-\mu_1)$
- $= \frac{1}{2}(\mathbf{x}^\top\Sigma^{-1}\mathbf{x} - \mathbf{x}^\top\Sigma^{-1}\mu_2 - \mu_2^\top\Sigma^{-1}\mathbf{x} + \mu_2^\top\Sigma^{-1}\mu_2)$
  $-\frac{1}{2}(\mathbf{x}^\top\Sigma^{-1}\mathbf{x} - \mathbf{x}^\top\Sigma^{-1}\mu_1 - \mu_1^\top\Sigma^{-1}\mathbf{x} + \mu_1^\top\Sigma^{-1}\mu_1)$
- $= \frac{1}{2}\left[\mathbf{x}^\top\Sigma^{-1}(\mu_1-\mu_2) + (\mu_1^\top - \mu_2^\top)\Sigma^{-1}\mathbf{x} + (\mu_2^\top + \mu_1^\top)\Sigma^{-1}(\mu_2-\mu_1)\right]$
- $= (\mu_1-\mu_2)^\top\Sigma^{-1}\mathbf{x} + \frac{1}{2}(\mu_2+\mu_1)^\top\Sigma^{-1}(\mu_2-\mu_1)$

Therefore, the exponential ξ can be rewritten as:

- $\xi = (\mu_1-\mu_2)^\top\Sigma^{-1}\mathbf{x} + \frac{1}{2}(\mu_2+\mu_1)^\top\Sigma^{-1}(\mu_2-\mu_1) + \ln\left[\frac{P(c_1)}{P(c_2)}\right]$
- ★ $= \left[\Sigma^{-1}(\mu_1-\mu_2)\right]^\top\mathbf{x} + \left(\frac{1}{2}(\mu_2+\mu_1)^\top\Sigma^{-1}(\mu_2-\mu_1) + \ln\left[\frac{P(c_1)}{P(c_2)}\right]\right)$, with $(\Sigma^{-1})^\top = \Sigma^{-1}$ known.
- $= \mathbf{w}^\top\mathbf{x} + b$

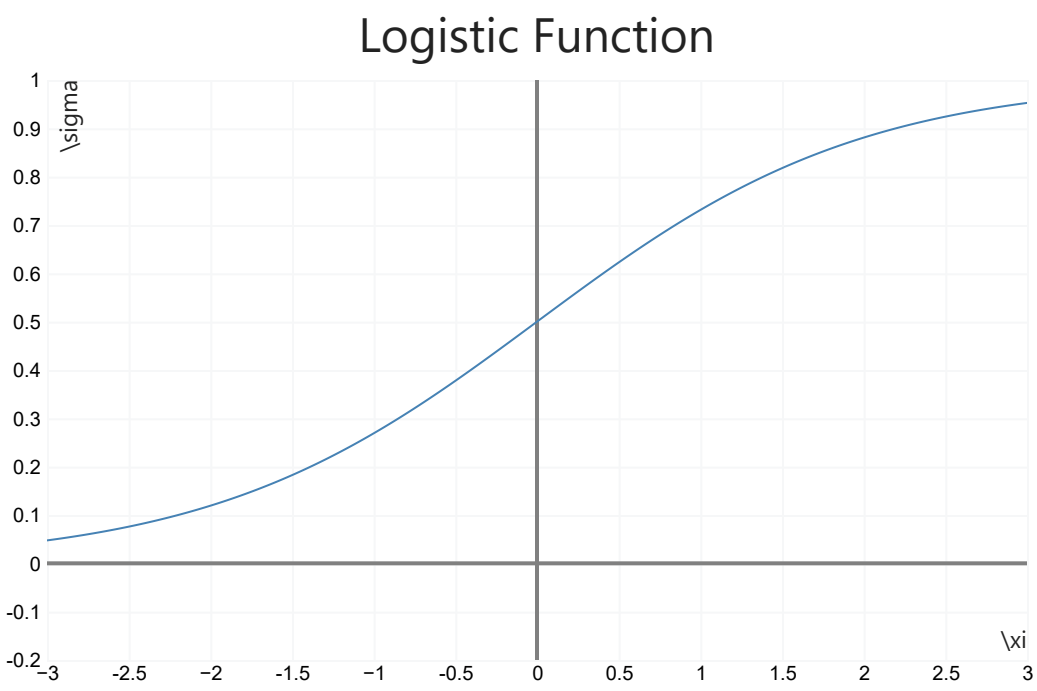
In conclusion,

$$P(c_1|\mathbf{x}) = \frac{1}{1+e^{-(\mathbf{w}^\top\mathbf{x}+b)}}$$

where

- $\mathbf{w} = \Sigma^{-1}(\mu_1-\mu_2)$, and
- $b = \frac{1}{2}(\mu_2+\mu_1)^\top\Sigma^{-1}(\mu_2-\mu_1) + \ln\left[\frac{P(c_1)}{P(c_2)}\right]$

Properties of Logistic Functions



1. Limits
    1. $\lim_{\xi \to -\infty} \sigma(\xi) = 0$
    2. $\lim_{\xi \to \infty} \sigma(\xi) = 1$
2. Central Symmetry:
    1. $\sigma(-\xi) = 1 - \sigma(\xi)$
3. Derivative:
    1. $\dfrac{d\Big[\sigma(\xi)\Big]}{d\xi} = \sigma(\xi)\sigma(-\xi) = \sigma(\xi)\Big(1 - \sigma(\xi)\Big)$

9.1.3 Maximum Likelihood Formulation

Recall: Bernoulli Distribution

Suppose that a variable Y confronts a Bernoulli Distribution,

- i.e., $Y \sim \text{Bernoulli}(p)$
- where p is the probabilistic of being Success,
  the probabilistic distribution function is

$$P(Y = y) = \begin{cases} p & \text{if } y = 1 \\ 1 - p & \text{if } y = 0 \end{cases}$$

Convert σ to Probability

We want to predict a binary output $y_n \in \{0, 1\}$ from an input $\mathbf{x}_n$. From above, we know that the logistic regression has the form:

$$y_n = \sigma(\mathbf{w}^\top \mathbf{x}_n) + \epsilon_n$$

where

$$\sigma(\xi) = \frac{1}{1+e^{-\xi}}$$

We model input-output by a conditional Bernoulli Distribution:

$$P(y_n = y|\mathbf{x}_n) = \begin{cases} \sigma(\mathbf{w}^\top \mathbf{x}_n) & \text{if } y = 1 \\ 1 - \sigma(\mathbf{w}^\top \mathbf{x}_n) & \text{if } y = 0 \end{cases}$$

Bernoulli Distribution Modelling

Given $\{(\mathbf{x}_n, y_n)|n = 1, \dots, N\}$, the likelihood is given by

$$P(\mathbf{y}|\mathbf{X}, \mathbf{w}) = \prod_{n=1}^{N} P(y_n|\mathbf{x}_n)$$

$$= \prod_{n=1}^{N} p(y_n = 1|\mathbf{x}_n)^{y_n} \Big(1 - P(y_n = 1|\mathbf{x}_n)\Big)^{1-y_n}$$

$$= \prod_{n=1}^{N} \sigma(\mathbf{w}^\top \mathbf{x}_n)^{y_n} \Big(1 - \sigma(\mathbf{w}^\top \mathbf{x}_n)\Big)^{1-y_n}$$

The log-likelihood function is thus given by:

$$\mathcal{L}(\mathbf{y}|\mathbf{X}, \mathbf{w}) = \sum_{n=1}^{n} \log P(y_n|\mathbf{x}_n)$$

$$= \sum_{n=1}^{N} \Big[y_n \log\Big(\sigma(\mathbf{w}^\top \mathbf{x}_n)\Big) + (1 - y_n) \log\Big(1 - \sigma(\mathbf{w}^\top \mathbf{x}_n)\Big) \Big]$$

We want to maximize this log-likelihood $\mathcal{L}$. However, the calculation of the maximum of nonlinear function of $\mathbf{w}$ cannot be done in a closed form. That is, it is very costly to directly compute:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = 0$$

Therefore, an iterated re-weighted least squares (IRLS) is then performed, derived from the Newton's method.

9.2 Math Basics

9.2.1 Gradient 梯度

Consider a real-valued function $f(x)$, which takes a real-valued vector $\mathbf{x} \in \mathbb{R}^d$ as an input:

$$f(\mathbf{x}) : \mathbb{R}^d \mapsto \mathbb{R}$$

The gradient of $f(\mathbf{x})$ is defined by:

$$\nabla f(\mathbf{x}) = \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_d} \end{bmatrix}$$

Which is the partial derivative of $f(\mathbf{x})$ with respect to all the dimensions of $\mathbf{x}$.

9.2.2 Hessian Matrix 海森矩阵

If $f(\mathbf{x})$ belongs to the class C^2, the Hessian matrix $\mathbf{H}$ is defined as the symmetric matrix with the combination of any two dimensions.

$$\begin{aligned} \mathbf{H} &= \nabla^2 f(\mathbf{x}) \\ &= \left[\frac{\partial^2 f(\mathbf{x})}{\partial x_i \partial x_j} \right] \\ &= \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_d} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_d} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_d \partial x_1} & \frac{\partial^2 f}{\partial x_d \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_d^2} \end{bmatrix} \\ &= \frac{\partial}{\partial \mathbf{x}} \left[\frac{\partial f}{\partial \mathbf{x}} \right]^\top \\ &= \frac{\partial}{\partial \mathbf{x}} [\nabla f(\mathbf{x})]^\top \end{aligned}$$

Namely,

$$\mathbf{H} = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_d} \end{bmatrix} \begin{bmatrix} \frac{\partial f}{\partial x_1} & \frac{\partial f}{\partial x_2} & \cdots & \frac{\partial f}{\partial x_d} \end{bmatrix}$$

The Hessian matrix can not only help us find the extreme points of the function (through the first-order derivative is 0), but also determine whether the point is a minimum, maximum or saddle point by analysing the curvature of the function near the extreme point. For example, if the Hessian matrix is positive definite, it means that the extreme point is a local minimum.

9.2.3 Gradient Descent/Ascent 梯度下降、上升

The gradient descent/ascent learning is a simple first-order iterative method for minimization/maximization.

Gradient Descent: Iterative Minimization

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} - \eta\left(\frac{\partial \mathcal{J}}{\partial \mathbf{w}}\right)$$

Gradient Ascent: Iterative Maximization

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \eta\left(\frac{\partial \mathcal{J}}{\partial \mathbf{w}}\right)$$

where the learning rate $\eta > 0$.

9.2.4 Newton's Method

The Basic idea of Newton's method is:

- To optimize the quadratic (二次的) approximation,
- of the objective function $\mathcal{J}(\mathbf{w})$,
- around the current point $\mathbf{w}^{(k)}$.

Tayler Series of $\mathcal{J}(\mathbf{w})$

The Taylor Series of a function $f(x) : \mathbb{R} \mapsto \mathbb{R}$ around a point a yields:

$$f(x) = f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \frac{f'''(a)}{3!}(x-a)^3 + \cdots = \sum_{i=0}^{\infty} \frac{f^{(i)}(a)}{i!}(x-a)^i$$

The second-order Taylor series expansion of $\mathcal{J}(\mathbf{w})$ at the current $\mathbf{w}^{(k)}$ gives:

$$\mathcal{J}_2(\mathbf{w}) = \mathcal{J}(\mathbf{w}^{(k)}) + \left[\nabla \mathcal{J}(\mathbf{w}^{(k)})\right]^\top (\mathbf{w} - \mathbf{w}^{(k)}) + \frac{1}{2}\left(\mathbf{w} - \mathbf{w}^{(k)}\right)^\top \left(\nabla^2 \mathcal{J}(\mathbf{w}^{(k)})\right)(\mathbf{w} - \mathbf{w}^{(k)})$$

Specifically:

- First term: $\mathcal{J}(\mathbf{w}^{(k)})$, is the value of the target function $\mathcal{J}$ with respect to the current $\mathbf{w}^{(k)}$.
- Second term: $\nabla \mathcal{J}(\mathbf{w}^{(k)})$ is the *gradient* of the target function at the current $\mathbf{w}^{(k)}$.
- Third Term: $\nabla^2 \mathcal{J}(\mathbf{w}^{(k)})$ is the *Hessian Matrix* describing the target function's 2nd order derivative at current $\mathbf{w}^{(k)}$.

Differentiation

Differentiate the above second-order Taylor Series with respect to $\mathbf{w}$:

$$\nabla \mathcal{J}(\mathbf{w}) = 0 + \nabla \mathcal{J}(\mathbf{w}^{(k)}) + \nabla^2 \mathcal{J}(\mathbf{w}^{(k)})(\mathbf{w} - \mathbf{w}^{(k)})$$

Set this equal to 0:

$$\nabla \mathcal{J}(\mathbf{w}^{(k)}) + \nabla^2 \mathcal{J}(\mathbf{w}^{(k)})(\mathbf{w} - \mathbf{w}^{(k)}) = 0$$

$$\implies \nabla \mathcal{J}(\mathbf{w}^{(k)}) + \nabla^2 \mathcal{J}(\mathbf{w}^{(k)})\mathbf{w} - \nabla^2 \mathcal{J}(\mathbf{w}^{(k)})\mathbf{w}^{(k)} = 0$$

$$\implies \nabla^2 \mathcal{J}(\mathbf{w}^{(k)})\mathbf{w} = \nabla^2 \mathcal{J}(\mathbf{w}^{(k)})\mathbf{w}^{(k)} - \nabla \mathcal{J}(\mathbf{w}^{(k)})$$

$$\implies \mathbf{w} = \mathbf{w}^{(k)} - \left[\nabla^2 \mathcal{J}(\mathbf{w}^{(k)})\right]^{-1} \nabla \mathcal{J}(\mathbf{w}^{(k)})$$

9.3 Logistic Regression Algorithms

Remark: We are learning a weight $\mathbf{w}$ such that:

- The log-likelihood $\mathcal{L}(\mathbf{y}|\mathbf{X},\mathbf{w})$ is minimize/maximized.
- That is $\dfrac{\partial \mathcal{L}}{\partial \mathbf{w}} = 0$.

The gradient Ascent Learning has the form

$$\mathbf{w}^{\text{new}} = \mathbf{w}^{\text{old}} + \eta\Big(\frac{\partial \mathcal{L}}{\partial \mathbf{w}}\Big)$$

9.3.1 Calculate Gradient

Recall the log-likelihood:

$$\mathcal{L} = \sum_{n=1}^{N}\Big[y_n \log\Big(\sigma(\mathbf{w}^\top \mathbf{x}_n)\Big) + (1-y_n)\log\Big(1-\sigma(\mathbf{w}^\top \mathbf{x}_n)\Big)\Big]$$

Calculate:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = \sum_{n=1}^{N}\Big[y_n \frac{\sigma_n'}{\sigma_n}\mathbf{x}_n + (1-y_n)\frac{-\sigma_n}{(1-\sigma_n)}\mathbf{x}_n\Big]$$

By using the 2nd and 3rd property of the logistic function σ, we obtain:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = \sum_{n=1}^{N}\Big[y_n \frac{\sigma_n(1-\sigma_n)}{\sigma_n}\mathbf{x}_n + (1-y_n)\frac{-\sigma_n(1-\sigma_n)}{1-\sigma_n}\mathbf{x}_n\Big]$$

$$= \sum_{n=1}^{N}\Big[y_n(1-\sigma_n)\mathbf{x}_n - (1-y_n)\sigma_n\mathbf{x}_n\Big]$$

$$= \sum_{n=1}^{N}\Big[y_n(1-\sigma_n) - (1-y_n)\sigma_n\Big]\mathbf{x}_n$$

$$= \sum_{n=1}^{N}\Big[y_n - y_n\sigma_n - \sigma_n + y_n\sigma_n\Big]\mathbf{x}_n$$

$$= \sum_{n=1}^{N}(y_n - \sigma_n)\mathbf{x}_n$$

Lastly, it could be concluded that:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = \sum_{n=1}^{N}\Big(y_n - \sigma(\mathbf{w}^\top \mathbf{x}_n)\Big)\mathbf{x}_n$$

As discussed before, it is a vector with the same shape of $\mathbf{x}_n$.

9.3.2 Calculate Hessian

Calculate the Hessian:

$$\mathbf{H} = \nabla^2 \mathcal{L}$$

Differentiate every term in the gradient:

$$\begin{aligned}
&\frac{\partial}{\partial \mathbf{w}}(y_n - \sigma(\mathbf{w}^\top \mathbf{x}_n))\mathbf{x_n} \\
&= \frac{\partial}{\partial \mathbf{w}} y_n \mathbf{x}_n - \frac{\partial}{\partial \mathbf{w}} \sigma_n \mathbf{x}_n \\
&= -\sigma_n(1-\sigma_n)\mathbf{x}_n \mathbf{x}_n^\top
\end{aligned}$$

Combining all the terms:

$$\nabla^2 \mathcal{L} = \sum_{n=1}^{N} -\sigma_n(1-\sigma_n)\mathbf{x}_n \mathbf{x}_n^\top$$

9.3.3 Objective Function

Notice that the original Log-Likelihood:

$$\mathcal{L}(\mathbf{y}|\mathbf{X}, \mathbf{w}) = \sum_{n=1}^{n} \log P(y_n|\mathbf{x}_n)$$

is *negative* since the probability $P(y_n|\mathbf{x}_n)$ is lower than 1.

Therefore, we set the objective function $\mathcal{J}(\mathbf{w})$ to be the negative log-likelihood:

$$\mathcal{J}(\mathbf{w}) = -\mathcal{L}(\mathbf{w}) = -\sum_{n=1}^{N} \Big[y_n \log\Big(\sigma(\mathbf{w}^\top \mathbf{x}_n)\Big) + (1-y_n)\log\Big(1-\sigma(\mathbf{w}^\top \mathbf{x}_n)\Big)\Big]$$

Therefore,

- The gradient: $\nabla \mathcal{J}(\mathbf{w}) = -\sum_{n=1}^{N}(y_n - \sigma_n)\mathbf{x}_n$
- The Hessian: $\nabla^2 \mathcal{J}(\mathbf{w}) = \sum_{n=1}^{N} \sigma_n(1-\sigma_n)\mathbf{x}_n \mathbf{x_n}^\top$

Thus, the *update* part of the Newton's method $\eta\Big(\frac{\partial \mathcal{L}}{\partial \mathbf{w}}\Big)$ has the form:

$$\Delta \mathbf{w} = -\Big[\sum_{n}^{N} \sigma_n(1-\sigma_n)\mathbf{x}_n \mathbf{x}_n^\top\Big]^{-1}\Big[-\sum_{n=1}^{N}(y_n - \sigma_n)\mathbf{x}_n\Big]$$

Namely,

$$\Delta \mathbf{w} = \Big(\mathbf{X} S \mathbf{X}^\top\Big)^{-1} S b$$

where:

- $S = \begin{bmatrix} \sigma_1(1-\sigma_1) & 0 & \cdots & 0 \\ 0 & \sigma_2(1-\sigma_2) & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & \sigma_n(1-\sigma_n) \end{bmatrix}$
- $b = \begin{bmatrix} \frac{y_1-\sigma_1}{\sigma_1(1-\sigma_1)} \\ \frac{y_2-\sigma_2}{\sigma_2(1-\sigma_2)} \\ \vdots \\ \frac{y_N-\sigma_N}{\sigma_N(1-\sigma_N)} \end{bmatrix}$

9.3.4 Recap: IRLS Algorithm

Input

- $\{(\mathbf{x}_n, y_n)|n = 1, 2, \cdots, N\}$

Do

1. Initialize $\mathbf{w} = 0$ and $w_0 = \log \frac{\bar{\mathbf{y}}}{1-\bar{\mathbf{y}}}$
2. Repeat until convergence:
   1. for $n = 1, 2, \cdots, N$ do:
   1. Compute $\sigma_n = \sigma(\mathbf{w}^\top \mathbf{x}_n + w_0)$
   2. Compute $s_n = \sigma_n(1-\sigma_n)$
   3. Compute $b_n = \frac{y_n-\sigma_n}{s_n}$
   2. Construct $S = diag(s_{1:N})$
   3. Update $\mathbf{w} = (XSX^\top)Sb$

Output

- $\mathbf{w}$