Week 10: Logistic Regression

Table of Contents

Soft Binary Classification

1. Similar form as (hard) binary classification, but interested in the **probability** rather than the exact ± 1
2. Target function $f(x) = P(+1|x) \in [0, 1]$
3. Can be thought of as hard binary classification, just with noise that shifts prediction away from ± 1 into range $[0, 1]$

Logistic hypothesis

1. For features $\mathbf{x} = (x_0, x_1, x_2, \ldots x_d)$, the goal is to obtain a *weighted* score s, where

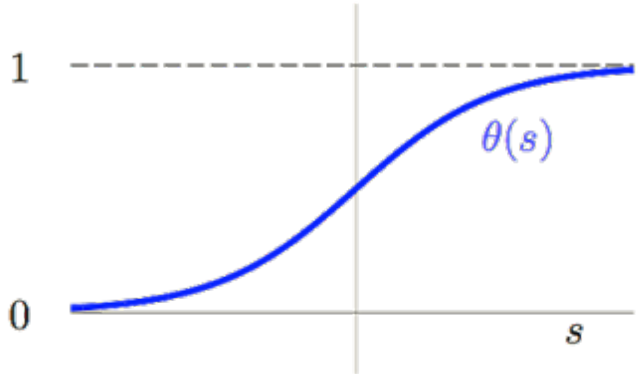
$$s = \sum_{i=0}^{d} w_i x_i = w^T x$$

    - **Logistic function** $\theta(s)$ converts the score into *estimated probability* between 0 and 1
2. For such target functions, the corresponding logistic hypotheis is

$$h(\mathbf{x}) = \theta(\mathbf{w}^{\mathbf{T}} \mathbf{x})$$

3. Given that score s has value range $(-\infty, \infty)$, and the target function demands a value mapped to $[0, 1]$. logistic function has form:

$$\theta(s) = \frac{e^s}{1 + e^s} = \frac{1}{1 + e^{-s}}$$



4. Logistic function $\theta(s)$ is:
   - Smooth
   - Sigmoid
   - Monotonic
5. Substituting logistic function into logistic hypothesis uses:

$$h(\mathbf{x}) = \frac{1}{1 + exp(-\mathbf{w}^T \mathbf{x})}$$

   to approximate target function $f(\mathbf{x}) = P(+1|\mathbf{x})$

Evaluating Logistic Regression Error

1. Optimizing logistic regression through squared error does not work because:
   - Under logistic regression hypothesis, squared error becomes

$$err(h, x_n, y_n) = \begin{cases} (\theta(w^T x) - 0)^2 & y_n = 0 \\ (1 - \theta(w^T x))^2 & y_n = 1 \end{cases}$$
$$= y_n(1 - \theta(w^T x))^2 + (1 - y_n)\theta^2(w^T x)$$

   - This is a **non-convex** function, for which a global minimum is difficult to find
2. Instead, error function of logistic regression is expressed as **likelihood**

Logistic Regression Likelihood

1. Given target function of logistic regression, $f(x) = P(+1|x)$

$$f(x) = P(+1|x) \Leftrightarrow P(y|x) = \begin{cases} f(x) & \text{for } y = +1 \\ 1 - f(x) & \text{for } y = -1 \end{cases}$$

2. For a given sample $\mathcal{D} = [(x_1, +1), (x_2, -1), \ldots, (x_N, -1)]$, the probability that target function f and hypothesis h give correct value of y for every point in $\mathcal{D}$ is:

probability that f generates $\mathcal{D}$

$$P(\mathbf{x}_1)P(\circ|\mathbf{x}_1) \times$$
$$P(\mathbf{x}_2)P(\times|\mathbf{x}_2) \times$$
$$\ldots$$
$$P(\mathbf{x}_N)P(\times|\mathbf{x}_N)$$

likelihood that h generates $\mathcal{D}$

$$P(\mathbf{x}_1)h(\mathbf{x}_1) \times$$
$$P(\mathbf{x}_2)(1 - h(\mathbf{x}_2)) \times$$
$$\ldots$$
$$P(\mathbf{x}_N)(1 - h(\mathbf{x}_N))$$

probability that f generates $\mathcal{D}$

$$P(\mathbf{x}_1)f(\mathbf{x}_1) \times$$
$$P(\mathbf{x}_2)(1 - f(\mathbf{x}_2)) \times$$
$$\ldots$$
$$P(\mathbf{x}_N)(1 - f(\mathbf{x}_N))$$

likelihood that h generates $\mathcal{D}$

$$P(\mathbf{x}_1)h(\mathbf{x}_1) \times$$
$$P(\mathbf{x}_2)(1 - h(\mathbf{x}_2)) \times$$
$$\ldots$$
$$P(\mathbf{x}_N)(1 - h(\mathbf{x}_N))$$

3. If hypothesis h is a *good approximation* of target function f, we expect the probability (f) and likelihood (h) with respect to training data set $\mathcal{D}$ to be similar, and large values (since f is the true representation of population, where $\mathcal{D}$ is sampled from)

$$\text{if } h \approx f, \text{ then } \; likelihood(h) \approx (\text{probability using } f) \approx \text{large}$$

4. Logistic regression can therefore be optimized by **maximizing likelihood function**

$$g = \underset{h}{argmax}\ likelihood(h)$$

5. Substituting in $h(x) = \theta(w^T x)$, and recall that logistic regression is symmetric $\rightarrow$ $1 - h(x) = h(-x)$, there is

$$\begin{aligned} likelihood(h) &= P(x_1)h(x_1) \times P(x_2)(1 - h(x_2)) \times \ldots \times P(x_N)(1 - h(x_N)) \\ &= P(x_1)h(x_1) \times P(x_2)(h(-x_2)) \times \ldots \times P(x_N)(h(-x_N)) \\ &= P(x_1)h(x_1) \times P(x_2)(h(y_2 x_2)) \times \ldots \times P(x_N)(h(y_N x_N)) \end{aligned}$$

6. The likelihood provided by logistic hypothesis h is therefore **proportional** to the **product** of all y and x

$$likelihood(logistic\ h) \propto \prod_{n=1}^{N} h(y_n x_n)$$

Cross-Entropy Error

1. Given that larger the likelihood for logistic hypothesis h to generate training set $\mathcal{D}$, the better it approximates target function f, the goal of optimization is to **maximize** $likelihood(logistic\ h)$

$$\max_{h} \ likelihood(logistic\ h) \propto \prod_{n=1}^{N} h(y_n x_n)$$

Substituting h with definition of logistic hypothesis $h(\mathbf{x}) = \theta(\mathbf{w}^{\mathbf{T}}\mathbf{x})$

$$\max_{w} \ likelihood(w) \propto \prod_{n=1}^{N} \theta(y_n w^T x_n)$$

Add log to convert product to sum through logarithm product rule

$$\propto ln \prod_{n=1}^{N} \theta(y_n w^T x_n)$$

$$\propto \frac{1}{N} \sum_{n=1}^{N} ln\, \theta(y_n w^T x_n)$$

Maximizing equation above is equivalent to minimizing its negative:

$$\min_{w} \frac{1}{N} \sum_{n=1}^{N} -ln\, \theta(y_n w^T x_n)$$

Substituting definition of logistic function function $\theta(s) = \dfrac{1}{1 + exp(-s)}$

$$\Rightarrow \min_{w} \frac{1}{N} \sum_{n=1}^{N} ln(1 + exp(-y_n w^T x_n))$$

$$\Rightarrow \min_{w} \frac{1}{N} \underbrace{\sum_{n=1}^{N} \mathrm{err}(w, x_n, y_n)}_{E_{in}(w)}$$

2. Error function derived above is known as **cross-entropy error**

$$err(w, x, y) = ln(1 + exp(-ywx))$$

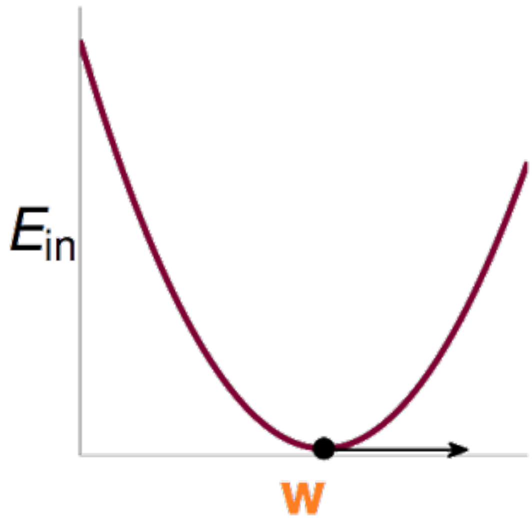
cross-entropy error

Minimize $E_{in}(w)$ for Logistic Regression

1. Given error function, the cost function $E_i n(w)$ of logistic regression is:

$$E_{in}(w) = \frac{1}{N} \sum_{n=1}^{N} ln(1 + exp(-y_n w^T x_n))$$

2. $E_{in}(w)$ is **continuous, differentiable, twice-differentiable, convex**



3. Gradient of the cost function, $\nabla E_{in}(w)$ can be found as partial derivative of $E_{in}(w)$

$$E_{in}(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N} \ln\left(\underbrace{1+\exp(\overbrace{-y_n\mathbf{w}^T\mathbf{x}_n}^{\bigcirc})}_{\square}\right)$$

$$\begin{aligned}\frac{\partial E_{in}(\mathbf{w})}{\partial w_i} &= \frac{1}{N}\sum_{n=1}^{N}\left(\frac{\partial \ln(\square)}{\partial \square}\right)\left(\frac{\partial(1+\exp(\bigcirc))}{\partial \bigcirc}\right)\left(\frac{\partial -y_n\mathbf{w}^T\mathbf{x}_n}{\partial w_i}\right)\\ &= \frac{1}{N}\sum_{n=1}^{N}\left(\frac{1}{\square}\right)\left(\exp(\bigcirc)\right)\left(-y_n x_{n,i}\right)\\ &= \frac{1}{N}\sum_{n=1}^{N}\left(\frac{\exp(\bigcirc)}{1+\exp(\bigcirc)}\right)\left(-y_n x_{n,i}\right) = \frac{1}{N}\sum_{n=1}^{N}\theta(\bigcirc)\left(-y_n x_{n,i}\right)\end{aligned}$$

Treating $x_{n,i}$ in equation above as element of x vector gives:

$$\nabla E_{in}(w) = \frac{1}{N}\sum_{n=1}^{N}\theta(-y_n w^T x_n)(-y_n x_n)$$

4. To minimize $E_{in}(w)$ for logistic regression optimization, we want

$$\nabla E_{in}(w) = \frac{1}{N}\sum_{n=1}^{N}\theta(-y_n w^T x_n)(-y_n x_n) = 0$$

which requires either:

- $\theta(\cdot) = 0$ for all n
  - Possilble only when $y_n w^T x_n \approx \infty \Rightarrow\ w^T x_n givescorrectpredictionof y_n$ **for every n**
    - Only possible when $\mathcal{D}$ is **linear separable**
- Weighted sum = 0
  - Non-linear equation of w without closed-form, analytical solution

5. Iterative optimization (similar to PLA) can be used to find approximation of optimized w
    - For t= 0, 1,..
    - Pick some n, and update w_t by

$$w_{t+1} \leftarrow w_t + \underbrace{1}_{\eta} \cdot \underbrace{(\|sign(w_t^T x_n) \neq y_n\| \cdot y_n x_n)}_{v}$$

        - η is **step size**
            - Step size is set to 1 in equation above, such that it resembles the iterative procedure defined by PLA
        - v is a vector reprsenting the **direction of correction*
            - Assumed to be a *unit vector*
    - When stop condition is reached, return last w as g
    - Choice of (η, v) and stopping condition defines **iterative optimization approach**

Gradient Descent

1. Greedy approach to iterative optimization
    - For some given $\eta > 0$, find

$$\min_{\|v\|=1} E_{in}(\underbrace{w_t + \eta w}_{w_{t+1}})$$

    - Still non-linear optimization, with added constraint
    - Hard to solve directly, handled instead through *local approximation by linear formula*
        - Non-linear curve can be approximated by linear segments within small range

$$E_{in}(w_t + \eta v) \approx E_{in}(w_t) + \eta v^T \nabla E_{in}(w_{in}) \ \text{ for very small } \eta \rightarrow \ \text{Taylor Expansion}$$

2. **Gradient descent** is an **approximate** greedy approach for some given small step size η

$$\min_{\|v\|=1} \underbrace{E_{in}(w_t)}_{known} + \underbrace{\eta}_{\text{given positive}} \ v^T \underbrace{\nabla E_{in}(w_t)}_{known}$$

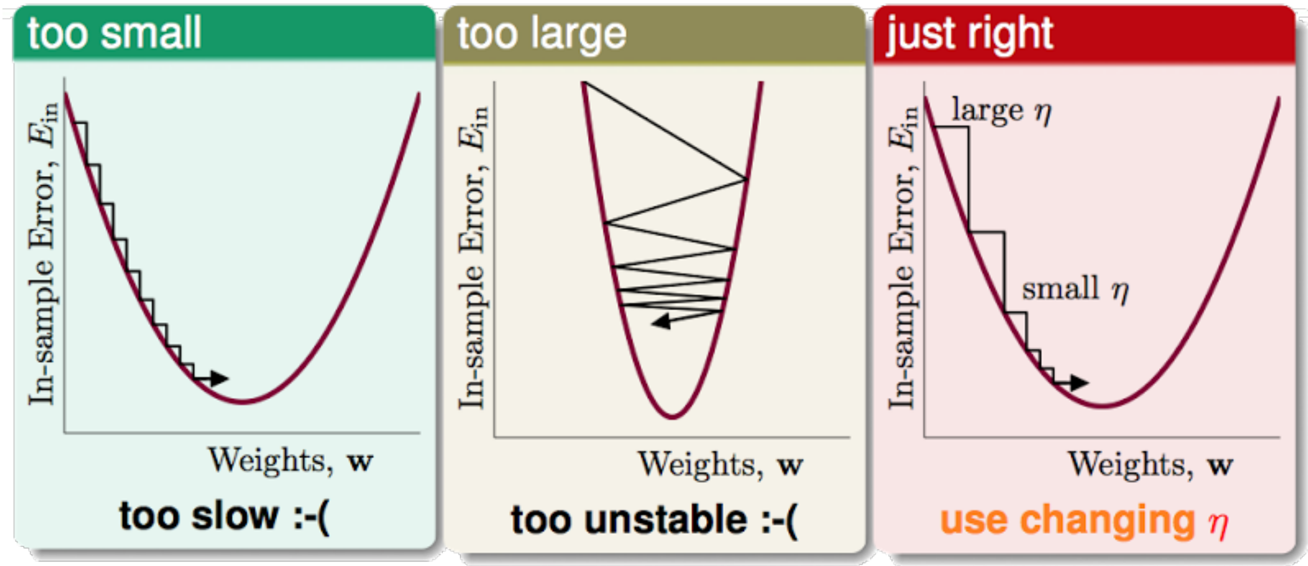
    - Since the end goal is to "correct" E_{in} as much as possible for the given step size, the correction direction should be **opposite** of $\nabla E_{in}(w_t)$, or

$$v = -\frac{\nabla E_{in}(w_t)}{\|\nabla E_{in}(w_t)\|}$$

    - In gradient descent, for **small** η:

$$w_{t+1} \leftarrow w_t - \eta \frac{\nabla E_{in}(w_t)}{\|\nabla E_{in}(w_t)\|}$$

3. Choices of step size η



- Step size too small → Takes long time to converge
- Step size too large → Unstable result, possibly missing out on global minimum and ends up in local minimum
- Step size **monotonic of** $\|\nabla E_{in}(w_t)\|$ → Allow faster convergence at beginning of iterations, and slow down as approaching stopping point

4. Fixed leaerning rate
   - Use *fixed learning rate* η as a way to keep *dynamic learning rate* η monotonic of $\|\nabla E_{in}(w_t)\|$

$$w_{t+1} \leftarrow w_t - \eta \frac{\nabla E_{in}(w_t)}{\|\nabla E_{in}(w_t)\|}$$

$$\|$$

$$w_t - \eta \nabla E_{in}(w_t)$$

5. Fixed learning rate gradient descent for logistic regression
   - Initialize w_0
   - For $t = 0, 1, \ldots$
     - Compute

$$\nabla E_{in}(w) = \frac{1}{N} \sum_{n=1}^{N} \theta(-y_n w^T x_n)(-y_n x_n)$$

     - Update w by

$$w_t - \eta \nabla E_{in}(w_t)$$

     until $\nabla E_{in}(w_{t+1}) = 0$ **or enough iterations**
     - Return *last* w_{t+1} as g

     Fixed learning rate gradient descent has time complexity similar to **Pocket PLA** per iteration