# Classification

We consider a dataset $\{x^{(i)}, t^{(i)}\}_{i=1}^{N}$

$t^{(i)} \in \{0, 2\}$

## probabilistic classifiers

→ logistic regression

$$h_\beta(x) = \sigma\left(\beta_0 + \beta_1 x_1 + \dots + \beta_0 x_0\right)$$

if we take $\sigma(x) = \dfrac{1}{1+e^{-x}}$ (sigmoid)

$\Rightarrow h_\beta(x) \in [0, 1]$ and can be interpreted as a probability

let us define $p(t(x)=1 \mid x) = h_\beta(x)$

$$p(t(x)=0 \mid x) = 1 - h_\beta(x)$$

to train our logistic regression classifier, let us consider

$$p(t(x)=t^{(i)} \mid x) = h_\beta(x)^{t^{(i)}} (1-h_\beta(x))^{1-t^{(i)}}$$

let us consider the whole dataset

$$p(\{t(x^{(i)})=t^{(i)}\}_{i=1}^{N} \mid x^{(i)}) = \prod_{i=1}^{N} h_\beta(x^{(i)})^{t^{(i)}} (1-h_\beta(x^{(i)}))^{1-t^{(i)}}$$

$$\log\left(p\left(\{t(x^{(i)}) = t^{(i)}\}_{i=1}^{N} \mid x^{(i)}\right)\right)$$

$$= \sum_{i=1}^{N} t^{(i)} \log h_\beta(x^{(i)}) + (1 - t^{(i)}) \log\left(1 - h_\beta(x^{(i)})\right)$$

We can then train the model by minimizing $-\log$,

$$\beta^* = \arg\min_\beta \underbrace{-\sum_{i=1}^{N} t^{(i)} \log h_\beta(x^{(i)}) + (1 - t^{(i)}) \log(1 - h_\beta(x^{(i)}))}_{\| \; \ell(\beta)}$$

→ can be solved through gradient descent

for any $\beta_j$, for any sample pair $\{t^{(i)}, x^{(i)}\}$

$$\frac{\partial \ell_i}{\partial \beta_j} = t^{(i)} \frac{\partial}{\partial \beta_j} \log\left(\sigma(\beta^T \tilde{x}^{(i)})\right) + (1-t^{(i)}) \frac{\partial}{\partial \beta_j} \log(1-\sigma($$

$$= t^{(i)} \left( \frac{\frac{\partial}{\partial \beta_j} \sigma(\beta^T \tilde{x}^{(i)})}{\sigma(\beta^T \tilde{x}^{(i)})} \right) + (1-t^{(i)}) \left( -\frac{\partial}{\partial \beta_j} \frac{\sigma(\beta^T \tilde{x}^{(i)})}{(1-\sigma(\dots))} \right)$$

using the chain rule

$$\frac{\partial}{\partial \beta_j} \sigma(\beta^T \tilde{x}^{(i)}) = \sigma'(\beta^T \tilde{x}^{(i)}) \cdot \tilde{x}_j^{(i)}$$

$$\frac{d\sigma(x)}{dx} = \frac{d}{dx} \frac{1}{1+e^{-x}} = \frac{e^{-x}}{(1+e^{-x})^2} = \left(1 - \frac{1}{1+e^{-x}}\right) \frac{1}{1+e^{-x}}$$

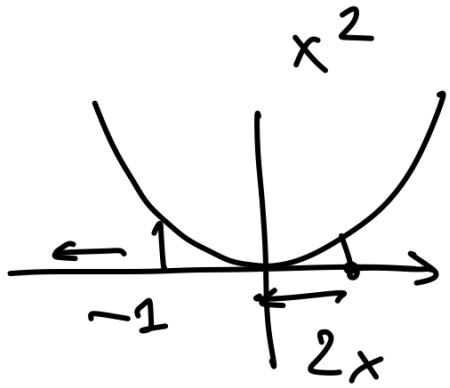
$$\sigma'(x) = \sigma(x)(1-\sigma(x))$$

$$\frac{\partial}{\partial \beta_j}\sigma(\beta^T \tilde{x}^{(i)}) = \tilde{x}_j^{(i)}\,\sigma(\beta^T \tilde{x}^{(i)})(1-\sigma(\beta^T \tilde{x}^{(i)}))$$

$$\frac{\partial \ell^{(i)}}{\partial \beta_j} = t^{(i)}\,\tilde{x}_j^{(i)}\,\frac{\sigma(1-\sigma)}{\sigma} - (1-t^{(i)})\cdot\frac{1}{1-\sigma}\,\tilde{x}_j^{(i)}\,\sigma(1-\sigma)$$

$$= t^{(i)}\,\tilde{x}_j^{(i)}(1-\sigma) - (1-t^{(i)})\,\sigma\,\tilde{x}_j^{(i)}$$

$$= (t^{(i)} - \sigma)\,\tilde{x}_j^{(i)}$$




$2x = -2$

Now we can use update of the form

$$\beta_j^{(k+1)} \leftarrow \beta_j^{(k)} - \eta\,(t^{(i)} - \sigma(\beta^T \tilde{x}^{(i)}))\,\tilde{x}_j^{(i)}$$

SGD (one sample)

$$\beta_j^{(k+1)} \leftarrow \beta_j^{(k)} - \eta \sum_{i=1}^{N} \left(t^{(i)} - \sigma(\beta^T \tilde{x}^{(i)})\right) \tilde{x}_j^{(i)}$$

BATCH GD

$\tilde{x} = [1, \vec{x}] \rightarrow \in \mathbb{R}^{D+1}$

$x \in \mathbb{R}^D$

Classification → least squares
→ logistic regression
→ Gaussian Discriminant Analysis
→ perceptron

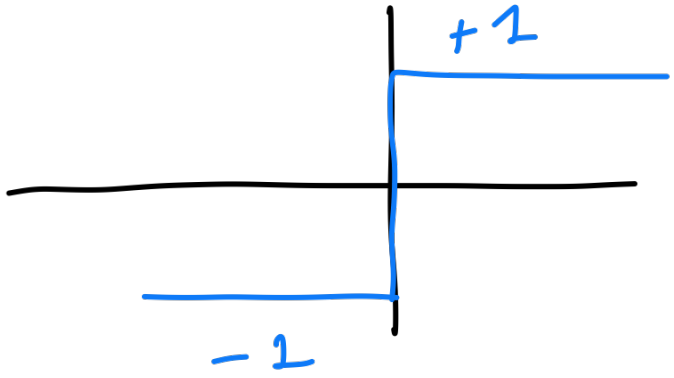
general formulation $\sigma(\beta_0 + \beta_1 x_1 + \cdots + \beta_D x_D)$

least square : $\sigma(x) = x$

logistic regression $\sigma(x) = \frac{1}{1+e^{-x}}$

perceptron $\sigma(x) = \begin{cases} +1 & x > 0 \\ -1 & x \leq 0 \end{cases}$

## Perceptron




→ derivative is zero almost everywhere

$$t^{(i)} = \begin{cases} +1 & x^{(i)} \in C_0 \\ -1 & x^{(i)} \in C_2 \end{cases}$$

→ Solution: take loss

$$l(\beta) = -\sum_{i \in \text{Misclassified}} \widehat{t^{(i)}} \cdot \widehat{(\beta^T \tilde{x}^{(i)})}$$

$$\min l(\beta) = \min -\sum_{i \in \text{Misclassified}} t^{(i)} (\beta^T \tilde{x}^{(i)})$$

From this we get

$$\operatorname*{grad}_{\beta} l = -\sum_{i \in M} t^{(i)} \tilde{x}^{(i)} \Bigg)$$

Perceptron update ( stochastic )

While $\exists$ misclassified points s.t $\sigma(\beta^T \tilde{x}^{(i)}) \neq t^{(i)}$

take a misclassified $x^{(i)}$ and do

$$\beta^{(k+1)} \leftarrow \beta^{(k)} + \eta\, t^{(i)} \tilde{x}^{(i)}$$

perceptron learning rule

Perceptron convergence theorem: if dataset is linearly separable perceptron will converge in a finite number of steps. → and $\eta$ small enough

Assume $\beta_0 = 0$