7.1.1. Linear Regression

Training set (x_i, y_i), $x_i \in \mathbb{R}^p$, $y_i \in \mathbb{R}$, $i = 1, \dots, n$

Hypothesis:

$$y_i = \vartheta_0 + x_{i1}\vartheta_1 + \dots + x_{ip}\vartheta_p + \varepsilon_i, \quad i = 1, \dots, n$$

$$\underline{y} = X\underline{\vartheta} + \underline{\varepsilon}, \quad x_{ij}, y_i \in \mathbb{R}, \quad X = \begin{pmatrix} 1 & \underline{x}_1^T \\ \vdots & \vdots \\ 1 & \underline{x}_n^T \end{pmatrix}$$

$$\hat{\underline{\vartheta}} = (X^T X)^{-1} X^T y \quad \left(\text{or } \hat{\underline{\vartheta}} = (X^T X)^{+} X^T y\right)$$

New observation

$$\hat{y} = (1, x^T)\hat{\underline{\vartheta}} \quad \text{best predictor.}$$

7.1.2. Logistic Regression

Training set (x_i, y_i), $x_i \in \mathbb{R}^p$, $y_i \in \{0, 1\}$

Example: Spam mail

$x_i \in \mathbb{R}^p$ feature vector of some email

$$y_i = \begin{cases} 0, & \text{spam} \\ 1, & \text{no spam} \end{cases}$$

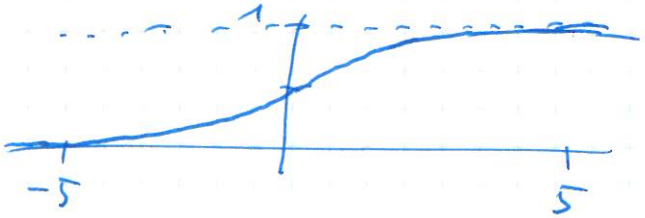
Hypothesis:

$$h_{\underline{\vartheta}}(\underline{x}) = g(\underline{\vartheta}^T \underline{x}) = \frac{1}{1 + e^{-\underline{\vartheta}^T \underline{x}}}$$

$g(z) = \frac{1}{1+e^{-z}}$ is called logistic or sigmoid function.

It holds $g'(z) = g(z)(1 - g(z))$, $z \in \mathbb{R}$

Plot:



Given training set

$$(x_i, y_i),\ i = 1, \dots, n,\ \underline{x}_i \in \mathbb{R}^p,\ y_i \in \{0, 1\}$$

Set $x_{i0} = 1$, s.t.

$$\underline{\theta}^T \underline{x}_i = \theta_0 + \sum_{j=1}^{p} \theta_j x_{ij},\ \underline{x}_i = (1, x_{i1}, \dots, x_{ip})$$

Objective: find the best $\underline{\theta}$!

Probabilistic interpretation:

Assume

$$P(y = 1 \mid \underline{x}, \underline{\theta}) = h_{\underline{\theta}}(\underline{x})$$

$$P(y = 0 \mid \underline{x}, \underline{\theta}) = 1 - h_{\underline{\theta}}(\underline{x})$$

Hence

$$p(y \mid \underline{x}, \underline{\theta}) = (h_{\underline{\theta}}(\underline{x}))^y (1 - h_{\underline{\theta}}(\underline{x}))^{1-y},\ y \in \{0, 1\}$$

Assume n independent training samples, set

$$X = \begin{pmatrix} 1 & \underline{x}_1^T \\ \vdots & \vdots \\ 1 & \underline{x}_n^T \end{pmatrix} = (x_{ij})_{\substack{1 \le i \le n \\ 0 \le j \le p}}$$

Likelihood fct.

$$L(\underline{\theta}) = p(\underline{y} \mid X, \underline{\theta}) = \prod_{i=1}^{n} p(y_i \mid X, \underline{\theta})$$

$$= \prod_{i=1}^{n} (h_{\underline{\theta}}(\underline{x}_i))^{y_i} (1 - h_{\underline{\theta}}(\underline{x}_i))^{1-y_i}$$

log-Likelihood fct.

$$\ell(\underline{\theta}) = \log L(\underline{\theta}) = \sum_{i=1}^{n} y_i \log h_{\underline{\theta}}(\underline{x}_i) + (1 - y_i) \log(1 - h_{\underline{\theta}}(\underline{x}_i)) \quad (*)$$

Objective: $\max_{\underline{\vartheta} \in \mathbb{R}^{p+1}} \ell(\underline{\vartheta})$

Algorithm to compute the optimum: gradient ascent

$$\underline{\vartheta}^{(k+1)} = \underline{\vartheta}^{(k)} + \alpha \nabla_{\underline{\vartheta}} \ell(\underline{\vartheta}^{(k)}), \quad \alpha: \text{"learning parameter"}$$

Compute $\nabla_{\underline{\vartheta}}$ for each addend in (*), set $(\underline{x}_i, y_i) = (\underline{x}, y)$

$$\frac{\partial}{\partial \vartheta_j} \left[y \log h_{\underline{\vartheta}}(\underline{x}) + (1-y) \log(1 - h_{\underline{\vartheta}}(\underline{x})) \right]$$

$$= \left(y \frac{1}{g(\underline{\vartheta}^T \underline{x})} - (1-y) \frac{1}{1 - g(\underline{\vartheta}^T \underline{x})} \right) \frac{\partial}{\partial \vartheta_j} g(\underline{\vartheta}^T \underline{x})$$

$$= \left(\dots \right) g(\underline{\vartheta}^T \underline{x}) (1 - g(\underline{\vartheta}^T \underline{x})) \frac{\partial}{\partial \vartheta_j} \underline{\vartheta}^T \underline{x}$$

$$= \left(y(1 - g(\underline{\vartheta}^T \underline{x})) - (1-y) g(\underline{\vartheta}^T \underline{x}) \right) x_j \quad , \quad \underline{x} = \begin{pmatrix} 1 \\ x_1 \\ \vdots \\ x_p \end{pmatrix}$$

$$= (y - h_{\underline{\vartheta}}(\underline{x})) x_j \quad , \quad j = 0, 1, \dots, p$$

Hence

$$\frac{\partial}{\partial \vartheta_j} \ell(\underline{\vartheta}) = \sum_{i=1}^{n} (y_i - h_{\underline{\vartheta}}(\underline{x}_i)) x_{ij} \quad , \quad j = 0, 1, \dots, p$$

Alternatively: Newton's method

$$\underline{\vartheta}^{(k+1)} = \underline{\vartheta}^{(k)} - H^{-1} \nabla_{\underline{\vartheta}} \ell(\underline{\vartheta}^{(k)})$$

Denote $\underline{\vartheta}^*$ the optimal ϑ, i.e., $\underline{\vartheta}^* = \arg\max_{\vartheta \in \mathbb{R}^p} \ell(\vartheta)$.

Let $\underline{x}$ be a future observation

Soft decision rule: $y = h_{\underline{\vartheta}^*}(\underline{x}) = \dfrac{1}{1 + e^{-\underline{\vartheta}^{*T} \underline{x}}}$

$$y \in (0, 1)$$

7.1.3. The perceptron learning algorithm

"Soft" decision in logistic regression is a value $y \in (0,1)$.

Force hard decision, values to be 0 or 1 by

$$g(z) = \begin{cases} 1, & z \geq 0 \\ 0, & z < 0 \end{cases}$$

$$h_{\underline{\theta}}(\underline{x}) = g(\underline{\theta}^T \underline{x})$$

Use the same update rule as above

$$\theta_j^{(k+1)} = \theta_j^{(k)} + \alpha \sum_{i=1}^{n} \left(y_i - h_{\underline{\theta}^{(k)}}(\underline{x}_i)\right) x_{ij}, \quad j = 0, 1, \dots, p.$$

This is the perceptron learning algorithm.

A rough model for how neurons in the brain work.

Lacking: meaningful probabilistic interpretation as MLE.