Lecture 9 Logistic Regression

- ► Binary Classification
  - ► Maximum Likelihood Formulation
  - ► Newton Method
  - ► Logistic regression algorithm
- ► Advanced topics
  - ► Generalized linear model
  - ► Multiple class extension

Binary Classification

Binary Classification

- Consider a binary classification problem, i.e., *C* = {C1,C2}
- Given a particular vector x, we wish to assign it to one of the two classes. To this end, we use Bayes' rule and calculate the relevant posterior probability:

$$
\begin{aligned}
P(\mathcal{C}_1|\mathbf{x}) &= \frac{P(\mathbf{x}|\mathcal{C}_1)P(\mathcal{C}_1)}{P(\mathbf{x})} \\
&= \frac{P(\mathbf{x}|\mathcal{C}_1)P(\mathcal{C}_1)}{P(\mathbf{x}|\mathcal{C}_1)P(\mathcal{C}_1) + P(\mathbf{x}|\mathcal{C}_2)P(\mathcal{C}_2)} \\
&= \frac{1}{1 + \frac{P(\mathbf{x}|\mathcal{C}_2)}{P(\mathbf{x}|\mathcal{C}_1)} \frac{P(\mathcal{C}_2)}{P(\mathcal{C}_1)}} \\
&= \frac{1}{1 + \exp\left\{ - \log\left[\frac{P(\mathbf{x}|\mathcal{C}_1)}{P(\mathbf{x}|\mathcal{C}_2)}\right] - \log\left[\frac{P(\mathcal{C}_1)}{P(\mathcal{C}_2)}\right] \right\}}
\end{aligned}
$$

- It can be written in the form of the logistic function:

$$P(\mathcal{C}_1|\mathbf{x}) = \frac{1}{1+e^{-\xi}},$$

where

$$\xi = \underbrace{\log\left[\frac{P(\mathbf{x}|\mathcal{C}_1)}{P(\mathbf{x}|\mathcal{C}_2)}\right]}_{\text{likelihood ratio}} + \underbrace{\log\left[\frac{P(\mathcal{C}_1)}{P(\mathcal{C}_2)}\right]}_{\text{prior ratio}}.$$

- Choose a particular form for the class-conditional density function P(x|C).

Multivariate Gaussian Model

Assume that the class-conditional densities are multivariate Gaussian with identical covariance matrix Σ:

$$P(\mathbf{x}|\mathcal{C}_i) = \frac{1}{(2\pi)^{\frac{D}{2}} \, |\boldsymbol{\Sigma}|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)\right\}$$

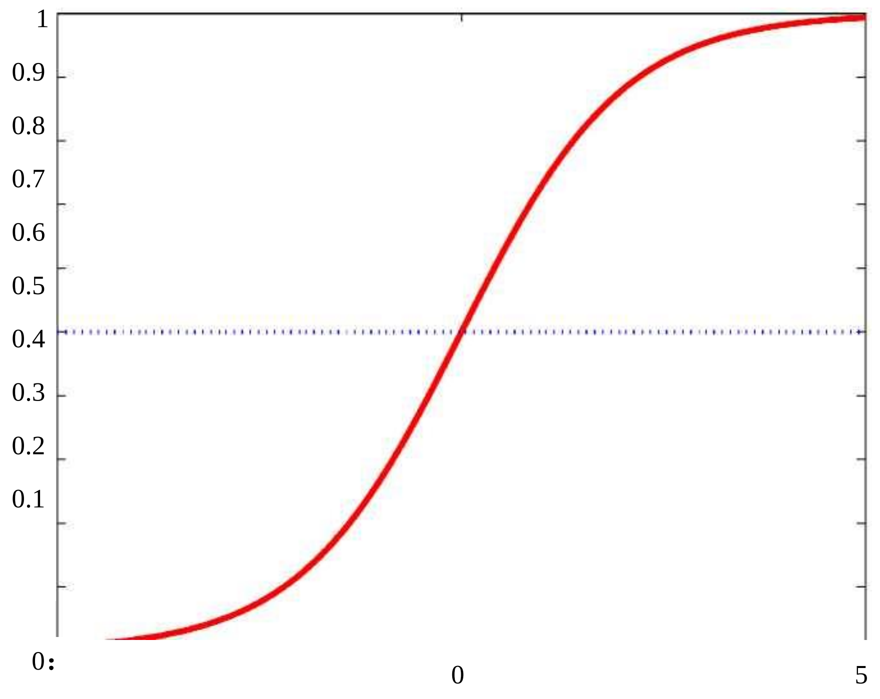
After a simple calculation, we have

$$P(\mathcal{C}_1|\mathbf{x}) = \frac{1}{1+e^{-(\mathbf{w}^\top\mathbf{x}+b)}},$$

where

$$\begin{aligned} \mathbf{w} &= \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1-\boldsymbol{\mu}_2), \\ b &= \frac{1}{2}(\boldsymbol{\mu}_2+\boldsymbol{\mu}_1)^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_2-\boldsymbol{\mu}_1) + \log\left[\frac{P(\mathcal{C}_1)}{P(\mathcal{C}_2)}\right]. \end{aligned}$$

Properties of Logistic Function



$\sigma(\xi) \to 0$ as $\xi \to -\infty$.

$\sigma(\xi) \to 1$ as $\xi \to \infty$.

$\sigma(-\xi) = 1 - \sigma(\xi)$.

$\frac{d}{d\xi}[\sigma(\xi)] = \sigma(\xi)\sigma(-\xi)$.

Maximum Likelihood Formulation

Logistic Regression

- Predict a binary output $y_n \in \{0,1\}$ from an input x_n.
- The logistic regression model has the form

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

Where

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

- Model input-output by a conditional Bernoulli distribution

$$\boxed{P(y_n = 1|\mathbf{x}_n) = \sigma(\mathbf{w}^\top \mathbf{x}_n).}$$

- Discriminative model: Directly model p(y|x).

Logistic Regression: MLE

- Given ((x_n,y_n) | n = 1,..., *N*}, the likelihood is given by

$$p(\mathbf{y}|\mathbf{X},\mathbf{w}) = \prod_{n=1}^{N} p(y_n = 1|\mathbf{x}_n)^{y_n} \left(1 - p(y_n = 1|\mathbf{x}_n)\right)^{1-y_n}$$
$$= \prod_{n=1}^{N} \sigma(\mathbf{w}^\top \mathbf{x}_n)^{y_n} \left(1 - \sigma(\mathbf{w}^\top \mathbf{x}_n)\right)^{1-y_n}.$$

- Then log-likelihood function is given by

$$\mathcal{L} = \sum_{n=1}^{N} \log p(y_n|\mathbf{x}_n) = \sum_{n=1}^{N} \left\{ y_n \log \sigma_n + (1 - y_n) \log(1 - \sigma_n) \right\},$$

where $\sigma_n = \sigma(\mathbf{w}^\top \mathbf{x}_n)$.

- This is a nonlinear function of w whose maximum cannot be computed in a closed form.
- Iterative re-weighted least squares (IRLS) is a popular algorithm, derived from Newton's method.

Newton’s Method

Mathematical Preliminaries

- Gradient
- Hessian matrix
- Gradient descent/ascent
- Newton's method

Gradient

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

$$f(\mathbf{x}) : \mathbb{R}^D \longrightarrow \mathbb{R}.$$

- The gradient of f(x) is defined by

$$\begin{aligned}\nabla f(\mathbf{x}) &= \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} \\ &= \left[\frac{\partial f}{\partial x_1}, \ \cdots, \ \frac{\partial f}{\partial x_D}\right]^\top .\end{aligned}$$

Hessian Matrix

If f(x) belongs to the class C^2, the Hessian matrix **H** is defined as the symmetric matrix with the (i,j)-element $\frac{\partial^2 f(\mathbf{x})}{\partial x_i \partial x_j}$,

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

Gradient Descent/Ascent

- 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

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

- Learning rate: $\eta > 0$

Newton's Method

The basic idea of Newton's method is to optimize the quadratic approximation of the objective function $\mathcal{J}(\mathbf{w})$ around the current point $\mathbf{w}^{(k)}$.

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

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

where $\nabla^2 \mathcal{J}(\mathbf{w}^{(k)})$ is the Hessian of $\mathcal{J}(\mathbf{w})$ evaluated at $\mathbf{w} = \mathbf{w}^{(k)}$
Differentiate this w.r.t. w and set it equal 0, which leads to

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

Thus we have

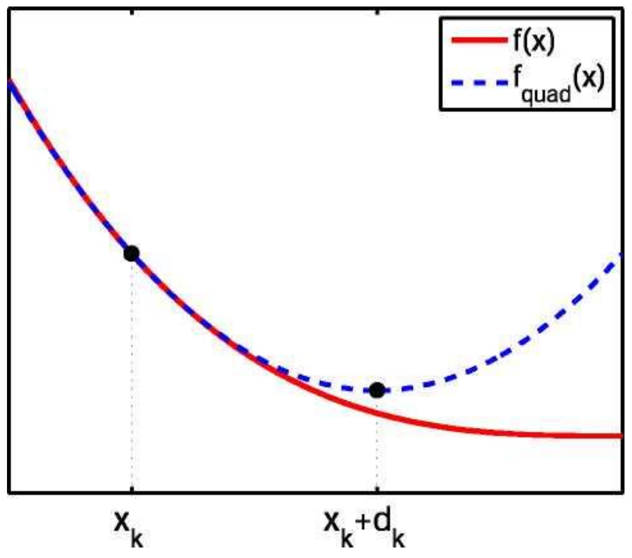
$$\mathbf{w} = \mathbf{w}^{(k)} - \left[\nabla^2 \mathcal{J}(\mathbf{w}^{(k)})\right]^{-1} \nabla \mathcal{J}(\mathbf{w}^{(k)}).$$

Hence the Newton's method is of the form

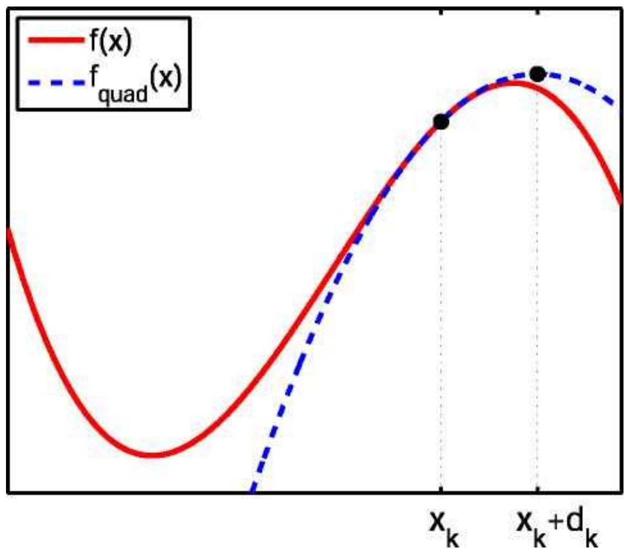
$$\boxed{\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} - \left[\nabla^2 \mathcal{J}(\mathbf{w}^{(k)})\right]^{-1} \nabla \mathcal{J}(\mathbf{w}^{(k)}).}$$

Remark: The Hessian $\nabla^2 \mathcal{J}(\mathbf{w}^{(k)})$ should be positive definite for all *t*.

Illustration of Newton's Method



(a) convex



(b) non-convex

[Figure source: Murphy's book]

Logistic Regression:
Algorithms

Logistic Regression: Gradient Ascent

The gradient ascent learning has the form

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

Calculate the gradient

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = \sum_{t} \{ y_n - \sigma(\mathbf{w}^\top \mathbf{x}_n) \} \mathbf{x}_n.$$

Hence, the gradient ascent update rule for **w** is

$$\mathbf{w}^{\text{new}} = \mathbf{w}^{\text{old}} + \eta \sum_{t} \left\{ y_n - \sigma \left(\left(\mathbf{w}^{\text{old}} \right)^\top \mathbf{x}_n \right) \right\} \mathbf{x}_n.$$

Detailed Calculation of Gradient

Recall the log-likelihood

$$\mathcal{L} = \sum_{n=1}^{N} \left[y_n \log \sigma_n + (1 - y_n) \log(1 - \sigma_n) \right].$$

Calculate

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \mathbf{w}} &= \sum_{n=1}^{N} \left[y_n \frac{\sigma_n'}{\sigma_n} \mathbf{x}_n + (1 - y_n) \frac{-\sigma_n'}{1 - \sigma_n} \mathbf{x}_n \right] \\
&= \sum_{n=1}^{N} \left[y_n \frac{\sigma_n(1 - \sigma_n)}{\sigma_n} - (1 - y_n) \frac{\sigma_n(1 - \sigma_n)}{1 - \sigma_n} \right] \mathbf{x}_n \\
&= \sum_{n=1}^{N} \left[y_n(1 - \sigma_n) - (1 - y_n)\sigma_n \right] \mathbf{x}_n \\
&= \sum_{n=1}^{N} (y_n - \sigma_n) \, \mathbf{x}_n.
\end{aligned}$$

Detailed Calculation of Hessian

Calculate the Hessian:

$$
\begin{aligned}
\nabla^2 \mathcal{L} &= \frac{\partial}{\partial \mathbf{w}} [\nabla \mathcal{L}]^\top \\
&= \frac{\partial}{\partial \mathbf{w}} \left[\sum_{n=1}^{N} (y_n - \sigma_n) \mathbf{x}_n^\top \right] \\
&= \sum_{n=1}^{N} -\sigma_n (1 - \sigma_n) \mathbf{x}_n \mathbf{x}_n^\top .
\end{aligned}
$$

We set the objective function $\mathcal{J}(\mathbf{w})$ as the negative log-likelihood:

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

Thus, the gradient and the Hessian are:

$$\begin{aligned} \nabla \mathcal{J}(\mathbf{w}) &= -\sum_{n=1}^{N} (y_n - \sigma_n)\, \mathbf{x}_n, \\ \nabla^2 \mathcal{J}(\mathbf{w}) &= \sum_{n=1}^{N} \sigma_n (1 - \sigma_n) \mathbf{x}_n \mathbf{x}_n^\top . \end{aligned}$$

Logistic Regression: IRLS

- Newton's update has the form

$$\Delta \mathbf{w} = -\underbrace{\left[\sum_{n=1}^{N} \sigma_n (1-\sigma_n)\, \mathbf{x}_n \mathbf{x}_n^\top\right]^{-1}}_{inverse\ \ of\ \ Hessian,\ [\nabla^2 \mathcal{J}(\mathbf{w})]^{-1}} \underbrace{\left[-\sum_{n=1}^{N} (y_n - \sigma_n)\, \mathbf{x}_n\right]}_{gradient,\ \nabla \mathcal{J}(\mathbf{w})},$$

- Newton's update reduces to iterative re-weighted least squares (IRLS):

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

where

$$\begin{aligned}
\mathbf{S} &= \begin{bmatrix} \sigma_1(1-\sigma_1) & & 0 \\ \vdots & \ddots & \vdots \\ 0 & & \sigma_N(1-\sigma_N) \end{bmatrix}, \\
\mathbf{b} &= \begin{bmatrix} \frac{y_1-\sigma_1}{\sigma_1(1-\sigma_1)} \\ \vdots \\ \frac{y_N-\sigma_N}{\sigma_N(1-\sigma_N)} \end{bmatrix}.
\end{aligned}$$

Algorithm Outline

Algorithm 1 IRLS

```
Input: ${(\mathbf{x}_n, y_n) \mid n = 1, \ldots, N}$
 1: Initialize $\mathbf{w} = \mathbf{0}$ and $w_0 = \log(\bar{y}/(1-\bar{y}))$
 2: repeat
 3:     for $n = 1, \ldots, N$ do
 4:         Compute $\sigma_n = \sigma(\mathbf{w}^\top \mathbf{x}_n + w_0)$
 5:         Compute $s_n = \sigma_n(1-\sigma_n)$
 6:         Compute $b_n = \frac{y_n - \sigma_n}{s_n}$
 7:     end for
 8:     Construct $\mathbf{S} = \mathrm{diag}(s_{1:N})$
 9:     Update $\mathbf{w} = \left(\mathbf{X}\mathbf{S}\mathbf{X}^\top\right)^{-1}\mathbf{S}\mathbf{b}$
10: until convergence
11: return w
```

Generalized Linear Models

Generalized Linear Models

The GLM is a flexible generalization of ordinary linear regression that allows for response variables that have error distribution models other than a normal distribution.

The GLM consists of three elements:

1. A probability distribution from the exponential family.
2. A linear predictor $\boldsymbol{\eta} = \mathbf{X}^\top \boldsymbol{\theta}$
3. A **link function** g(-) such that $\mathbb{E}[\mathbf{y}|\mathbf{X}] = \boldsymbol{\mu} = g^{-1}(\boldsymbol{\eta})$.

Logistic Regression: Generalized Linear Model

- Bernoulli distribution for a univariate binary random variable $y \in \{0,1\}$ with mean μ,

$$p(y|\mu) = \mu^y (1-\mu)^{1-y}.$$

- **Logit transform (log-odds)** $\theta = \mathbf{w}^\top \mathbf{x} = \log\left(\frac{\mu}{1-\mu}\right)$, leads to $[0,1] \mapsto \mathbb{R}$.
- The **logistic function,** which is the **inverse-logit,** gives $\mu = \frac{1}{1+e^{-\theta}} = \sigma(\theta)$.
- Thus, we have

$$\begin{aligned} p(y|\theta) &= \sigma(\theta)^y \left(1-\sigma(\theta)\right)^{1-y} \\ &= \sigma(\theta)^y \sigma(-\theta)^{1-y} \end{aligned}$$

Multiclass Extension

Multiclass Logistic Regression (Multinomial Logistic Regression)

- Model input-output by a softmax transformation of logits $\theta_k = \mathbf{w}_k^\top \mathbf{x}_n$:

$$\begin{aligned} p(y_n = k|\mathbf{x}_n) &= \text{softmax}(\theta_k) \\ &= \frac{\exp\{\mathbf{w}_k^\top \mathbf{x}_n\}}{\sum_{j=1}^K \exp\{\mathbf{w}_j^\top \mathbf{x}_n\}} \end{aligned}$$

- Given $\mathbf{Y} \in \mathbb{R}^{K\times N}$ (each column $\mathbf{y}_n \in \mathbb{R}^K$ follows the 1-of-K coding) and $\mathbf{X} \in \mathbb{R}^{D\times N}$ the likelihood is given by

$$p(\mathbf{Y}|\mathbf{X}, \mathbf{w}_1, \ldots, \mathbf{w}_K) = \prod_{n=1}^N \prod_{k=1}^K p(y_n = k|\mathbf{x}_n)^{Y_{k,n}}.$$

- The log-likelihood is given by

$$\mathcal{L} = \sum_{n=1}^N \sum_{k=1}^K Y_{k,n} \log\left[p(y_n = k|\mathbf{x}_n)\right].$$

- One can apply Newton's update to derive IRLS, as in logistic regression.