Iterative Reweighted Least Squares

Sargur N. Srihari
University at Buffalo, State University of New York
USA

Topics in Linear Classification using Probabilistic Discriminative Models

- Generative vs Discriminative

1. Fixed basis functions in linear classification
2. Logistic Regression (two-class)
3. Iterative Reweighted Least Squares (IRLS)
4. Multiclass Logistic Regression
5. Probit Regression
6. Canonical Link Functions

Topics in IRLS

- What is IRLS
- Linear and Logistic Regression
- IRLS for Linear Regression
- IRLS for Logistic Regression

What is IRLS?

- An iterative method to find solution $\boldsymbol{w}^*$
  - for linear regression and logistic regression
    - assuming least squares objective
- While simple gradient descent has the form

$$\boxed{\boldsymbol{w}^{(\text{new})} = \boldsymbol{w}^{(\text{old})} - \eta \nabla E(\boldsymbol{w})}$$

- IRLS uses the second derivative and has the form $\boxed{\boldsymbol{w}^{(\text{new})} = \boldsymbol{w}^{(\text{old})} - H^{-1} \nabla E(\boldsymbol{w})}$
  - It is derived from Newton-Raphson method
    - where H is the Hessian matrix whose elements are the second derivatives of $E(\boldsymbol{w})$ wrt $\boldsymbol{w}$

Newton-Raphson Method (1-D)

- Based on second derivatives
  - Derivative at point x of a function is the slope of its tangent at that point
- Illustration of second derivative
  - Derivatives of Gaussian $p(x) \sim N(0, \sigma)$

$$\frac{\partial}{\partial x}\left[\frac{1}{\sqrt{2\pi}\sigma}e^{-x^2/(2\sigma^2)}\right] = \frac{-x}{\sqrt{2\pi}\sigma^3}e^{-x^2/(2\sigma^2)} = \frac{-x}{\sigma^2}p(x)$$

$$\frac{\partial^2}{\partial x^2}\left[\frac{1}{\sqrt{2\pi}\sigma}e^{-x^2/(2\sigma^2)}\right] = \frac{1}{\sqrt{2\pi}\sigma^5}\left(-\sigma^2+x^2\right)e^{-x^2/(2\sigma^2)} = \frac{-\sigma^2+x^2}{\sigma^4}p(x)$$

$$\frac{\partial^3}{\partial x^3}\left[\frac{1}{\sqrt{2\pi}\sigma}e^{-x^2/(2\sigma^2)}\right] = \frac{1}{\sqrt{2\pi}\sigma^7}\left(3x\sigma^2-x^3\right)e^{-x^2/(2\sigma^2)} = \frac{-3x\sigma^2-x^3}{\sigma^6}p(x),$$

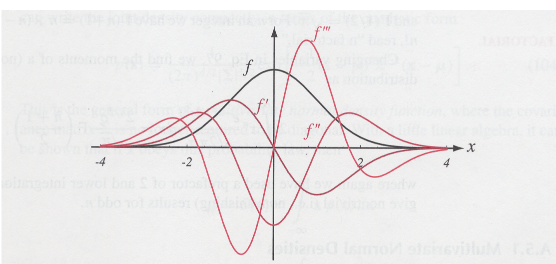
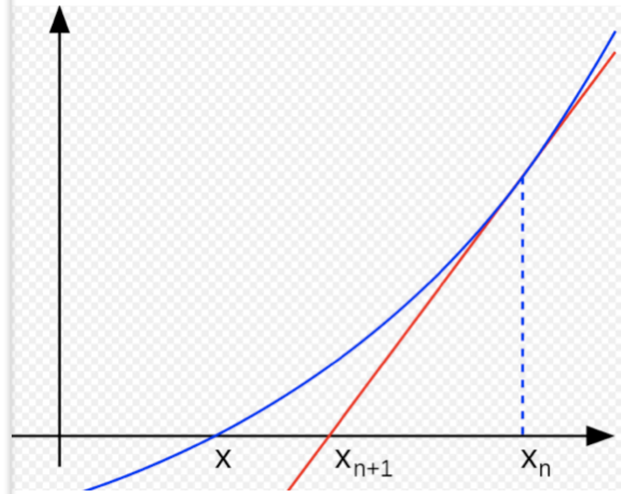


FIGURE A.2. A one-dimensional Gaussian distribution and its first three derivatives, shown for $f(x) \sim N(0, 1)$.



$$f'(x_n) = \frac{\text{rise}}{\text{run}} = \frac{\Delta y}{\Delta x} = \frac{f(x_n) - 0}{x_n - x_{n+1}}.$$

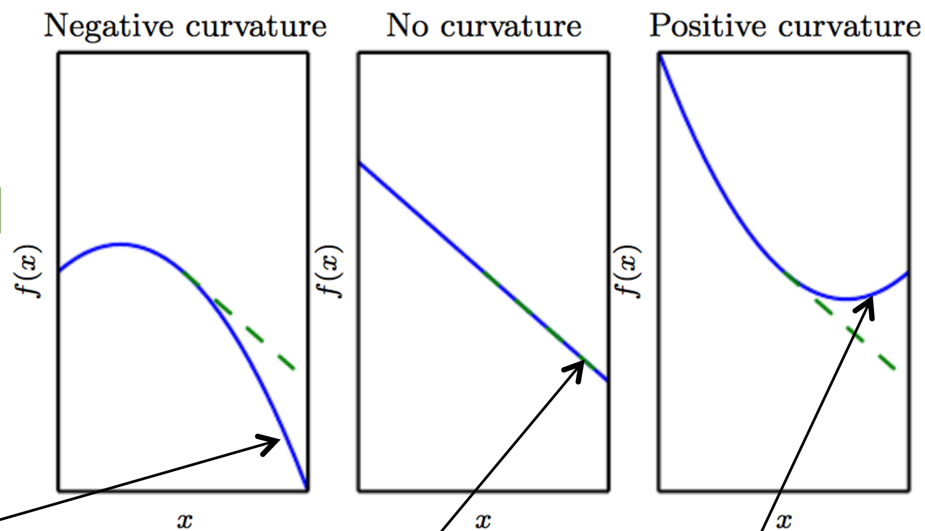
$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}.$$

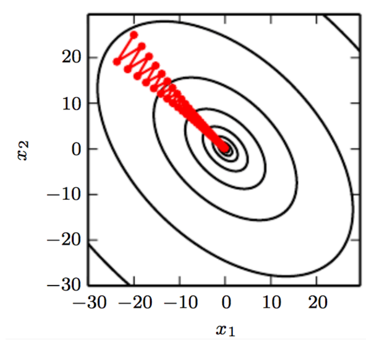
Since we are solving for derivative of $E(\boldsymbol{w})$
Need second derivative

Second derivative measures curvature

- Derivative of a derivative
- Quadratic functions with different curvatures

Dashed line is value of cost function predicted by gradient alone





Decrease is faster than predicted by Gradient Descent

Gradient Predicts decrease correctly

Decrease is slower than expected Actually increases

Learning rate from Hessian

- Taylor's series of $f(\boldsymbol{x})$ around current point $\boldsymbol{x}^{(0)}$

$$\boxed{f(\boldsymbol{x}) \approx f(\boldsymbol{x}^{(0)}) + (\boldsymbol{x} - \boldsymbol{x}^{(0)})^T \boldsymbol{g} + \frac{1}{2}(\boldsymbol{x} - \boldsymbol{x}^{(0)})^T H(\boldsymbol{x} - \boldsymbol{x}^{(0)})}$$

  - where $\boldsymbol{g}$ is the gradient and H is the Hessian at $\boldsymbol{x}^{(0)}$

– If we use learning rate ε the new point $\boldsymbol{x}$ is given by $\boldsymbol{x}^{(0)}-\varepsilon\boldsymbol{g}$. Thus we get $\boxed{f(\boldsymbol{x}^{(0)} - \varepsilon\boldsymbol{g}) \approx f(\boldsymbol{x}^{(0)}) - \varepsilon\boldsymbol{g}^T\boldsymbol{g} + \frac{1}{2}\varepsilon^2\boldsymbol{g}^T H\boldsymbol{g}}$

  - There are three terms:
    - original value of f,
    - expected improvement due to slope, and
    - correction to be applied due to curvature
  - Solving for step size when correction is least gives

$$\boxed{\varepsilon^* \approx \frac{\boldsymbol{g}^T\boldsymbol{g}}{\boldsymbol{g}^T H\boldsymbol{g}}}$$

Another 2nd Derivative Method

- Using Taylor's series of $f(\boldsymbol{x})$ around current $\boldsymbol{x}^{(0)}$

$$\boxed{f(\boldsymbol{x}) \approx f(\boldsymbol{x}^{(0)}) + (\boldsymbol{x} - \boldsymbol{x}^{(0)})^T \nabla_{\boldsymbol{x}} f(\boldsymbol{x}^{(0)}) + \frac{1}{2}(\boldsymbol{x} - \boldsymbol{x}^{(0)})^T H(f)(\boldsymbol{x} - \boldsymbol{x}^{(0)})(\boldsymbol{x} - \boldsymbol{x}^{(0)})}$$

  - solve for the critical point of this function to give

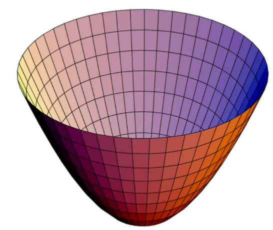
$$\boxed{\boldsymbol{x}^* = \boldsymbol{x}^{(0)} - H(f)(\boldsymbol{x}^{(0)})^{-1} \nabla_{\boldsymbol{x}} f(\boldsymbol{x}^{(0)})}$$

  - When f is a quadratic (positive definite) function use solution to jump to the minimum function directly
  - When not quadratic apply solution iteratively
- Can reach critical point much faster than gradient descent
  - But useful only when nearby point is a minimum

Linear and Logistic Regression

- In linear regression there is a closed-form max likelihood solution for determining $\boldsymbol{w}$ $\boxed{\mathrm{w}_{ML} = \Phi^{+}\mathrm{t}}$
  - on the assumption of Gaussian noise model
  - Due to quadratic dependence of log-likelihood on $\boldsymbol{w}$

$$\boxed{E_D(\boldsymbol{w}) = \frac{1}{2}\sum_{n=1}^{N}\left\{\, t_n - \boldsymbol{w}^T\phi(\boldsymbol{x}_n)\,\right\}^2}$$



- For logistic regression: No closed-form maximum likelihood solution
  - Due to nonlinearity of logistic sigmoid

$$\boxed{E(\boldsymbol{w}) = -\ln p(t \mid \boldsymbol{w}) = -\sum_{n=1}^{N}\left\{t_n \ln y_n + (1-t_n)\ln(1-y_n)\right\}} \quad \boxed{y_n = \sigma(\boldsymbol{w}^{\mathrm{T}}\phi_n)}$$

- But departure from quadratic is not substantial
  - Error function is concave, i.e., unique minimum

Two applications of IRLS

- IRLS is applicable to both Linear Regression and Logistic Regression
- We discuss both, for each we need
  1. Model $\boxed{y(\boldsymbol{x},\boldsymbol{w}) = \sum_{j=0}^{M-1} w_j \phi_j(\boldsymbol{x}) = \boldsymbol{w}^T \phi(\boldsymbol{x})}$ $\boxed{p(C_1|\boldsymbol{\phi}) = y(\boldsymbol{\phi}) = \sigma(\boldsymbol{w}^T\boldsymbol{\phi})}$
  2. Objective function $E(\mathrm{w})$
     - Linear Regression: Sum of Squared Errors $\boxed{E(\boldsymbol{w}) = \frac{1}{2}\sum_{n=1}^{N}\left\{t_n - \boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}_n)\right\}^2}$
     - Logistic Regression: Bernoulli Likelihood Function $\boxed{p(\mathrm{t}\,|\,\boldsymbol{w}) = \prod_{n=1}^{N} y_n^{t_n}\left\{1-y_n\right\}^{1-t_n}}$
  3. Gradient $\boxed{\nabla E(\boldsymbol{w})}$
  4. Hessian $\boxed{H = \nabla\nabla E(\boldsymbol{w})}$
  5. Newton-Raphson update

$$\boxed{\boldsymbol{w}^{(\mathrm{new})} = \boldsymbol{w}^{(\mathrm{old})} - H^{-1}\nabla E(\boldsymbol{w})}$$

IRLS for Linear Regression

1. **Model:** $$y(\boldsymbol{x},\boldsymbol{w}) = \sum_{j=0}^{M-1} w_j \phi_j(\boldsymbol{x}) = \boldsymbol{w}^T \phi(\boldsymbol{x})$$

2. **Error Function:** Sum of Squares: $$E(\boldsymbol{w}) = \frac{1}{2}\sum_{n=1}^{N}\left\{t_n - \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}_n)\right\}^2$$

   for data set $X=\{\boldsymbol{x}_n, t_n\}$ $n=1,..N$

3. **Gradient of Error Function is:**

$$\nabla E(\boldsymbol{w}) = \sum_{n=1}^{N}(\boldsymbol{w}^T\phi_n - t_n)\phi_n = \Phi^T\Phi\boldsymbol{w} - \Phi^T \mathrm{t}$$

where Φ is the N x M design matrix whose n^{th} row is given by $\boldsymbol{\phi}_n^{\ T}$

$$\Phi = \begin{pmatrix} \phi_0(\mathrm{x}_1) & \phi_1(\mathrm{x}_1) & \ldots & \phi_{M-1}(\mathrm{x}_1) \\ \phi_0(\mathrm{x}_2) & & & \\ & & & \\ \phi_0(\mathrm{x}_N) & & & \phi_{M-1}(\mathrm{x}_N) \end{pmatrix}$$

4. **Hessian is:** $$H = \nabla\nabla E(\boldsymbol{w}) = \sum_{n=1}^{N}\phi_n\phi_n^T = \Phi^T\Phi$$

5. **Newton-Raphson Update:** $$\boldsymbol{w}^{(\mathrm{new})} = \boldsymbol{w}^{(\mathrm{old})} - H^{-1}\nabla E(\boldsymbol{w})$$

Substituting: $H = \Phi^T\Phi$ and $\nabla E(\boldsymbol{w}) = \Phi^T\Phi\boldsymbol{w} - \Phi^T\mathrm{t}$

$$\boldsymbol{w}^{(\mathrm{new})} = \boldsymbol{w}^{(\mathrm{old})} - (\Phi^{\mathrm{T}}\Phi)^{-1}\{\Phi^{\mathrm{T}}\Phi\boldsymbol{w}^{(\mathrm{old})} - \Phi^{\mathrm{T}}\mathrm{t}\}$$
$$= (\Phi^{\mathrm{T}}\Phi)^{-1}\Phi^{\mathrm{T}}\mathrm{t}$$

which is the standard least squares solution

Since it is independent of $\boldsymbol{w}$, Newton-Raphson gives exact solution in one step

IRLS for Logistic Regression

1. **Model:** $p(C_1|\phi) = y(\phi) = \sigma(\boldsymbol{w}^T\phi)$ **Likelihood:** $p(\mathrm{t}\,|\,\boldsymbol{w}) = \prod_{n=1}^{N} y_n^{t_n}\{1-y_n\}^{1-t_n}$

   for data set $\{\phi_n, t_n\}$, $t_n \in \{0,1\}$, $y_n = \phi(\boldsymbol{x}_n)$

2. **Objective Function:** Negative log-likelihood: $E(\boldsymbol{w}) = -\sum_{n=1}^{N}\{t_n \ln y_n + (1-t_n)\ln(1-y_n)\}$

   Cross-entropy error function

3. **Gradient of Error Function is:**

$$\nabla E(\boldsymbol{w}) = \sum_{n=1}^{N}(y_n - t_n)\phi_n = \Phi^T(\boldsymbol{y}-\boldsymbol{t})$$

   where Φ is the N x M design matrix whose n^{th} row is given by $\boldsymbol{\phi}_n^T$

$$\Phi = \begin{pmatrix} \phi_0(\mathrm{x}_1) & \phi_1(\mathrm{x}_1) & \ldots & \phi_{M-1}(\mathrm{x}_1) \\ \phi_0(\mathrm{x}_2) & & & \\ & & & \\ \phi_0(\mathrm{x}_N) & & & \phi_{M-1}(\mathrm{x}_N) \end{pmatrix}$$

4. **Hessian is:**

$$\mathrm{H} = \nabla\nabla E(\boldsymbol{w}) = \sum_{n=1}^{N} y_n(1-y_n)\phi_n\phi_n^T = \Phi^T \mathrm{R} \Phi$$

   R is N x N diagonal matrix with elements $R_{nn} = y_n(1-y_n) = \boldsymbol{w}^T\phi_n(1-\boldsymbol{w}^T\phi_n)$

   Hessian is not constant and depends on $\boldsymbol{w}$ through R. Since H is positive-definite (i.e., for arbitrary $\boldsymbol{u}$, $\boldsymbol{u}^T\mathrm{H}\boldsymbol{u}>0$) error function is a concave function of $\boldsymbol{w}$ and so has a unique minimum

5. **Newton-Raphson Update:** $\boldsymbol{w}^{(\text{new})} = \boldsymbol{w}^{(\text{old})} - H^{-1}\nabla E(\boldsymbol{w})$

   Substituting $\mathrm{H} = \Phi^T \mathrm{R}\Phi$ and $\nabla E(\boldsymbol{w}) = \Phi^T(\mathrm{y}-\mathrm{t})$

$$\begin{aligned}\boldsymbol{w}^{(\text{new})} &= \boldsymbol{w}^{(\text{old})} - (\Phi^T\mathrm{R}\Phi)^{-1}\Phi^T(\boldsymbol{y}-\boldsymbol{t}) \\ &= (\Phi^T\mathrm{R}\Phi)^{-1}\{\Phi\Phi\boldsymbol{w}^{(\text{old})} - \Phi^T(\mathrm{y}-\mathrm{t})\} \\ &= (\Phi^T\mathrm{R}\Phi)^{-1}\Phi^T\mathrm{R}\boldsymbol{z}\end{aligned}$$

   where $\boldsymbol{z}$ is a N-dimensional vector with elements $\boldsymbol{z} = \Phi\boldsymbol{w}^{(\text{old})} - \mathrm{R}^{-1}(\boldsymbol{y}-\boldsymbol{t})$

   Update formula is a set of normal equations.
   Since Hessian depends on $\boldsymbol{w}$ apply them iteratively each time using the new weight vector