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

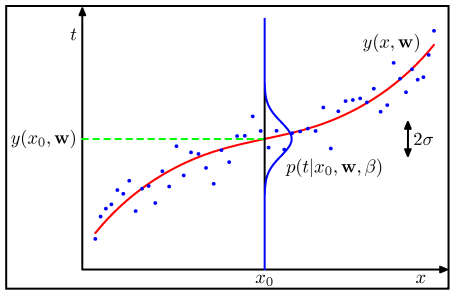
Topics in IRLS

- Linear and Logistic Regression
- Newton-Raphson Method
- Hessian
- IRLS for Linear Regression
- IRLS for Logistic Regression
- Numerical Example
- Hessian in Deep Learning

Recall Linear Regression

- Assuming Gaussian noise $p(t|\mathrm{x},\mathrm{w},\beta)=N(t \mid y(\mathrm{x},\mathrm{w}),\beta^{-1})$
- And a linear model

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



- Likelihood has the form

$$p(\mathrm{t} \mid X,\mathrm{w},\beta) = \prod_{n=1}^{N} N\left(t_n \mid \mathrm{w}^T \boldsymbol{\phi}(\mathrm{x}_n), \beta^{-1}\right)$$

- Log-likelihood becomes

$$\ln p(\mathrm{t} \mid \mathrm{w},\beta) = \sum_{n=1}^{N} \ln N\left(t_n \mid \mathrm{w}^T \phi(\mathrm{x}_n), \beta^{-1}\right) = \frac{N}{2}\ln\beta - \frac{N}{2}\ln 2\pi - \beta E_D(\mathrm{w})$$

$$E_D(\mathrm{w}) = \frac{1}{2}\sum_{n=1}^{N}\left\{ t_n - \mathrm{w}^T \boldsymbol{\varphi}(x_n) \right\}^2$$

  - Which is quadratic in input
- Derivative has the form

$$\nabla \ln p(\mathrm{t} \mid \mathrm{w},\beta) = \beta \sum_{n=1}^{N}\left\{ t_n - \mathrm{w}^T \boldsymbol{\phi}(\mathrm{x}_n) \right\} \boldsymbol{\phi}(\mathrm{x}_n)^T$$

- Setting equal to zero and solving we get

$$\mathrm{w}_{ML} = \boldsymbol{\Phi}^{+}\mathrm{t} \qquad \boldsymbol{\Phi}^{+} = (\boldsymbol{\Phi}^T\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^T$$

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

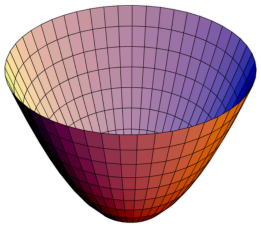
Linear and Logistic Regression

- In linear regression there is a closed-form max likelihood solution for determining $\boldsymbol{w}$

$$\mathrm{w}_{ML} = \Phi^{+}\mathrm{t}$$

  - on the assumption of Gaussian noise model
  - Due to quadratic dependence of log-likelihood on $\boldsymbol{w}$

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



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

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

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

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

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

- IRLS uses the second derivative and has the form

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

  - 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 of a function at x is the slope of its tangent at that point
- Illustration of 2nd and higher derivatives
  - Derivatives of Gaussian $p(x) \sim N(0, \sigma)$

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

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

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

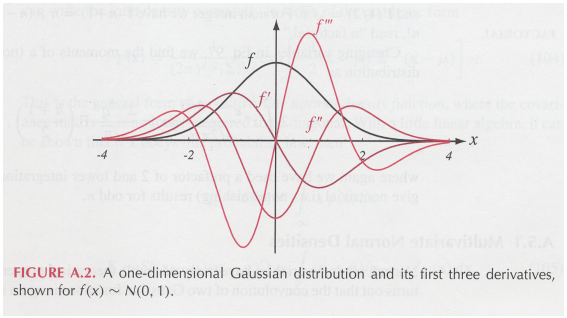
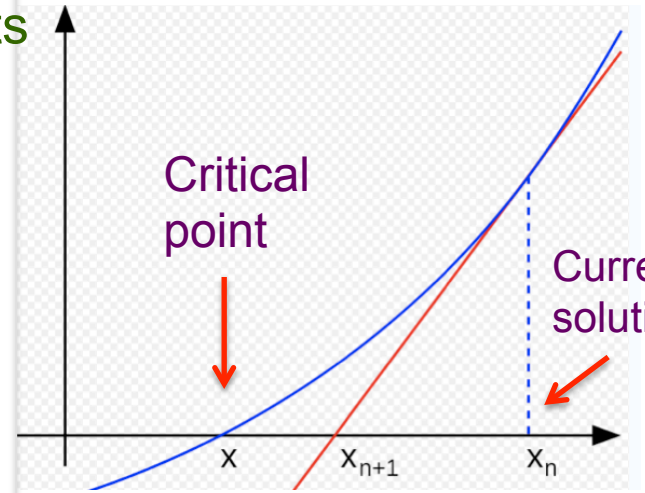


FIGURE A.2. A one-dimensional Gaussian distribution and its first three derivatives, shown for $f(x) \sim N(0, 1)$.



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

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

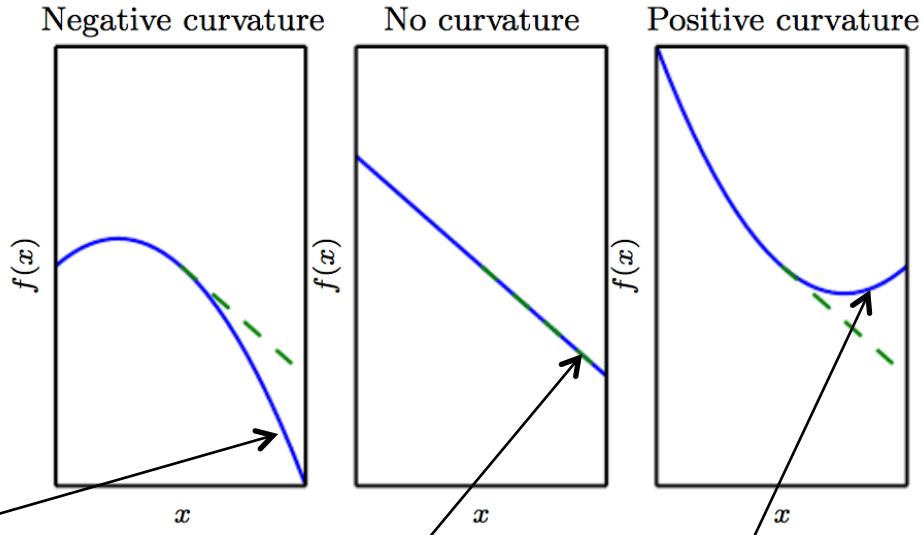
Since we are solving for derivative of $E(\boldsymbol{w})$
Need second derivative

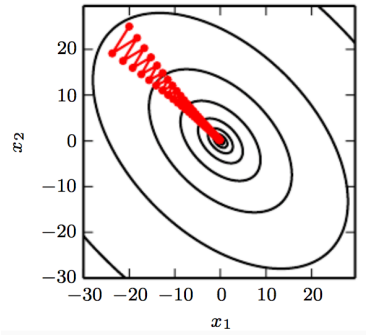
7

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

Beyond Gradient: Jacobian and Hessian matrices

- Sometimes we need to find all derivatives of a function whose input and output are both vectors
- If we have function $f: R^m \rightarrow R^n$
  - Then the matrix of partial derivatives is known as the Jacobian matrix J defined as

$$\boxed{J_{i,j} = \frac{\partial}{\partial x_j} f(x)_i}$$

Second derivative

- Derivative of a derivative
- For a function f: $R^n \rightarrow R$ the derivative wrt x_i of the derivative of f wrt x_j is denoted as $\frac{\partial^2}{\partial x_i \partial x_j} f$
- In a single dimension we can denote $\frac{\partial^2}{\partial x^2} f$ by $f''(\boldsymbol{x})$
- Tells us how the first derivative will change as we vary the input
- This important as it tells us whether a gradient step will cause as much of an improvement as based on gradient alone

Hessian

- Second derivative with many dimensions
- $H(f)(\boldsymbol{x})$ is defined as $$H(f)(\boldsymbol{x})_{i,j} = \frac{\partial^2}{\partial x_i \partial x_j} f(\boldsymbol{x})$$
- Hessian is the Jacobian of the gradient
- Hessian matrix is symmetric, i.e., $H_{i,j} = H_{j,i}$
  - anywhere that the second partial derivatives are continuous
  - So the Hessian matrix can be decomposed into a set of real eigenvalues and an orthogonal basis of eigenvectors
    - Eigenvalues of H are useful to determine learning rate as seen in next two slides

Role of eigenvalues of Hessian

- Second derivative in direction d is $\mathrm{d}^T H \mathrm{d}$
  - If d is an eigenvector, second derivative in that direction is given by its eigenvalue
  - For other directions, weighted average of eigenvalues (weights of 0 to 1, with eigenvectors with smallest angle with d receiving more value)
- Maximum eigenvalue determines maximum second derivative and minimum eigenvalue determines minimum second derivative

Learning rate from Hessian

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

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

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

  - 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

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

Another 2nd Derivative Method

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

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

  - solve for the critical point of this function to give

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

  - 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

Two applications of IRLS

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

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

IRLS for Linear Regression

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

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 $\mathrm{X}=\{\boldsymbol{x}_n, t_n\}$ $n=1,..N$

3. Gradient of Error Function is: $$\nabla E(\boldsymbol{w}) = \sum_{n=1}^{N}(\boldsymbol{w}^T\boldsymbol{\phi}_n - t_n)\boldsymbol{\phi}_n = \boldsymbol{\Phi}^T\boldsymbol{\Phi}\boldsymbol{w} - \boldsymbol{\Phi}^T\mathrm{t}$$

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

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

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

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

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

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

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

Srihari

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

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

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:

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

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

$$\boldsymbol{\Phi} = \begin{pmatrix} \phi_0(x_1) & \phi_1(x_1) & \cdots & \phi_{M-1}(x_1) \\ \phi_0(x_2) & & & \\ & & & \\ \phi_0(x_N) & & & \phi_{M-1}(x_N) \end{pmatrix}$$

4. Hessian is:

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

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

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

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

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

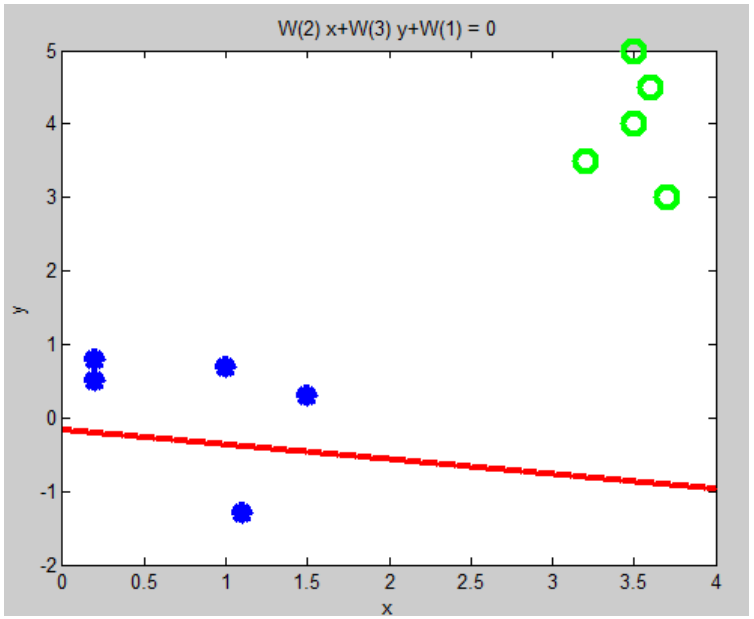
Update formula is a set of normal equations.
Since Hessian depends on $\boldsymbol{w}$ apply them iteratively each time using the new weight vector

Initial Weight Vector, Gradient and Hessian (2-class)

- Weight vector

```
W =

    0.1117
    0.1363
    0.6787
```



- Gradient

```
Delta_E =

         0
    6.7500
    9.5000
```

- Hessian

```
H =

    3.5000    5.3750    5.2500
    5.3750   17.4825   17.4950
    5.2500   17.4950   22.4150
```

Final Weight Vector, Gradient and Hessian (2-class)

- Weight Vector

```
W =

  704.5915
  -20.9086
 -337.6170
```

- Gradient

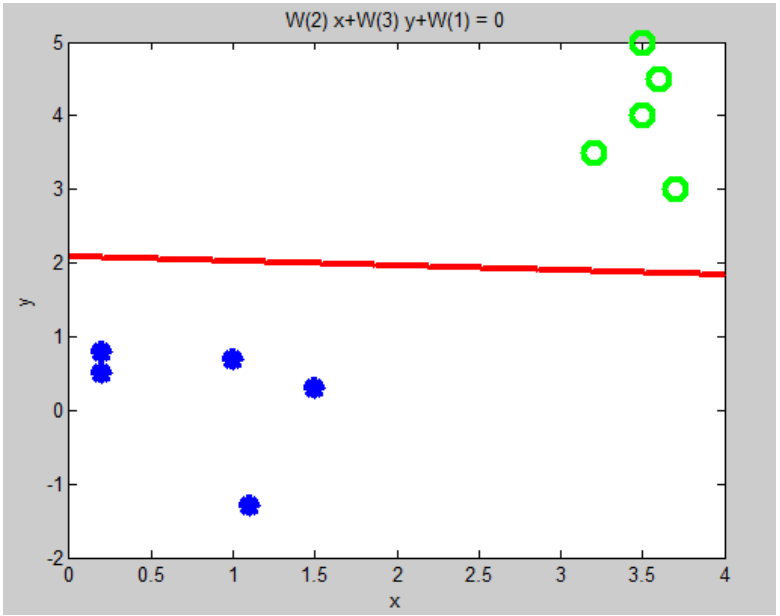
```
Delta_E =

  -12.3917
   -1.6321
    4.9025
```

- Hessian

```
H =

    1.0000    0.0000    0.0000
    0.0000    1.0000    0.0000
    0.0000    0.0000    1.0000
```



Number of iterations : 10

Error (Initial and Final): 15.0642, 1.0000e-009

Hessian in Deep Learning

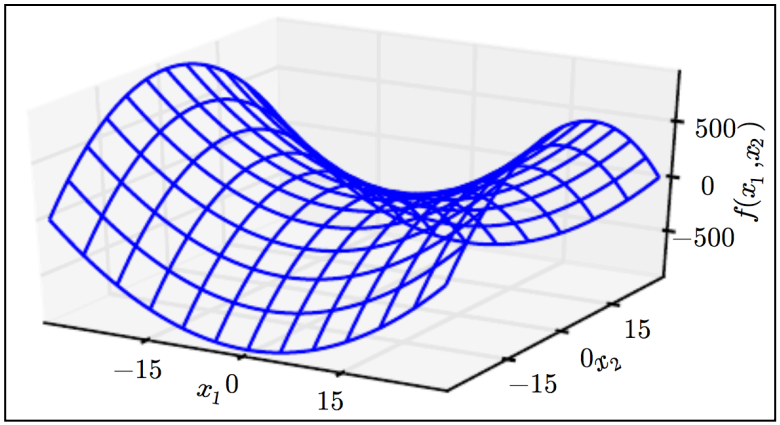
- The second derivative can be used to determine whether a critical point is a local minimum, maximum or a saddle point
  - On a critical point $f'(x)=0$
  - When $f''(x)>0$ the first derivative $f'(x)$ increases as we move to the right and decreases as we move left
    - We conclude that x is a local minimum
  - For local maximum, $f'(x)=0$ and $f''(x)<0$
  - When $f''(x)=0$ test is inconclusive: x may be a saddle point or part of a flat region

Multidimensional Second derivative test

- In multiple dimensions, we need to examine second derivatives of all dimensions
- Eigendecomposition generalizes the test
- Test eigenvalues of Hessian to determine whether critical point is a local maximum, local minimum or saddle point
- When H is positive definite (all eigenvalues are positive) the point is a local minimum
- Similarly negative definite implies a maximum

Saddle point

- Contains both positive and negative curvature
- Function is $f(\boldsymbol{x}) = x_1^2 - x_2^2$



  - Along axis x_1, function curves upwards: this axis is an eigenvector of H and has a positive value
  - Along x_2, function corves downwards; its direction is an eigenvector of H with negative eigenvalue
  - At a saddle point eigen values are both positive and negative