Gradient Decent and such

Schwartz

April 13, 2017

A brief history of Optimization

- 300 bc Euclid considers minimal distance from point to a line & proves square is the greatest area rectangle
- 1615 Kepler optimizes dimensions of wine barrel & formulates an early version of the (classical) secretary problem while looking for a new wife
- 1636 Fermat shows derivatives vanish at extremes & light travels between two points in minimal time
- 1660s Newton & Leibniz create the mathematical basis of calculus and hence optimization calculus
- 1696 Johann & Jacob Bernoulli study Brachistochrone problem – calculus optimization is born
- 1712 König shows that the shape of honeycomb is optimal. The French Academy of Sciences declares the phenomenon as divine guidance
- 1740 Euler's publication begins the research on a general theory of calculus optimization
- 1754 Lagrange makes his first of many findings regarding calculus optimization at age 19
- 1900's The first optimization algorithms are presented by Weierstrass, Steiner, Hamilton and Jacobi
- 1806 Legendre presents the least square method, which also Gauss claims to have invented
- 1815 "The Law of Diminishing Returns" (introduced simultaneously by Malthus, Torrens, West, and Ricardo) uses a (quasi) concave function
- 1826 Fourier formulates linear programming (LP) for solving mechanics and probability problems
- 1847 Cauchy presents the gradient method
- 1857 Gibbs shows chemical equilibrium is minimum energy
- 1870s The marginalist revolution in economics shifts the focus of economists to maximizing individuals utility
- 1880s Convexity theory created – Jensen introduces convex functions in 1905 – Minkowski convex sets in 1911
- 1917 Hancock publishes the first text book on optimization: "Theory of Minima and Maxima"
- 1917 Thompson's "On Growth and Form" applies optimization to analyze the forms of living organisms
- 1928 Ramsey studies optimal economic growth which becomes optimal growth theory in the 1950's
- 1932 Menger generalizes the traveling salesman problem
- 1939 Kantorovich presents LP-model & solution algorithm and receives Nobel prize in 1975 with Koopmans
- 1944 Neuman and Morgenstern, and Wald (1947) solve sequential problems w/ dynamic programming (DP)
- 1947 Dantzig (USAF) presents the Simplex method for LP-problems, Neumann establishes duality theory
- 1950's Electronic calculation initiates algorithmic research
- 1951 Markowitz presents portfolio theory using quadratic programing (QP) and receives the 1990 Nobel prize
- 1954 Ford & Fulkerson introduce combinatorial optimization for network research problems
- 1960's Space race sparks optimal control theory research
- 1970s Complexity analysis influences optimization theory
- 1980's Heuristic global optimization algorithms for large scale problems gain popularity as computers improve

Objectives

- Loss Function
- Cost Function
- Objective Function

- Gradient Decent
- Stochastic Gradient Descent
- Newton's Method

Exercise of the day

Find parameters that maximizes the fit of model to data

Exercise of the day

Find parameters that maximizes
the fit of model to data

$$\Longleftrightarrow$$

minimize distance between
predicted and observed values

Exercise of the day

Find parameters that maximizes the fit of model to data

$$\iff$$

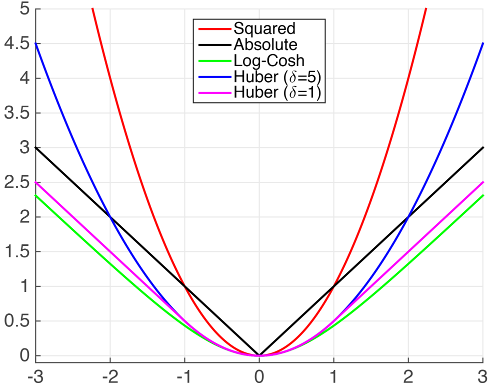
minimize distance between predicted and observed values

$$\iff$$

$$\hat{Y}_i \approx Y_i$$

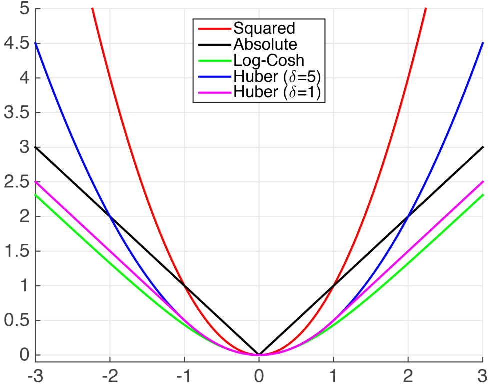
Regression Loss Functions

$$Y_i - \hat{Y}_i$$



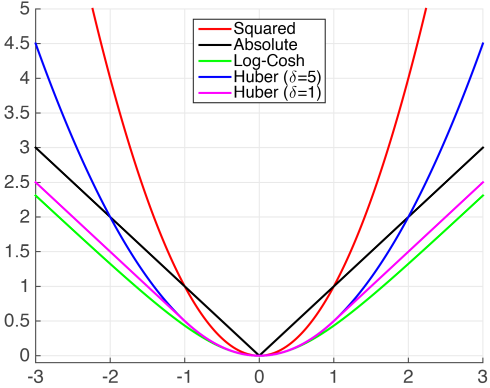
Regression Loss Functions

$$(Y_i - \hat{Y}_i)^2$$



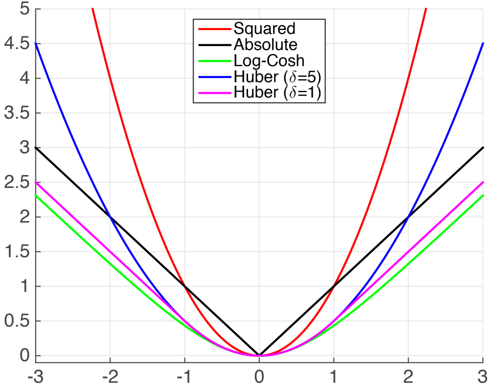
Regression Loss Functions

$$|Y_i - \hat{Y}_i|$$



Regression Loss Functions

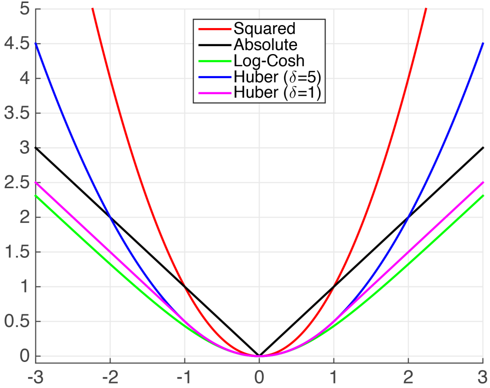
$$ln(cosh(Y_i - \hat{Y}_i))$$



Regression Loss Functions

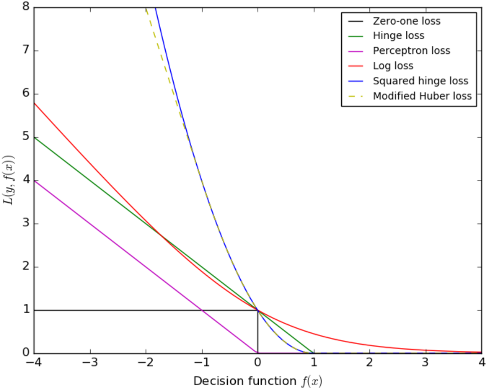
$$L_\delta(Y_i - \hat{Y}_i)$$

$$L_\delta(a) = \begin{cases} \frac{1}{2}a^2 & : |a| < \delta \\ \delta(|a| - \frac{1}{2}\delta) & : o.w. \end{cases}$$



Classification Loss Functions

$$Y \in [0,1] : \min_p - Y \log p - (1 - Y) \log(1 - p) \qquad p = \frac{1}{1 + e^{-x^T \beta}}$$

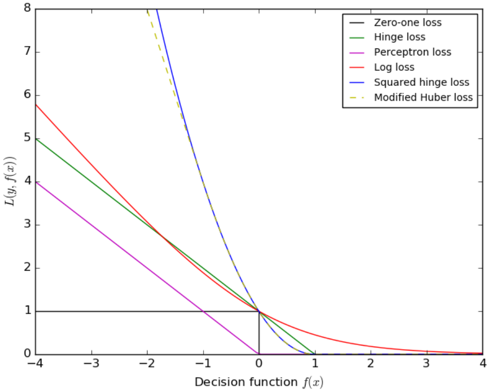


Classification Loss Functions

$$Y \in [0,1] : \min_p - Y \log p - (1-Y)\log(1-p) \qquad p = \frac{1}{1+e^{-x^T\beta}}$$

$$\Longleftrightarrow$$

$$Y \in [-1,1] : \min_p \frac{1}{ln2}\left(1 + e^{-Yln\left(\frac{p}{1-p}\right)}\right) \qquad ln\left(\frac{p}{1-p}\right) = x^T\beta$$



Cost function

- $\sum(Y_i - x_i^T\beta)^2$

- $-\sum Y_i \log g^{-1}(x_i^T\beta) + (1 - Y_i)\log\left(1 - g^{-1}(x_i^T\beta)\right)$

$$g^{-1}(z) = \frac{1}{1 + e^{-z}}$$

Cost function

- $\sum(Y_i - x_i^T\beta)^2 = (\boldsymbol{Y} - \boldsymbol{x}\beta)^T(\boldsymbol{Y} - \boldsymbol{x}\beta) = ||\boldsymbol{Y} - \boldsymbol{x}\beta||^2$

- $-\sum Y_i \log g^{-1}(x_i^T\beta) + (1 - Y_i)\log\left(1 - g^{-1}(x_i^T\beta)\right)$

$$g^{-1}(z) = \frac{1}{1 + e^{-z}}$$

Cost function

- $\sum(Y_i - x_i^T\beta)^2 = (\boldsymbol{Y} - \boldsymbol{x}\beta)^T(\boldsymbol{Y} - \boldsymbol{x}\beta) = ||\boldsymbol{Y} - \boldsymbol{x}\beta||^2$
  This cost function is minimized at $\left(\boldsymbol{x}^T\boldsymbol{x}\right)^{-1}\boldsymbol{x}^T\boldsymbol{Y}$

- $-\sum Y_i \log g^{-1}(x_i^T\beta) + (1 - Y_i)\log\left(1 - g^{-1}(x_i^T\beta)\right)$

$$g^{-1}(z) = \frac{1}{1 + e^{-z}}$$

Cost function

- $\sum(Y_i - x_i^T\beta)^2 = (\boldsymbol{Y} - \boldsymbol{x}\beta)^T(\boldsymbol{Y} - \boldsymbol{x}\beta) = ||\boldsymbol{Y} - \boldsymbol{x}\beta||^2$
  This cost function is minimized at $\left(\boldsymbol{x}^T\boldsymbol{x}\right)^{-1}\boldsymbol{x}^T\boldsymbol{Y}$

- $-\sum Y_i \log g^{-1}(x_i^T\beta) + (1 - Y_i)\log\left(1 - g^{-1}(x_i^T\beta)\right)$

$$g^{-1}(z) = \frac{1}{1 + e^{-z}}$$

  This cost function has no analytical solution...

Cost function

- $\sum(Y_i - x_i^T \beta)^2 = (\boldsymbol{Y} - \boldsymbol{x}\beta)^T(\boldsymbol{Y} - \boldsymbol{x}\beta) = ||\boldsymbol{Y} - \boldsymbol{x}\beta||^2 + \lambda||\beta||$

  This cost function is minimized at $\left(\boldsymbol{x}^T\boldsymbol{x}\right)^{-1}\boldsymbol{x}^T\boldsymbol{Y}$

- $-\sum Y_i \log g^{-1}(x_i^T\beta) + (1 - Y_i)\log\left(1 - g^{-1}(x_i^T\beta)\right) + \lambda||\beta||$

  $$g^{-1}(z) = \frac{1}{1 + e^{-z}}$$

  This cost function has no analytical solution...

  Regularized cost functions also don't have closed form solutions

Objective function

- An objective function is <u>a target of an optimization procedure</u>
- A cost function *IS* an objective function

Objective function

- An objective function is a target of an optimization procedure
- A cost function *IS* an objective function
- But there are obviously *other* objective functions
- E.g., a likelihood is an objective function

Objective function

- An objective function is a target of an optimization procedure
- A cost function *IS* an objective function
- But there are obviously *other* objective functions
- E.g., a likelihood is an objective function
- MLE maximizes the (log) likelihood, e.g.,

$$\underset{\beta}{argmax}\ (2\pi\sigma^2)^{-\frac{n}{2}} e^{-\frac{1}{2\sigma^2}(\mathbf{Y}-\mathbf{x}\beta)^T(\mathbf{Y}-\mathbf{x}\beta)}$$

or

$$\underset{\beta}{argmax}\ \prod \left(\frac{1}{1+e^{-\mathbf{x}_i^T\beta}}\right)^{Y_i}\left(\frac{1}{1+e^{\mathbf{x}_i^T\beta}}\right)^{1-Y_i}$$

Objective function

- An objective function is a target of an optimization procedure
- A cost function *IS* an objective function
- But there are obviously *other* objective functions
- E.g., a likelihood is an objective function
- MLE maximizes the (log) likelihood, e.g.,

$$\underset{\beta}{argmax}\ (2\pi\sigma^2)^{-\frac{n}{2}} e^{-\frac{1}{2\sigma^2}(\mathbf{Y}-\mathbf{x}\beta)^T(\mathbf{Y}-\mathbf{x}\beta)}$$

or

$$\underset{\beta}{argmax}\ \prod \left(\frac{1}{1+e^{-\mathbf{x}_i^T\beta}}\right)^{Y_i}\left(\frac{1}{1+e^{\mathbf{x}_i^T\beta}}\right)^{1-Y_i}$$

- But in Machine Learning
  the standard orientation and nomenclature is "minimize cost"

Exercise of the day

Find parameters that maximizes the fit of model to data

Exercise of the day

Find parameters that maximizes the fit of model to data

$$\iff$$

minimize distance between predicted and observed values

Exercise of the day

Find parameters that maximizes the fit of model to data

$$\Longleftrightarrow$$

minimize distance between predicted and observed values

$$\Longleftrightarrow$$

$$\hat{Y}_i \approx Y_i$$

Gradients

For some function $f(\boldsymbol{x})$, the gradient

$$\nabla f(\boldsymbol{a}) = \left(\frac{\partial f}{\partial x_1}(\boldsymbol{a}), \frac{\partial f}{\partial x_2}(\boldsymbol{a}), \cdots, \frac{\partial f}{\partial x_p}(\boldsymbol{a}) \right)$$

collects all the instantaneous slopes (derivatives)
with respect to each variable of $\boldsymbol{x}$ and evaluates them at point $\boldsymbol{a}$

Gradients

For some function $f(\boldsymbol{x})$, the gradient

$$\nabla f(\boldsymbol{a}) = \left(\frac{\partial f}{\partial x_1}(\boldsymbol{a}), \frac{\partial f}{\partial x_2}(\boldsymbol{a}), \cdots, \frac{\partial f}{\partial x_p}(\boldsymbol{a}) \right)$$

collects all the instantaneous slopes (derivatives)
with respect to each variable of $\boldsymbol{x}$ and evaluates them at point $\boldsymbol{a}$

$\nabla f(\boldsymbol{a})$ points in the direction of greatest increase in f at position $\boldsymbol{a}$

Gradients

For some function $f(\boldsymbol{x})$, the gradient

$$\nabla f(\boldsymbol{a}) = \left(\frac{\partial f}{\partial x_1}(\boldsymbol{a}), \frac{\partial f}{\partial x_2}(\boldsymbol{a}), \cdots, \frac{\partial f}{\partial x_p}(\boldsymbol{a}) \right)$$

collects all the instantaneous slopes (derivatives)
with respect to each variable of $\boldsymbol{x}$ and evaluates them at point $\boldsymbol{a}$

$\nabla f(\boldsymbol{a})$ points in the direction of greatest increase in f at position $\boldsymbol{a}$
The magnitude $|\nabla f(\boldsymbol{a})|$ is proportional to the steepness in f at $\boldsymbol{a}$

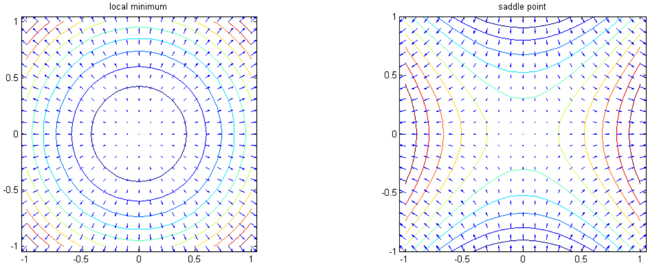
Gradients

For some function $f(\boldsymbol{x})$, the gradient

$$\nabla f(\boldsymbol{a}) = \left(\frac{\partial f}{\partial x_1}(\boldsymbol{a}), \frac{\partial f}{\partial x_2}(\boldsymbol{a}), \cdots, \frac{\partial f}{\partial x_p}(\boldsymbol{a}) \right)$$

collects all the instantaneous slopes (derivatives)
with respect to each variable of $\boldsymbol{x}$ and evaluates them at point $\boldsymbol{a}$

$\nabla f(\boldsymbol{a})$ points in the direction of greatest increase in f at position $\boldsymbol{a}$
The magnitude $|\nabla f(\boldsymbol{a})|$ is proportional to the steepness in f at $\boldsymbol{a}$



Gradient Descent

http://vis.supstat.com/2013/03/gradient-descent-algorithm-with-r/

0. Choose step size α and precision threshold ϵ
1. Select starting point $\boldsymbol{x}^{(0)}$, set $i = 1$
2. Update $\boldsymbol{x}^{(t)} = \boldsymbol{x}^{(t-1)} - \alpha \nabla f(\boldsymbol{x}^{(t-1)})$
3. If $\frac{|f(\boldsymbol{x}^{(t-1)})| - |f(\boldsymbol{x}^{(t)})|}{|f(\boldsymbol{x}^{(t-1)})|} < \epsilon$, return *min* $|f(\boldsymbol{x}^{(t)})|$ & *argmin* $\boldsymbol{x}^{(t)}$
4. else, return to step 2.

Gradient Descent

http://vis.supstat.com/2013/03/gradient-descent-algorithm-with-r/

0. Choose step size α and precision threshold ϵ
1. Select starting point $\boldsymbol{x}^{(0)}$, set $i = 1$
2. Update $\boldsymbol{x}^{(t)} = \boldsymbol{x}^{(t-1)} - \alpha \nabla f(\boldsymbol{x}^{(t-1)})$
3. If $\frac{|f(\boldsymbol{x}^{(t-1)})| - |f(\boldsymbol{x}^{(t)})|}{|f(\boldsymbol{x}^{(t-1)})|} < \epsilon$, return *min* $|f(\boldsymbol{x}^{(t)})|$ & *argmin* $\boldsymbol{x}^{(t)}$
4. else, return to step 2. [how do we choose α and ϵ?]

Step Size α and Stopping Criterion ϵ

Here's a way to choose α:

Step Size α and Stopping Criterion ϵ

Here's a way to choose α:

- If $\frac{|\nabla f(\boldsymbol{a}) - \nabla f(\boldsymbol{a}')|}{|a - a'|} < c \in \mathbb{R}$, choose $\alpha \leq 1/c$

Step Size α and Stopping Criterion ϵ

Here's a way to choose α:

- If $\frac{|\nabla f(\mathbf{a}) - \nabla f(\mathbf{a}')|}{|a-a'|} < c \in \mathbb{R}$, choose $\alpha \leq 1/c$
- E.g., if $f(x) = x^2$, then $\nabla f(x) = f'(x) = 2x$

Step Size α and Stopping Criterion ϵ

Here's a way to choose α:

- If $\frac{|\nabla f(\boldsymbol{a})-\nabla f(\boldsymbol{a}')|}{|a-a'|} < c \in \mathbb{R}$, choose $\alpha \leq 1/c$
- E.g., if $f(x) = x^2$, then $\nabla f(x) = f'(x) = 2x$
- So $\frac{|\nabla f(\boldsymbol{a})-\nabla f(\boldsymbol{a}')|}{|a-a'|} = \frac{2(a-a')}{(a-a')} = 2$, and $\alpha = 1/2$ is optimal

Step Size α and Stopping Criterion ϵ

Here's a way to choose α:

- If $\frac{|\nabla f(\mathbf{a})-\nabla f(\mathbf{a}')|}{|a-a'|} < c \in \mathbb{R}$, choose $\alpha \leq 1/c$
- E.g., if $f(x) = x^2$, then $\nabla f(x) = f'(x) = 2x$
- So $\frac{|\nabla f(\mathbf{a})-\nabla f(\mathbf{a}')|}{|a-a'|} = \frac{2(a-a')}{(a-a')} = 2$, and $\alpha = 1/2$ is optimal

Here's *another* (adaptive) way to choose α

Step Size α and Stopping Criterion ϵ

Here's a way to choose α:

- If $\frac{|\nabla f(\boldsymbol{a}) - \nabla f(\boldsymbol{a}')|}{|a-a'|} < c \in \mathbb{R}$, choose $\alpha \leq 1/c$
- E.g., if $f(x) = x^2$, then $\nabla f(x) = f'(x) = 2x$
- So $\frac{|\nabla f(\boldsymbol{a}) - \nabla f(\boldsymbol{a}')|}{|a-a'|} = \frac{2(a-a')}{(a-a')} = 2$, and $\alpha = 1/2$ is optimal

Here's *another* (adaptive) way to choose α

- Let $\Delta \boldsymbol{x}^{(t)} = \boldsymbol{x}^{(t)} - \boldsymbol{x}^{(t-1)}$

Step Size α and Stopping Criterion ϵ

Here's a way to choose α:

- If $\frac{|\nabla f(\boldsymbol{a}) - \nabla f(\boldsymbol{a}')|}{|a - a'|} < c \in \mathbb{R}$, choose $\alpha \leq 1/c$
- E.g., if $f(x) = x^2$, then $\nabla f(x) = f'(x) = 2x$
- So $\frac{|\nabla f(\boldsymbol{a}) - \nabla f(\boldsymbol{a}')|}{|a - a'|} = \frac{2(a - a')}{(a - a')} = 2$, and $\alpha = 1/2$ is optimal

Here's *another* (adaptive) way to choose α

- Let $\Delta \boldsymbol{x}^{(t)} = \boldsymbol{x}^{(t)} - \boldsymbol{x}^{(t-1)}$

  and let $\Delta \nabla f(\boldsymbol{x}^{(t)}) = \nabla f(\boldsymbol{x}^{(t)}) - \nabla f(\boldsymbol{x}^{(t-1)})$

Step Size α and Stopping Criterion ϵ

Here's a way to choose α:

- If $\frac{|\nabla f(\boldsymbol{a}) - \nabla f(\boldsymbol{a}')|}{|a - a'|} < c \in \mathbb{R}$, choose $\alpha \leq 1/c$
- E.g., if $f(x) = x^2$, then $\nabla f(x) = f'(x) = 2x$
- So $\frac{|\nabla f(\boldsymbol{a}) - \nabla f(\boldsymbol{a}')|}{|a - a'|} = \frac{2(a - a')}{(a - a')} = 2$, and $\alpha = 1/2$ is optimal

Here's *another* (adaptive) way to choose α

- Let $\Delta \boldsymbol{x}^{(t)} = \boldsymbol{x}^{(t)} - \boldsymbol{x}^{(t-1)}$

  and let $\Delta \nabla f(\boldsymbol{x}^{(t)}) = \nabla f(\boldsymbol{x}^{(t)}) - \nabla f(\boldsymbol{x}^{(t-1)})$
- For step $t + 1$, let $\alpha = \frac{\Delta \nabla f(\boldsymbol{x}^{(t)})^T \Delta \boldsymbol{x}^{(t)}}{||\Delta \nabla f(\boldsymbol{x}^{(t)})||^2}$ (if gradient isn't changing keep going!)

Step Size α and Stopping Criterion ϵ

Here's a way to choose α:

- If $\frac{|\nabla f(\boldsymbol{a}) - \nabla f(\boldsymbol{a}')|}{|a-a'|} < c \in \mathbb{R}$, choose $\alpha \leq 1/c$
- E.g., if $f(x) = x^2$, then $\nabla f(x) = f'(x) = 2x$
- So $\frac{|\nabla f(\boldsymbol{a}) - \nabla f(\boldsymbol{a}')|}{|a-a'|} = \frac{2(a-a')}{(a-a')} = 2$, and $\alpha = 1/2$ is optimal

Here's *another* (adaptive) way to choose α

- Let $\Delta \boldsymbol{x}^{(t)} = \boldsymbol{x}^{(t)} - \boldsymbol{x}^{(t-1)}$

  and let $\Delta \nabla f(\boldsymbol{x}^{(t)}) = \nabla f(\boldsymbol{x}^{(t)}) - \nabla f(\boldsymbol{x}^{(t-1)})$
- For step $t+1$, let $\alpha = \frac{\Delta \nabla f(\boldsymbol{x}^{(t)})^T \Delta \boldsymbol{x}^{(t)}}{||\Delta \nabla f(\boldsymbol{x}^{(t)})||^2}$ (if gradient isn't changing keep going!)

Here's some potential stopping criterion:

Step Size α and Stopping Criterion ϵ

Here's a way to choose α:

- If $\frac{|\nabla f(\boldsymbol{a}) - \nabla f(\boldsymbol{a}')|}{|a-a'|} < c \in \mathbb{R}$, choose $\alpha \leq 1/c$
- E.g., if $f(x) = x^2$, then $\nabla f(x) = f'(x) = 2x$
- So $\frac{|\nabla f(\boldsymbol{a}) - \nabla f(\boldsymbol{a}')|}{|a-a'|} = \frac{2(a-a')}{(a-a')} = 2$, and $\alpha = 1/2$ is optimal

Here's *another* (adaptive) way to choose α

- Let $\Delta \boldsymbol{x}^{(t)} = \boldsymbol{x}^{(t)} - \boldsymbol{x}^{(t-1)}$

  and let $\Delta \nabla f(\boldsymbol{x}^{(t)}) = \nabla f(\boldsymbol{x}^{(t)}) - \nabla f(\boldsymbol{x}^{(t-1)})$
- For step $t+1$, let $\alpha = \frac{\Delta \nabla f(\boldsymbol{x}^{(t)})^T \Delta \boldsymbol{x}^{(t)}}{||\Delta \nabla f(\boldsymbol{x}^{(t)})||^2}$ (if gradient isn't changing keep going!)

Here's some potential stopping criterion:

a. $\frac{|f(\boldsymbol{x}^{(t-1)})| - |f(\boldsymbol{x}^{(t)})|}{|f(\boldsymbol{x}^{(t-1)})|} < \epsilon$

Step Size α and Stopping Criterion ϵ

Here's a way to choose α:

- If $\frac{|\nabla f(\boldsymbol{a}) - \nabla f(\boldsymbol{a}')|}{|a - a'|} < c \in \mathbb{R}$, choose $\alpha \leq 1/c$
- E.g., if $f(x) = x^2$, then $\nabla f(x) = f'(x) = 2x$
- So $\frac{|\nabla f(\boldsymbol{a}) - \nabla f(\boldsymbol{a}')|}{|a - a'|} = \frac{2(a - a')}{(a - a')} = 2$, and $\alpha = 1/2$ is optimal

Here's *another* (adaptive) way to choose α

- Let $\Delta \boldsymbol{x}^{(t)} = \boldsymbol{x}^{(t)} - \boldsymbol{x}^{(t-1)}$

  and let $\Delta \nabla f(\boldsymbol{x}^{(t)}) = \nabla f(\boldsymbol{x}^{(t)}) - \nabla f(\boldsymbol{x}^{(t-1)})$
- For step $t + 1$, let $\alpha = \frac{\Delta \nabla f(\boldsymbol{x}^{(t)})^T \Delta \boldsymbol{x}^{(t)}}{||\Delta \nabla f(\boldsymbol{x}^{(t)})||^2}$ (if gradient isn't changing keep going!)

Here's some potential stopping criterion:

a. $\frac{|f(\boldsymbol{x}^{(t-1)})| - |f(\boldsymbol{x}^{(t)})|}{|f(\boldsymbol{x}^{(t-1)})|} < \epsilon$

b. Max number of iterations

Step Size α and Stopping Criterion ϵ

Here's a way to choose α:

- If $\frac{|\nabla f(\mathbf{a}) - \nabla f(\mathbf{a}')|}{|a - a'|} < c \in \mathbb{R}$, choose $\alpha \leq 1/c$
- E.g., if $f(x) = x^2$, then $\nabla f(x) = f'(x) = 2x$
- So $\frac{|\nabla f(\mathbf{a}) - \nabla f(\mathbf{a}')|}{|a - a'|} = \frac{2(a-a')}{(a-a')} = 2$, and $\alpha = 1/2$ is optimal

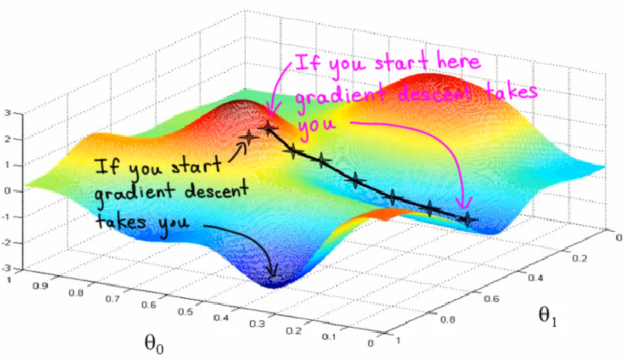
Here's *another* (adaptive) way to choose α

- Let $\Delta \boldsymbol{x}^{(t)} = \boldsymbol{x}^{(t)} - \boldsymbol{x}^{(t-1)}$
  and let $\Delta \nabla f(\boldsymbol{x}^{(t)}) = \nabla f(\boldsymbol{x}^{(t)}) - \nabla f(\boldsymbol{x}^{(t-1)})$
- For step $t+1$, let $\alpha = \frac{\Delta \nabla f(\boldsymbol{x}^{(t)})^T \Delta \boldsymbol{x}^{(t)}}{||\Delta \nabla f(\boldsymbol{x}^{(t)})||^2}$ (if gradient isn't changing keep going!)

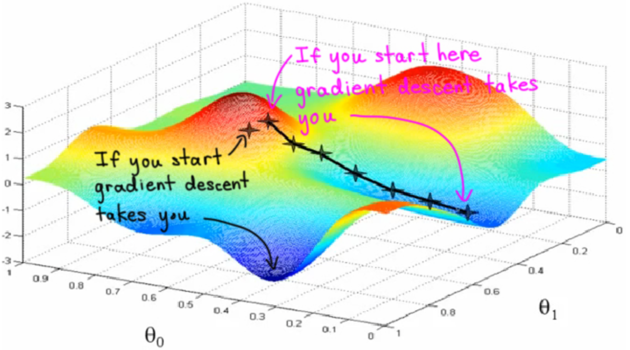
Here's some potential stopping criterion:

a. $\frac{|f(\boldsymbol{x}^{(t-1)})| - |f(\boldsymbol{x}^{(t)})|}{|f(\boldsymbol{x}^{(t-1)})|} < \epsilon$
b. Max number of iterations
c. $|\nabla f| < \epsilon$

Pitfalls (literally)

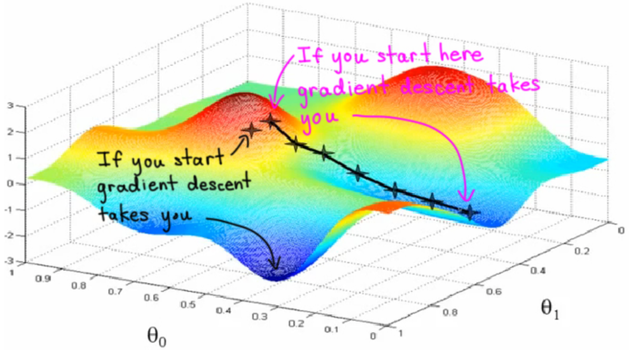


Pitfalls (literally)



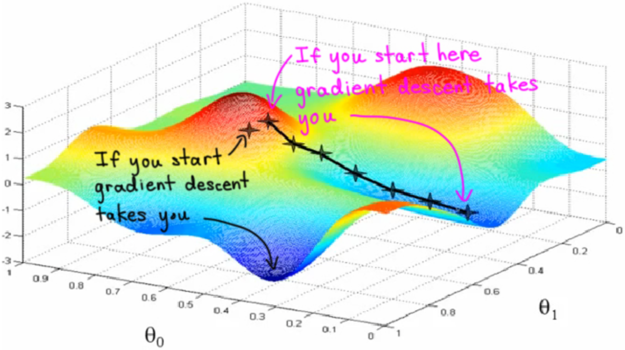
- convexity to guarantee global maximum

Pitfalls (literally)



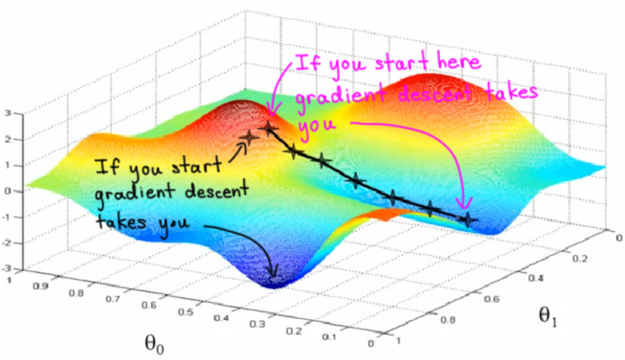
- convexity to guarantee global maximum
- Need differentiable cost function

Pitfalls (literally)



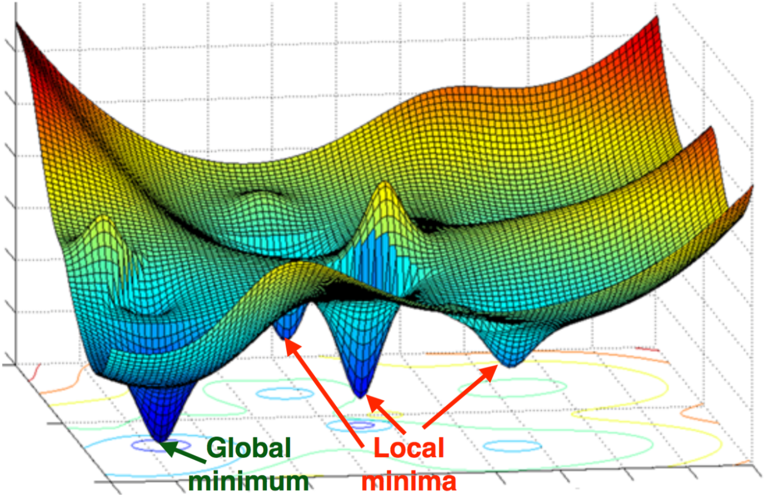
- convexity to guarantee global maximum
- Need differentiable cost function
- converges asymptotically

Pitfalls (literally)



- convexity to guarantee global maximum
- Need differentiable cost function
- converges asymptotically
- good performance requires feature scaling & parameter tuning

Here's another one...



http:
//vis.supstat.com/2013/03/gradient-descent-algorithm-with-r/

Logistic regression

$$L(\beta) = \sum Y_i \log g^{-1}(x_i^T \beta) + (1 - Y_i) \log \left(1 - g^{-1}(x_i^T \beta)\right)$$

Logistic regression

$$L(\beta) = \sum Y_i \log g^{-1}(x_i^T \beta) + (1 - Y_i) \log \left(1 - g^{-1}(x_i^T \beta)\right)$$

$$\nabla L(\beta) = ?$$

Logistic regression

$$L(\beta) = \sum Y_i \log g^{-1}(x_i^T \beta) + (1 - Y_i) \log \left(1 - g^{-1}(x_i^T \beta)\right)$$

$$\nabla L(\beta) = ?$$

$$\frac{d}{dz} g^{-1}(z) = \frac{d}{dz} \frac{1}{1 + e^{-z}}$$

Logistic regression

$$L(\beta) = \sum Y_i \log g^{-1}(x_i^T \beta) + (1 - Y_i) \log \left(1 - g^{-1}(x_i^T \beta)\right)$$

$$\nabla L(\beta) = ?$$

$$\begin{aligned} \frac{d}{dz} g^{-1}(z) &= \frac{d}{dz} \frac{1}{1 + e^{-z}} \\ &= \frac{d}{dz} (1 + e^{-z})^{-1} \end{aligned}$$

Logistic regression

$$L(\beta) = \sum Y_i \log g^{-1}(x_i^T \beta) + (1 - Y_i) \log \left(1 - g^{-1}(x_i^T \beta)\right)$$

$$\nabla L(\beta) = ?$$

$$\begin{aligned}
\frac{d}{dz} g^{-1}(z) &= \frac{d}{dz} \frac{1}{1 + e^{-z}} \\
&= \frac{d}{dz} (1 + e^{-z})^{-1} \\
&= -(1 + e^{-z})^{-2} (-e^{-z})
\end{aligned}$$

Logistic regression

$$L(\beta) = \sum Y_i \log g^{-1}(x_i^T \beta) + (1 - Y_i) \log \left(1 - g^{-1}(x_i^T \beta)\right)$$

$$\nabla L(\beta) = ?$$

$$\begin{aligned}
\frac{d}{dz} g^{-1}(z) &= \frac{d}{dz} \frac{1}{1 + e^{-z}} \\
&= \frac{d}{dz} (1 + e^{-z})^{-1} \\
&= -(1 + e^{-z})^{-2} (-e^{-z}) \\
&= \frac{1}{1 + e^{-z}} \frac{e^{-z}}{1 + e^{-z}}
\end{aligned}$$

Logistic regression

$$L(\beta) = \sum Y_i \log g^{-1}(x_i^T \beta) + (1 - Y_i) \log \left(1 - g^{-1}(x_i^T \beta)\right)$$

$$\nabla L(\beta) = ?$$

$$\begin{aligned}
\frac{d}{dz} g^{-1}(z) &= \frac{d}{dz} \frac{1}{1 + e^{-z}} \\
&= \frac{d}{dz} (1 + e^{-z})^{-1} \\
&= -(1 + e^{-z})^{-2} (-e^{-z}) \\
&= \frac{1}{1 + e^{-z}} \frac{e^{-z}}{1 + e^{-z}} \\
&= g^{-1}(z) \left(1 - g^{-1}(z)\right)
\end{aligned}$$

Logistic regression

$$L(\beta) = \sum Y_i \log g^{-1}(x_i^T \beta) + (1 - Y_i) \log \left(1 - g^{-1}(x_i^T \beta)\right)$$

$$\nabla L(\beta) = ?$$

$$\frac{\partial}{\partial \beta_{\color{red}{j}}} L(\beta)$$

$$= \frac{\partial}{\partial \beta_j} \left(\sum Y_i \log g^{-1}(x_i^T \beta) + (1 - Y_i) \log \left(1 - g^{-1}(x_i^T \beta)\right)\right)$$

Logistic regression

$$L(\beta) = \sum Y_i \log g^{-1}(x_i^T \beta) + (1 - Y_i) \log \left(1 - g^{-1}(x_i^T \beta)\right)$$

$$\nabla L(\beta) = ?$$

$$\begin{aligned}
&\frac{\partial}{\partial \beta_{j}} L(\beta) \\
&= \frac{\partial}{\partial \beta_j} \left(\sum Y_i \log g^{-1}(x_i^T \beta) + (1 - Y_i) \log \left(1 - g^{-1}(x_i^T \beta)\right)\right) \\
&= \sum \frac{\partial}{\partial \beta_j} g^{-1}(x_i^T \beta) \left(\frac{Y_i}{g^{-1}(x_i^T \beta)} - \frac{(1 - Y_i)}{(1 - g^{-1}(x_i^T \beta))}\right)
\end{aligned}$$

Logistic regression

$$L(\beta) = \sum Y_i \log g^{-1}(x_i^T \beta) + (1 - Y_i) \log \left(1 - g^{-1}(x_i^T \beta)\right)$$

$$\nabla L(\beta) = ?$$

$$\begin{aligned}
&\frac{\partial}{\partial \beta_j} L(\beta) \\
&= \frac{\partial}{\partial \beta_j} \left(\sum Y_i \log g^{-1}(x_i^T \beta) + (1 - Y_i) \log \left(1 - g^{-1}(x_i^T \beta)\right)\right) \\
&= \sum \frac{\partial}{\partial \beta_j} g^{-1}(x_i^T \beta) \left(\frac{Y_i}{g^{-1}(x_i^T \beta)} - \frac{(1 - Y_i)}{(1 - g^{-1}(x_i^T \beta))} \right) \\
&= \sum g^{-1}(x_i^T \beta) \left(1 - g^{-1}(x_i^T \beta)\right) x_{ij} \left(\frac{Y_i}{g^{-1}(x_i^T \beta)} - \frac{(1 - Y_i)}{(1 - g^{-1}(x_i^T \beta))} \right)
\end{aligned}$$

Logistic regression

$$L(\beta) = \sum Y_i \log g^{-1}(x_i^T \beta) + (1 - Y_i) \log \left(1 - g^{-1}(x_i^T \beta)\right)$$

$$\nabla L(\beta) = ?$$

$$\begin{aligned}
&\frac{\partial}{\partial \beta_j} L(\beta) \\
&= \frac{\partial}{\partial \beta_j} \left(\sum Y_i \log g^{-1}(x_i^T \beta) + (1 - Y_i) \log \left(1 - g^{-1}(x_i^T \beta)\right)\right) \\
&= \sum \frac{\partial}{\partial \beta_j} g^{-1}(x_i^T \beta) \left(\frac{Y_i}{g^{-1}(x_i^T \beta)} - \frac{(1 - Y_i)}{(1 - g^{-1}(x_i^T \beta))}\right) \\
&= \sum g^{-1}(x_i^T \beta) \left(1 - g^{-1}(x_i^T \beta)\right) x_{ij} \left(\frac{Y_i}{g^{-1}(x_i^T \beta)} - \frac{(1 - Y_i)}{(1 - g^{-1}(x_i^T \beta))}\right) \\
&= \sum x_{ij} \left(Y_i \left(1 - g^{-1}(x_i^T \beta)\right) - (1 - Y_i) g^{-1}(x_i^T \beta)\right)
\end{aligned}$$

Logistic regression

$$L(\beta) = \sum Y_i \log g^{-1}(x_i^T\beta) + (1 - Y_i)\log\left(1 - g^{-1}(x_i^T\beta)\right)$$

$$\nabla L(\beta) = ?$$

$$\begin{aligned}
&\frac{\partial}{\partial \beta_j} L(\beta) \\
&= \frac{\partial}{\partial \beta_j}\left(\sum Y_i \log g^{-1}(x_i^T\beta) + (1 - Y_i)\log\left(1 - g^{-1}(x_i^T\beta)\right)\right) \\
&= \sum \frac{\partial}{\partial \beta_j} g^{-1}(x_i^T\beta)\left(\frac{Y_i}{g^{-1}(x_i^T\beta)} - \frac{(1 - Y_i)}{(1 - g^{-1}(x_i^T\beta))}\right) \\
&= \sum g^{-1}(x_i^T\beta)\left(1 - g^{-1}(x_i^T\beta)\right) x_{ij}\left(\frac{Y_i}{g^{-1}(x_i^T\beta)} - \frac{(1 - Y_i)}{(1 - g^{-1}(x_i^T\beta))}\right) \\
&= \sum x_{ij}\left(Y_i\left(1 - g^{-1}(x_i^T\beta)\right) - (1 - Y_i)g^{-1}(x_i^T\beta)\right) \\
&= \sum x_{ij}\left(Y_i - g^{-1}(x_i^T\beta)\right) = \sum_{i=1}^{n} x_{ij}\left(Y_i - \frac{1}{1 + e^{-x_i^T\beta}}\right) = \sum_{i=1}^{n} x_{ij}(Y_i - \hat{Y}_i)
\end{aligned}$$

Logistic regression

$$L(\beta) = \sum Y_i \log g^{-1}(x_i^T\beta) + (1 - Y_i)\log\left(1 - g^{-1}(x_i^T\beta)\right) \; - \lambda\frac{1}{2}||\beta||^2$$

$$\nabla L(\beta) = ?$$

$$\frac{\partial}{\partial \beta_j} L(\beta)$$

$$= \frac{\partial}{\partial \beta_j}\left(\sum Y_i \log g^{-1}(x_i^T\beta) + (1 - Y_i)\log\left(1 - g^{-1}(x_i^T\beta)\right)\right)$$

$$= \sum \frac{\partial}{\partial \beta_j} g^{-1}(x_i^T\beta)\left(\frac{Y_i}{g^{-1}(x_i^T\beta)} - \frac{(1 - Y_i)}{(1 - g^{-1}(x_i^T\beta))}\right)$$

$$= \sum g^{-1}(x_i^T\beta)\left(1 - g^{-1}(x_i^T\beta)\right) x_{ij} \left(\frac{Y_i}{g^{-1}(x_i^T\beta)} - \frac{(1 - Y_i)}{(1 - g^{-1}(x_i^T\beta))}\right)$$

$$= \sum x_{ij}\left(Y_i\left(1 - g^{-1}(x_i^T\beta)\right) - (1 - Y_i)g^{-1}(x_i^T\beta)\right)$$

$$= \sum x_{ij}\left(Y_i - g^{-1}(x_i^T\beta)\right) = \sum_{i=1}^{n} x_{ij}\left(Y_i - \frac{1}{1 + e^{-x_i^T\beta}}\right) = \sum_{i=1}^{n} x_{ij}(Y_i - \hat{Y}_i)$$

Logistic regression

$$L(\beta) = \sum Y_i \log g^{-1}(x_i^T\beta) + (1 - Y_i)\log\left(1 - g^{-1}(x_i^T\beta)\right) - \lambda|\beta|$$

$$\nabla L(\beta) = ?$$

$$\begin{aligned}
&\frac{\partial}{\partial \beta_j} L(\beta)\\
&= \frac{\partial}{\partial \beta_j}\left(\sum Y_i \log g^{-1}(x_i^T\beta) + (1 - Y_i)\log\left(1 - g^{-1}(x_i^T\beta)\right)\right)\\
&= \sum \frac{\partial}{\partial \beta_j} g^{-1}(x_i^T\beta)\left(\frac{Y_i}{g^{-1}(x_i^T\beta)} - \frac{(1 - Y_i)}{(1 - g^{-1}(x_i^T\beta))}\right)\\
&= \sum g^{-1}(x_i^T\beta)\left(1 - g^{-1}(x_i^T\beta)\right) x_{ij}\left(\frac{Y_i}{g^{-1}(x_i^T\beta)} - \frac{(1 - Y_i)}{(1 - g^{-1}(x_i^T\beta))}\right)\\
&= \sum x_{ij}\left(Y_i\left(1 - g^{-1}(x_i^T\beta)\right) - (1 - Y_i)g^{-1}(x_i^T\beta)\right)\\
&= \sum x_{ij}\left(Y_i - g^{-1}(x_i^T\beta)\right) = \sum_{i=1}^{n} x_{ij}\left(Y_i - \frac{1}{1 + e^{-x_i^T\beta}}\right) = \sum_{i=1}^{n} x_{ij}(Y_i - \hat{Y}_i)
\end{aligned}$$

Logistic regression

$$L(\beta) = \sum Y_i \log g^{-1}(x_i^T \beta) + (1 - Y_i) \log \left(1 - g^{-1}(x_i^T \beta)\right)$$

$$\nabla L(\beta) = \begin{bmatrix} \sum x_{i1} \left(Y_i - \frac{1}{1+e^{-x_i^T \beta}} \right) \\ \sum x_{i2} \left(Y_i - \frac{1}{1+e^{-x_i^T \beta}} \right) \\ \vdots \\ \sum x_{ip} \left(Y_i - \frac{1}{1+e^{-x_i^T \beta}} \right) \end{bmatrix}$$

$$\beta^{(t)} = \beta^{(k-1)} + \alpha \nabla f(\beta^{(k-1)})$$

Potential Gradient Decent Drawbacks

Potential Gradient Decent Drawbacks

- Memory (data needs to fit)
- Processor (cost function over all rows is expensive)

Potential Gradient Decent Drawbacks *Solutions*

- Observations contribute equal weight to the gradient

$$\frac{\partial L_{1,\cdots n}}{\partial \beta_j}(\beta) = \sum_{i=1}^{n} x_{ij} \left(Y_i - \frac{1}{1 + e^{-x_i^T \beta}} \right)$$

Potential Gradient Decent Drawbacks *Solutions*

- Observations contribute equal weight to the gradient

$$\frac{\partial L_{1,\cdots n}}{\partial \beta_j}(\beta) = \sum_{i=1}^{n} x_{ij} \left(Y_i - \frac{1}{1 + e^{-x_i^T \beta}} \right)$$

- We could just use only a single data point at each iteration?

$$\frac{\partial L_i}{\partial \beta_j}(\beta) = x_{ij} \left(Y_i - \frac{1}{1 + e^{-x_i^T \beta}} \right)$$

Potential Gradient Decent Drawbacks *Solutions*

- Observations contribute equal weight to the gradient

$$\frac{\partial L_{1,\cdots n}}{\partial \beta_j}(\beta) = \sum_{i=1}^{n} x_{ij} \left(Y_i - \frac{1}{1+e^{-x_i^T \beta}} \right)$$

- We could just use only a single data point at each iteration?

$$\frac{\partial L_i}{\partial \beta_j}(\beta) = x_{ij} \left(Y_i - \frac{1}{1+e^{-x_i^T \beta}} \right)$$

- The expected direction of the gradient would stay the same

$$\frac{1}{n} \mathrm{E}\left[\frac{\partial L_{1,\cdots n}}{\partial \beta_j}(\beta) \right] = \mathrm{E}\left[\frac{\partial L_i}{\partial \beta_j}(\beta) \right]$$

Potential Gradient Decent Drawbacks *Solutions*

- Observations contribute equal weight to the gradient

$$\frac{\partial L_{1,\cdots n}}{\partial \beta_j}(\beta) = \sum_{i=1}^{n} x_{ij}\left(Y_i - \frac{1}{1+e^{-x_i^T\beta}}\right)$$

- We could just use only a single data point at each iteration?

$$\frac{\partial L_i}{\partial \beta_j}(\beta) = x_{ij}\left(Y_i - \frac{1}{1+e^{-x_i^T\beta}}\right)$$

- The expected direction of the gradient would stay the same

$$\frac{1}{n}\mathrm{E}\left[\frac{\partial L_{1,\cdots n}}{\partial \beta_j}(\beta)\right] = \mathrm{E}\left[\frac{\partial L_i}{\partial \beta_j}(\beta)\right]$$

- We could also use a batch of data points at each iteration

Potential Gradient Decent Drawbacks *Solutions*

- Observations contribute equal weight to the gradient

$$\frac{\partial L_{1,\cdots n}}{\partial \beta_j}(\beta) = \sum_{i=1}^{n} x_{ij} \left(Y_i - \frac{1}{1+e^{-x_i^T \beta}} \right)$$

- We could just use only a single data point at each iteration?

$$\frac{\partial L_i}{\partial \beta_j}(\beta) = x_{ij} \left(Y_i - \frac{1}{1+e^{-x_i^T \beta}} \right)$$

- The expected direction of the gradient would stay the same

$$\frac{1}{n}\mathrm{E}\left[\frac{\partial L_{1,\cdots n}}{\partial \beta_j}(\beta)\right] = \mathrm{E}\left[\frac{\partial L_i}{\partial \beta_j}(\beta)\right]$$

- We could also use a batch of data points at each iteration

  This would address memory/processing limitations

Potential Gradient Decent Drawbacks *Solutions*

- Observations contribute equal weight to the gradient

$$\frac{\partial L_{1,\cdots n}}{\partial \beta_j}(\beta) = \sum_{i=1}^{n} x_{ij} \left(Y_i - \frac{1}{1 + e^{-x_i^T \beta}} \right)$$

- We could just use only a single data point at each iteration?

$$\frac{\partial L_i}{\partial \beta_j}(\beta) = x_{ij} \left(Y_i - \frac{1}{1 + e^{-x_i^T \beta}} \right)$$

- The expected direction of the gradient would stay the same

$$\frac{1}{n} \mathrm{E}\left[\frac{\partial L_{1,\cdots n}}{\partial \beta_j}(\beta) \right] = \mathrm{E}\left[\frac{\partial L_i}{\partial \beta_j}(\beta) \right]$$

- We could also use a batch of data points at each iteration

  This would address memory/processing limitations
  It has been empirically proven to also often converge faster!

Potential Gradient Decent Drawbacks *Solutions*

- Observations contribute equal weight to the gradient

$$\frac{\partial L_{1,\cdots n}}{\partial \beta_j}(\beta) = \sum_{i=1}^{n} x_{ij} \left(Y_i - \frac{1}{1 + e^{-x_i^T \beta}} \right)$$

- We could just use only a single data point at each iteration?

$$\frac{\partial L_i}{\partial \beta_j}(\beta) = x_{ij} \left(Y_i - \frac{1}{1 + e^{-x_i^T \beta}} \right)$$

- The expected direction of the gradient would stay the same

$$\frac{1}{n} \mathrm{E}\left[\frac{\partial L_{1,\cdots n}}{\partial \beta_j}(\beta) \right] = \mathrm{E}\left[\frac{\partial L_i}{\partial \beta_j}(\beta) \right]$$

- We could also use a batch of data points at each iteration

  This would address memory/processing limitations
  It has been empirically proven to also often converge faster!
  It does tend to oscillate around as it nears the minimum...

Newton-Raphson

- Gradient Decent

$$x^{(t)} = x^{(t-1)} - \alpha f'(x^{(t-1)})$$

$$\boldsymbol{x}^{(t)} = \boldsymbol{x}^{(t-1)} - \alpha \nabla f(\boldsymbol{x}^{(t-1)})$$

Newton-Raphson

- Gradient Decent

$$x^{(t)} = x^{(t-1)} - \alpha f'(x^{(t-1)})$$

$$\boldsymbol{x}^{(t)} = \boldsymbol{x}^{(t-1)} - \alpha \nabla f(\boldsymbol{x}^{(t-1)})$$

- Newton-Raphson

$$x^{(t)} = x^{(t-1)} - \frac{f'(x^{(t-1)})}{f''(x^{(t-1)})}$$

$$\boldsymbol{x}^{(t)} = \boldsymbol{x}^{(t-1)} - [Hf(\boldsymbol{x}^{(t-1)})]^{-1} \nabla f(\boldsymbol{x}^{(t-1)})$$

Newton-Raphson

- Gradient Decent

$$x^{(t)} = x^{(t-1)} - \alpha f'(x^{(t-1)})$$

$$\boldsymbol{x}^{(t)} = \boldsymbol{x}^{(t-1)} - \alpha \nabla f(\boldsymbol{x}^{(t-1)})$$

- Newton-Raphson

$$x^{(t)} = x^{(t-1)} - \frac{f'(x^{(t-1)})}{f''(x^{(t-1)})}$$

$$\boldsymbol{x}^{(t)} = \boldsymbol{x}^{(t-1)} - [Hf(\boldsymbol{x}^{(t-1)})]^{-1} \nabla f(\boldsymbol{x}^{(t-1)})$$

$$Hf(\boldsymbol{x}) = \boldsymbol{H}, \text{ where } \boldsymbol{H}_{ij} = \frac{\partial f}{\partial a_i \partial a_j}(\boldsymbol{x})$$

Newton-Raphson

- Gradient Decent

$$x^{(t)} = x^{(t-1)} - \alpha f'(x^{(t-1)})$$

$$\boldsymbol{x}^{(t)} = \boldsymbol{x}^{(t-1)} - \alpha \nabla f(\boldsymbol{x}^{(t-1)})$$

- Newton-Raphson

$$x^{(t)} = x^{(t-1)} - \frac{f'(x^{(t-1)})}{f''(x^{(t-1)})}$$

$$\boldsymbol{x}^{(t)} = \boldsymbol{x}^{(t-1)} - [Hf(\boldsymbol{x}^{(t-1)})]^{-1} \nabla f(\boldsymbol{x}^{(t-1)})$$

$$Hf(\boldsymbol{x}) = \boldsymbol{H}, \text{ where } \boldsymbol{H}_{ij} = \frac{\partial f}{\partial a_i \partial a_j}(\boldsymbol{x})$$

2nd order information makes Newton-Raphson faster

Newton-Raphson

- Gradient Decent

$$x^{(t)} = x^{(t-1)} - \alpha f'(x^{(t-1)})$$

$$\boldsymbol{x}^{(t)} = \boldsymbol{x}^{(t-1)} - \alpha \nabla f(\boldsymbol{x}^{(t-1)})$$

- Newton-Raphson

$$x^{(t)} = x^{(t-1)} - \frac{f'(x^{(t-1)})}{f''(x^{(t-1)})}$$

$$\boldsymbol{x}^{(t)} = \boldsymbol{x}^{(t-1)} - [Hf(\boldsymbol{x}^{(t-1)})]^{-1} \nabla f(\boldsymbol{x}^{(t-1)})$$

$$Hf(\boldsymbol{x}) = \boldsymbol{H}, \text{ where } \boldsymbol{H}_{ij} = \frac{\partial f}{\partial a_i \partial a_j}(\boldsymbol{x})$$

2nd order information makes Newton-Raphson faster
Inverting the Hessian matrix can be costly (or impossible)

Newton-Raphson

- Gradient Decent

$$x^{(t)} = x^{(t-1)} - \alpha f'(x^{(t-1)})$$

$$\boldsymbol{x}^{(t)} = \boldsymbol{x}^{(t-1)} - \alpha \nabla f(\boldsymbol{x}^{(t-1)})$$

- Newton-Raphson

$$x^{(t)} = x^{(t-1)} - \frac{f'(x^{(t-1)})}{f''(x^{(t-1)})}$$

$$\boldsymbol{x}^{(t)} = \boldsymbol{x}^{(t-1)} - [Hf(\boldsymbol{x}^{(t-1)})]^{-1} \nabla f(\boldsymbol{x}^{(t-1)})$$

$$Hf(\boldsymbol{x}) = \boldsymbol{H}, \text{ where } \boldsymbol{H}_{ij} = \frac{\partial f}{\partial a_i \partial a_j}(\boldsymbol{x})$$

2nd order information makes Newton-Raphson faster
Inverting the Hessian matrix can be costly (or impossible)
Newton-Raphson can diverge with an initial bad guess

Conclusions

- Best Method?

Conclusions

- Best Method?

  Depends

Conclusions

- Best Method?

  Depends
- Inventing New Cost functions?

Conclusions

- Best Method?

  Depends
- Inventing New Cost functions?

  Probably not

Conclusions

- Best Method?

  Depends
- Inventing New Cost functions?

  Probably not
- So why are we doing this?

Conclusions

- Best Method?

  Depends
- Inventing New Cost functions?

  Probably not
- So why are we doing this?

  Fitting models is just optimizing... knowing how it works could help you apply model fitting procedures more effectively

Bonus

http://cs229.stanford.edu/notes/cs229-notes1.pdf