Logistic Regression

Il-Chul Moon
Dept. of Industrial and Systems Engineering
KAIST
icmoon@kaist.ac.kr

Weekly Objectives

- Learn the logistic regression classifier
  - Understand why the logistic regression is better suited than the linear regression for classification tasks
  - Understand the logistic function
  - Understand the logistic regression classifier
  - Understand the approximation approach for the open form solutions
- Learn the gradient descent algorithm
  - Know the tailor expansion
  - Understand the gradient descent/ascent algorithm
- Learn the different between the naïve Bayes and the logistic regression
  - Understand the similarity of the two classifiers
  - Understand the differences of the two classifiers
  - Understand the performance differences

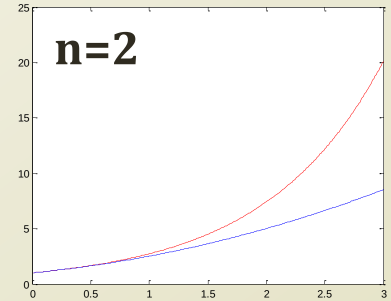
GRADIENT METHOD

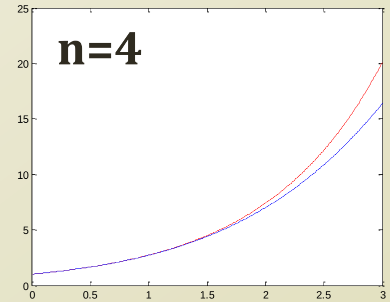
Taylor Expansion

- Taylor series is a representation of a function
  - as a infinite sum of terms calculated from the values of the function's derivatives at a fixed point.
  - $f(x) = f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \cdots$
    $= \sum_{n=0}^{\infty} \frac{f^{(n)}(a)}{n!}(x-a)^n$
  - a = a constant value
- Taylor series is possible when
  - Infinitely differentiable at a real or complex number of a
- Taylor expansion is a process of generating the Taylor series

$when\ a = 0,$

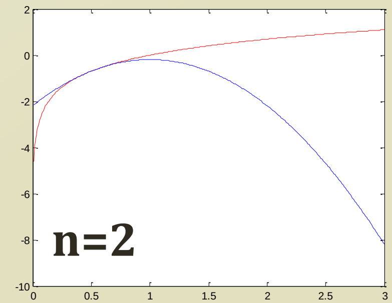
$$e^x = 1 + \frac{e^0}{1!}(x-0)^1 + \frac{e^0}{2!}(x-0)^2 + \cdots$$

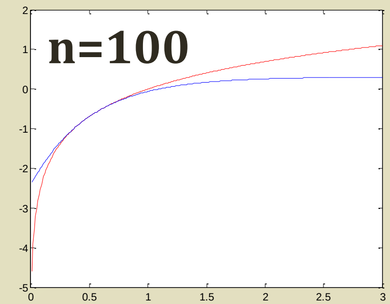




$when\ a = 0.5,$

$$logx = \log(0.5) + \frac{\frac{1}{0.5}}{1!}(x-0.5)^1 + \frac{\frac{1}{0.5^2}}{2!}(x-0.5)^2 + \cdots$$

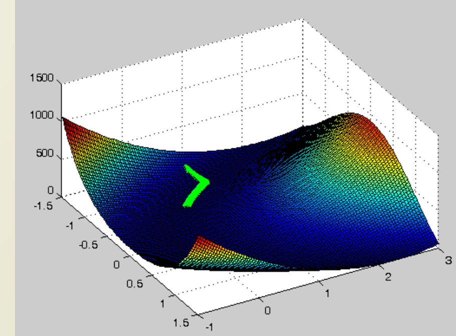


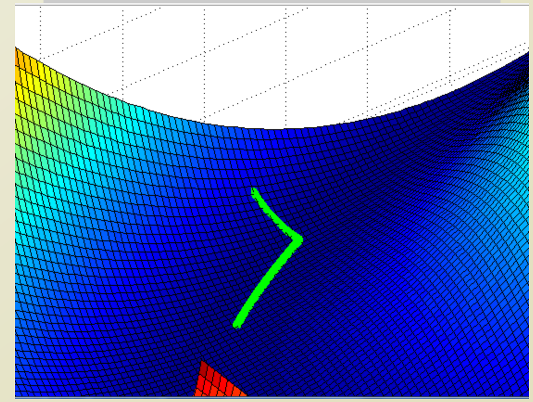


Gradient Descent/Ascent

- Gradient descent/ascent method is
  - Given a differentiable function of *f(x)* and an initial parameter of x_1
  - Iteratively moving the parameter to the lower/higher value of *f(x)*
  - By taking the direction of the negative/positive gradient of *f(x)*
- Why this works?
  - $f(x) = f(a) + \frac{f'(a)}{1!}(x-a) + O(||x-a||^2)$ — Useful Big-Oh Notation
    - Assume $a=x_1$ and $x=x_1+h\mathbf{u}$, $\mathbf{u}$ is the unit direction vector for the partial deriv.
    - $f(x_1 + h\mathbf{u}) = f(x_1) + hf'(x_1)\mathbf{u} + h^2O(1)$
    - $f(x_1 + h\mathbf{u}) - f(x_1) \approx hf'(x_1)\mathbf{u}$ — Always???
    - $\mathbf{u}^* = argmin_{\mathbf{u}}\{f(x_1 + h\mathbf{u}) - f(x_1)\} = argmin_{\mathbf{u}} hf'(x_1)\mathbf{u} = -\frac{f'(x_1)}{|f'(x_1)|}$
    - $\because f(x_1 + h\mathbf{u}) \leq f(x_1), \vec{a}\cdot\vec{b} = |\vec{a}||\vec{b}|cos\alpha$
  - $x_{t+1} \leftarrow x_t + h\mathbf{u}^* = x_t - h\frac{f'(x_1)}{|f'(x_1)|}$ — Gradient Descent
- Perfectly applicable to $\hat{\theta} = argmax_\theta \sum_{1\leq i\leq N} log(P(Y_i|X_i;\theta))$ — Gradient Ascent
  - $f(\theta) = \sum_{1\leq i\leq N} log(P(Y_i|X_i;\theta))$
  - Setup an initial parameter of θ_1
  - Iteratively moving θ_t to the higher value of $f(\theta_t)$
  - By taking the direction of the ***positive*** gradient of $f(\theta_t)$

How Gradient Descent Works

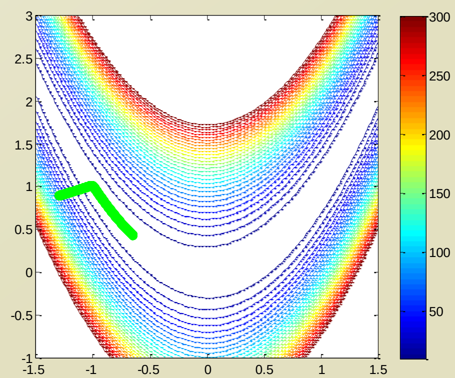


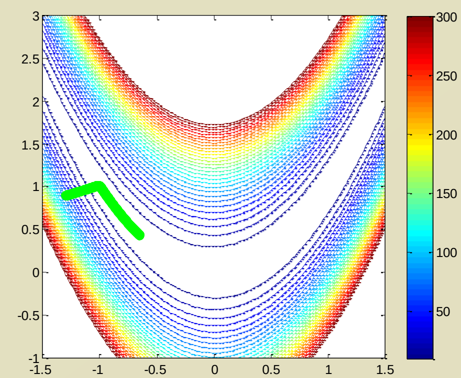


- Example function: Rosenbrock function
  - $f(x_1, x_2) = (1 - x_1)^2 + 100(x_2 - x_1^2)^2$
  - $\frac{\partial}{\partial x_1} f(x_1, x_2) = -2(1 - x_1) - 400x_1(x_2 - x_1^2)$
  - $\frac{\partial}{\partial x_2} f(x_1, x_2) = 200(x_2 - x_1^2)$
- Assume the initial point
  - $\mathbf{x}^0 = (x_1^0, x_2^0) = (-1.3, 0.9)$

Global Minimum=0 at (1,1)

- Partial derivative vector at the point
  - $\boldsymbol{f}'(\mathbf{x}^0) = \left(\frac{\partial}{\partial x_1} f(x_1, x_2), \frac{\partial}{\partial x_2} f(x_1, x_2)\right) = (-415.4, -158)$
- Update the point with the negative partial derivative in a small scale, h=0.001
  - $\mathbf{x}^1 \leftarrow \mathbf{x}^0 - h \frac{f'(\mathbf{x}^0)}{|f'(\mathbf{x}^0)|}$
  - $\mathbf{x}^1 = \begin{pmatrix} -1.3 - 0.001 \times -415.4/444.4335, \\ 0.9 - 0.001 \times -158/444.4335 \end{pmatrix}$
  - $= (-1.2991, 0.9004)$





- Repeat the update until converges

$$P(y=1|x) = \frac{e^{X\theta}}{1+e^{X\theta}}$$

Finding θ with Gradient Ascent

- $\hat{\theta} = argmax_\theta \sum_{1\le i\le N} log(P(Y_i|X_i;\theta))$
  - $f(\theta) = \sum_{1\le i\le N} log(P(Y_i|X_i;\theta))$
  - $\frac{\partial f(\theta)}{\partial \theta_j} = \frac{\partial}{\partial \theta_j}\{\sum_{1\le i\le N} log(P(Y_i|X_i;\theta))\} = \sum_{1\le i\le N} X_{i,j}(Y_i - P(y=1|x;\theta))$
- To utilize the gradient method
  - We need to know *f'(x)* which are above
    - Case of ascent: $x_{t+1} \leftarrow x_t + h\mathbf{u}^* = x_t + h\frac{f'(x_t)}{|f'(x_t)|}$
  - Then, how to iteratively update the parameter, $\boldsymbol{\theta}$
  - $\theta_j^{t+1} \leftarrow \theta_j^t + h\frac{\partial f(\theta^t)}{\partial \theta_j^t} = \theta_j^t + h\{\sum_{1\le i\le N} X_{i,j}\left(Y_i - P(Y=1|X_i;\theta^t)\right)\}$

$$= \theta_j^t + \frac{h}{C}\{\sum_{1\le i\le N} X_{i,j}\left(Y_i - \frac{e^{X_i\theta^t}}{1+e^{X_i\theta^t}}\right)\}$$

C=Normalization to the unit vector

  - θ_j^0 can be arbitrarily chosen.

Logistic Regression Matlab Exercise

- Let's do some coding…

Linear Regression Revisited

- Previously,
  - $\hat{\theta} = argmin_{\theta}(f - \hat{f})^2 = argmin_{\theta}(Y - X\theta)^2$
    $= argmin_{\theta}(Y - X\theta)^T\,(Y - X\theta) = argmin_{\theta}(Y - X\theta)^T\,(Y - X\theta)$
    $= argmin_{\theta}(\theta^T X^T X\theta - 2\theta^T X^T Y + Y^T Y) = argmin_{\theta}(\theta^T X^T X\theta - 2\theta^T X^T Y)$
  - $\nabla_{\theta}(\theta^T X^T X\theta - 2\theta^T X^T Y) = 0$
    - $2X^T X\theta - 2X^T Y = 0$
  - $\theta = (X^T X)^{-1} X^T Y$
- Any problem???
- Gradient descent can be a solution
  - $\hat{\theta} = argmin_{\theta}(f - \hat{f})^2 = argmin_{\theta}(Y - X\theta)^2 =$
    $argmin_{\theta} \sum_{1 \le i \le N}(Y^i - \sum_{1 \le j \le d} X_j^i \theta_j)^2$
  - $\frac{\partial}{\partial \theta_k} \sum_{1 \le i \le N}(Y^i - \sum_{1 \le j \le d} X_j^i \theta_j)^2 = -\sum_{1 \le i \le N} 2(Y^i - \sum_{1 \le j \le d} X_j^i \theta_j) X_k^i$
  - $\theta_k^{t+1} \leftarrow \theta_k^t - h \frac{\partial f(\theta^t)}{\partial \theta_k^t} = \theta_k^t + h \sum_{1 \le i \le N} 2(Y^i - \sum_{1 \le j \le d} X_j^i \theta_j) X_k^i$