UCLA Samueli Computer Science

CS M146 Discussion: Week 3

Perceptron, Linear Models, Optimization

Junheng Hao

Friday, 01/22/2021

UCLA
Engineer Change.

Roadmap

- Announcement
- Perceptron & Linear Models
- Optimization, MLE

UCLA
Engineer Change.

Announcements

- 5:00 pm PST, Jan. 22: Weekly Quiz 3 released on Gradescope.
- **11:59 pm PST, Jan. 24 (Sunday):** Weekly quiz 3 closed on Gradescope!
  - Start the quiz before **11:00 pm PST, Jan. 24** to have the full 60-minute time
- Problem set 1 released on campuswire/CCLE, submission on Gradescope.
  - Please assign pages of your submission with corresponding problem set outline items on GradeScope.
  - You do not need to submit code, only the results required by the problem set
  - Due on **11:59pm PST, Jan. 29 (Friday)**

UCLA
Engineer Change.

About Quiz 3

- Quiz release date and time: **Jan 22, 2021 (Friday) 05:00 PM PST**
- Quiz due/close date and time: **Jan 24, 2021 (Sunday) 11:59 PM PST**
- You will have up to **60 minutes** to take this exam. → Start before **11:00 PM** Sunday
- You can find the exam entry named "Week 3 Quiz" on GradeScope.
- Topics: **Perceptron, Linear Models**
- Question Types
  - True/false, multiple choices, and auto-graded short answers (fill blanks)
  - Some questions may include several subquestions.
- Some light calculations are expected. Some scratch paper and one scientific calculator (physical or online) are recommended for preparation.

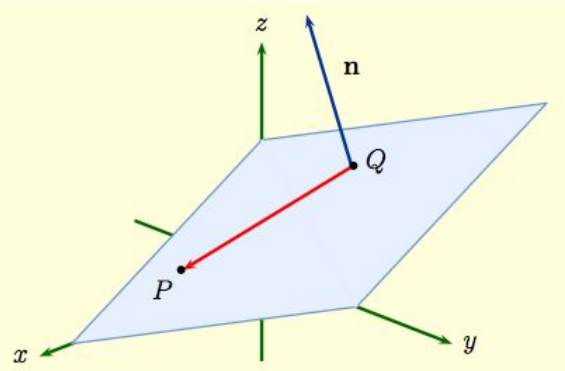
UCLA
Engineer Change.

*One more quiz of K-NN

- True/False: The training error of K-NN will be zero when K = 1, irrespective of the dataset.

Math Reminder: Normal vector and plane

- A normal vector is a vector perpendicular to another object, such as a surface or plane.



Let $Q(a,\ b,\ c)$ be a fixed point in the plane, $P(x,\ y,\ z)$ an arbitrary point in the plane, and $\mathbf{n} = \langle A,\ B,\ C \rangle$ the normal to the plane. If

$$\mathbf{b} = \langle a,\ b,\ c \rangle, \qquad \mathbf{r} = \langle x,\ y,\ z \rangle,$$

the vector

$$\overrightarrow{QP} = \mathbf{r} - \mathbf{b} = \langle x - a,\ y - b,\ z - c \rangle$$

lies in the plane, and is **perpendicular** to $\mathbf{n}$.

Thus $\mathbf{n} \cdot (\mathbf{r} - \mathbf{b}) = 0$. In terms of coordinates, this becomes

$$\langle A,\ B,\ C \rangle \cdot \langle x - a,\ y - b,\ z - c \rangle = 0,$$

where $\mathbf{n} = \langle A,\ B,\ C \rangle$. In other words, we get the **point-normal** equation

$$A(x - a) + B(y - b) + C(z - c) = 0.$$

for a plane.

Credit: https://web.ma.utexas.edu/users/m408m/Display12-5-4.shtml

Math Reminder: Normal vector and plane

- A normal vector is a vector perpendicular to another object, such as a surface or plane.

As promised, we return the the question of finding the equation for a plane from the location of three points, say

$$Q(x_1,\, y_1,\, z_1)\,, \qquad R(x_2,\, y_2,\, z_2)\,, \qquad S(x_3,\, y_3,\, z_3)$$

The fact that the cross-product $\mathbf{a} \times \mathbf{b}$ is perpendicular to both $\mathbf{a}$ and $\mathbf{b}$ makes it very useful when dealing with normals to planes.

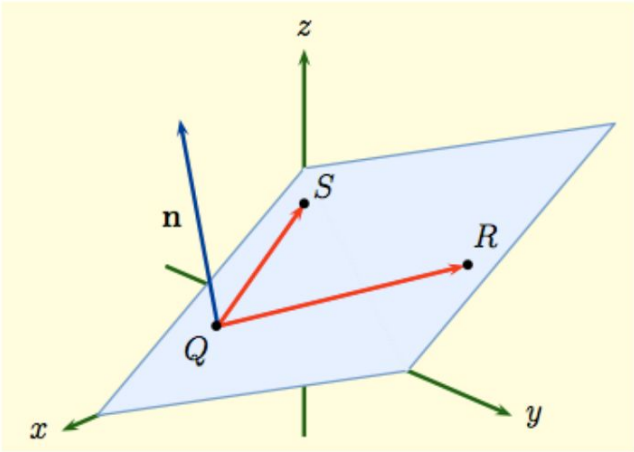
Let

$$\mathbf{b} = \langle\, x_1,\, y_1,\, z_1 \,\rangle\,, \quad \mathbf{r} = \langle\, x_2,\, y_2,\, z_2 \,\rangle\,, \quad \mathbf{s} = \langle\, x_3,\, y_3,\, z_3 \,\rangle.$$

The vectors

$$\overrightarrow{QR} \;=\; \mathbf{r} - \mathbf{b}\,, \qquad \overrightarrow{QS} \;=\; \mathbf{s} - \mathbf{b}\,,$$

then lie in the plane. The normal to the plane is given by the cross product $\mathbf{n} = (\mathbf{r} - \mathbf{b}) \times (\mathbf{s} - \mathbf{b})$. Once this normal has been calculated, we can then use the point-normal form to get the equation of the plane passing through Q, R, and S.



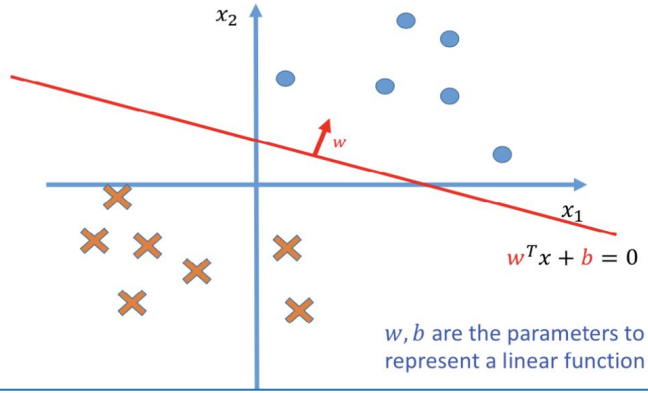
UCLA
Engineer Change.

Math Reminder: Normal vector and plane

- Demo Calculation Example

Perceptron: Overview

- Instance (feature vectors): $\boldsymbol{x} \in \mathbb{R}^{\mathrm{D}}$
- Label: $y \in \{-1, +1\}$
- Model/Hypotheses:
  $H = \{h | h : \mathbb{X} \to \{-1, +1\}, h(\boldsymbol{x}) = sign(\sum_{d=1}^{D} w_d x_d + b)\}$.
- Learning goal: $y = h(\boldsymbol{x})$
  - Learn $w_1, \ldots, w_D, b$.
  - Parameters: $w_1, \ldots, w_D, b$.
  - $\boldsymbol{w}$: weights, b: bias



Iteratively solving one case at a time

- REPEAT
- Pick a data point $\boldsymbol{x}_n$
- Compute $a = \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_n$ using the *current* $\boldsymbol{w}$
- If $ay_n > 0$, do nothing. Else,

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + y_n \boldsymbol{x}_n$$

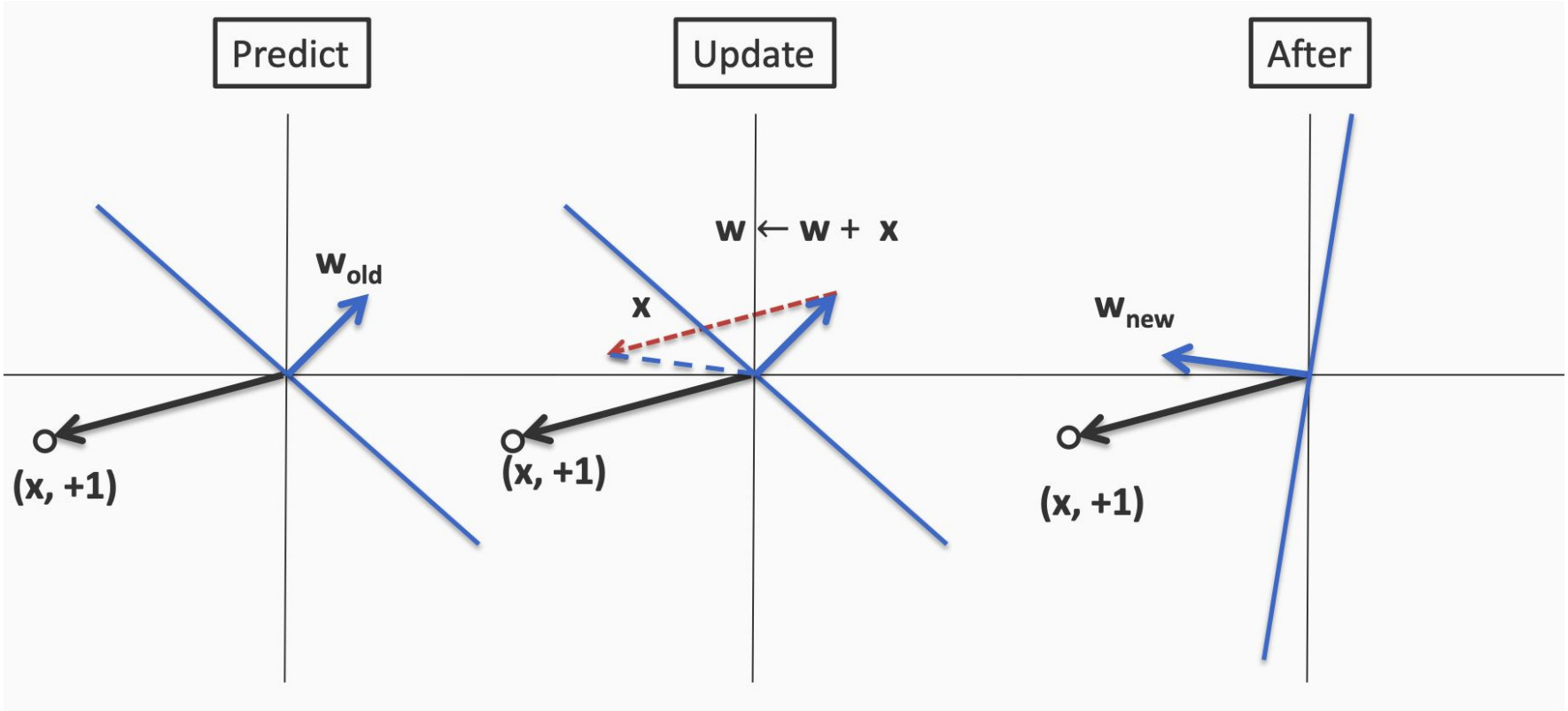
- UNTIL converged.

Perceptron: Convergence

- If training data is **not linearly separable**, the algorithm does not converge.
- If the training data is **linearly separable**, the algorithm stops in a finite number of steps (converges).

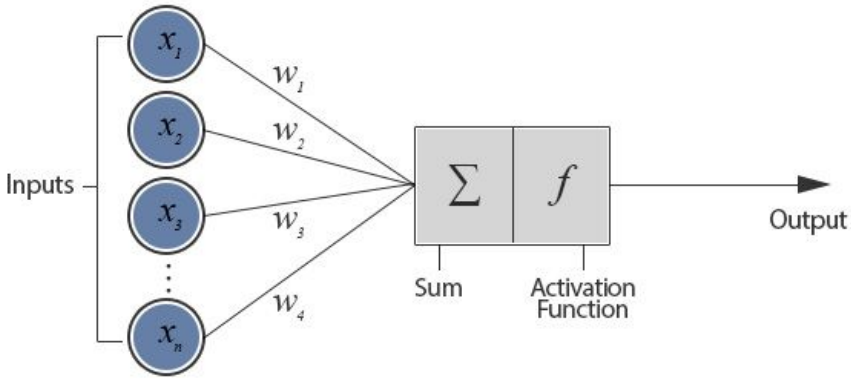
- Let $\{(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_\mathsf{N}, y_\mathsf{N})\}$ be a sequence of training examples such that $\|\boldsymbol{x}_n\|_2 \leq R$ and label $y_n \in \{-1, +1\}$.
- Suppose there exists a unit vector $\boldsymbol{u} \in \mathbb{R}^D$ such that for some $\gamma > 0$, we have $y_n \boldsymbol{u}^\mathrm{T} \boldsymbol{x}_n \geq \gamma$.
- Then the Perceptron algorithm will make at most $\frac{R^2}{\gamma^2}$ mistakes on the training sequence.

Perceptron: Update (Geometry)

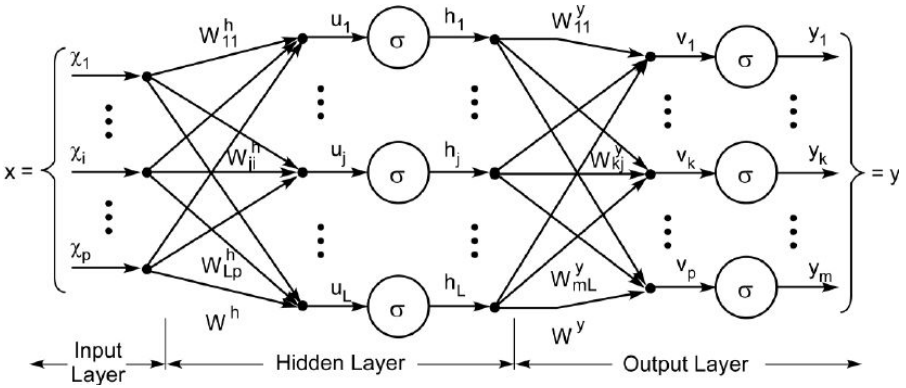


Perceptron: Connect to Neural Network

A Single Perceptron



Multi-layer Neural Network



Question: Can a single perceptron classify XOR data? How about 2-layer perceptrons?

Logistic Regression: Overview

Probability of a single training sample $(\boldsymbol{x}_n, y_n)$

$$p(y_n|\boldsymbol{x}_n; b, \boldsymbol{w}) = \begin{cases} h_{\boldsymbol{w},b}(\boldsymbol{x}_n) = \sigma(b + \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_n) & \text{if} \quad y_n = 1 \\ = 1 - h_{\boldsymbol{w},b}(\boldsymbol{x}_n) = 1 - \sigma(b + \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_n) & \text{otherwise} \end{cases}$$

Compact expression, exploring that y_n is either 1 or 0

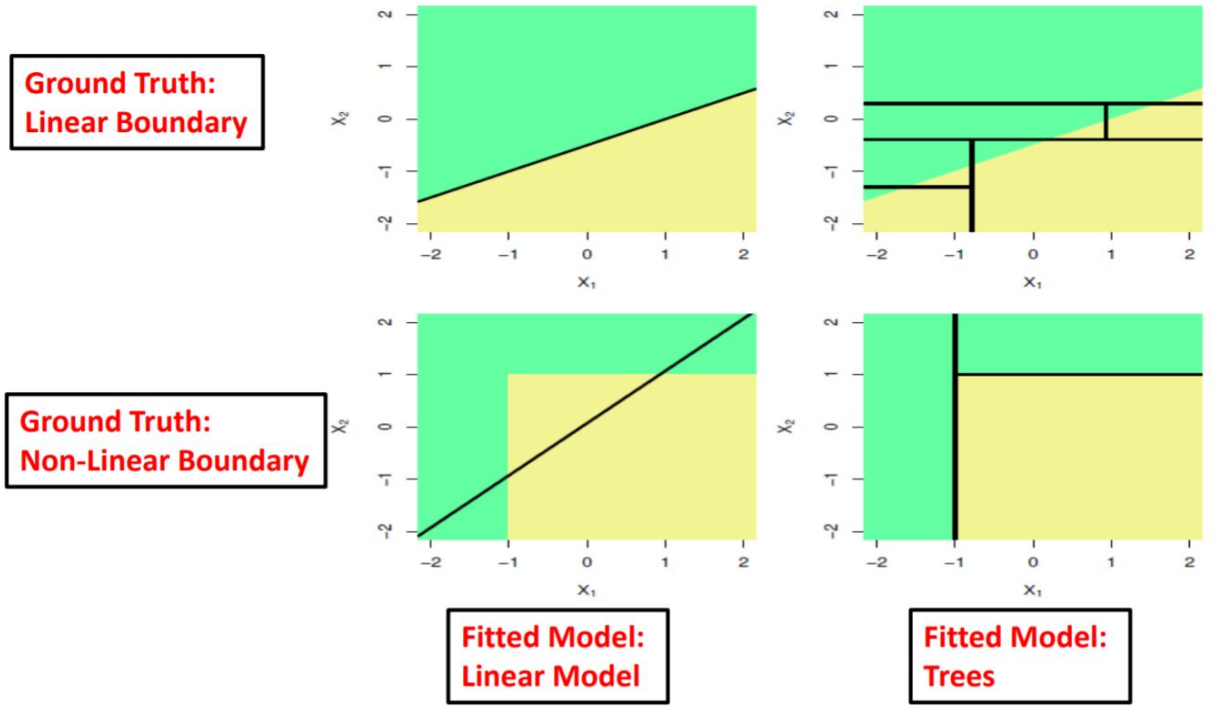
$$p(y_n|\boldsymbol{x}_n; b; \boldsymbol{w}) = h_{\boldsymbol{w},b}(\boldsymbol{x}_n)^{y_n}[1 - h_{\boldsymbol{w},b}(\boldsymbol{x}_n)]^{1-y_n}$$

Log-likelihood of the whole training data $\mathcal{D}$

$$l(\boldsymbol{w}, b) = \sum_n \{y_n \log h_{\boldsymbol{w},\boldsymbol{b}}(\boldsymbol{x}_n) + (1 - y_n) \log[1 - h_{\boldsymbol{w},b}(\boldsymbol{x}_n)]\}$$

Linear Models

- Compare: Decision Tree, Nearest Neighbors, Perceptron



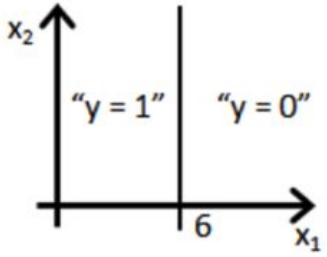
UCLA
Engineer Change.

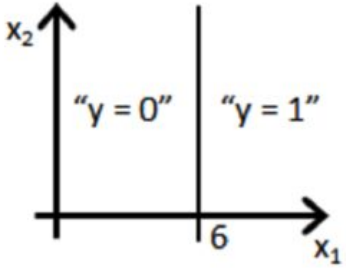
Decision Boundary: Quiz

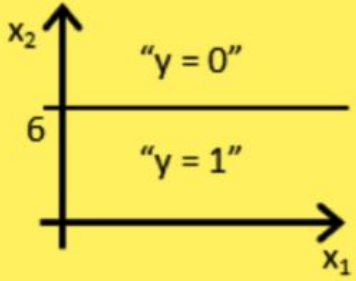
Suppose you train a logistic classifier $h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$. Suppose $\theta_0 = 6, \theta_1 = 0, \theta_2 = -1$. Which of the following figures represents the decision boundary found by your classifier?

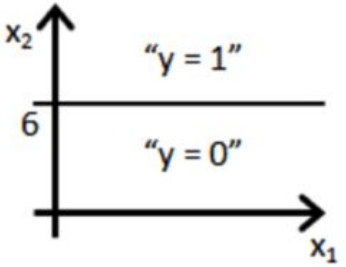
Decision Boundary: Quiz

Suppose you train a logistic classifier $h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$. Suppose $\theta_0 = 6, \theta_1 = 0, \theta_2 = -1$. Which of the following figures represents the decision boundary found by your classifier?









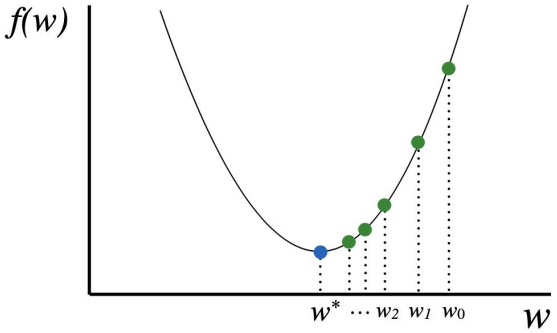
UCLA
Engineer Change.

Unconstrained Optimization

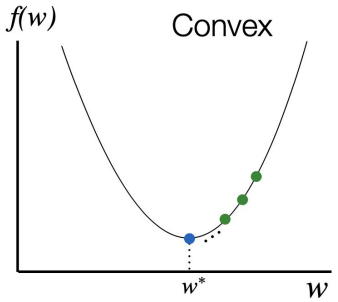
- Convex Function and Convexity
- Closed-form solution
- Gradient Descent
- Newton’s methods

Gradient Descent

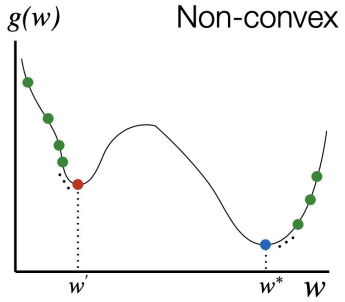
Start at a random point
Repeat
    Determine a descent direction
    Choose a step size
    Update
Until stopping criterion is satisfied



Where Will We Converge?



Any local minimum is a global minimum



Multiple local minima may exist

Newton's Method (Optional)

Definition:

$$\beta^{\text{new}} = \beta^{\text{old}} - \left(\frac{\partial^2 L(\beta)}{\partial\beta\partial\beta^T}\right)^{-1} \frac{\partial L(\beta)}{\partial\beta}$$

Apply Newton's methods on single variable to find minima:

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

From single variable to Multivariate Newton-Raphson Method

UCLA
Engineer Change.

Newton's Method: Steps (Optional)

1. Initialize $x^{(0)}$
2. Calculate $\nabla f(x)$
3. Calculate $F(x)$
4. Initialize step$n = 0$ and start loops
   a. Calculate $\nabla f(x^{(n)})$
   b. Calculate $F(x^{(n)})$
   c. Calculate $[F(x^{(n)})]^{-1}$
   d. Update: $x^{(n+1)} = x^{(n)} - [F(x^{(n)})]^{-1} \cdot \nabla f(x^{(n)})$
   e. Update: $n = n + 1$
5. Exit Loop

Newton's Method: Example (Optional)

$$x^{(0)} = [3, -1, 0]$$

$$f(x_1, x_2, x_3) = (x_1 + 10x_2)^2 + 5(x_1 - x_3)^2 + (x_2 - 2x_3)^4$$

Newton's Method Example in one step:

- ➔ Calculate $\nabla f(x^{(n)})$
- ➔ Calculate $F(x^{(n)})$
- ➔ Calculate $[F(x^{(n)})]^{-1}$
- ➔ Update: $x^{(n+1)}$
- ➔ Update: $n = n + 1$

$$\nabla f(x^{(0)}) = \left[\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \frac{\partial f}{\partial x_3}\right] = [16, -144, 22]$$

$$F(x^{(0)}) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \frac{\partial^2 f}{\partial x_1 \partial x_3} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \frac{\partial^2 f}{\partial x_2 \partial x_3} \\ \frac{\partial^2 f}{\partial x_3 \partial x_1} & \frac{\partial^2 f}{\partial x_3 \partial x_2} & \frac{\partial^2 f}{\partial x_3^2} \end{bmatrix} = \begin{bmatrix} 12 & 20 & -10 \\ 20 & 22 & -24 \\ -10 & -24 & 48 \end{bmatrix}$$

$$[F(x^{(0)})]^{-1} = \begin{bmatrix} -0.079 & 0.119 & 0.043 \\ 0.119 & -0.079 & -0.015 \\ 0.043 & -0.015 & 0.023 \end{bmatrix}$$

$$x^{(1)} = x^{(0)} - [F(x^{(0)})]^{-1} \cdot \nabla f(x^{(0)})$$

$$n = 1$$

Maximum Likelihood Estimation

Definition: The maximum likelihood estimator (MLE) $\hat{\theta}$, is the value of θ that maximizes $L(\theta)$.

The log-likelihood function is defined by $l(\theta) = \log L(\theta)$. Its maximum occurs at the same place as that of the likelihood function.

- Using logs simplifies mathemetical expressions (converts exponents to products and products to sums)
- Using logs helps with numerical stabilitity

The same is true of the likelihood function times any constant. Thus we shall often drop constants in the likelihood function.

UCLA
Engineer Change.

MLE: Logistic Regression

- Model

$$y = \sigma(X) = \frac{1}{1 + e^{-X^T \beta}}$$

- Original Objective

$$J(\beta) = -\frac{1}{n} \sum_i \left(y_i x_i^T \beta - \log\left(1 + \exp\{x_i^T \beta\}\right)\right)$$

MLE: Logistic Regression

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}},$$

where

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

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

MLE: Logistic Regression

$$
\begin{aligned}
P(y = 1 \mid x; \theta) &= h_\theta(x) \\
P(y = 0 \mid x; \theta) &= 1 - h_\theta(x)
\end{aligned}
$$

$$p(y \mid x; \theta) = (h_\theta(x))^y \, (1 - h_\theta(x))^{1-y}$$

Assuming that the m training examples were generated independently, we can then write down the likelihood of the parameters as

$$
\begin{aligned}
L(\theta) &= p(\vec{y} \mid X; \theta) \\
&= \prod_{i=1}^{m} p(y^{(i)} \mid x^{(i)}; \theta) \\
&= \prod_{i=1}^{m} \left(h_\theta(x^{(i)})\right)^{y^{(i)}} \left(1 - h_\theta(x^{(i)})\right)^{1-y^{(i)}}
\end{aligned}
$$

As before, it will be easier to maximize the log likelihood:

$$
\begin{aligned}
\ell(\theta) &= \log L(\theta) \\
&= \sum_{i=1}^{m} y^{(i)} \log h(x^{(i)}) + (1 - y^{(i)}) \log(1 - h(x^{(i)}))
\end{aligned}
$$

Constrained Optimization

- Lagrange Multiplier

$$\mathcal{L}(x, y, \lambda) = f(x, y) - \lambda g(x, y)$$

maximize $f(x, y)$
subject to: $g(x, y) = 0$

$$\nabla_{x,y,\lambda} \mathcal{L}(x, y, \lambda) = 0 \iff \begin{cases} \nabla_{x,y} f(x, y) = \lambda \nabla_{x,y} g(x, y) \\ g(x, y) = 0 \end{cases}$$

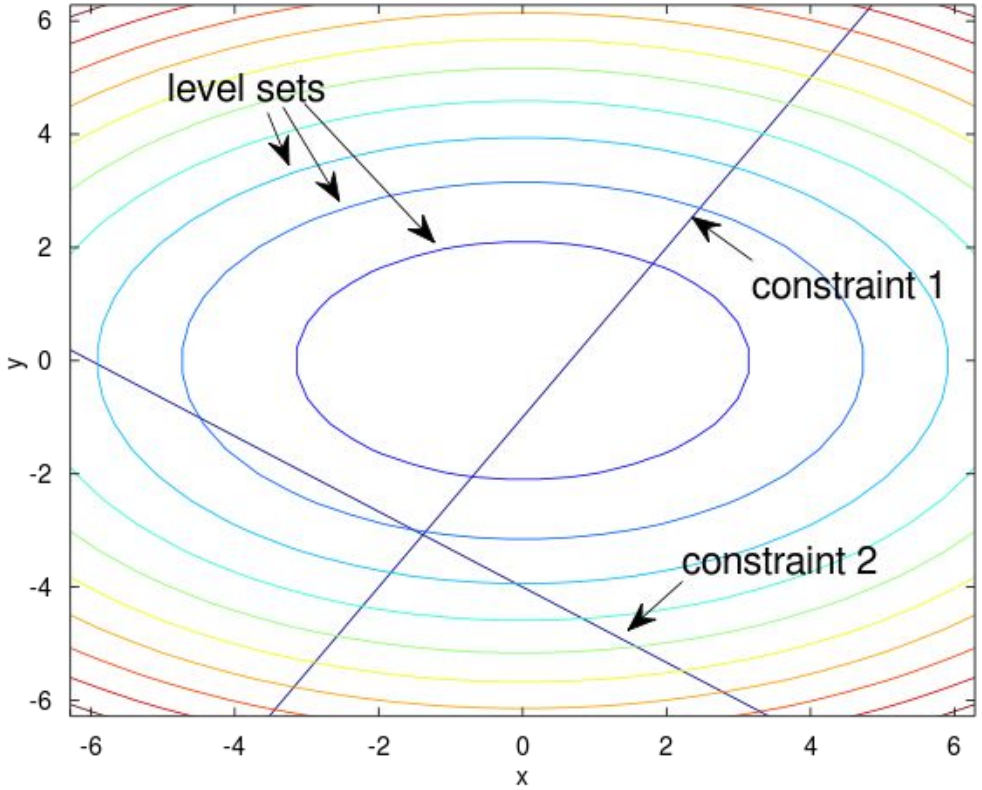
$$\nabla f(\mathbf{x}) = \sum_{k=1}^{M} \lambda_k \nabla g_k(\mathbf{x}) \iff \nabla f(\mathbf{x}) - \sum_{k=1}^{M} \lambda_k \nabla g_k(\mathbf{x}) = 0.$$

- Considering multiple constraints

$$\mathcal{L}\left(x_1, \ldots, x_n, \lambda_1, \ldots, \lambda_M\right) = f\left(x_1, \ldots, x_n\right) - \sum_{k=1}^{M} \lambda_k g_k\left(x_1, \ldots, x_n\right)$$

$$\nabla_{x_1,\ldots,x_n,\lambda_1,\ldots,\lambda_M} \mathcal{L}(x_1, \ldots, x_n, \lambda_1, \ldots, \lambda_M) = 0 \iff \begin{cases} \nabla f(\mathbf{x}) - \sum_{k=1}^{M} \lambda_k \nabla g_k(\mathbf{x}) = 0 \\ g_1(\mathbf{x}) = \cdots = g_M(\mathbf{x}) = 0 \end{cases}$$

Constrained Optimization

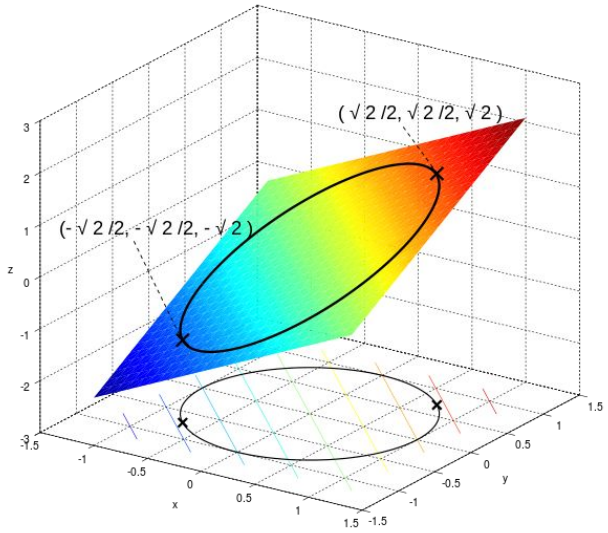


Lagrange Multiplier

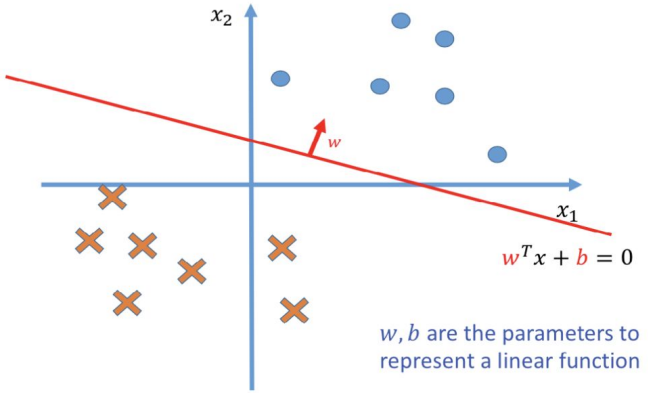
- Example:

$$f(x, y) = x + y$$

$$\text{Constraint} : g(x, y) = x^2 + y^2 = 1$$



Lagrange Multiplier: Connect to SVM *



Original optimization problem:

$$\text{minimize}_{\mathbf{w},b} \quad \frac{1}{2}\mathbf{w}.\mathbf{w}$$
$$\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j \geq 1, \ \forall j$$

Rewrite constraints

One Lagrange multiplier per example

Lagrangian:

$$L(\mathbf{w}, \alpha) = \frac{1}{2}\mathbf{w}.\mathbf{w} - \sum_j \alpha_j \left[\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j - 1\right]$$
$$\alpha_j \geq 0, \ \forall j$$

UCLA
Engineer Change.

What's next?

- In next week's discussion, we will discuss:
  - Logistic Regression (Continued)
  - Naive Bayes, Linear Regression (Planned)

UCLA Samueli Computer Science

Thank you!

Q & A