Linear Models and Perceptrons (with more gradient optimization)

CMSC 678

UMBC

Some slides adapted from Hamed Pirsiavash

Outline

Recap: Decision Theory and ERM

Gradient Optimization

Linear Models & Surrogate Loss

- Example 1: Linear Regression
- Example 2: Linear Classification Model
- Example 3: Perceptrons

Recap from last time…

Probability Prerequisites

Basic probability axioms and definitions

Probability chain rule

Joint probability

Common distributions

Probabilistic Independence

Expected Value (of a function) of a Random Variable

$$\mathbb{E}[f(X)] = \sum_x f(x)\, p(x)$$

Marginal probability

Conditional probabilities $p(\cdot \mid Y)$ are still probabilities

Definition of conditional probability

Bayes rule

$$\underbrace{p(X \mid Y)}_{\textit{posterior}} = \frac{\overbrace{p(Y \mid X)}^{\textit{likelihood}} * \overbrace{p(X)}^{\textit{prior}}}{\underbrace{p(Y)}_{\textit{marginal}}}$$

Decision Theory → Empirical Risk Minimization

"Decision theory is trivial, apart from the computational details" – MacKay, ITILA, Ch 36

Input: **x** ("state of the world")

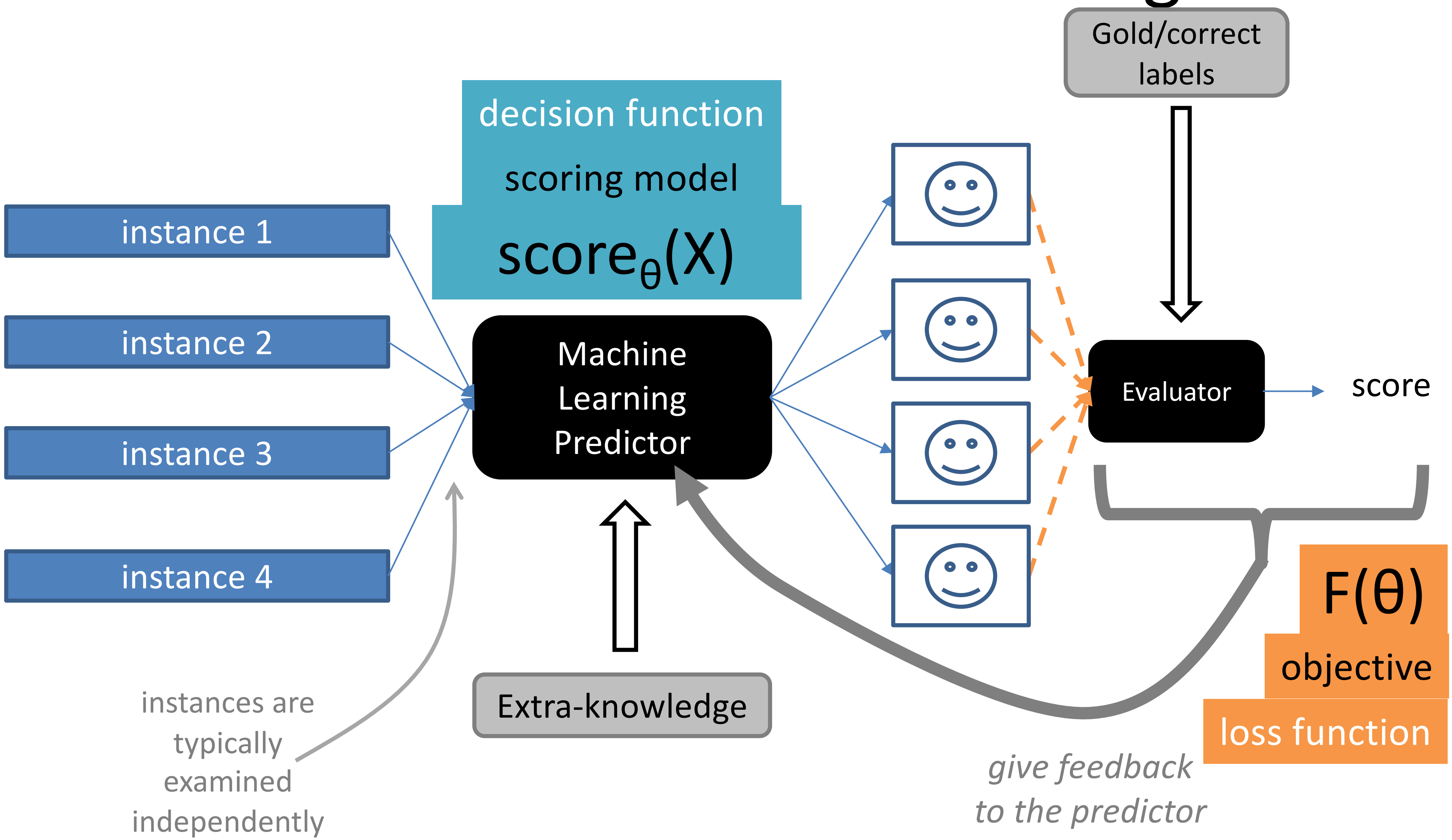
Output: a decision ŷ

Requirement 1: a decision (hypothesis) function h(**x**) to produce ŷ

Requirement 2: a loss function ℓ(y, ŷ) telling us how wrong we are

Goal: minimize expected loss across any possible input → minimize expected loss across our observed input

Machine Learning Framework: Decision Theoretic Learning



Empirical Risk Minimization

minimize expected loss across our observed input

$$\underset{h}{\operatorname{argmin}} \sum_{i=1}^{N} \ell\left(y_i, h(\boldsymbol{x}_i)\right)$$

our classifier/predictor

controlled by our parameters θ

change θ →
change the behavior of the classifier

Best (Practical) Case:
Optimize Empirical Risk with Gradients

$$\underbrace{\operatorname*{argmin}_{h} \sum_{i=1}^{N} \ell\big(y_i, h_\theta(\boldsymbol{x}_i)\big)}_{F}$$

$$\Downarrow$$

$$\nabla_\theta F = \sum_i \frac{\partial \ell\big(y_i, \hat{y} = h_\theta(\boldsymbol{x}_i)\big)}{\partial \hat{y}} \nabla_\theta h_\theta(\boldsymbol{x_i})$$

differentiating might not always work: "… apart from the computational details"

Regression Loss Function Examples

squared loss

$$\ell(y, \hat{y}) = (\mathrm{y} - \hat{y})^2$$

absolute loss

$$\ell(y, \hat{y}) = |y - \hat{y}|$$

Classification Loss Function: 0-1 Loss

$$\ell(y, \hat{y}) = \begin{cases} 0, & \text{if } y = \hat{y} \\ 1, & \text{if } y \neq \hat{y} \end{cases}$$

Problem: not differentiable wrt $\hat{y}$ (or θ)

Solution 1: is h(x) a conditional distribution p(y | x)? Use MAP

Solution 2: use a surrogate loss that approximates 0-1

Solution 3: is the data linearly separable? Perceptron can work

Outline

Recap: Decision Theory and ERM

Gradient Optimization

Linear Models & Surrogate Loss

- Example 1: Linear Regression
- Example 2: Linear Classification Model
- Example 3: Perceptrons

Learning the Model's Parameters

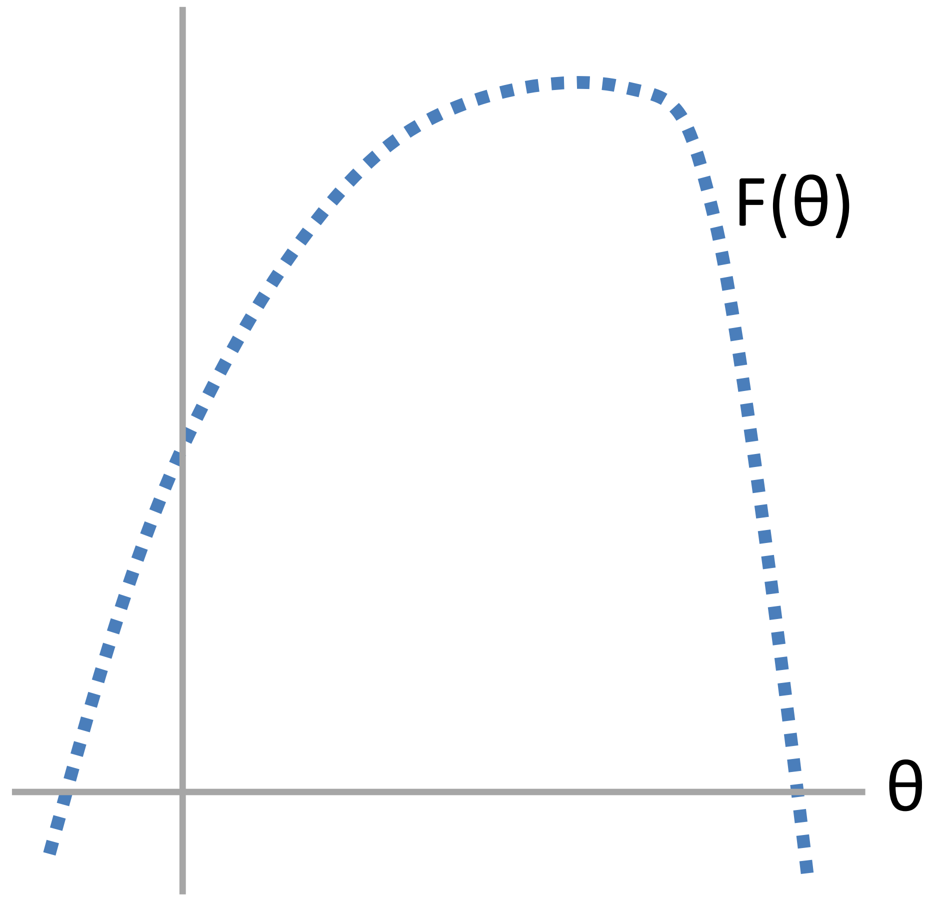
$$\text{score}_\theta(x)$$

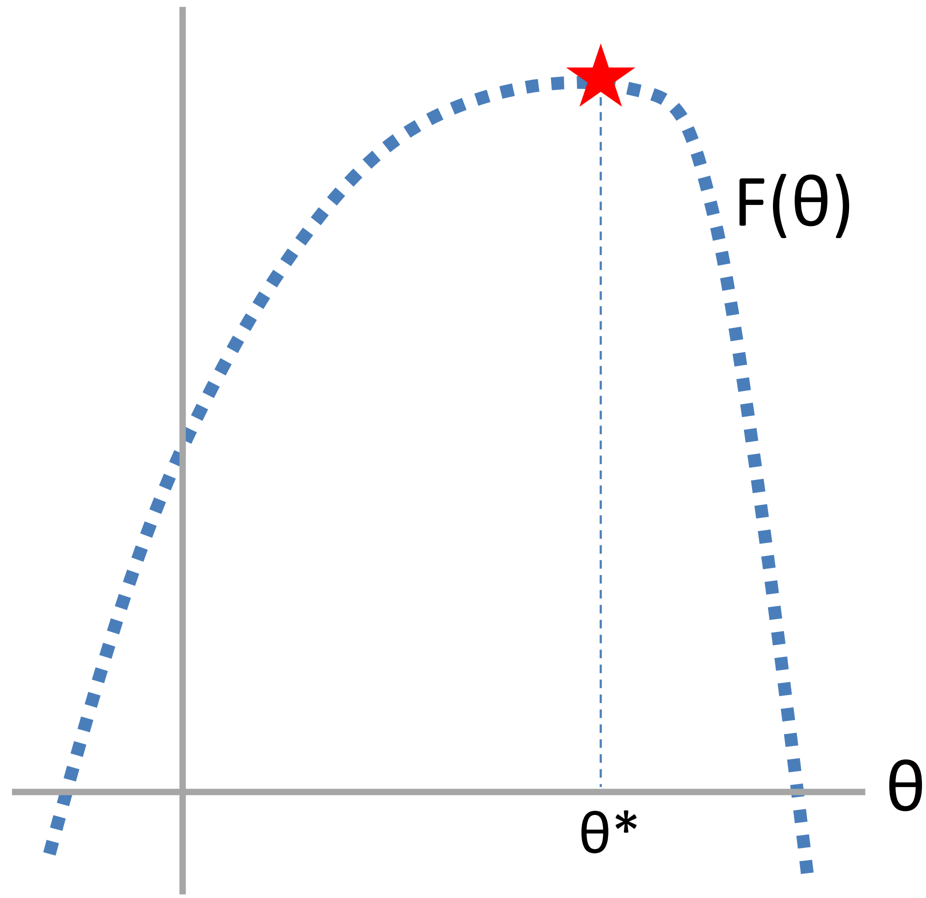
model (probabilistic, neural, other)

$$\Downarrow$$

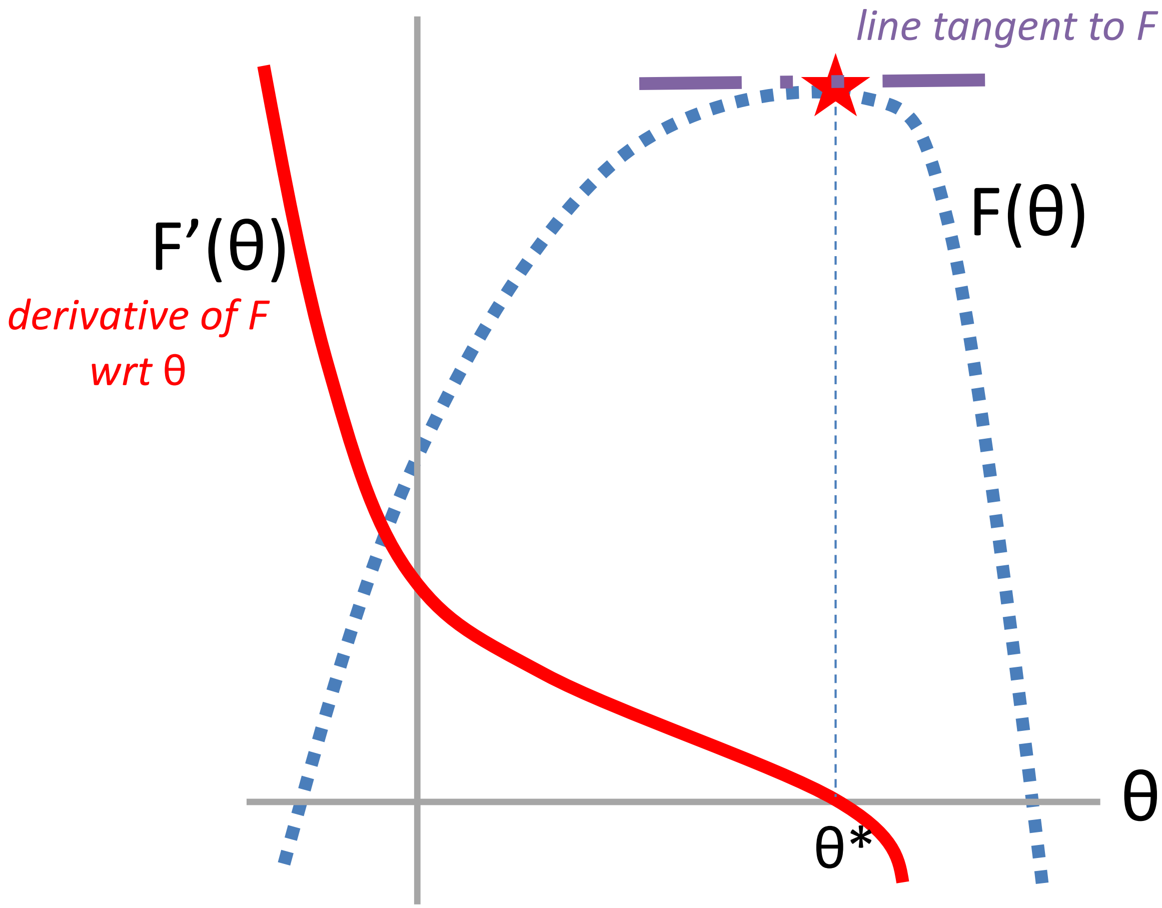
$$F(\theta; \{\boldsymbol{x}\})$$

objective
(given observations)


F(θ)
θ



F(θ)
θ
θ*


line tangent to F
F(θ)
F’(θ)
derivative of F
wrt θ
θ
θ*

Example

$$F(x) = -(x-2)^2$$

differentiate

$$F'(x) = -2x + 4$$

Solve $F'(x) = 0$

$$x = 2$$

Common Derivative Rules

$$\frac{d\exp x}{dx} = \exp x$$

$$\frac{df(x)g(x)}{dx} = \frac{df(x)}{dx}g(x) + \frac{dg(x)}{dx}f(x)$$

$$\frac{d\log x}{dx} = \frac{1}{x}$$

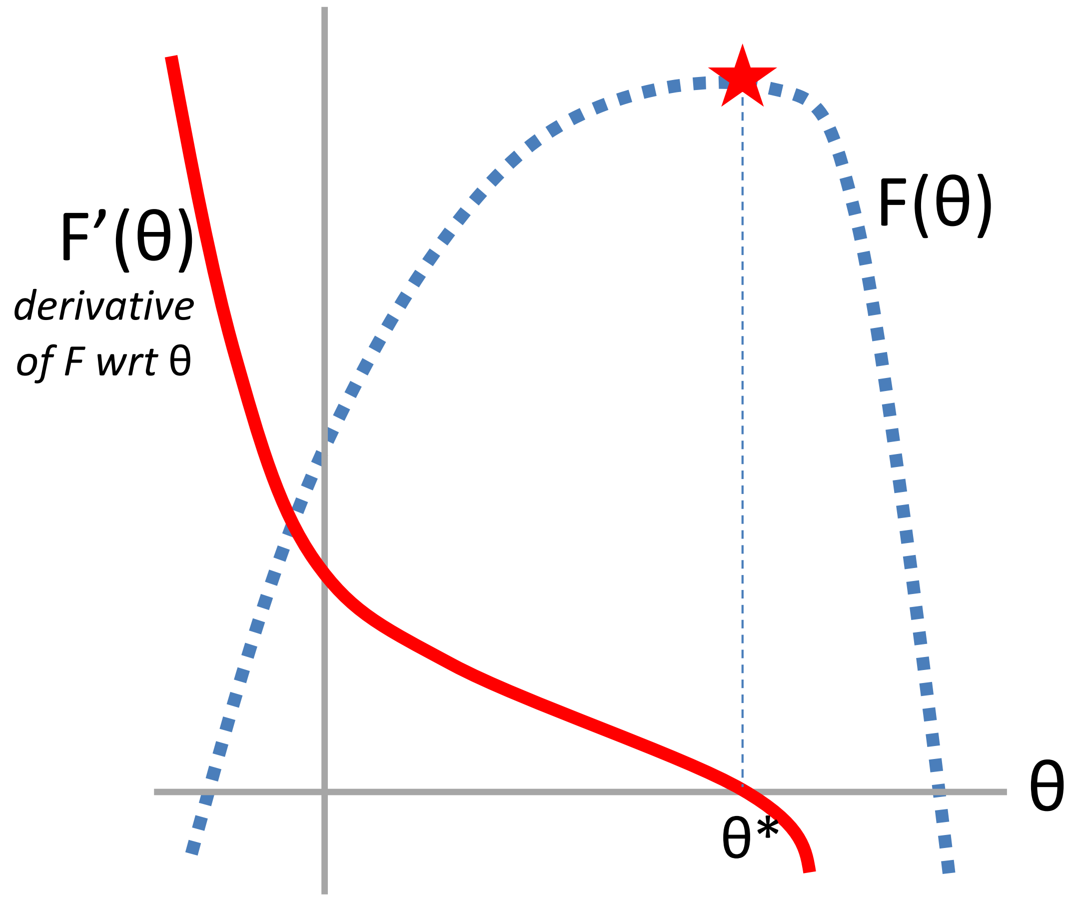
$$\frac{df(g(x))}{dx} = \frac{df(g(x))}{dg(x)}\frac{dg(x)}{dx}$$

Optimizing F(θ) through Calculus

1. Directly solving for the gradient (derivative)

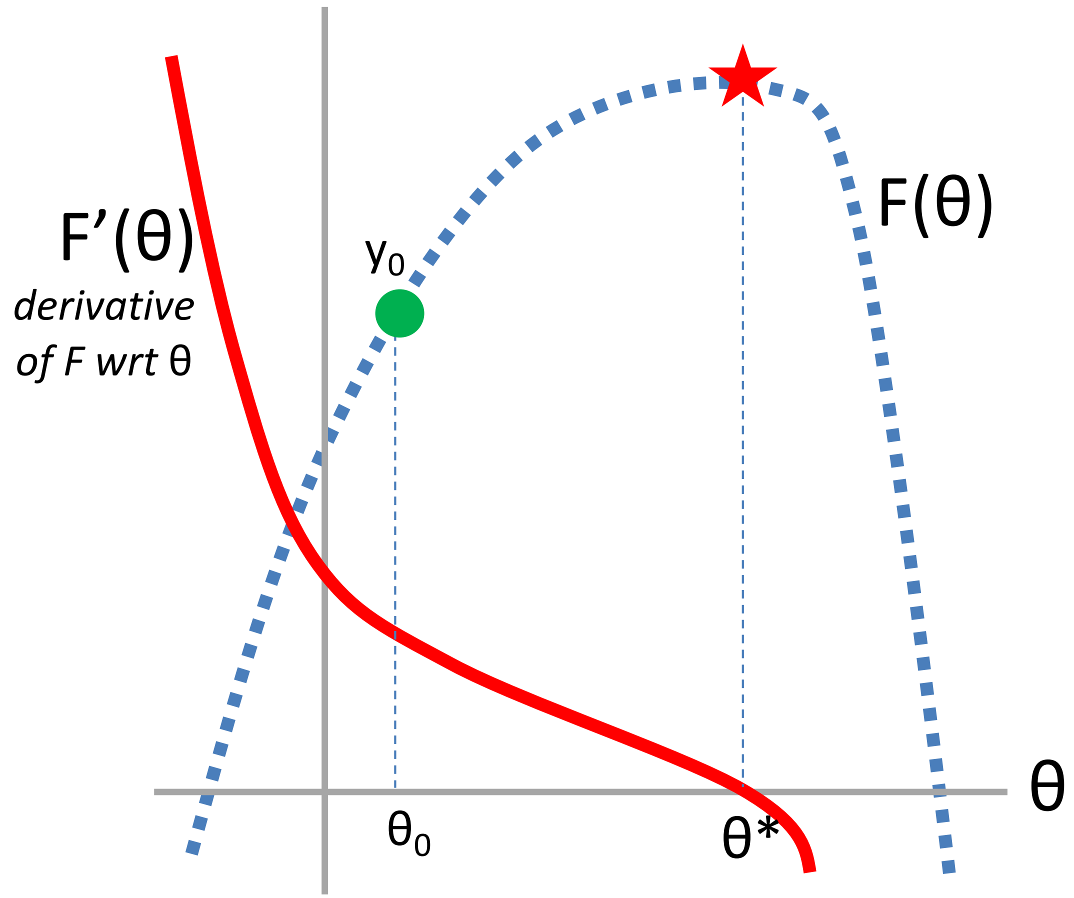
2. Gradient descent/ascent: incremental hill-climbing

What if you can't find the roots? Follow the derivative



What if you can't find the roots? Follow the derivative

Set $t = 0$
Pick a starting value θ_t
Until converged:
1. Get value $y_t = F(\theta_t)$



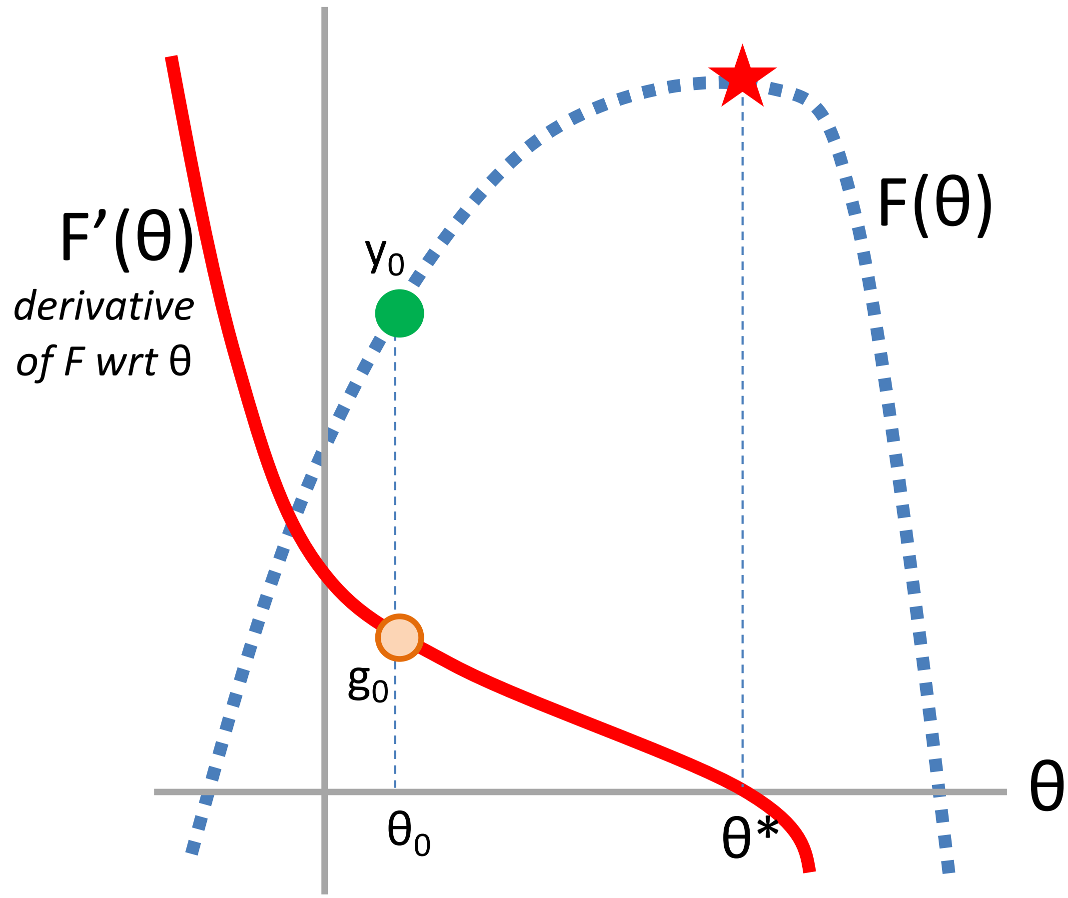
What if you can't find the roots? Follow the derivative

Set $t = 0$

Pick a starting value θ_t

Until converged:

1. Get value $y_t = F(\theta_t)$
2. Get derivative $g_t = F'(\theta_t)$



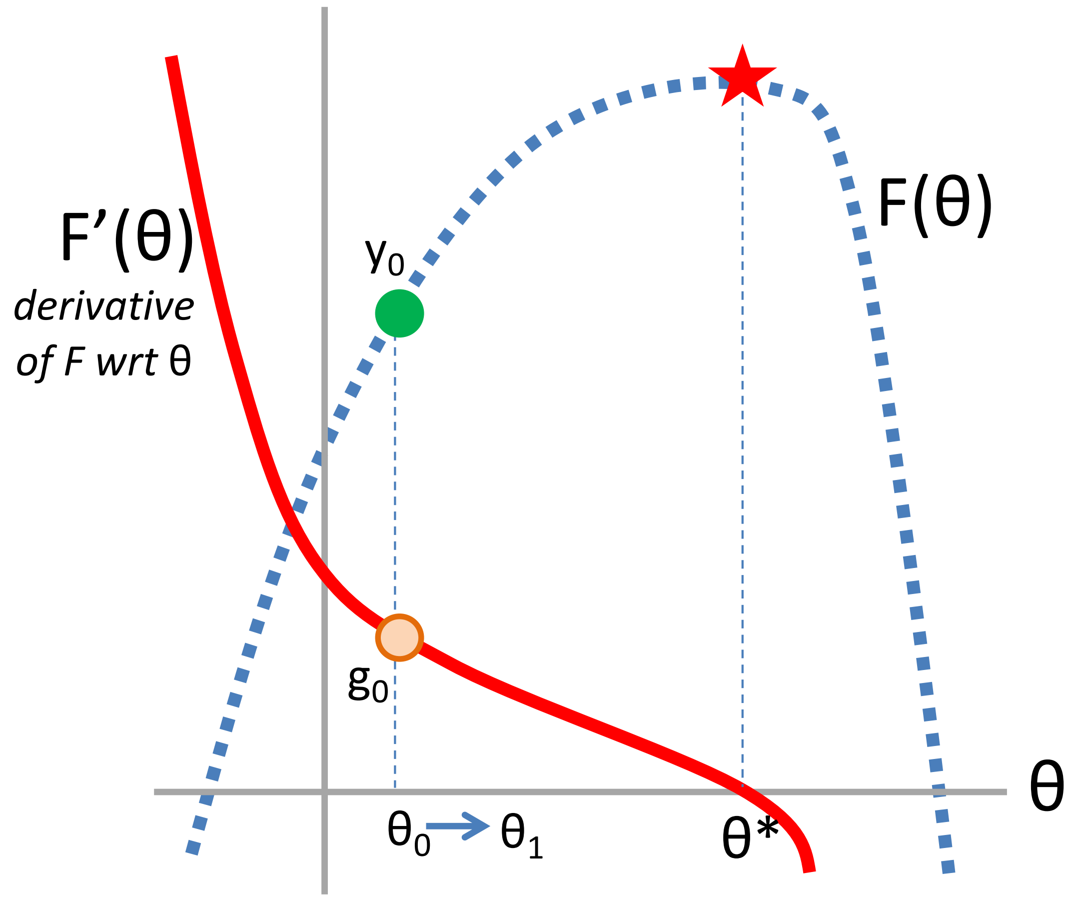
What if you can't find the roots? Follow the derivative

Set $t = 0$

Pick a starting value θ_t

Until converged:

1. Get value $y_t = F(\theta_t)$
2. Get derivative $g_t = F'(\theta_t)$
3. Get scaling factor ρ_t
4. Set $\theta_{t+1} = \theta_t + \rho_t * g_t$
5. Set t += 1



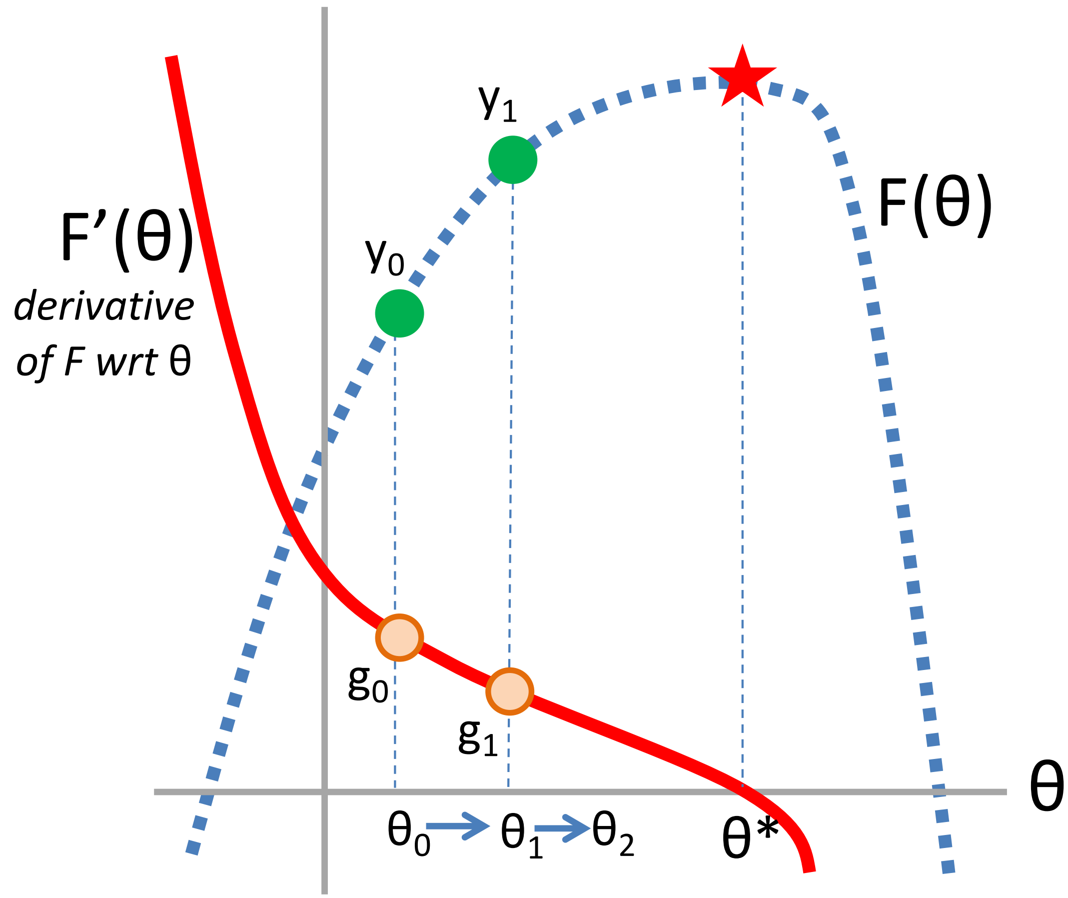
What if you can't find the roots? Follow the derivative

Set $t = 0$

Pick a starting value θ_t

Until converged:

1. Get value $y_t = F(\theta_t)$
2. Get derivative $g_t = F'(\theta_t)$
3. Get scaling factor ρ_t
4. Set $\theta_{t+1} = \theta_t + \rho_t * g_t$
5. Set t += 1



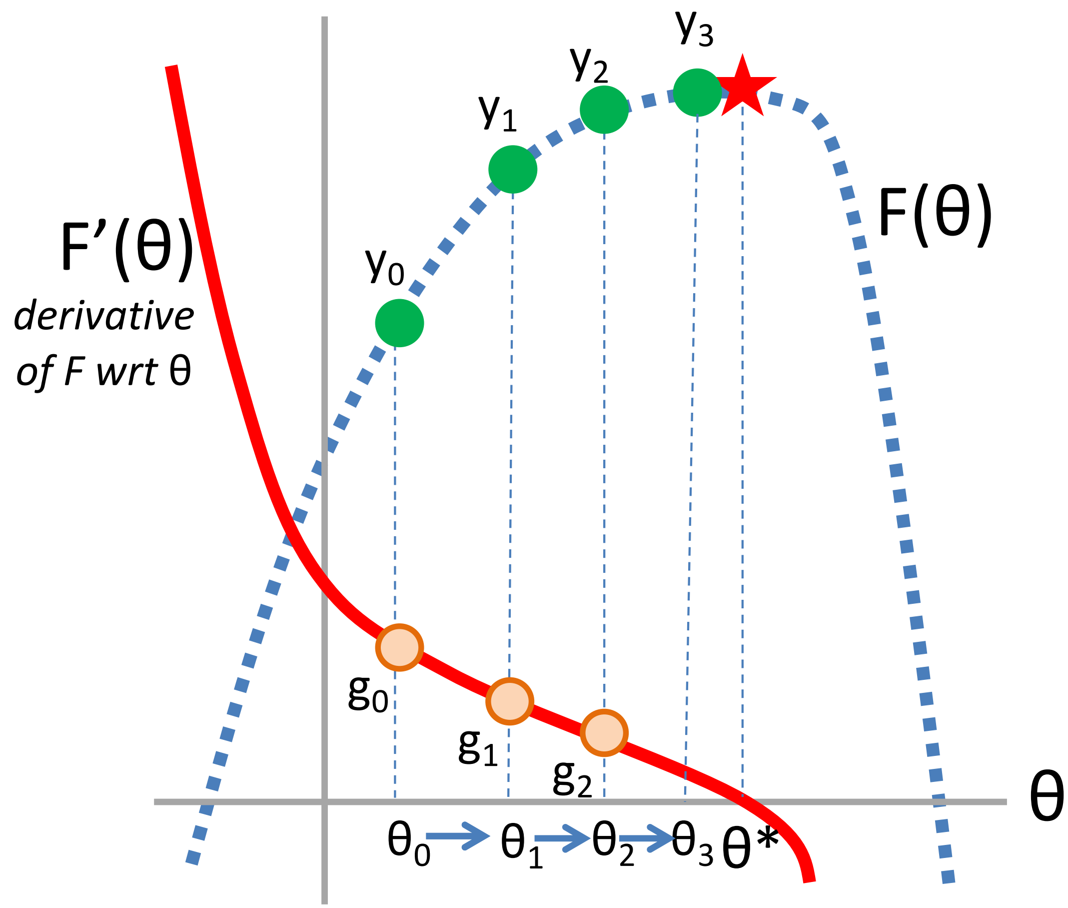
What if you can't find the roots? Follow the derivative

Set $t = 0$

Pick a starting value θ_t

Until converged:

1. Get value $y_t = F(\theta_t)$
2. Get derivative $g_t = F'(\theta_t)$
3. Get scaling factor ρ_t
4. Set $\theta_{t+1} = \theta_t + \rho_t * g_t$
5. Set t += 1



Example

Set t = 0

Pick a starting value θ_t

Until converged:

1. Get value $y_t = F(\theta_t)$
2. Get derivative $g_t = F'(\theta_t)$
3. Get scaling factor ρ_t
4. Set $\theta_{t+1} = \theta_t + \rho_t * g_t$
5. Set t += 1

Model parameter(s)

$$F(\theta) = -(c - h_\theta(X))^2$$

$$\frac{\partial F}{\partial \theta} = ???$$

Data

Remember:

$$\text{score}_\theta(\text{x})$$

$$\Downarrow$$

$$F(\theta; \{x\})$$

Example

Set $t = 0$
Pick a starting value θ_t
Until converged:
1. Get value $y_t = F(\theta_t)$
2. Get derivative $g_t = F'(\theta_t)$
3. Get scaling factor ρ_t
4. Set $\theta_{t+1} = \theta_t + \rho_t * g_t$
5. Set $t += 1$

$$F(\theta) = -\left(c - h_\theta(X)\right)^2$$

$$\frac{\partial F}{\partial \theta} = 2(c - h_\theta(X))\frac{\partial h_\theta}{\partial \theta}$$

Example

Set t = 0
Pick a starting value θ_t
Until converged:

1. Get value $y_t = F(\theta_t)$
2. Get derivative $g_t = F'(\theta_t)$
3. Get scaling factor ρ_t
4. Set $\theta_{t+1} = \theta_t + \rho_t * g_t$
5. Set t += 1

$$F(\theta) = -\left(c - h_\theta(X)\right)^2$$

$$\frac{\partial F}{\partial \theta} = 2\left(c - h_\theta(X)\right)\frac{\partial h_\theta}{\partial \theta}$$

$$h_\theta(X) = X * \theta$$

Example

$$F(\theta) = -\left(c - h_\theta(X)\right)^2$$

$$\frac{\partial F}{\partial \theta} = 2\left(c - h_\theta(X)\right)\frac{\partial h_\theta}{\partial \theta}$$

$$h_\theta(X) = X * \theta$$

$X = 0.2\ c = 1, \rho_t = 2^{-(t+1)}$

Learning rate

Set t = 0
Pick a starting value θ_t
Until converged:

1. Get value $y_t = F(\theta_t)$
2. Get derivative $g_t = F'(\theta_t)$
3. Get scaling factor ρ_t
4. Set $\theta_{t+1} = \theta_t + \rho_t * g_t$
5. Set t += 1

Example

$$F(\theta) = -\big(c - h_\theta(X)\big)^2$$

$$\frac{\partial F}{\partial \theta} = 2\big(c - h_\theta(X)\big)\frac{\partial h_\theta}{\partial \theta}$$

$$h_\theta(X) = X * \theta$$

$$X = 0.2\ c = 1, \rho_t = 2^{-(t+1)}$$

Set t = 0
Pick a starting value θ_t
Until converged:
1. Get value $y_t = F(\theta_t)$
2. Get derivative $g_t = F'(\theta_t)$
3. Get scaling factor ρ_t
4. Set $\theta_{t+1} = \theta_t + \rho_t * g_t$
5. Set t += 1

t=0
$\theta_0 = 2$
$F(\theta_0) = -(1 - 0.2 * 2)^2 = -0.36$

Example

$$F(\theta) = -\left(c - h_\theta(X)\right)^2$$

$$\frac{\partial F}{\partial \theta} = 2\left(c - h_\theta(X)\right)\frac{\partial h_\theta}{\partial \theta}$$

$$h_\theta(X) = X * \theta$$

$$X = 0.2\ c = 1, \rho_t = 2^{-(t+1)}$$

Set t = 0
Pick a starting value θ_t
Until converged:
1. Get value $y_t = F(\theta_t)$
2. Get derivative $g_t = F'(\theta_t)$
3. Get scaling factor ρ_t
4. Set $\theta_{t+1} = \theta_t + \rho_t * g_t$
5. Set t += 1

t=0
$\theta_0 = 2$
$F(\theta_0) = -(1 - 0.2 * 2)^2 = -0.36$
$\frac{\partial F}{\partial \theta} = 2(1 - 0.2 * 2) * 0.2 = 0.24$

Example

$$F(\theta) = -\left(c - h_\theta(X)\right)^2$$

$$\frac{\partial F}{\partial \theta} = 2\left(c - h_\theta(X)\right)\frac{\partial h_\theta}{\partial \theta}$$

$$h_\theta(X) = X * \theta$$

$$X = 0.2\ c = 1, \rho_t = 2^{-(t+1)}$$

Set t = 0
Pick a starting value θ_t
Until converged:

1. Get value $y_t = F(\theta_t)$
2. Get derivative $g_t = F'(\theta_t)$
3. Get scaling factor ρ_t
4. Set $\theta_{t+1} = \theta_t + \rho_t * g_t$
5. Set t += 1

t=0
$\theta_0 = 2$
$F(\theta_0) = -(1 - 0.2 * 2)^2 = -0.36$
$\frac{\partial F}{\partial \theta} = 2(1 - 0.2 * 2) * 0.2 = 0.24$
$\rho_0 = 2^{-(0+1)} = 0.5$
$\theta_1 = \theta_0 + \rho_0 * g_0 = 2 + 0.5 * 0.24 = 2.12$

Example

$$F(\theta) = -\left(c - h_\theta(X)\right)^2$$

$$\frac{\partial F}{\partial \theta} = 2\left(c - h_\theta(X)\right)\frac{\partial h_\theta}{\partial \theta}$$

$$h_\theta(X) = X * \theta$$

$$X = 0.2\ c = 1, \rho_t = 2^{-(t+1)}$$

Set t = 0
Pick a starting value θ_t
Until converged:

1. Get value $y_t = F(\theta_t)$
2. Get derivative $g_t = F'(\theta_t)$
3. Get scaling factor ρ_t
4. Set $\theta_{t+1} = \theta_t + \rho_t * g_t$
5. Set t += 1

t=0	t=1
$\theta_0 = 2$	$\theta_1 = 2.12$
$F(\theta_0) = -(1 - 0.2 * 2)^2 = -0.36$	$F(\theta_1) = -(1 - 0.2 * 2.12)^2 = -0.33$
$\frac{\partial F}{\partial \theta} = 2(1 - 0.2 * 2) * 0.2 = 0.24$	
$\rho_0 = 2^{-(0+1)} = 0.5$	
$\theta_1 = \theta_0 + \rho_0 * g_0 = 2 + 0.5 * 0.24 = 2.12$	

Example

$$F(\theta) = -\left(c - h_\theta(X)\right)^2$$

$$\frac{\partial F}{\partial \theta} = 2\left(c - h_\theta(X)\right)\frac{\partial h_\theta}{\partial \theta}$$

$$h_\theta(X) = X * \theta$$

$$X = 0.2\ c = 1, \rho_t = 2^{-(t+1)}$$

Set t = 0
Pick a starting value θ_t
Until converged:

1. Get value $y_t = F(\theta_t)$
2. Get derivative $g_t = F'(\theta_t)$
3. Get scaling factor ρ_t
4. Set $\theta_{t+1} = \theta_t + \rho_t * g_t$
5. Set t += 1

t=0	t=1
$\theta_0 = 2$	$\theta_1 = 2.12$
$F(\theta_0) = -(1 - 0.2 * 2)^2 = -0.36$	$F(\theta_1) = -(1 - 0.2 * 2.12)^2$ $= -0.33$
$\frac{\partial F}{\partial \theta} = 2(1 - 0.2 * 2) * 0.2 = 0.24$	$\frac{\partial F}{\partial \theta} = 2(1 - 0.2 * 2.12) * 0.2 = 0.2304$
$\rho_0 = 2^{-(0+1)} = 0.5$	$\rho_1 = 2^{-(1+1)} = 0.25$
$\theta_1 = \theta_0 + \rho_0 * g_0 = 2 + 0.5 * 0.24$ $= 2.12$	$\theta_2 = \theta_1 + \rho_1 * g_1$ $= 2.12 + 0.25 * 0.2304 = 2.1776$

Example

$$F(\theta) = -\left(c - h_\theta(X)\right)^2$$

$$\frac{\partial F}{\partial \theta} = 2\left(c - h_\theta(X)\right)\frac{\partial h_\theta}{\partial \theta}$$

$$h_\theta(X) = X * \theta$$

$$X = 0.2\ c = 1, \rho_t = 2^{-(t+1)}$$

Set t = 0
Pick a starting value θ_t
Until converged:

1. Get value $y_t = F(\theta_t)$
2. Get derivative $g_t = F'(\theta_t)$
3. Get scaling factor ρ_t
4. Set $\theta_{t+1} = \theta_t + \rho_t * g_t$
5. Set t += 1

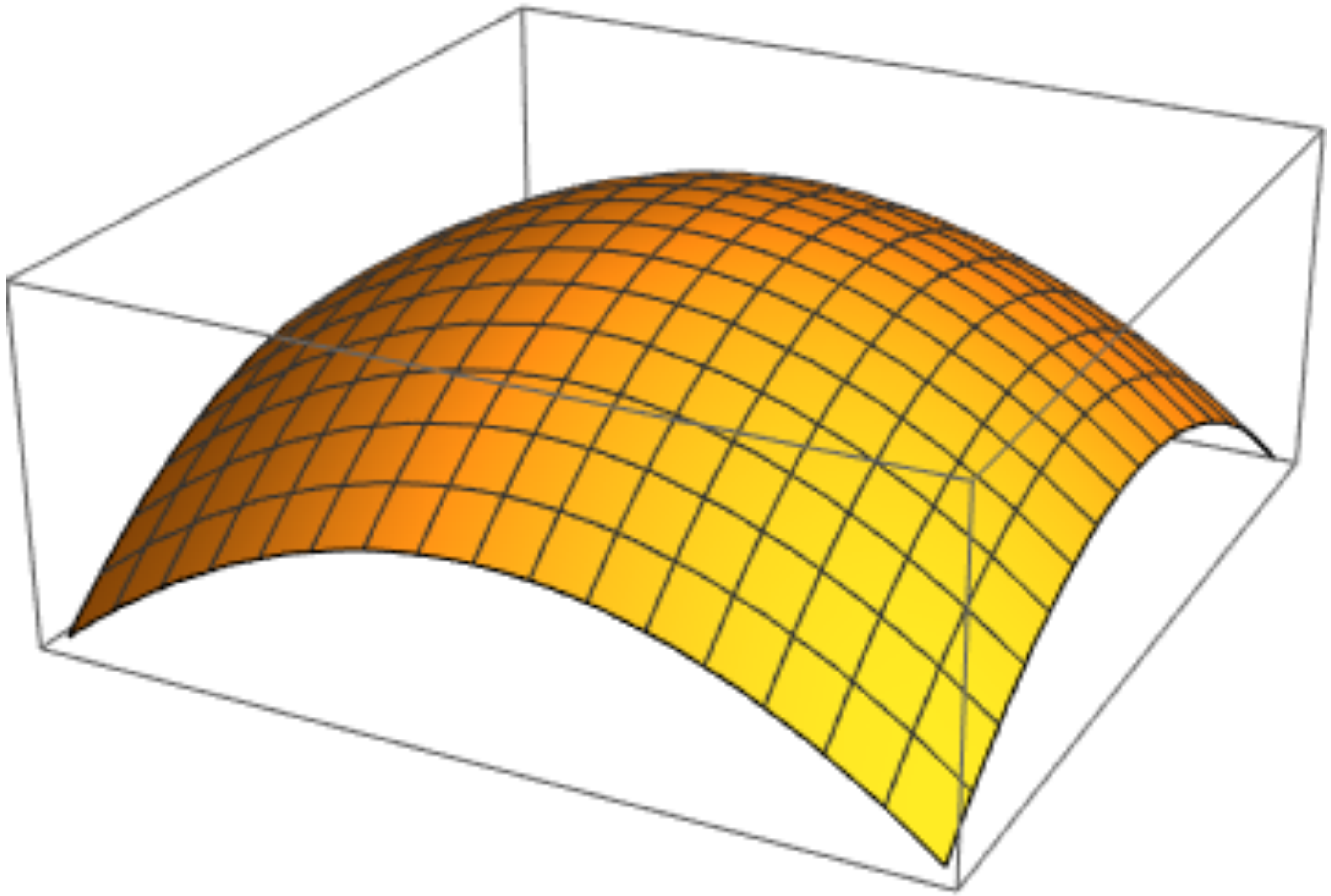
t=0	t=1	t=2
$\theta_0 = 2$	$\theta_1 = 2.12$	$\theta_2 = 2.1776$
$F(\theta_0) = -(1 - 0.2 * 2)^2 = -0.36$	$F(\theta_1) = -(1 - 0.2 * 2.12)^2 = -0.33$	$F(\theta_2) = -(1 - 0.2 * 2.1776)^2 = -0.3186$
$\frac{\partial F}{\partial \theta} = 2(1 - 0.2 * 2) * 0.2 = 0.24$	$\frac{\partial F}{\partial \theta} = 2(1 - 0.2 * 2.12) * 0.2 = 0.2304$	…
$\rho_0 = 2^{-(0+1)} = 0.5$	$\rho_1 = 2^{-(1+1)} = 0.25$	…
$\theta_1 = \theta_0 + \rho_0 * g_0 = 2 + 0.5 * 0.24 = 2.12$	$\theta_2 = \theta_1 + \rho_1 * g_1 = 2.12 + 0.25 * 0.2304 = 2.1776$	…

Gradient = Multi-variable derivative

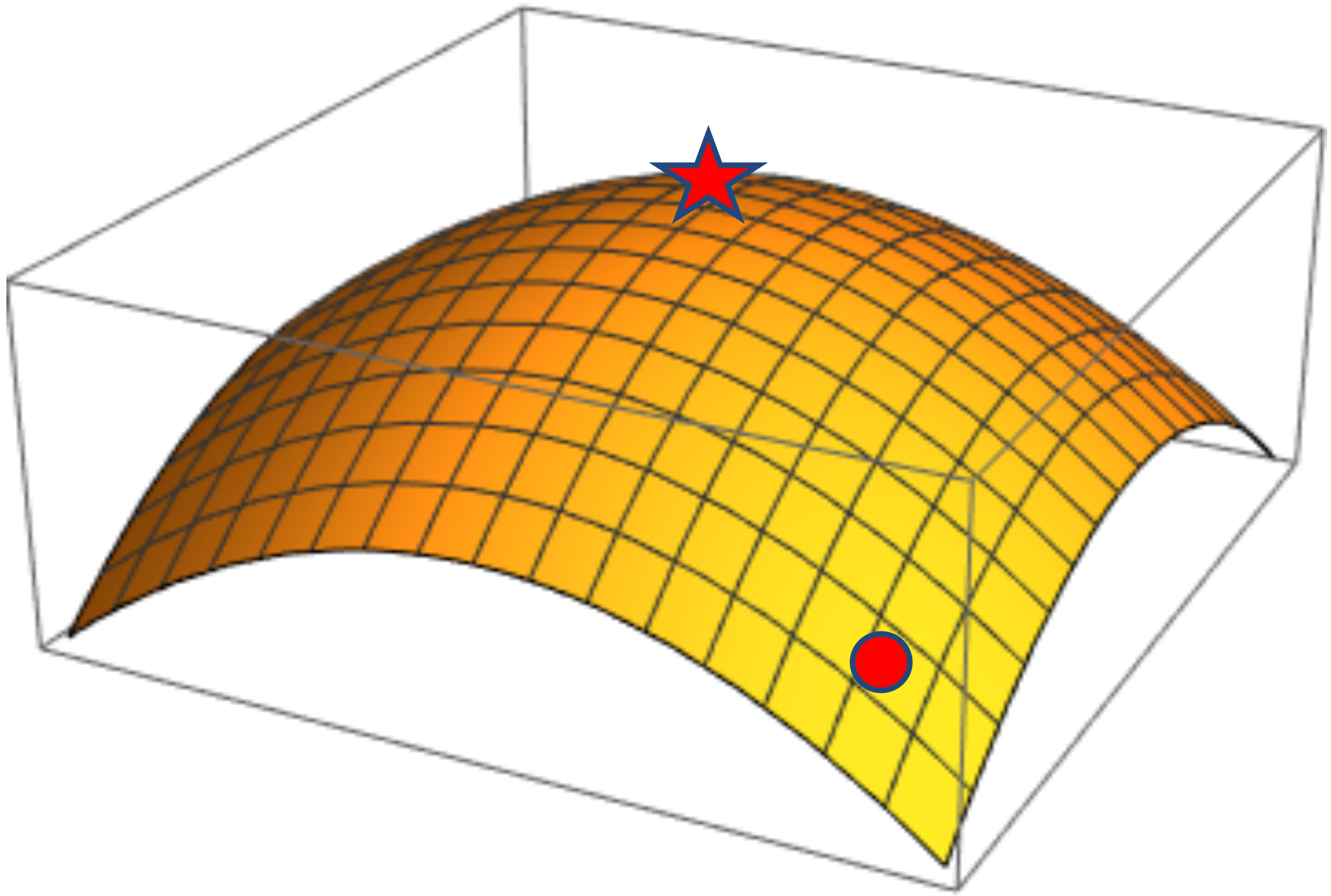
K-dimensional input

$$\nabla_{\theta} F(\theta) = \underbrace{\left(\frac{\partial F}{\partial \theta_1}, \frac{\partial F}{\partial \theta_2}, \ldots, \frac{\partial F}{\partial \theta_K} \right)}_{\text{K-dimensional output}}$$

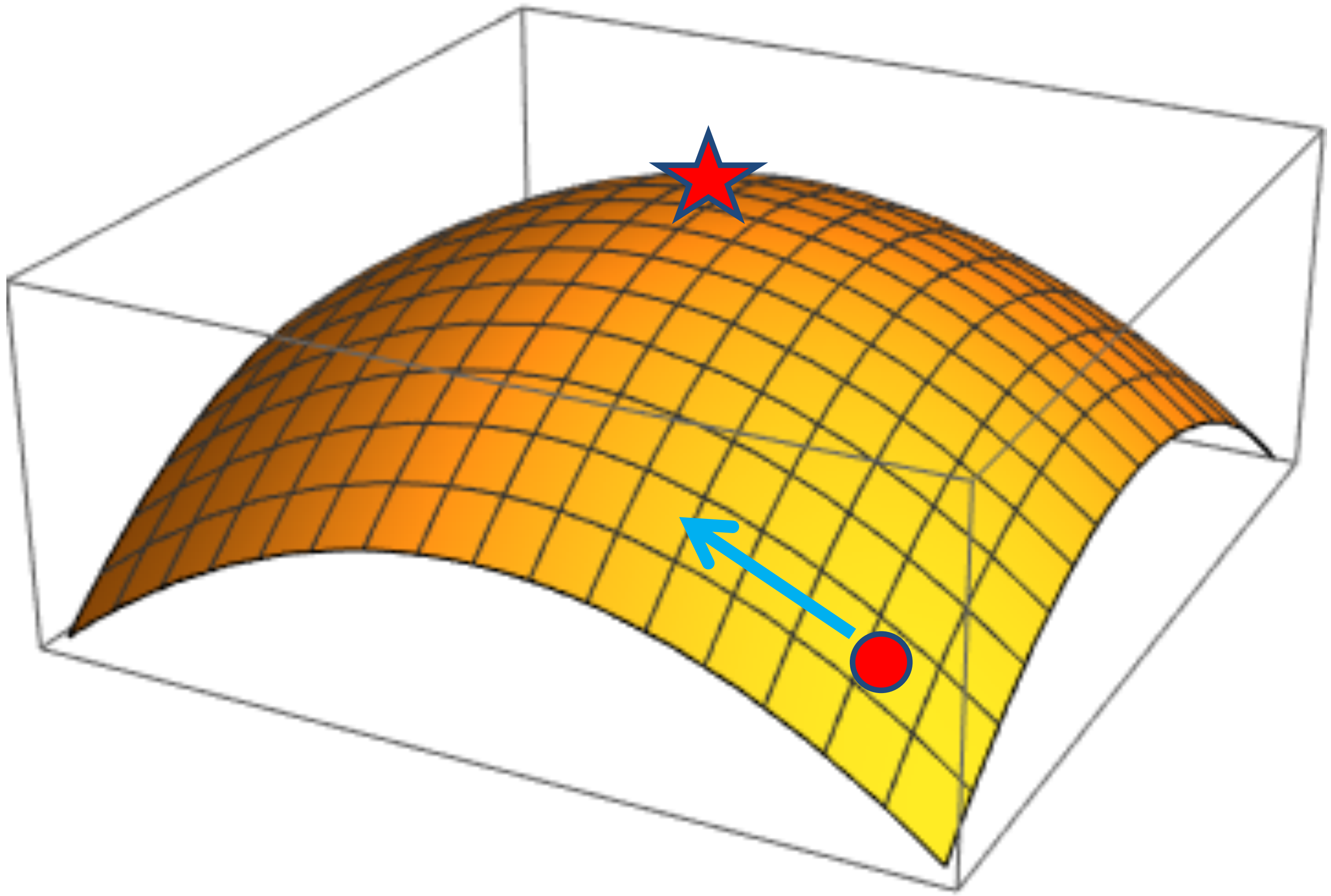
Gradient Ascent



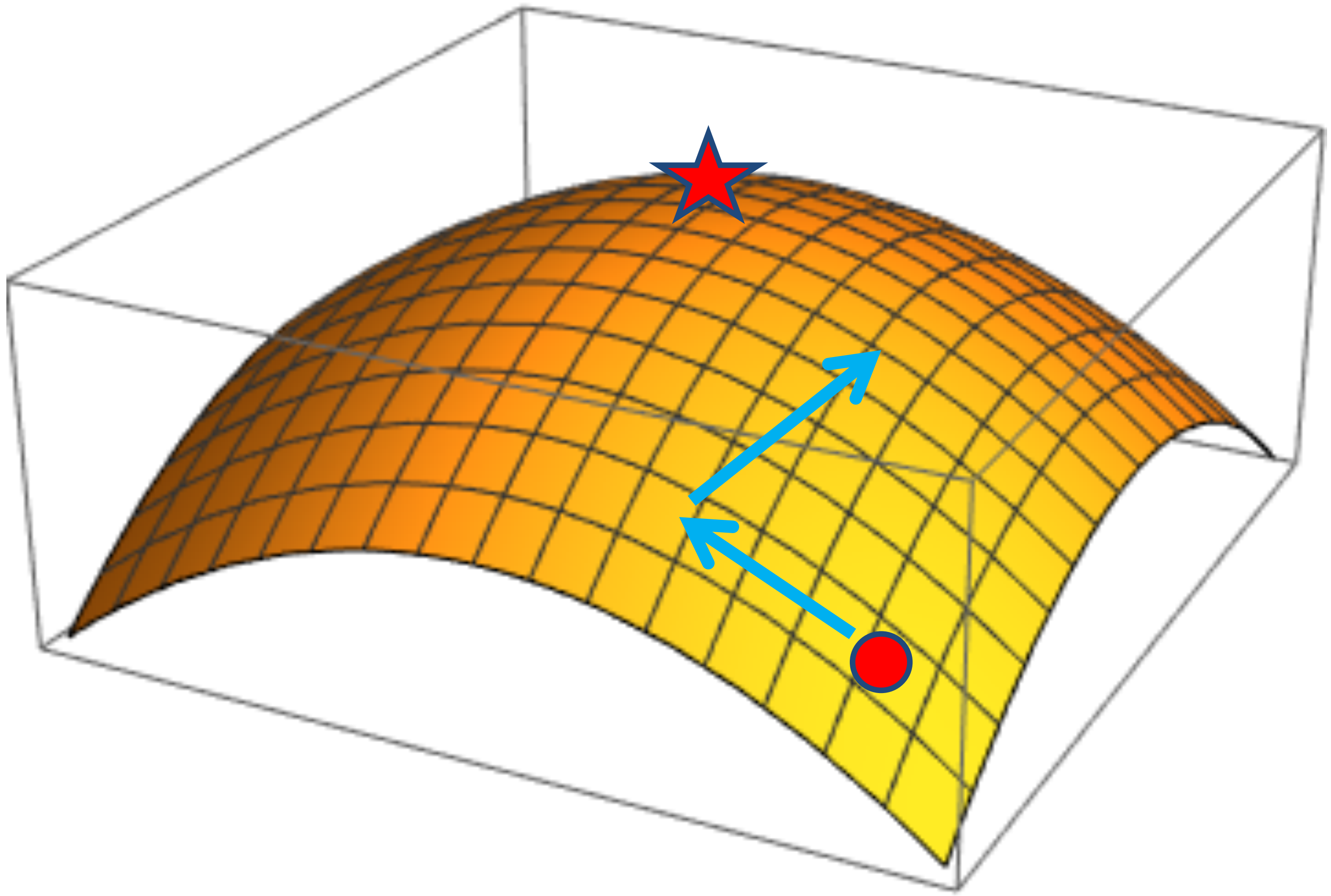
Gradient Ascent



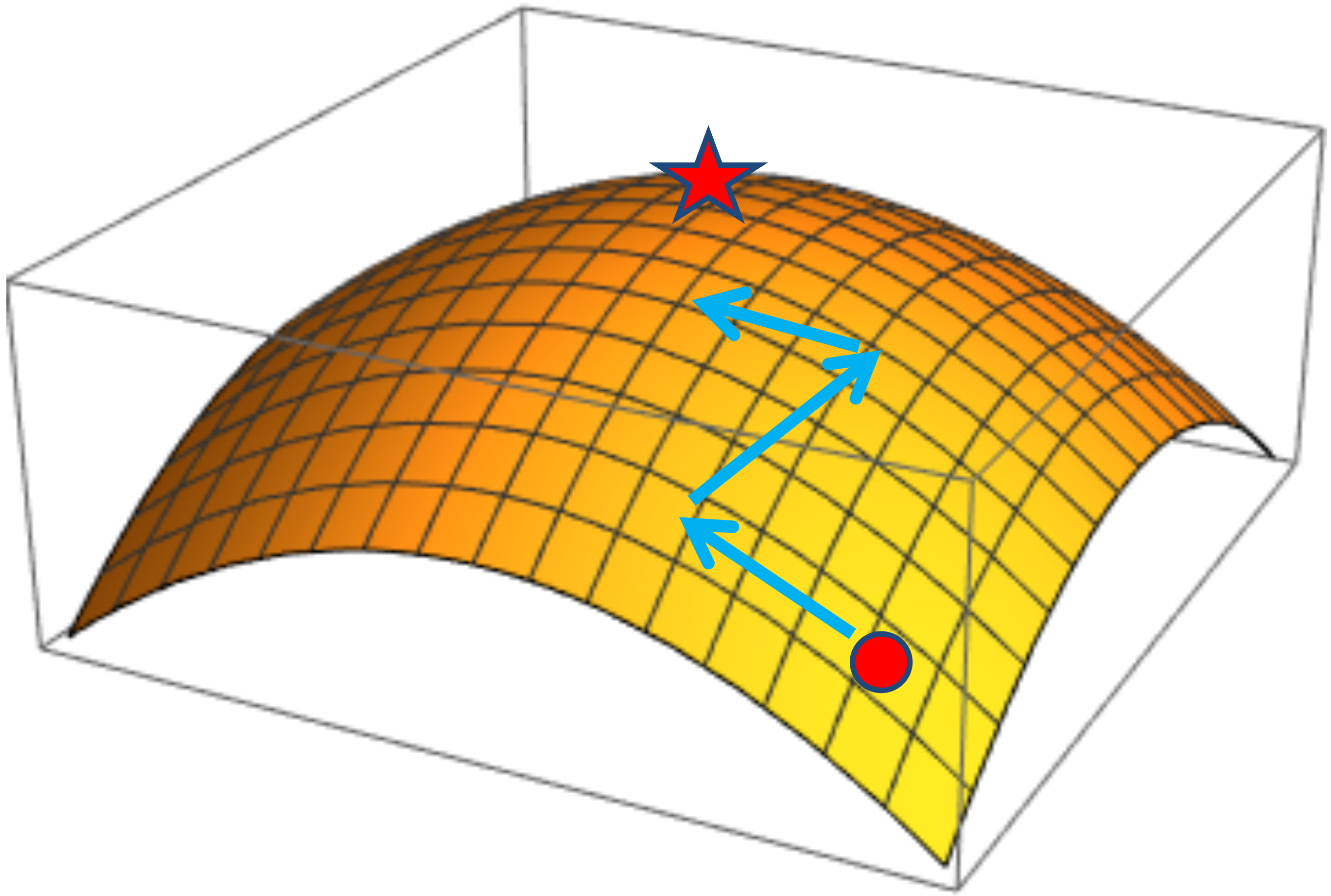
Gradient Ascent



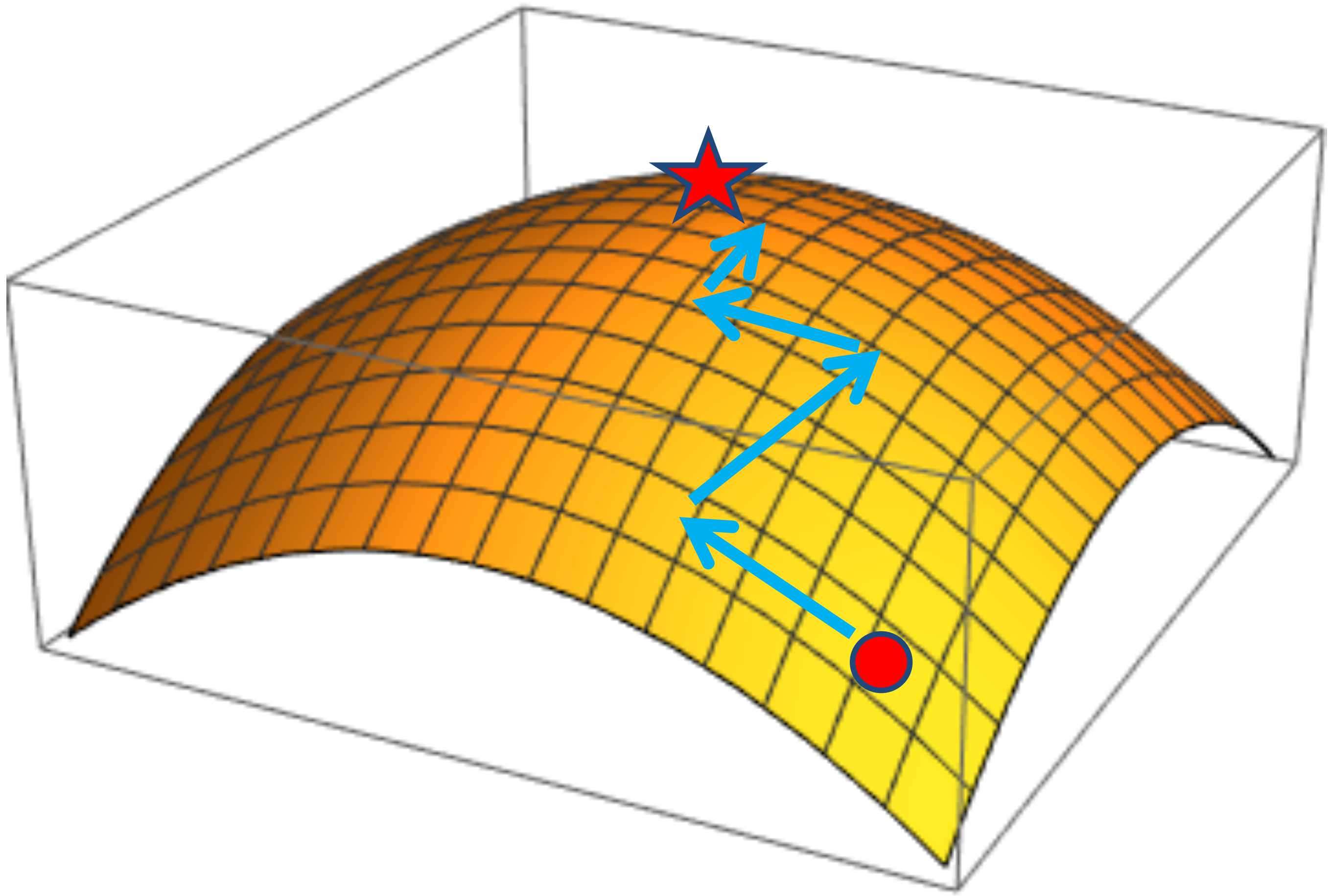
Gradient Ascent



Gradient Ascent



Gradient Ascent



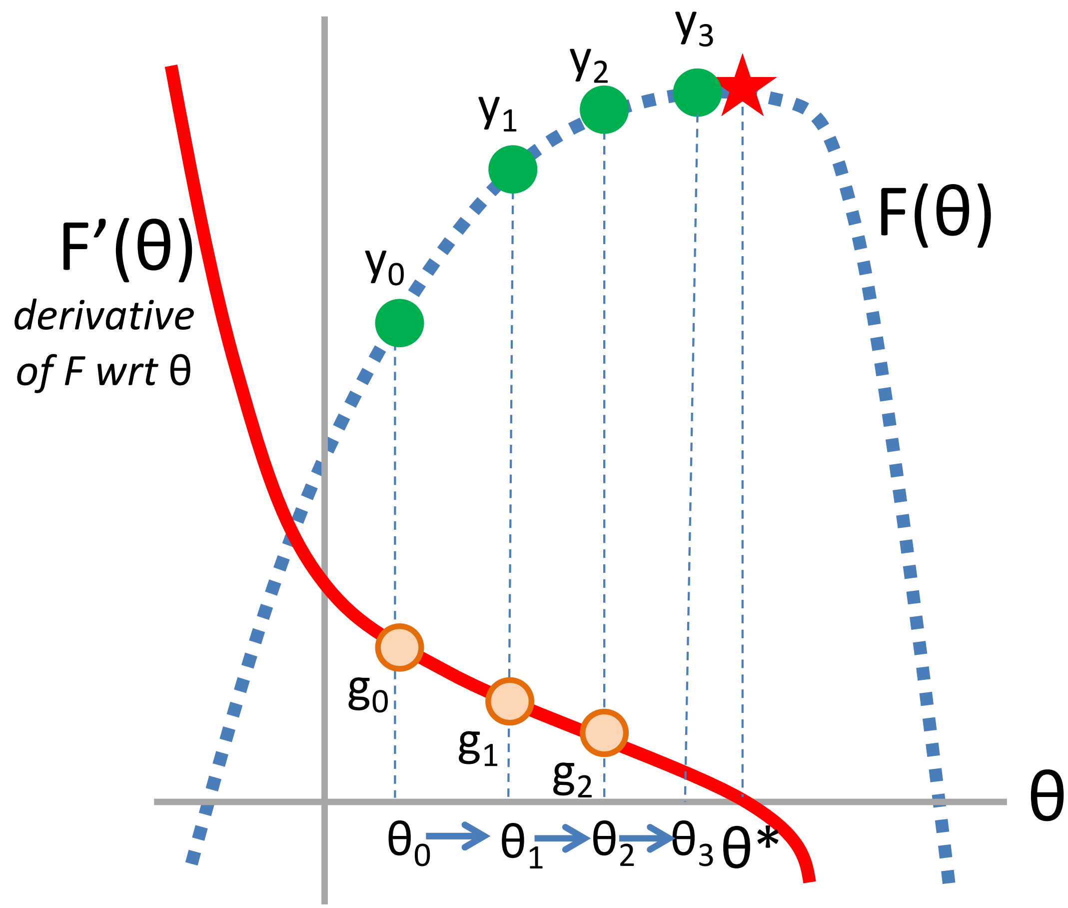
Optimizing a Function can be an Art

Set $t = 0$

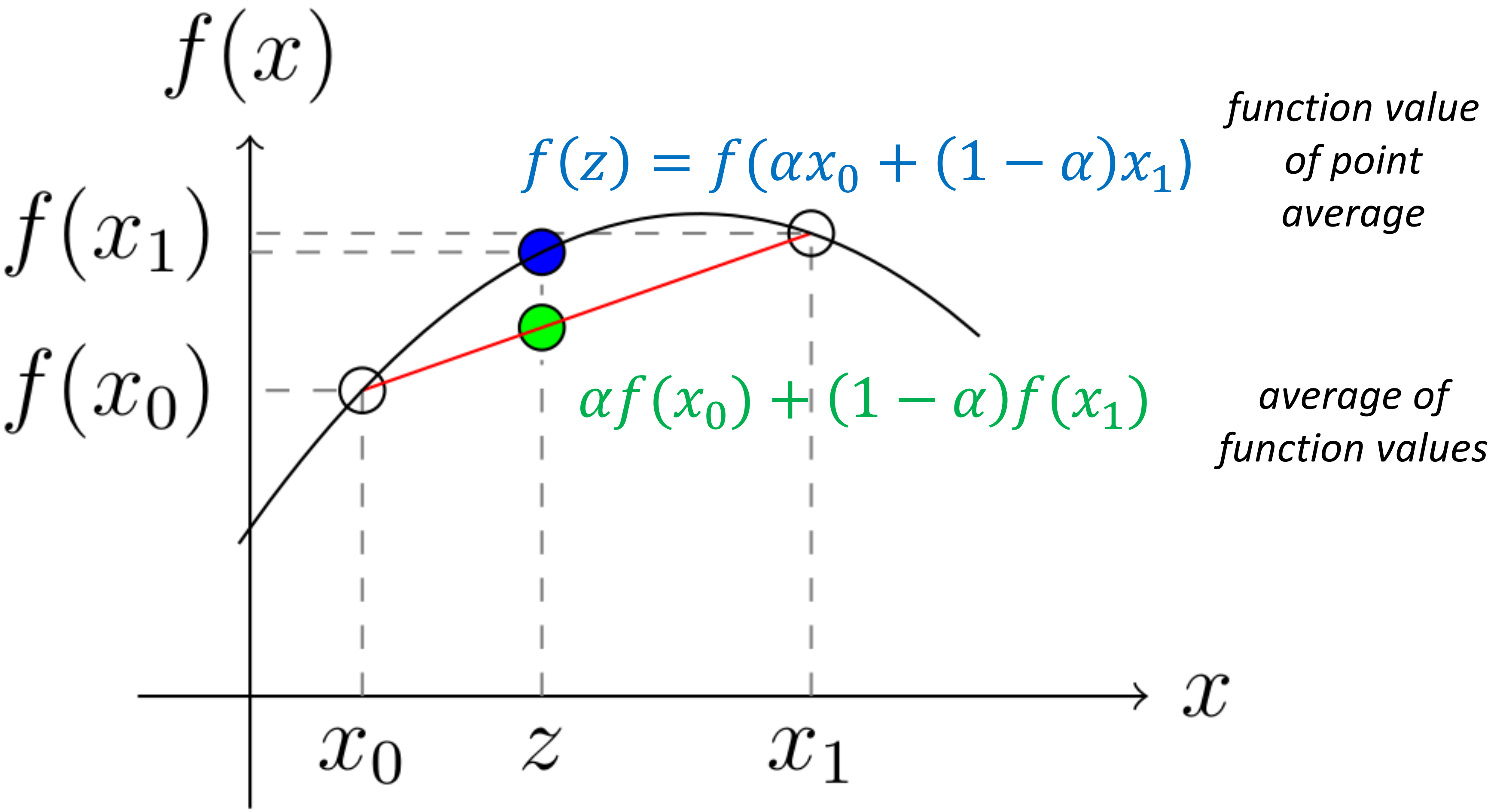
Pick a starting value θ_t

Until **converged**:

1. Get value $y_t = F(\theta_t)$
2. Get derivative $g_t = F'(\theta_t)$
3. Get **scaling factor ρ_t**
4. Set $\theta_{t+1} = \theta_t + \rho_t * g_t$
5. Set t += 1



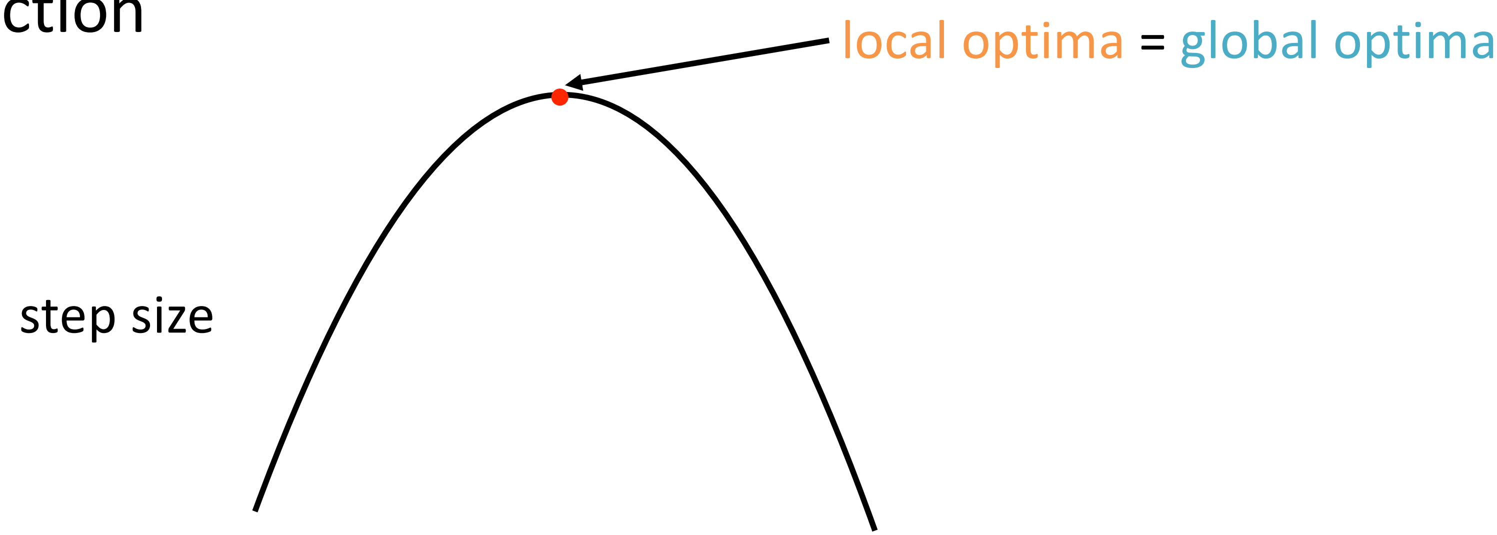
Convergence to Global Optima with Convex (Concave) Functions



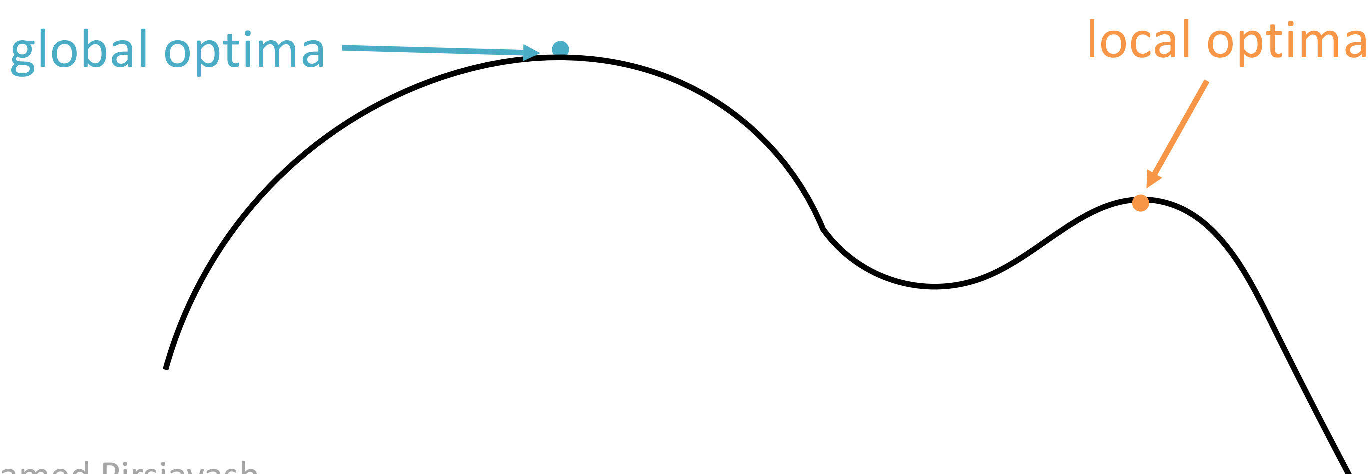
$$z = \alpha x_0 + (1-\alpha)x_1 \qquad 0 \leq \alpha \leq 1$$

Road Bumps in Gradient Ascent

Convex function



Non-convex function

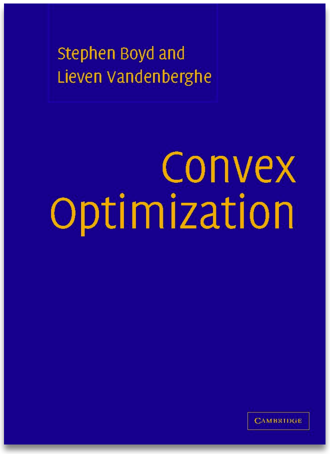


Adapted from Hamed Pirsiavash

Choice of step size

Too small → slow convergence

Too large → no convergence



http://stanford.edu/~boyd/cvxbook/

A1, Q4: You explore two strategies of setting the step size

Minimizing vs. Maximizing

Maximizing F

Set $t = 0$

Pick a starting value θ_t

Until converged:

1. Get value $y_t = F(\theta_t)$
2. Get derivative $g_t = F'(\theta_t)$
3. Get scaling factor ρ_t
4. Set $\theta_{t+1} = \theta_t + \rho_t * g_t$
5. Set t += 1

Minimizing vs. Maximizing

Maximizing F

Set $t = 0$

Pick a starting value θ_t

Until converged:

1. Get value $y_t = F(\theta_t)$
2. Get derivative $g_t = F'(\theta_t)$
3. Get scaling factor ρ_t
4. Set $\theta_{t+1} = \theta_t + \rho_t * g_t$
5. Set t += 1

Minimizing F

Set $t = 0$

Pick a starting value θ_t

Until converged:

1. Get value $y_t = F(\theta_t)$
2. Get derivative $g_t = F'(\theta_t)$
3. Get scaling factor ρ_t
4. Set $\theta_{t+1} = \theta_t - \rho_t * g_t$
5. Set t += 1

Outline

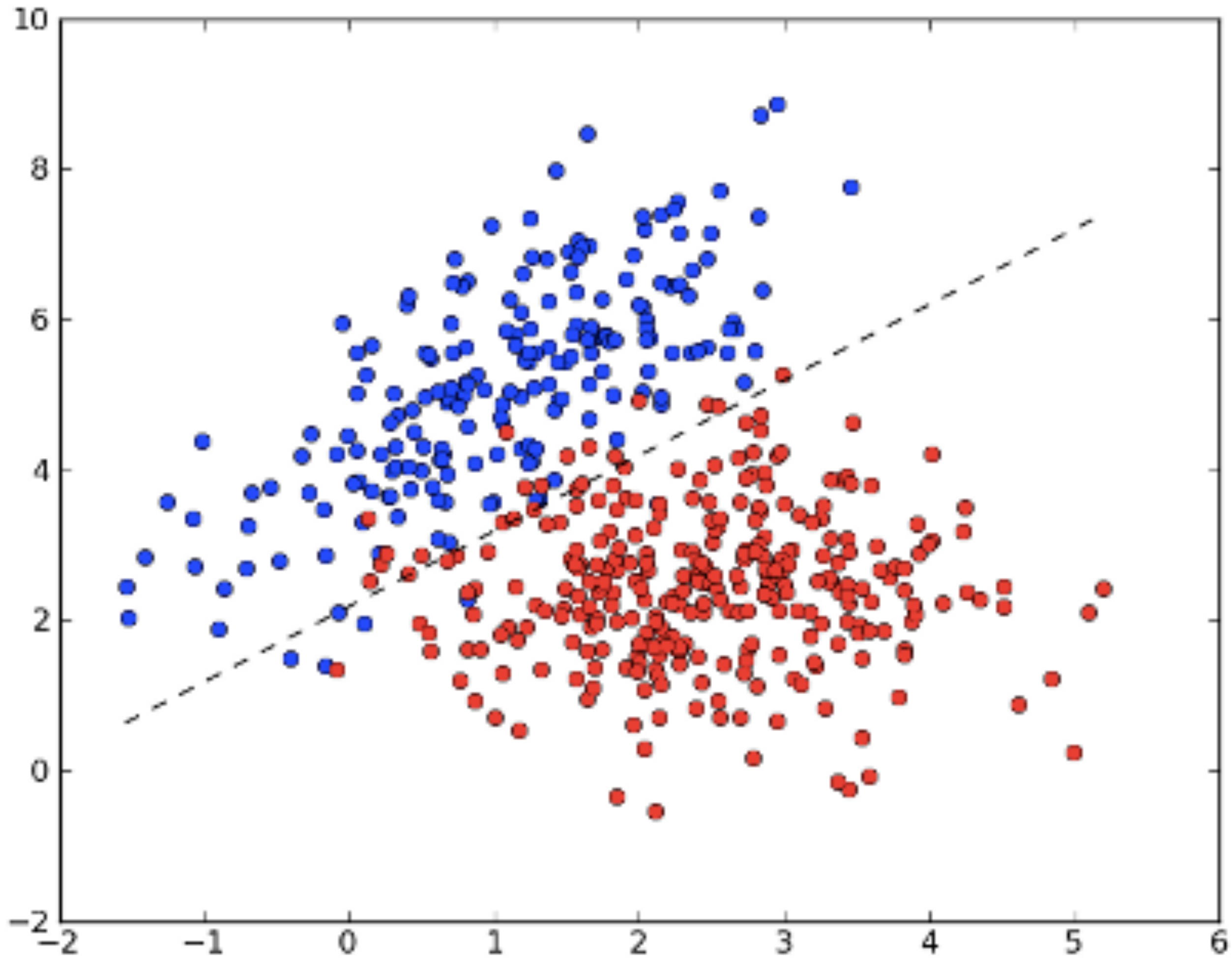
Recap: Decision Theory and ERM

Gradient Optimization

Linear Models & Surrogate Loss

- Example 1: Linear Regression
- Example 2: Linear Classification Model
- Example 3: Perceptrons

Linear Models



Classification as (Linear) Separability

Image courtesy Hamed Pirsiavash

A Simple Linear Model for Regression

$$\text{predict } y_i \text{ from } \mathbf{x_i}$$

value y_i

data point x_i, as a vector of features

A Simple Linear Model for Regression

vector w of weights

$$y_i = \mathbf{w^T x_i}$$

value y_i

data point x_i, as a vector of features

A Simple Linear Model for Regression

vector w of weights

bias b (WLOG, 0)

$$y_i = \mathbf{w}^{\mathbf{T}}\mathbf{x}_{\mathbf{i}} + b$$

value y_i

data point x_i, as a vector of features

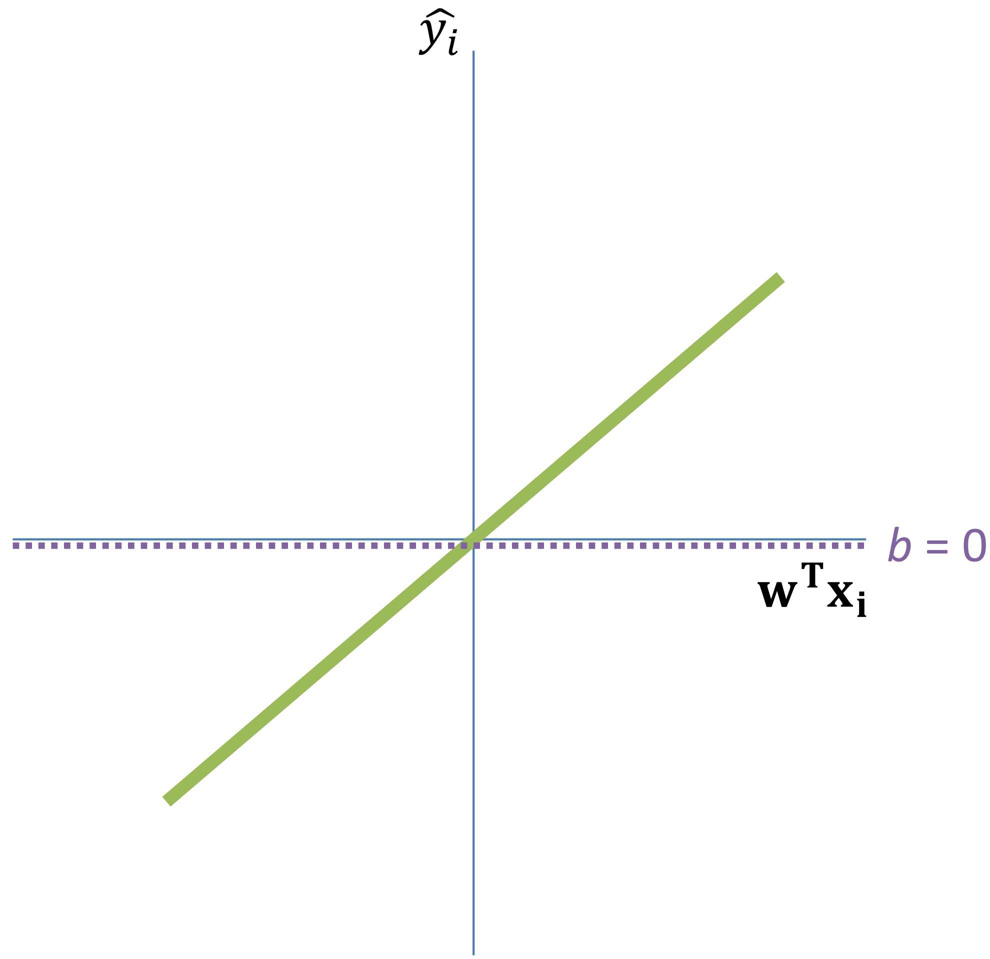
A Simple Linear Model for Regression

vector w of weights

$$y_i = \mathbf{w}^{\mathbf{T}}\mathbf{x_i} + 0$$

value y_i

data point x_i, as a vector of features



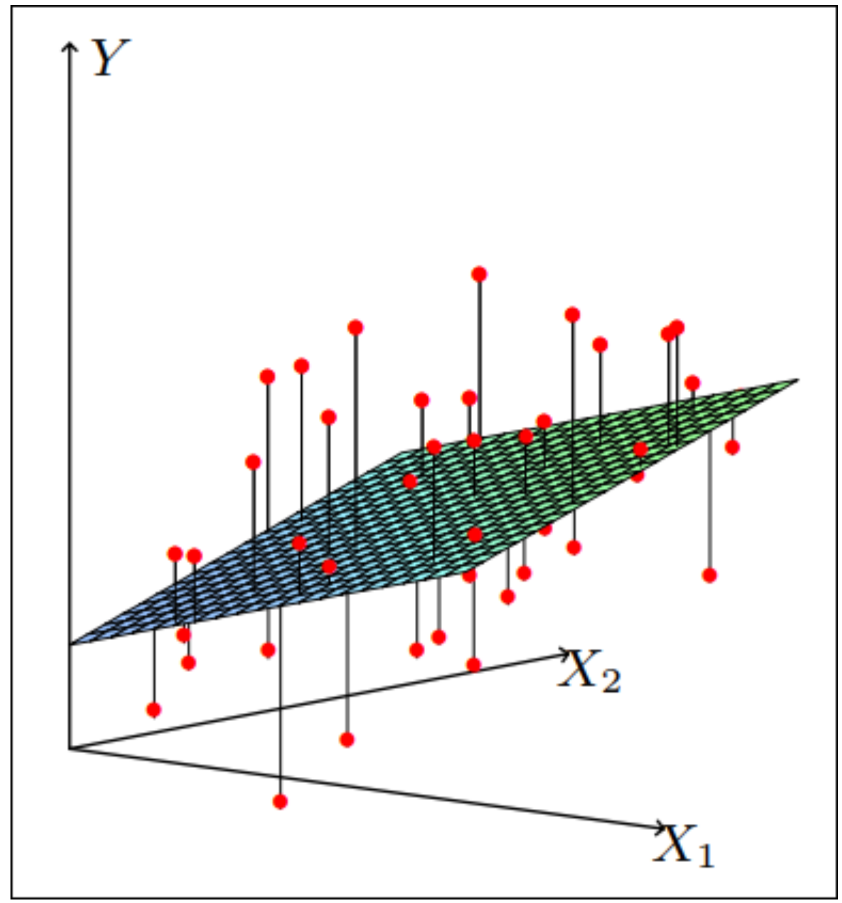
A Simple Linear Regression Model: From ERM

Common regression loss function:

$$\ell = (y_i - \hat{y}_i)^2$$

Start with the ERM objective…

$$\operatorname*{argmin}_{h} \sum_{i=1}^{N} \ell(y_i, h(\boldsymbol{x}_i))$$



ESL, Fig 3.1

A Simple Linear Regression Model: From ERM

Common regression loss function: $\ell = (y_i - \hat{y}_i)^2$

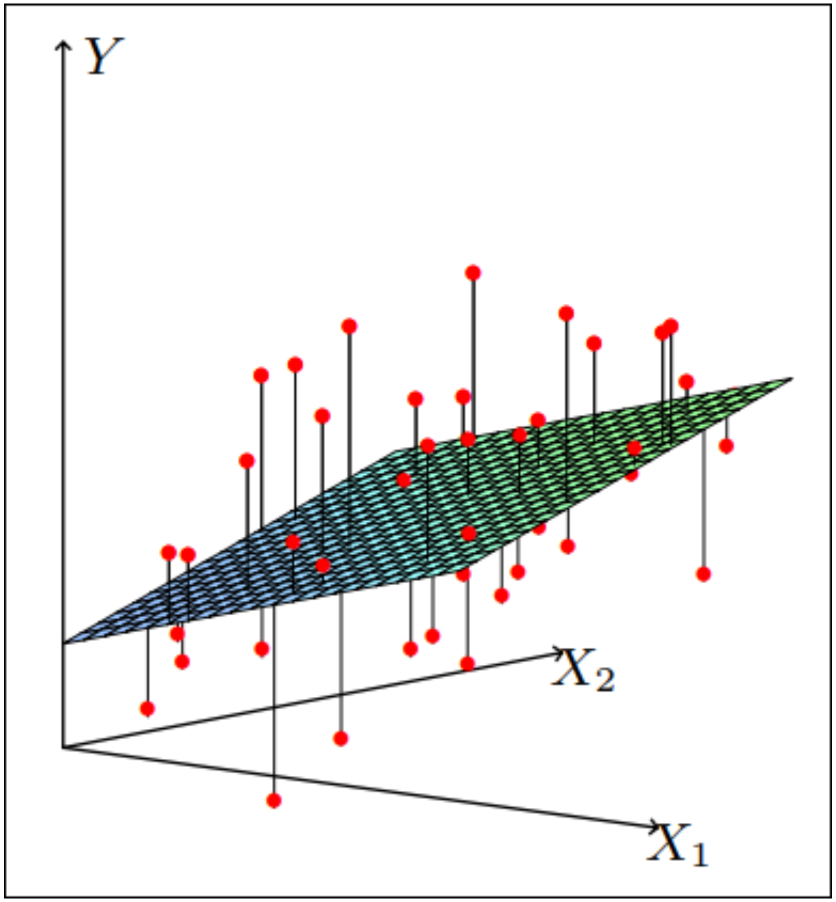
Start with the ERM objective…

$$\underset{h}{\operatorname{argmin}} \sum_{i=1}^{N} \ell(y_i, h(\boldsymbol{x}_i))$$

and get…

$$\underset{w}{\operatorname{argmin}} \sum_{i} (y_i - \mathbf{w}^{\mathrm{T}} x_i - b)^2$$

least squares estimation



ESL, Fig 3.1

A Simple Linear Regression Model: From ERM

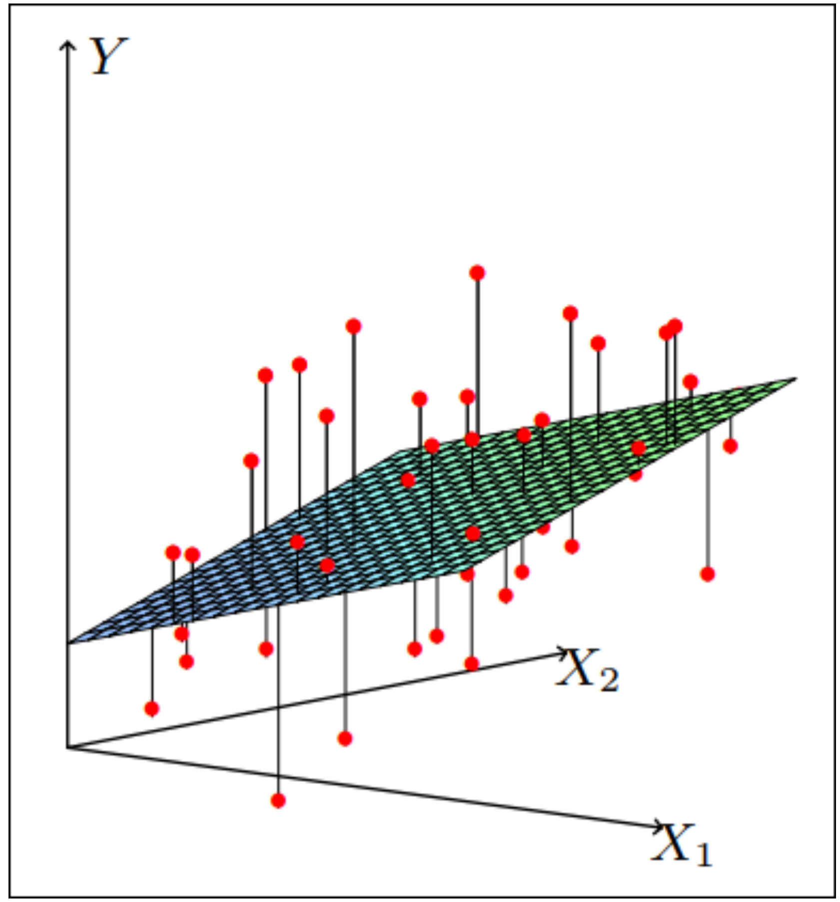
Common regression loss function:

$$\ell = (y_i - \hat{y}_i)^2$$

$$\underset{w}{\operatorname{argmin}} \sum_i \left(y_i - \mathbf{w}^{\mathrm{T}} x_i - b\right)^2 =$$

$$\min_w (\boldsymbol{y} - \mathbf{X}\mathbf{w} - \boldsymbol{b})^T (\boldsymbol{y} - \mathbf{X}\mathbf{w} - \boldsymbol{b})$$

least squares estimation



ESL, Fig 3.1

A Simple Linear Regression Model: From ERM

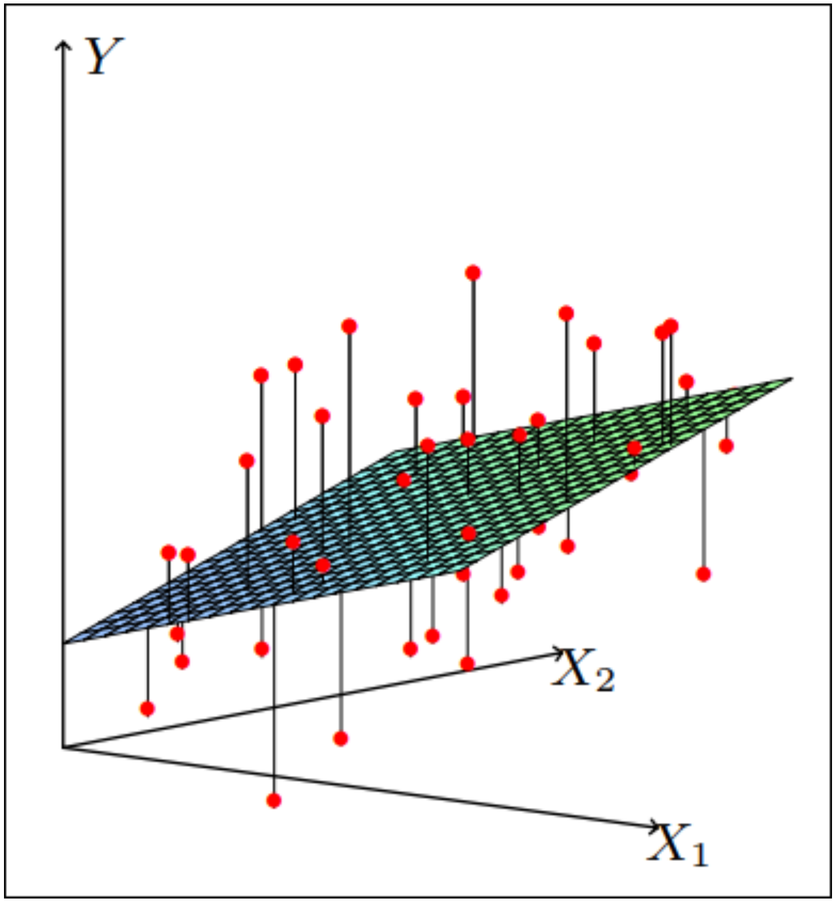
loss function: $\ell = (y_i - \hat{y}_i)^2$

$$\underset{w}{\operatorname{argmin}} \sum_i (y_i - \mathbf{w}^{\mathrm{T}} x_i - b)^2 =$$

$$\min_w \underbrace{(\boldsymbol{y} - \mathbf{X}\mathbf{w} - \boldsymbol{b})^T (\boldsymbol{y} - \mathbf{X}\mathbf{w} - \boldsymbol{b})}_{\mathcal{L}(\boldsymbol{w})}$$

A1, Q1 {E, F}

least squares estimation



ESL, Fig 3.1

Outline

Recap: Decision Theory and ERM

Gradient Optimization

Linear Models & Surrogate Loss

- Example 1: Linear Regression
- Example 2: Linear Classification Model
- Example 3: Perceptrons

A Simple Linear Model for Classification

vector w of weights

bias b (WLOG, 0)

$$y_i = \mathbf{w}^{\mathbf{T}}\mathbf{x}_{\mathbf{i}} + b$$

label y_i, (WLOG, binary {0, 1} value)

data point x_i, as a vector of features

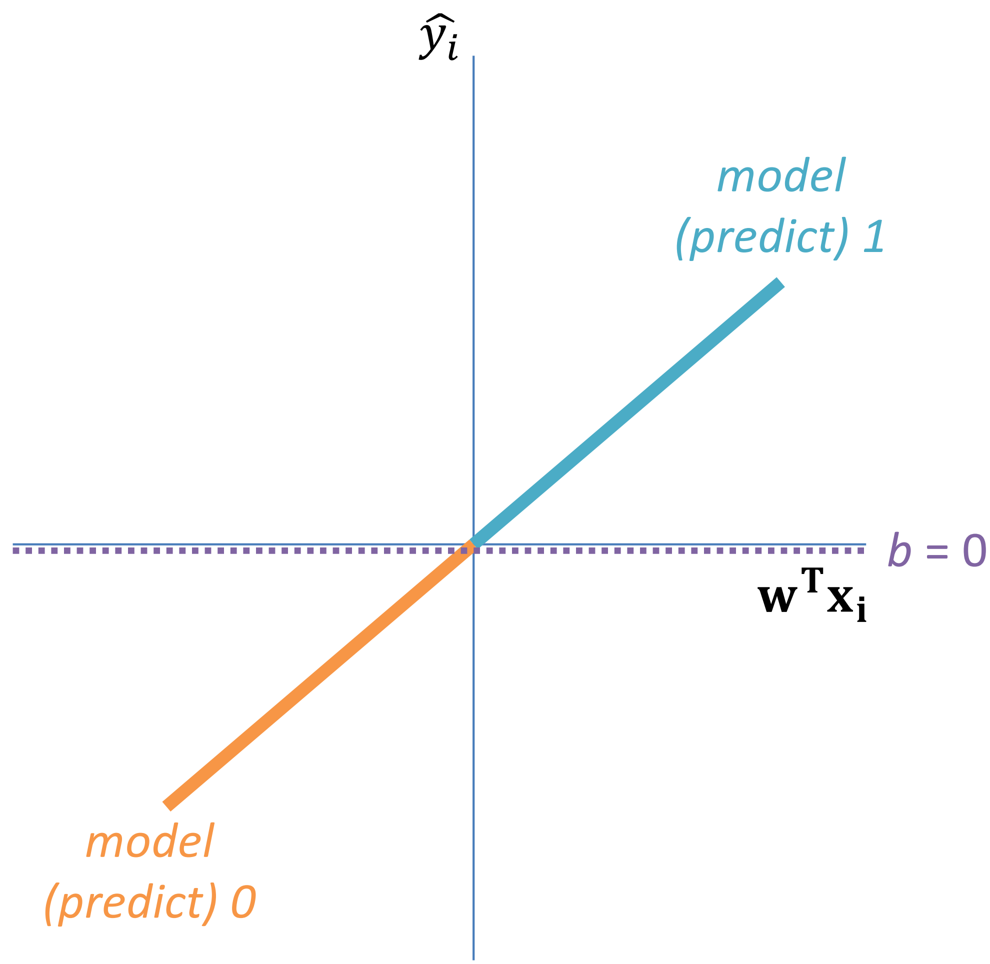
A Simple Linear Model for Classification

vector w of weights

$$y_i = \mathbf{w}^{\mathbf{T}}\mathbf{x_i}$$

label y_i, (WLOG, binary {0, 1} value)

data point x_i, as a vector of features



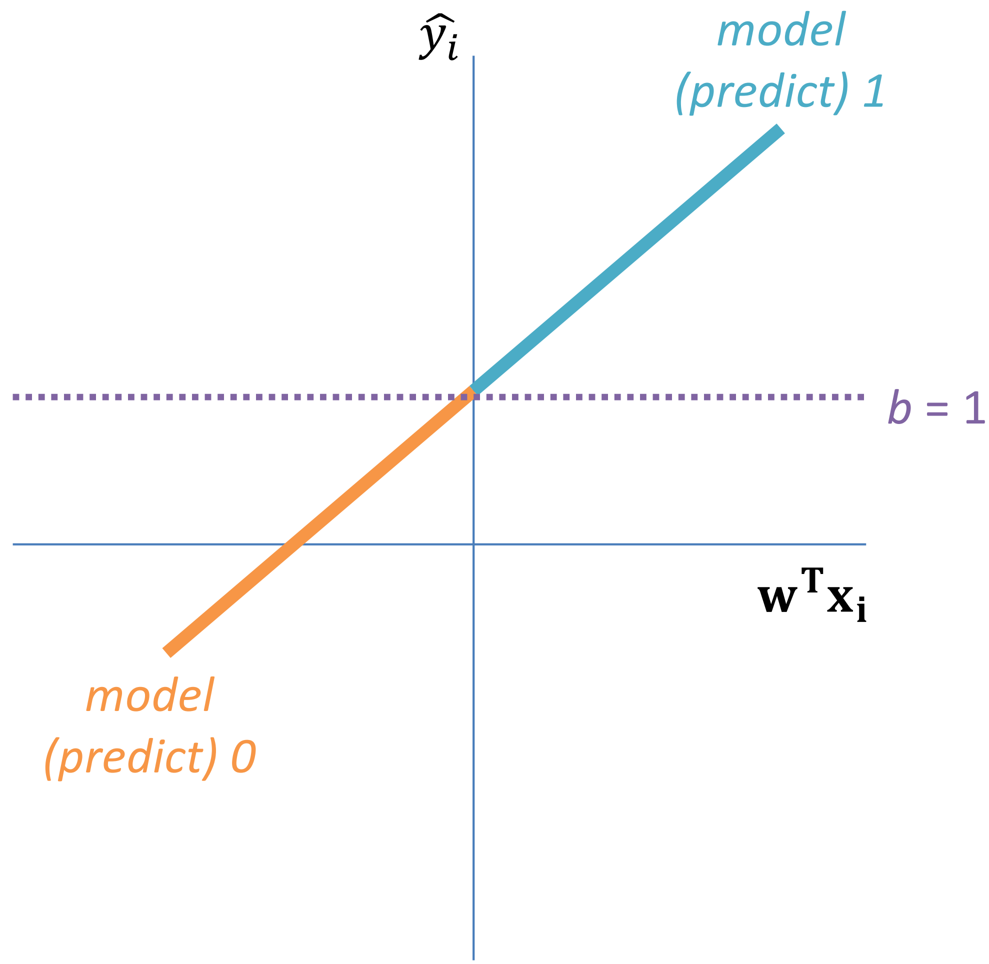
A Simple Linear Model for Classification

vector w of weights

$$y_i = \mathbf{w}^{\mathbf{T}}\mathbf{x_i} + 1$$

label y_i, (WLOG, binary {0, 1} value)

data point x_i, as a vector of features



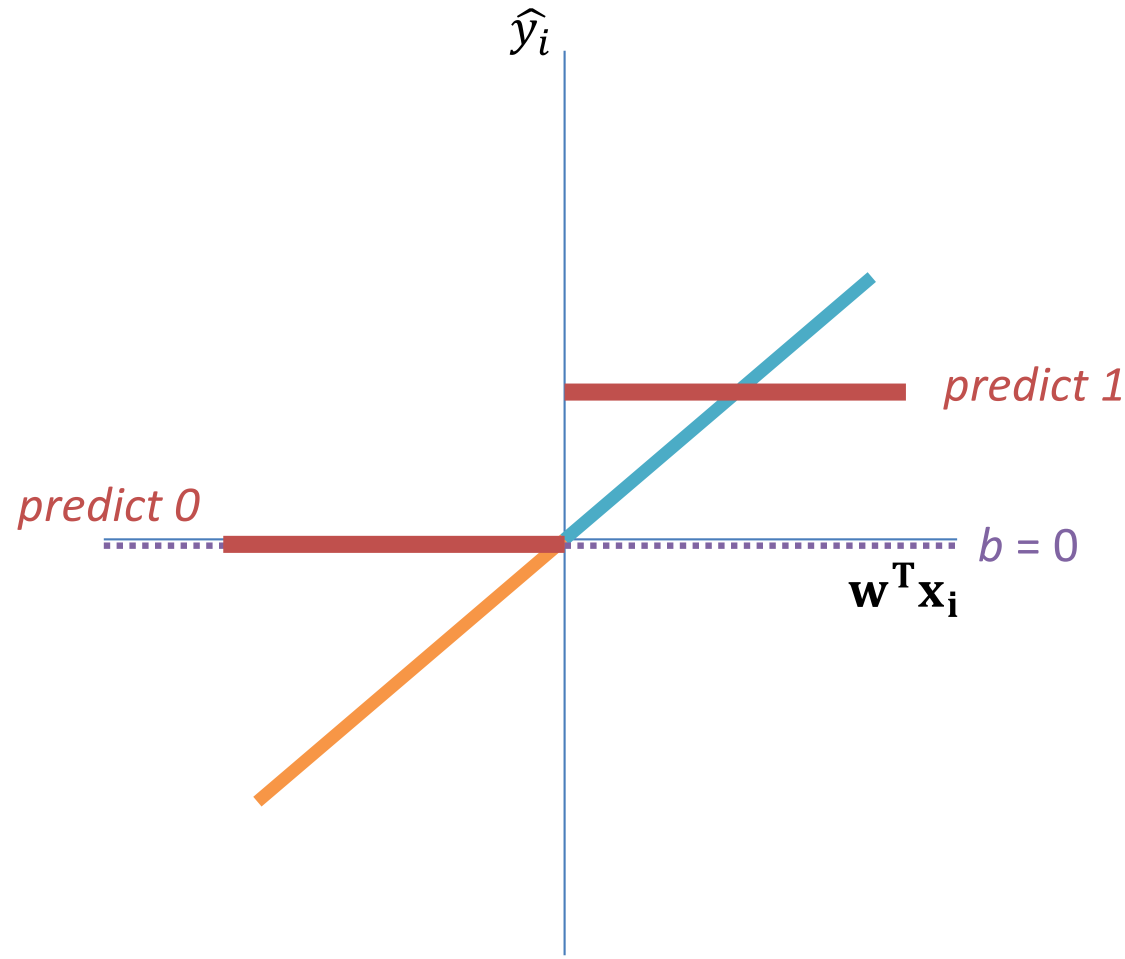
A Simple Linear Model for Classification

vector w of weights

$$y_i = \mathbf{w}^{\mathbf{T}}\mathbf{x_i}$$

label y_i, (WLOG, binary {0, 1} value)

data point x_i, as a vector of features



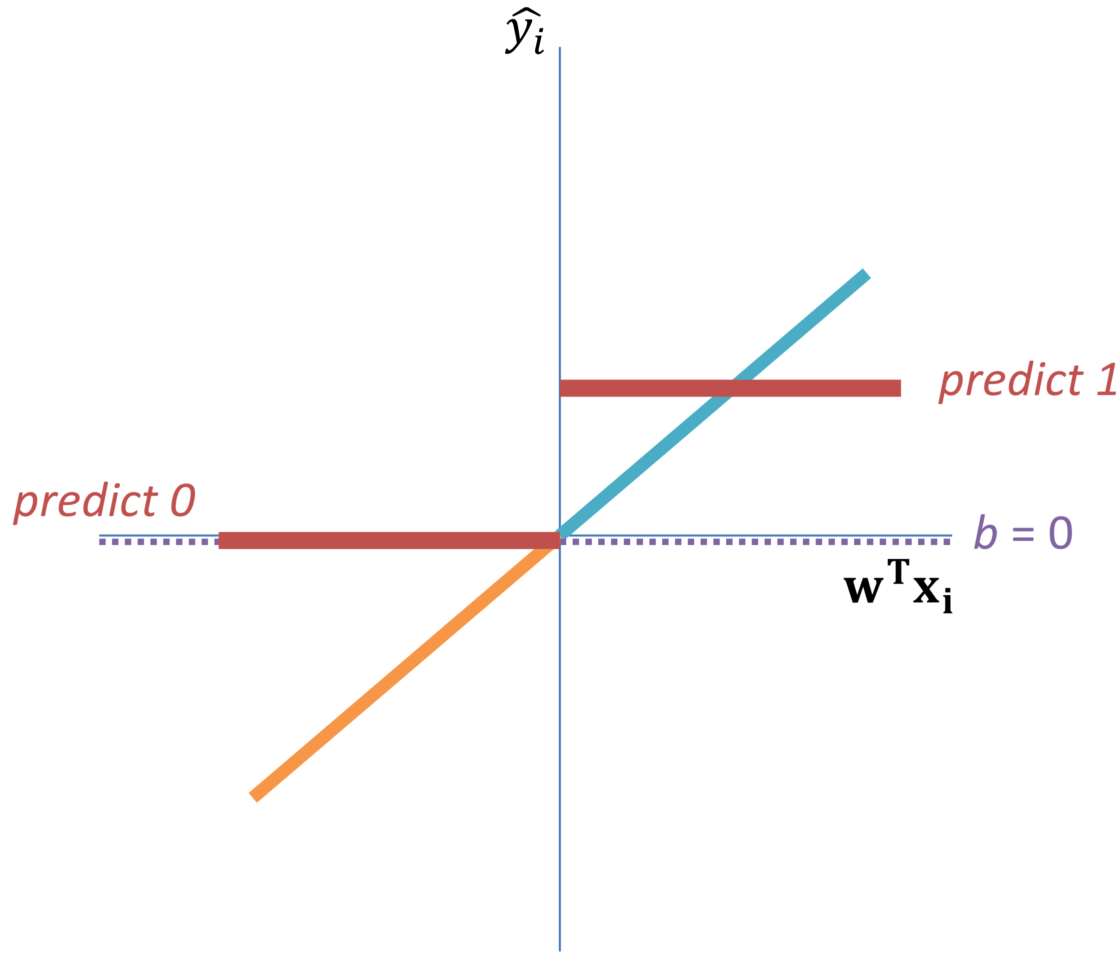
A Simple Linear Model for Classification

vector w of weights

$$y_i = \mathbf{w}^{\mathbf{T}}\mathbf{x_i}$$

label y_i, (WLOG, binary {0, 1} value)

data point x_i, as a vector of features



decision rule: $\widehat{y}_i = \begin{cases} 0, & \mathbf{w}^{\mathbf{T}}\mathbf{x_i} < 0 \\ 1, & \mathbf{w}^{\mathbf{T}}\mathbf{x_i} \geq 0 \end{cases}$

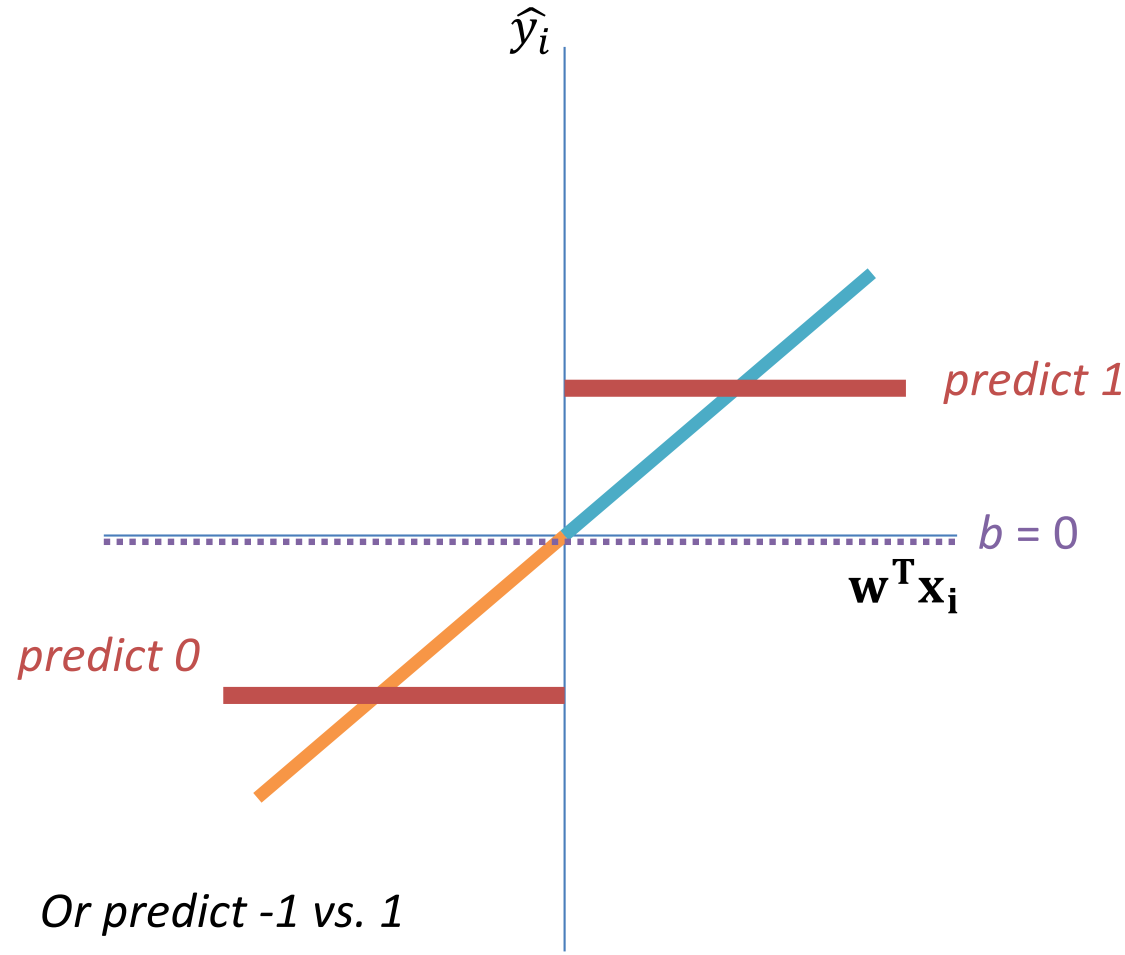
A Simple Linear Model for Classification

vector w of weights

$$y_i = \mathbf{w}^{\mathbf{T}}\mathbf{x_i}$$

label y_i, (WLOG, binary {0, 1} value)

data point x_i, as a vector of features



decision rule: $\hat{y}_i = \begin{cases} -1, & \mathbf{w}^{\mathbf{T}}\mathbf{x_i} < 0 \\ 1, & \mathbf{w}^{\mathbf{T}}\mathbf{x_i} \geq 0 \end{cases}$

A Simple Linear Classifier: From ERM

decision rule: $\hat{y}_i = \begin{cases} 0, & \mathbf{w}^{\mathbf{T}}\mathbf{x_i} < 0 \\ 1, & \mathbf{w}^{\mathbf{T}}\mathbf{x_i} \geq 0 \end{cases}$

loss function: $\ell = \begin{cases} 1, & y_i\mathbf{w}^{\mathbf{T}}\mathbf{x_i} < 0 \\ 0, & y_i\mathbf{w}^{\mathbf{T}}\mathbf{x_i} \geq 0 \end{cases}$

Q: Are there any issues?

A Simple Linear Classifier: From ERM

decision rule: $\hat{y}_i = \begin{cases} 0, & \mathbf{w^T x_i} < 0 \\ 1, & \mathbf{w^T x_i} \geq 0 \end{cases}$

loss function: $\ell = \begin{cases} 1, & y_i \mathbf{w^T x_i} < 0 \\ 0, & y_i \mathbf{w^T x_i} \geq 0 \end{cases}$

Q: Are there any issues?

A: Objective is piecewise constant wrt weights **w**

Loss Function Example: 0-1 Loss

$$\ell(y, \hat{y}) = \begin{cases} 0, & \text{if } y = \hat{y} \\ 1, & \text{if } y \neq \hat{y} \end{cases}$$

Problem: not differentiable wrt $\hat{y}$ (or θ)

Solution 1: is h(x) a conditional distribution p(y | x)? Use MAP

Solution 2: use a surrogate loss that approximates 0-1

Solution 3: is the data linearly separable? Perceptron can work

Why Do We Care about Probabilities? Classification

- Assigning subject categories, topics, or genres
- Spam detection
- Authorship identification
- Age/gender identification
- Language Identification
- Sentiment analysis
- …

class

observed data

class-based likelihood (language model)

prior probability of class

$$p(Y \mid X) = \frac{p(X \mid Y) * p(Y)}{p(X)}$$

observation likelihood (averaged over all classes)

Classify with Bayes Rule

$$\mathrm{argmax}_Y \, p(Y \mid X)$$

Classify with Bayes Rule

$$\operatorname{argmax}_Y \frac{p(X \mid Y) * p(Y)}{p(X)}$$

Classify with Bayes Rule

$$\text{argmax}_Y \frac{p(X \mid Y) * p(Y)}{p(X)}$$

constant with respect to X

Classify with Bayes Rule

$$\operatorname{argmax}_Y p(X \mid Y) * p(Y)$$

Classify with Bayes Rule

$$\text{argmax}_Y \log p(X \mid Y) + \log p(Y)$$

Classify with Bayes Rule

$$\operatorname{argmax}_Y \log p(X \mid Y) + \log p(Y)$$

Classify with Bayes Rule

how likely is label Y *overall?*

$$\text{argmax}_Y \log p(X \mid Y) + \log p(Y)$$

how well does blob X *represent label* Y*?*

Classify with Bayes Rule

how likely is label Y *overall?*

$$\text{argmax}_Y \log p(X \mid Y) + \log p(Y)$$

how well does blob X *represent label* Y*?*

For “simple” or “flat” labels:

* iterate through labels
* evaluate score for each label, keeping only the best (n best)
* return the best (or n best) label and score

"Solution" 1: A Simple Probabilistic (Linear*) Classifier

decision rule:

$$\hat{y}_i = \begin{cases} 0, & p(\hat{y}_i = 1 \mid \mathbf{x_i}) < .5 \\ 1, & p(\hat{y}_i = 1 \mid \mathbf{x_i}) \geq .5 \end{cases}$$

turn responses
into probabilities

minimize posterior 0-1 loss:

$$\min_{\mathbf{w}} \sum_i \mathbb{E}_{\widehat{y_i} \sim p(\cdot|x_i)}[\ell(y_i, \hat{y}_i)] =$$

*linear not strictly required

“Solution” 1: A Simple Probabilistic (Linear*) Classifier

decision rule:

$$\widehat{y_i} = \begin{cases} 0, & p(\widehat{y_i} = 1 \mid \mathbf{x_i}) < .5 \\ 1, & p(\widehat{y_i} = 1 \mid \mathbf{x_i}) \geq .5 \end{cases}$$

turn responses into probabilities

minimize posterior 0-1 loss:

$$\min_{\mathbf{w}} \sum_i \mathbb{E}_{\widehat{y_i} \sim p(\cdot|x_i)}[\ell(y_i, \widehat{y_i})] =$$

$$\min_{\mathbf{w}} \sum_i \sum_j (1 - \delta_{y_i j}) p(\widehat{y_i} = j | x_i) =$$

Kronecker delta:
1 if $y_i = j$, 0 otherwise

*linear not strictly required

"Solution" 1: A Simple Probabilistic (Linear*) Classifier

decision rule:

$$\hat{y}_i = \begin{cases} 0, & p(\hat{y}_i = 1 \mid \mathbf{x_i}) < .5 \\ 1, & p(\hat{y}_i = 1 \mid \mathbf{x_i}) \geq .5 \end{cases}$$

turn responses into probabilities

minimize posterior 0-1 loss:

$$\min_{\mathbf{w}} \sum_i \mathbb{E}_{\widehat{y_i} \sim p(\cdot|x_i)}[\ell(y_i, \hat{y}_i)] =$$

$$\min_{\mathbf{w}} \sum_i \sum_j (1 - \delta_{y_i j}) p(\hat{y}_i = j | x_i) =$$

$$\min_{\mathbf{w}} \sum_i \left[1 - \sum_j \delta_{y_i j} p(\hat{y}_i = j | x_i)\right]$$

*linear not strictly required

“Solution” 1: A Simple Probabilistic (Linear*) Classifier

decision rule:

$$\hat{y}_i = \begin{cases} 0, & p(\hat{y}_i = 1 \mid \mathbf{x_i}) < .5 \\ 1, & p(\hat{y}_i = 1 \mid \mathbf{x_i}) \geq .5 \end{cases}$$

turn responses into probabilities

minimize posterior 0-1 loss:

$$\min_{\mathbf{w}} \sum_i \mathbb{E}_{\widehat{y_i} \sim p(\cdot|x_i)}[\ell(y_i, \hat{y}_i)] =$$

$$\min_{\mathbf{w}} \sum_i \sum_j (1 - \delta_{y_i j}) p(\hat{y}_i = j | x_i) =$$

$$\min_{\mathbf{w}} \sum_i \left[1 - \sum_j \delta_{y_i j} p(\hat{y}_i = j | x_i)\right]$$

why MAP classifiers are reasonable

$$\max_{\mathbf{w}} \sum_i p(\hat{y}_i = y_i | x_i)$$

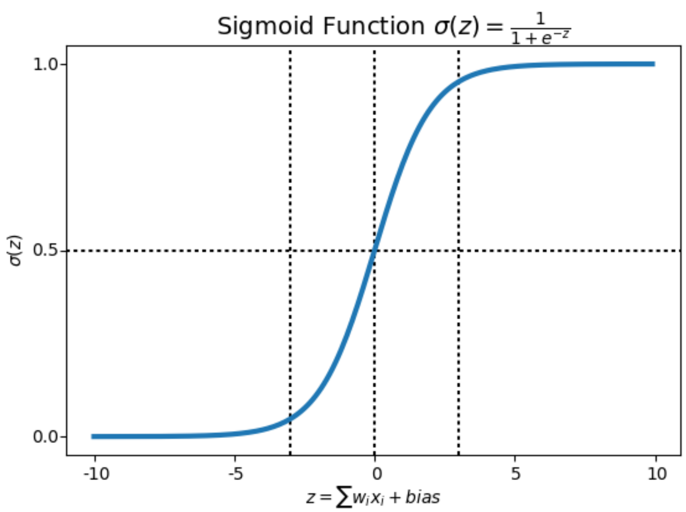
*linear not strictly required

"Solution" 1: A Simple Probabilistic (Linear*) Classifier

decision rule:

$$\widehat{y_i} = \begin{cases} 0, & \sigma(\mathbf{w}^{\mathbf{T}}\mathbf{x_i} + b) < .5 \\ 1, & \sigma(\mathbf{w}^{\mathbf{T}}\mathbf{x_i} + b) \geq .5 \end{cases}$$

turn responses into probabilities



minimize posterior 0-1 loss:

$$\min_{\mathbf{w}} \sum_i \mathbb{E}_{\widehat{y_i}}[\ell^{0/1}(y, \widehat{y_i})] =$$

$$\max_{\mathbf{w}} \sum_i p(\widehat{y_i} = y_i | x_i)$$

why MAP classifiers are reasonable

Plot from https://towardsdatascience.com/multi-layer-neural-networks-with-sigmoid-function-deep-learning-for-rookies-2-bf464f09eb7f

*linear not strictly required

Loss Function Example: 0-1 Loss

$$\ell(y, \hat{y}) = \begin{cases} 0, & \text{if } y = \hat{y} \\ 1, & \text{if } y \neq \hat{y} \end{cases}$$

Problem: not differentiable wrt $\hat{y}$ (or θ)

Solution 1: is h(x) a conditional distribution p(y | x)? Use MAP

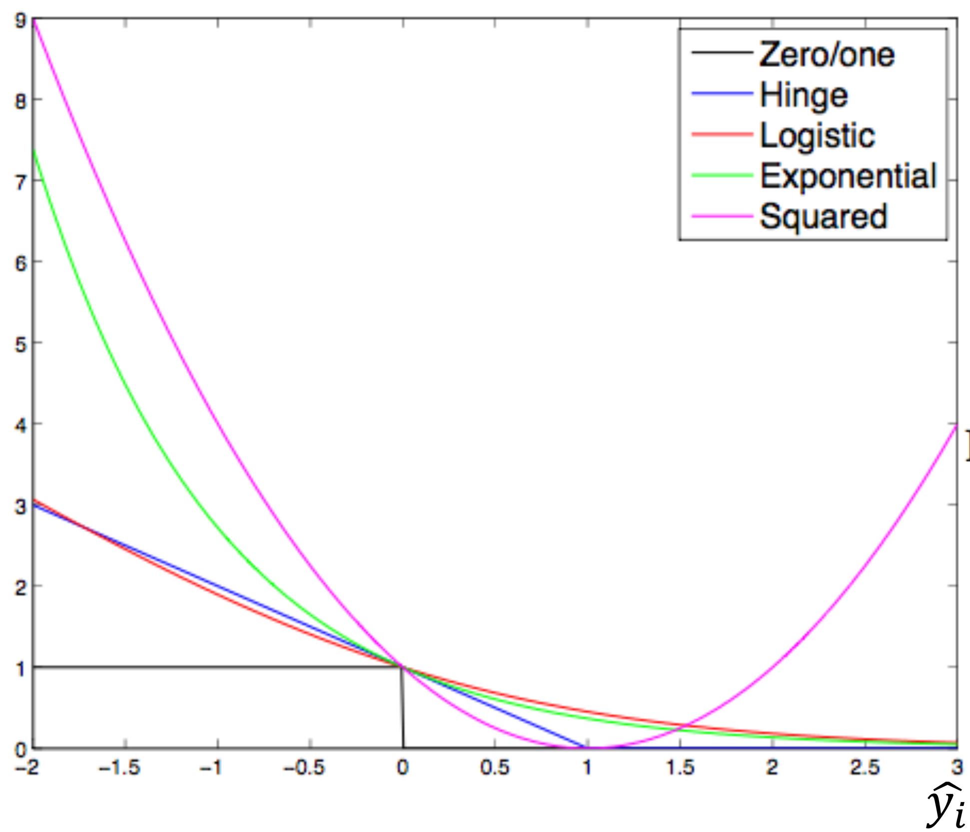
Solution 2: use a surrogate loss that approximates 0-1

Solution 3: is the data linearly separable? Perceptron can work

Solution 2:
Convex surrogate loss functions

Surrogate loss: replace Zero/one loss by a smooth function

Easier to optimize if the surrogate loss is convex



$$y = +1 \qquad \hat{y} \leftarrow \mathbf{w}^T \mathbf{x}$$

$$\begin{aligned}
\text{Zero/one:} \quad & \ell^{(0/1)}(y, \hat{y}) = \mathbf{1}[y\hat{y} \leq 0] \\
\text{Hinge:} \quad & \ell^{(\text{hin})}(y, \hat{y}) = \max\{0, 1 - y\hat{y}\} \\
\text{Logistic:} \quad & \ell^{(\text{log})}(y, \hat{y}) = \frac{1}{\log 2} \log\left(1 + \exp[-y\hat{y}]\right) \\
\text{Exponential:} \quad & \ell^{(\text{exp})}(y, \hat{y}) = \exp[-y\hat{y}] \\
\text{Squared:} \quad & \ell^{(\text{sqr})}(y, \hat{y}) = (y - \hat{y})^2
\end{aligned}$$

Courtesy Hamed Pirsiavash, CIML

Example: Exponential loss

$$\mathcal{L}(\mathbf{w}) = \sum_n \exp(-y_n \mathbf{w}^T \mathbf{x}_n)$$

objective

Courtesy Hamed Pirsiavash

Example: Exponential loss

$$\mathcal{L}(\mathbf{w}) = \sum_n \exp(-y_n \mathbf{w}^T \mathbf{x}_n)$$ objective

$$\frac{d\mathcal{L}}{d\mathbf{w}} = \sum_n -y_n \mathbf{x}_n \exp(-y_n \mathbf{w}^T \mathbf{x}_n)$$ gradient

Courtesy Hamed Pirsiavash

Example: Exponential loss

$$\mathcal{L}(\mathbf{w}) = \sum_n \exp(-y_n \mathbf{w}^T \mathbf{x}_n)$$ objective

$$\frac{d\mathcal{L}}{d\mathbf{w}} = \sum_n -y_n \mathbf{x}_n \exp(-y_n \mathbf{w}^T \mathbf{x}_n)$$ gradient

$$\mathbf{w} \leftarrow \mathbf{w} - \eta \sum_n -y_n \mathbf{x}_n \exp(-y_n \mathbf{w}^T \mathbf{x}_n)$$ update

loss term

$$\mathbf{w} \leftarrow \mathbf{w} + c y_n \mathbf{x}_n$$

high for misclassified points

Courtesy Hamed Pirsiavash

Outline

Recap: Decision Theory and ERM

Gradient Optimization

Linear Models & Surrogate Loss

- Example 1: Linear Regression
- Example 2: Linear Classification Model with Surrogate
- Example 3: Perceptrons

Loss Function Example: 0-1 Loss

$$\ell(y, \hat{y}) = \begin{cases} 0, & \text{if } y = \hat{y} \\ 1, & \text{if } y \neq \hat{y} \end{cases}$$

Problem: not differentiable wrt $\hat{y}$ (or θ)

Solution 1: is h(x) a conditional distribution p(y | x)? Use MAP

Solution 2: use a surrogate loss that approximates 0-1

Solution 3: is the data linearly separable? Perceptron can work

Ingredients for classification

Inject *your* knowledge into a learning system

Feature representation

Training data:
labeled examples

Model

Courtesy Hamed Pirsiavash

Ingredients for classification

Inject *your* knowledge into a learning system

Problem specific

Difficult to learn from bad ones

Feature representation

Training data: labeled examples

Model

Courtesy Hamed Pirsiavash

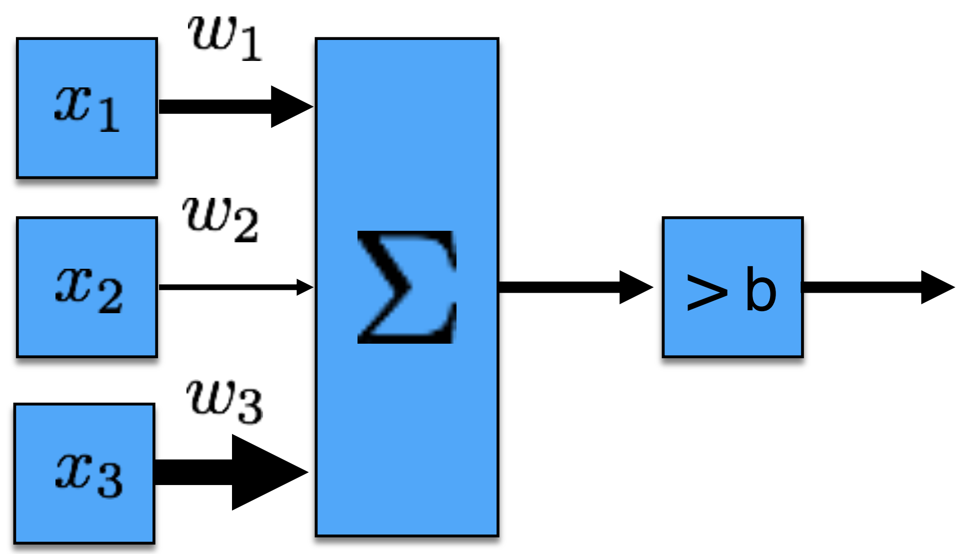
Perceptron

Inputs are feature values

Each feature has a weight

Sum in the activation

$$\text{activation}(\mathbf{w}, \mathbf{x}) = \sum_i w_i x_i = \mathbf{w}^T \mathbf{x}$$



If the activation is:

> b, output *class 1 ("positive")*

otherwise, output *class 2 ("negative")*

$$\mathbf{x} \to (\mathbf{x}, 1)$$
$$\mathbf{w}^T \mathbf{x} + b \to (\mathbf{w}, b)^T (\mathbf{x}, 1)$$

Slide courtesy Hamed Pirsiavash

Example: Document Classification

Electronic alerts have been used to assist the authorities in moments of chaos and potential danger: after the Boston bombing in 2013, when the Boston suspects were still at large, and last month in Los Angeles, during an active shooter scare at the airport.

TECH

NOT TECH

preprocessing/ feature extraction

word	count
alerts	1
assist	1
bombing	1
Boston	2
…	

x: “bag of words”

Example: Document Classification

Electronic alerts have been used to assist the authorities in moments of chaos and potential danger: after the Boston bombing in 2013, when the Boston suspects were still at large, and last month in Los Angeles, during an active shooter scare at the airport.

TECH

NOT TECH

feature	weight
alerts	.043
assist	-0.25
bombing	0.8
Boston	-0.00001
…	

w: weights

word	count
alerts	1
assist	1
bombing	1
Boston	2
…	

x: "bag of words"

Example: Document Classification

Electronic alerts have been used to assist the authorities in moments of chaos and potential danger: after the Boston bombing in 2013, when the Boston suspects were still at large, and last month in Los Angeles, during an active shooter scare at the airport.

TECH

NOT TECH

feature	weight
alerts	.043
assist	-0.25
bombing	0.8
Boston	-0.00001
…	

w: weights

T

word	count
alerts	1
assist	1
bombing	1
Boston	2
…	

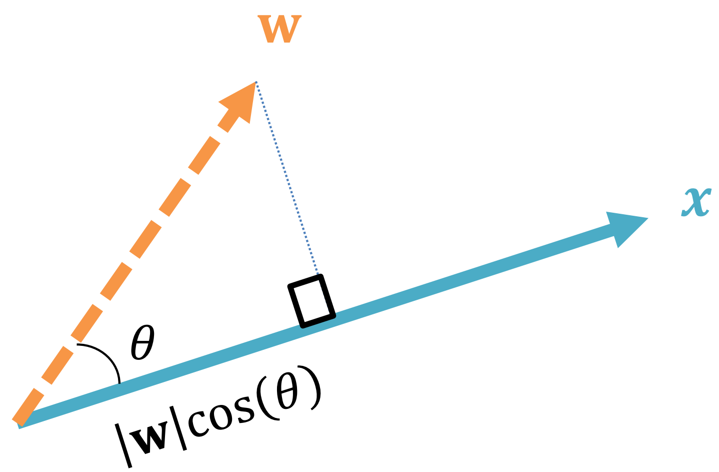
x: "bag of words"

$> b$

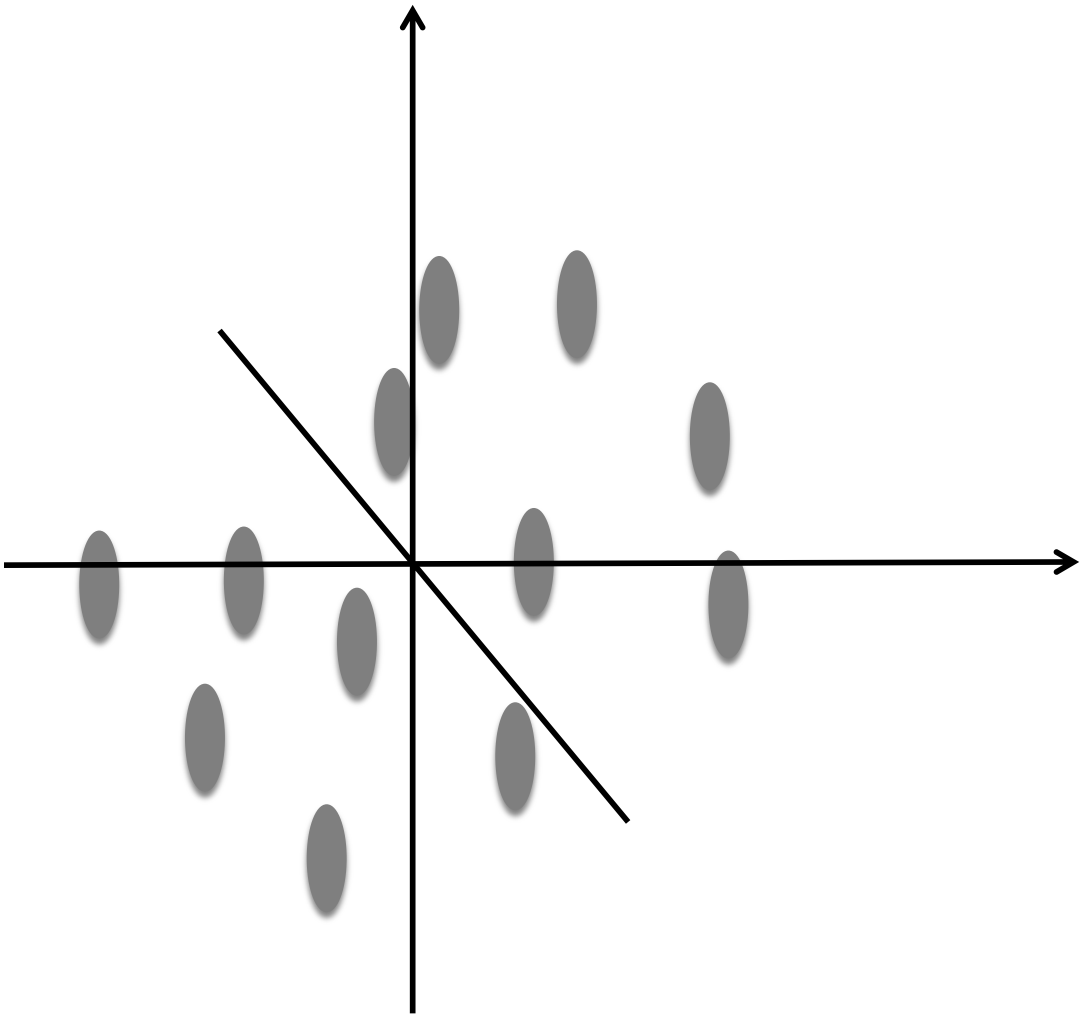
Perceptron action

Reminder: Dot Product & Cosine

$$\mathbf{w}^{\mathrm{T}}\boldsymbol{x} = |\mathbf{w}|\ |\boldsymbol{x}|\cos(\theta)$$



Geometry of the Perceptron

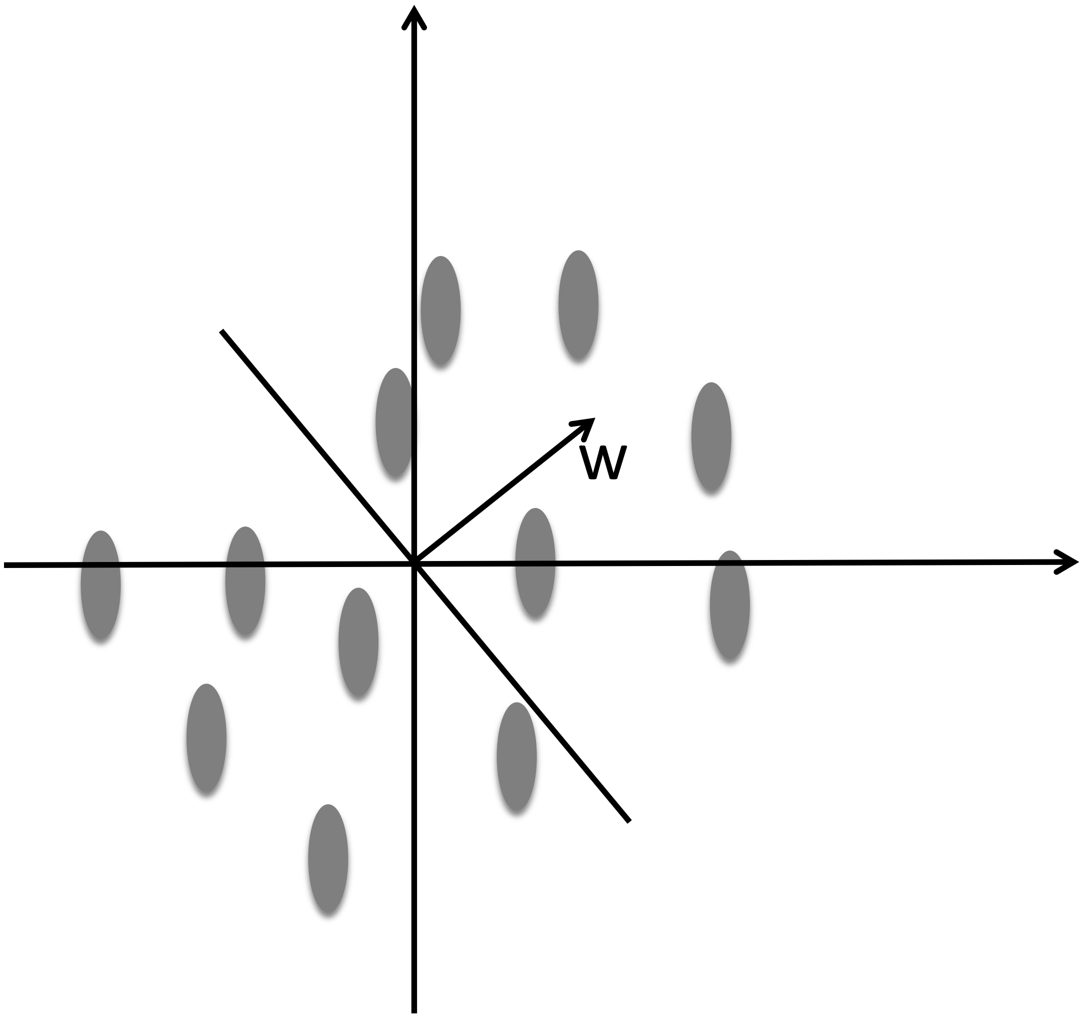


In the space of feature vectors

- examples are points (in D dimensions)

Adapted from Hamed Pirsiavash

Geometry of the Perceptron

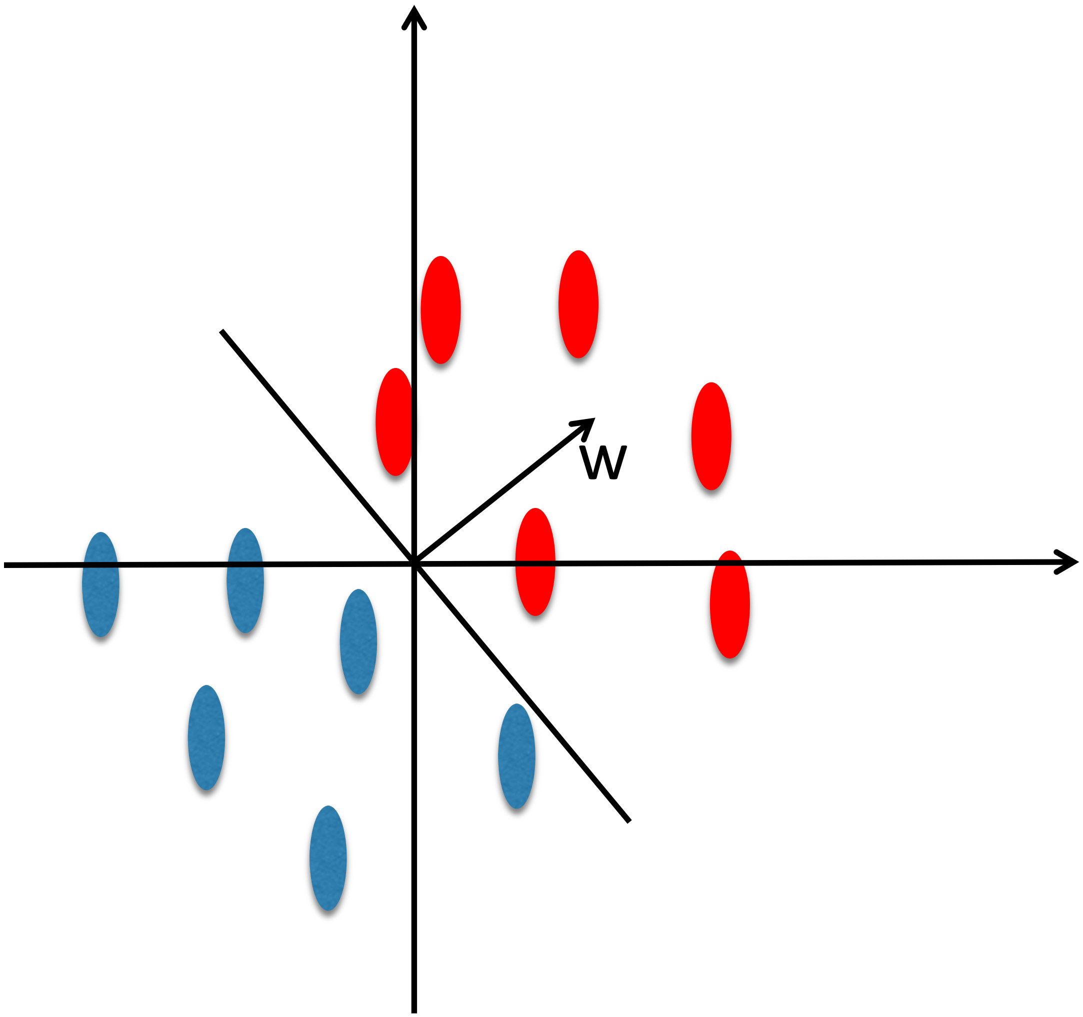


In the space of feature vectors

- examples are points (in D dimensions)
- a weight vector is a hyperplane (a D-1 dimensional object)

Adapted from Hamed Pirsiavash

Geometry of the Perceptron



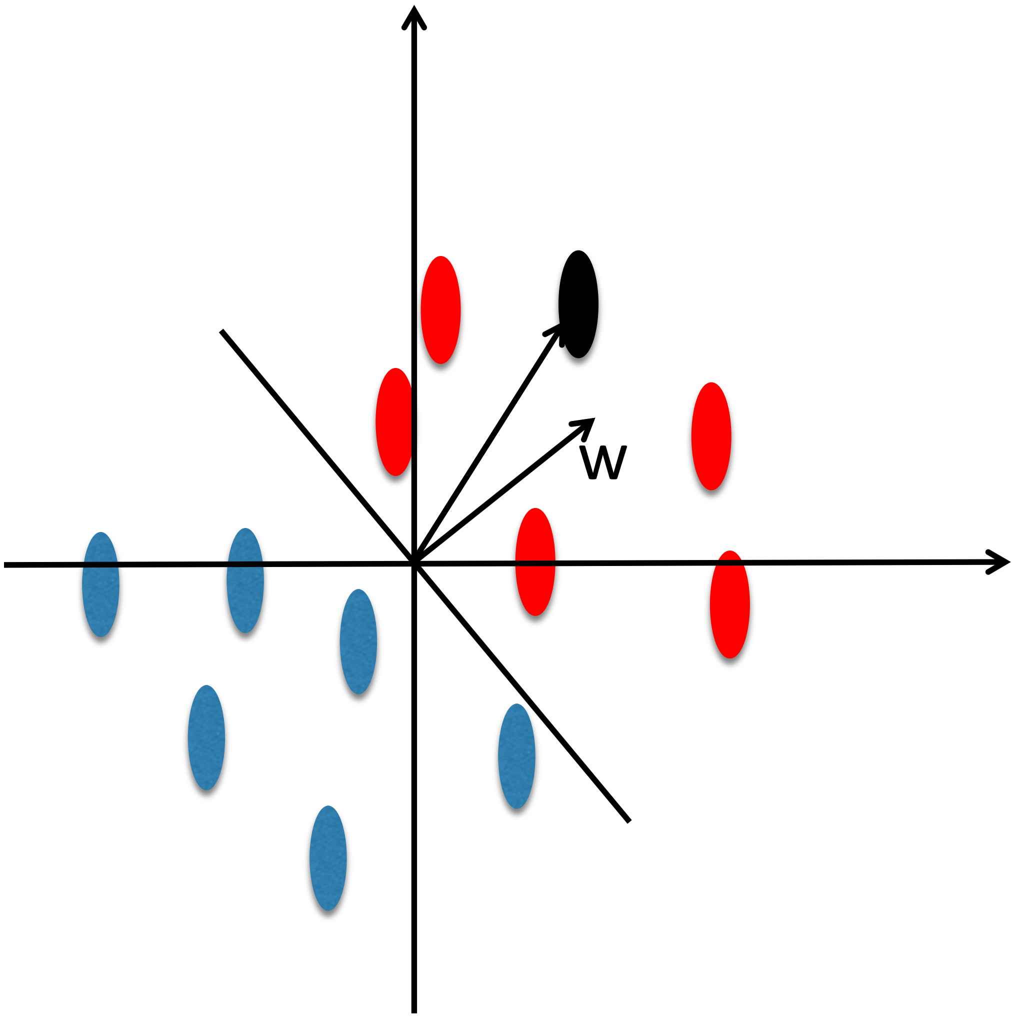
In the space of feature vectors

- examples are points (in D dimensions)
- a weight vector is a hyperplane (a D-1 dimensional object)
- One side corresponds to y=+1
- Other side corresponds to y=-1

$$\text{activation}(\mathbf{w}, \mathbf{x}) = \sum_i w_i x_i = \mathbf{w}^T \mathbf{x}$$

Adapted from Hamed Pirsiavash

Geometry of the Perceptron



In the space of feature vectors

- examples are points (in D dimensions)
- a weight vector is a hyperplane (a D-1 dimensional object)
- One side corresponds to y=+1
- Other side corresponds to y=-1

Perceptrons are also called as linear classifiers

$$\text{activation}(\mathbf{w}, \mathbf{x}) = \sum_i w_i x_i = \mathbf{w}^T \mathbf{x}$$

Adapted from Hamed Pirsiavash

Learning a Perceptron

Input: training data $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)$

Perceptron training algorithm (Rosenblatt, 57)

Initialize $\mathbf{w} \leftarrow [0, \ldots, 0]$

for iteration = 1,…,T

Learning a Perceptron

Input: training data $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)$

Perceptron training algorithm (Rosenblatt, 57)

Initialize $\mathbf{w} \leftarrow [0, \ldots, 0]$

for iteration = 1,…,T

for example i = 1,.., N

Slide courtesy Hamed Pirsiavash

Learning a Perceptron

Input: training data $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)$

Perceptron training algorithm (Rosenblatt, 57)

Initialize $\mathbf{w} \leftarrow [0, \ldots, 0]$

for iteration = 1,…,T

for example i = 1,.., N

predict according to the current model

$$\hat{y}_i = \begin{cases} +1 & \text{if } \mathbf{w}^T\mathbf{x}_i > 0 \\ -1 & \text{if } \mathbf{w}^T\mathbf{x}_i \leq 0 \end{cases}$$

Slide courtesy Hamed Pirsiavash

Learning a Perceptron

Input: training data $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)$

Perceptron training algorithm (Rosenblatt, 57)

Initialize $\mathbf{w} \leftarrow [0, \ldots, 0]$

for iteration = 1,…,T

for example i = 1,.., N

predict according to the current model

$$\hat{y}_i = \begin{cases} +1 & \text{if } \mathbf{w}^T \mathbf{x}_i > 0 \\ -1 & \text{if } \mathbf{w}^T \mathbf{x}_i \leq 0 \end{cases}$$

if $y_i = \hat{y}_i$, no change

else, $\mathbf{w} \leftarrow \mathbf{w} + y_i \mathbf{x}_i$

Slide courtesy Hamed Pirsiavash

Learning a Perceptron

Input: training data $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)$

Perceptron training algorithm (Rosenblatt, 57)

Initialize $\mathbf{w} \leftarrow [0, \ldots, 0]$

for iteration = 1,…,T

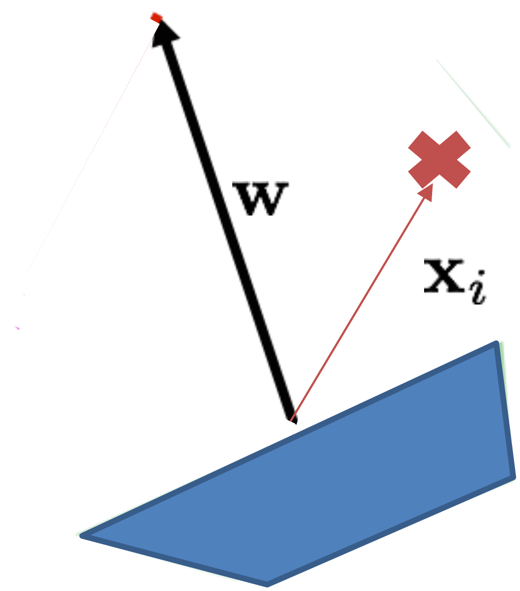
for example i = 1,.., N

predict according to the current model

$$\hat{y}_i = \begin{cases} +1 & \text{if } \mathbf{w}^T \mathbf{x}_i > 0 \\ -1 & \text{if } \mathbf{w}^T \mathbf{x}_i \leq 0 \end{cases}$$

if $y_i = \hat{y}_i$, no change

else, $\mathbf{w} \leftarrow \mathbf{w} + y_i \mathbf{x}_i$



$y_i = -1$

Slide courtesy Hamed Pirsiavash

Learning a Perceptron

Input: training data $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)$

Perceptron training algorithm (Rosenblatt, 57)

Initialize $\mathbf{w} \leftarrow [0, \ldots, 0]$

for iteration = 1,…,T

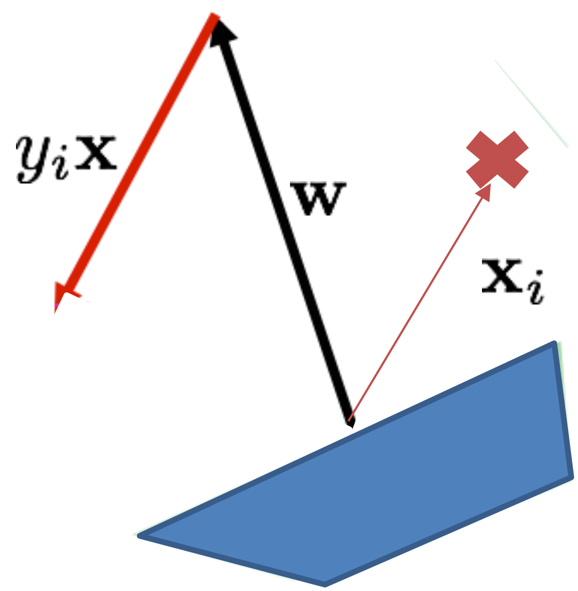
for example i = 1,.., N

predict according to the current model

$$\hat{y}_i = \begin{cases} +1 & \text{if } \mathbf{w}^T\mathbf{x}_i > 0 \\ -1 & \text{if } \mathbf{w}^T\mathbf{x}_i \leq 0 \end{cases}$$

if $y_i = \hat{y}_i$, no change

else, $\mathbf{w} \leftarrow \mathbf{w} + y_i\mathbf{x}_i$



$y_i = -1$

Learning a Perceptron

Input: training data $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)$

Perceptron training algorithm (Rosenblatt, 57)

Initialize $\mathbf{w} \leftarrow [0, \ldots, 0]$

for iteration = 1,…,T

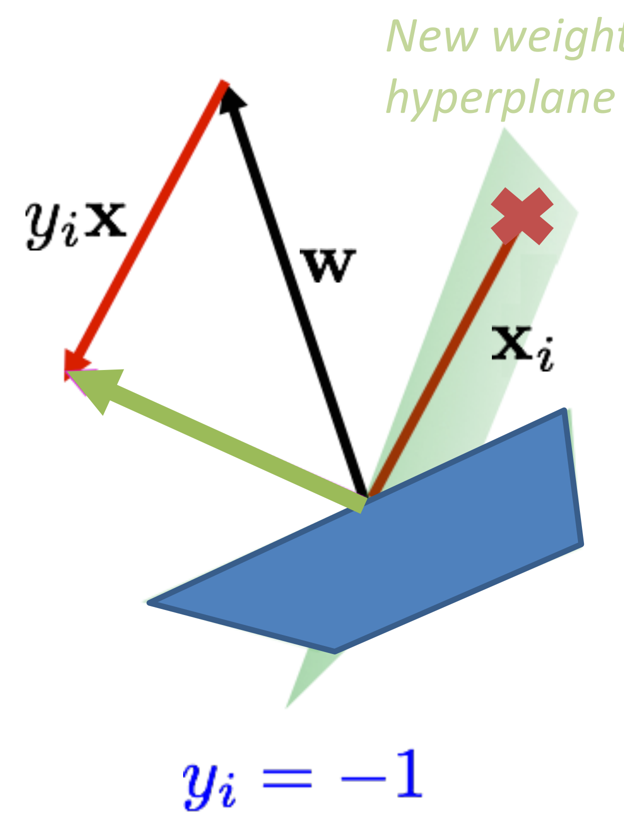
for example i = 1,.., N

predict according to the current model

$$\hat{y}_i = \begin{cases} +1 & \text{if } \mathbf{w}^T\mathbf{x}_i > 0 \\ -1 & \text{if } \mathbf{w}^T\mathbf{x}_i \leq 0 \end{cases}$$

if $y_i = \hat{y}_i$, no change

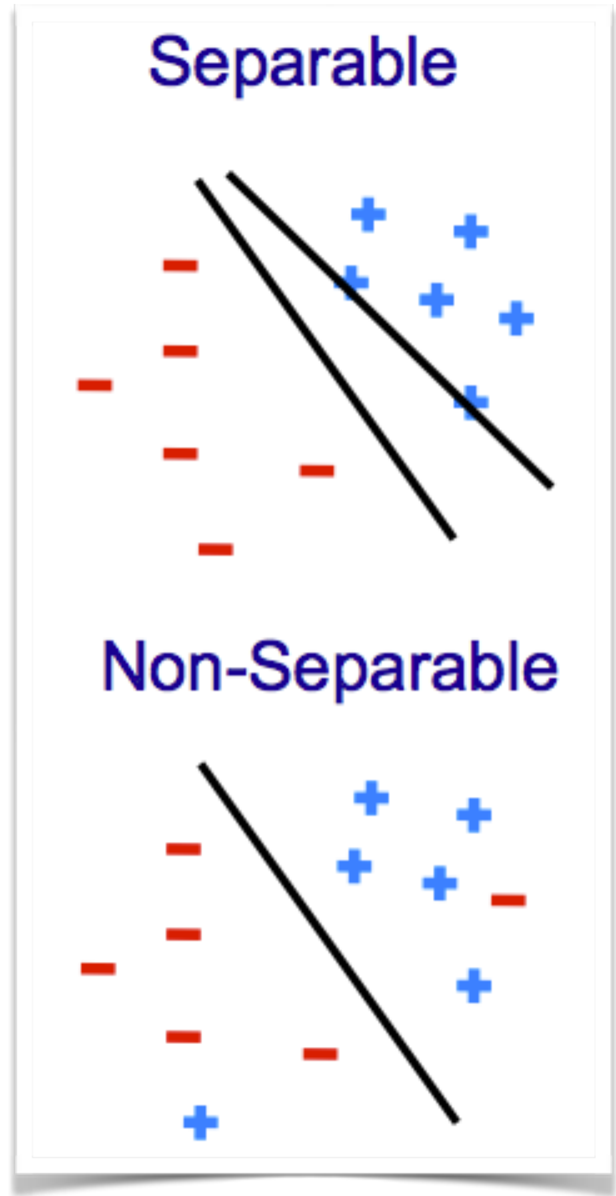
else, $\mathbf{w} \leftarrow \mathbf{w} + y_i\mathbf{x}_i$



Slide courtesy Hamed Pirsiavash

Properties of perceptrons

Separability: some parameters will classify the training data perfectly

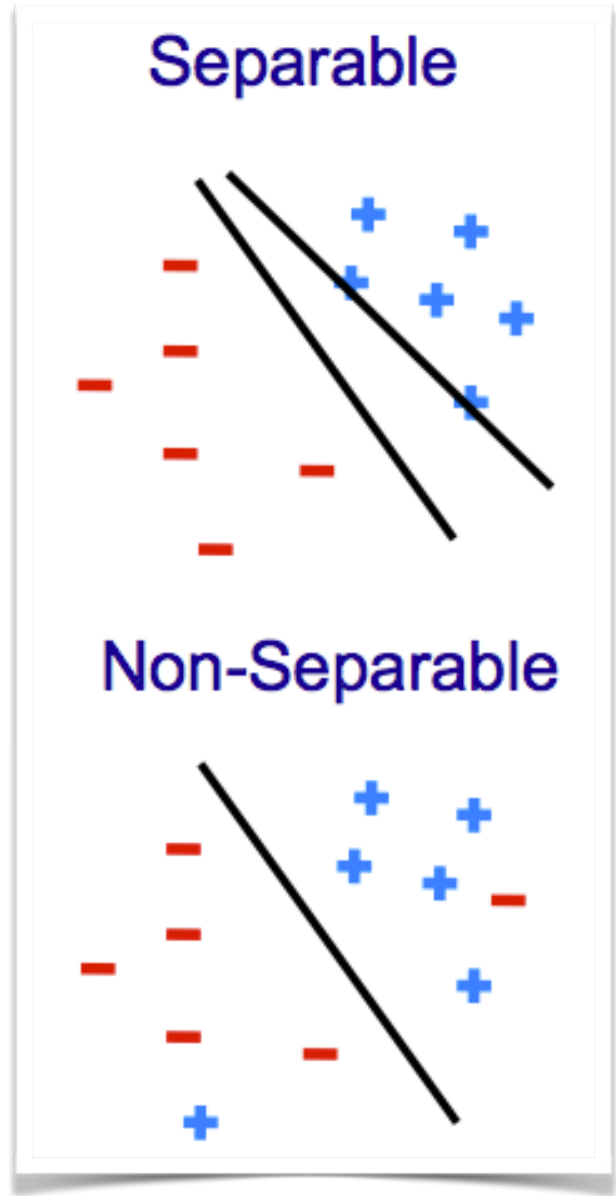


Slide courtesy Hamed Pirsiavash

Properties of perceptrons

Separability: some parameters will classify the training data perfectly

Convergence: if the training data is separable then the perceptron training will eventually converge



Slide courtesy Hamed Pirsiavash

Properties of perceptrons

Separability: some parameters will classify the training data perfectly

Convergence: if the training data is separable then the perceptron training will eventually converge

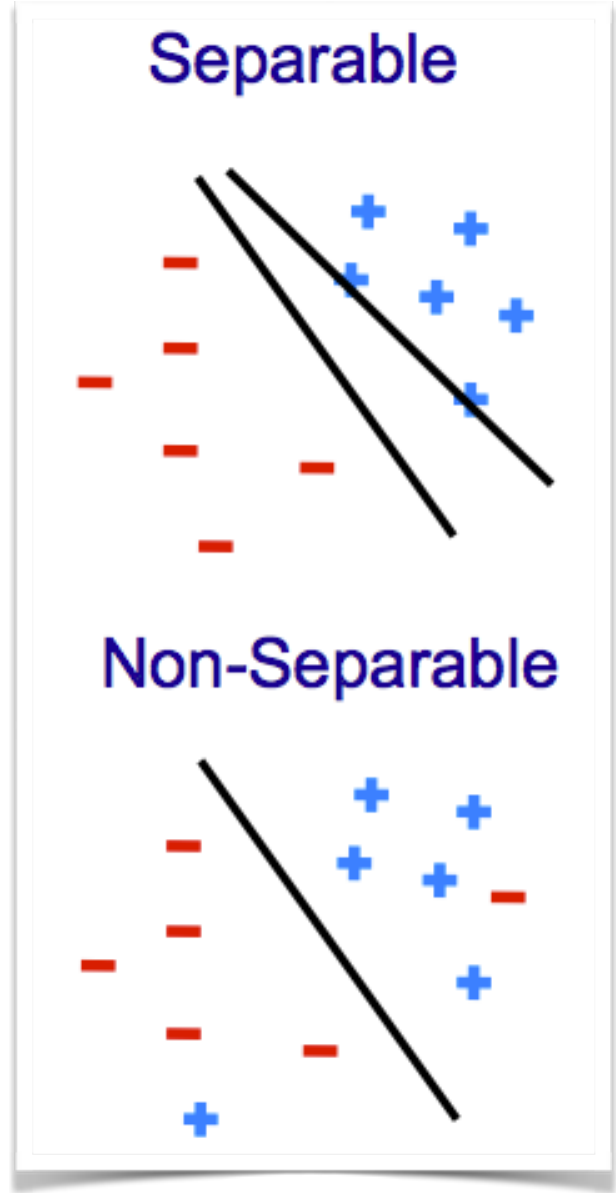
Mistake bound: the maximum number of mistakes is related to the **margin**

$$\text{assuming, } ||\mathbf{x}_i|| \leq 1$$

$$\#\text{mistakes} < \frac{1}{\delta^2}$$

$$\delta = \max_{\mathbf{w}} \min_{(\mathbf{x}_i, y_i)} \left[y_i \mathbf{w}^T \mathbf{x}_i \right]$$

$$\text{such that, } ||\mathbf{w}|| = 1$$



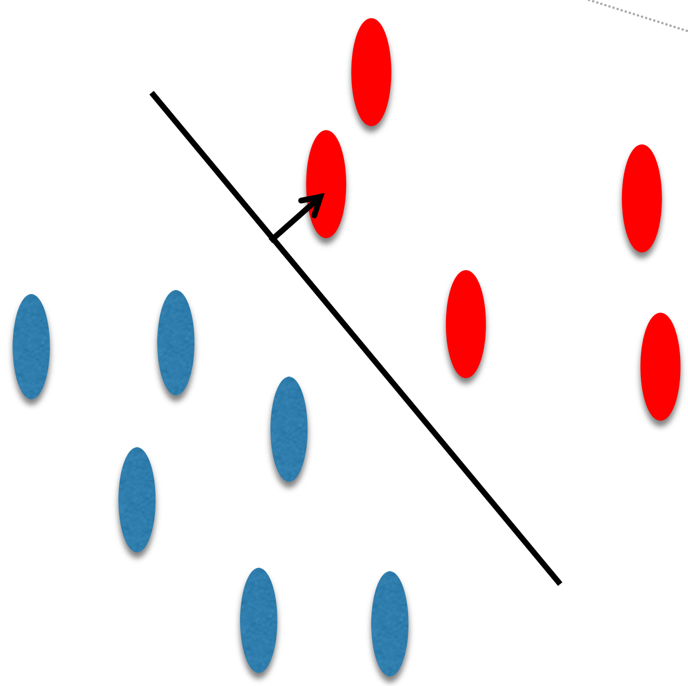
Slide courtesy Hamed Pirsiavash

Margin

$$margin(\mathbf{D}, \boldsymbol{w}, b) = \begin{cases} \min_{(x,y) \in \mathbf{D}} y(\boldsymbol{w} \cdot \boldsymbol{x} + b) & \text{if } \boldsymbol{w} \text{ separates } \mathbf{D} \\ -\infty & \text{otherwise} \end{cases}$$

$$margin(\mathbf{D}) = \sup_{\boldsymbol{w}, b} margin(\mathbf{D}, \boldsymbol{w}, b)$$

(mathematically correct “max”)



Slide courtesy Hamed Pirsiavash

Proof of convergence

separating hyperplane with margin=δ

$$\hat{\mathbf{w}}^T\mathbf{w}^{(k)} = \hat{\mathbf{w}}^T\left(\mathbf{w}^{(k-1)} + y_i\mathbf{x}_i\right)$$ update rule

w is getting closer

$$= \hat{\mathbf{w}}^T\mathbf{w}^{(k-1)} + \hat{\mathbf{w}}^T y_i\mathbf{x}_i$$ algebra

$$\geq \hat{\mathbf{w}}^T\mathbf{w}^{(k-1)} + \delta$$ definition of margin

$$\geq k\delta$$

$$||\mathbf{w}^{(k)}|| \geq k\delta$$

$$||\mathbf{w}^{(k)}||^2 = ||\mathbf{w}^{(k-1)} + y_i\mathbf{x}_i||^2$$ update rule

bound the norm

$$\leq ||\mathbf{w}^{(k-1)}||^2 + ||y_i\mathbf{x}_i||^2$$ triangle inequality

$$\leq ||\mathbf{w}^{(k-1)}||^2 + 1$$ norm

$$\leq k$$

$$||\mathbf{w}^{(k)}|| \leq \sqrt{k}$$

$$k\delta \leq ||\mathbf{w}^{(k)}|| \leq \sqrt{k} \longrightarrow k \leq \frac{1}{\delta^2}$$

Limitations of Perceptrons

Convergence: if the data isn't separable, the training algorithm may not terminate

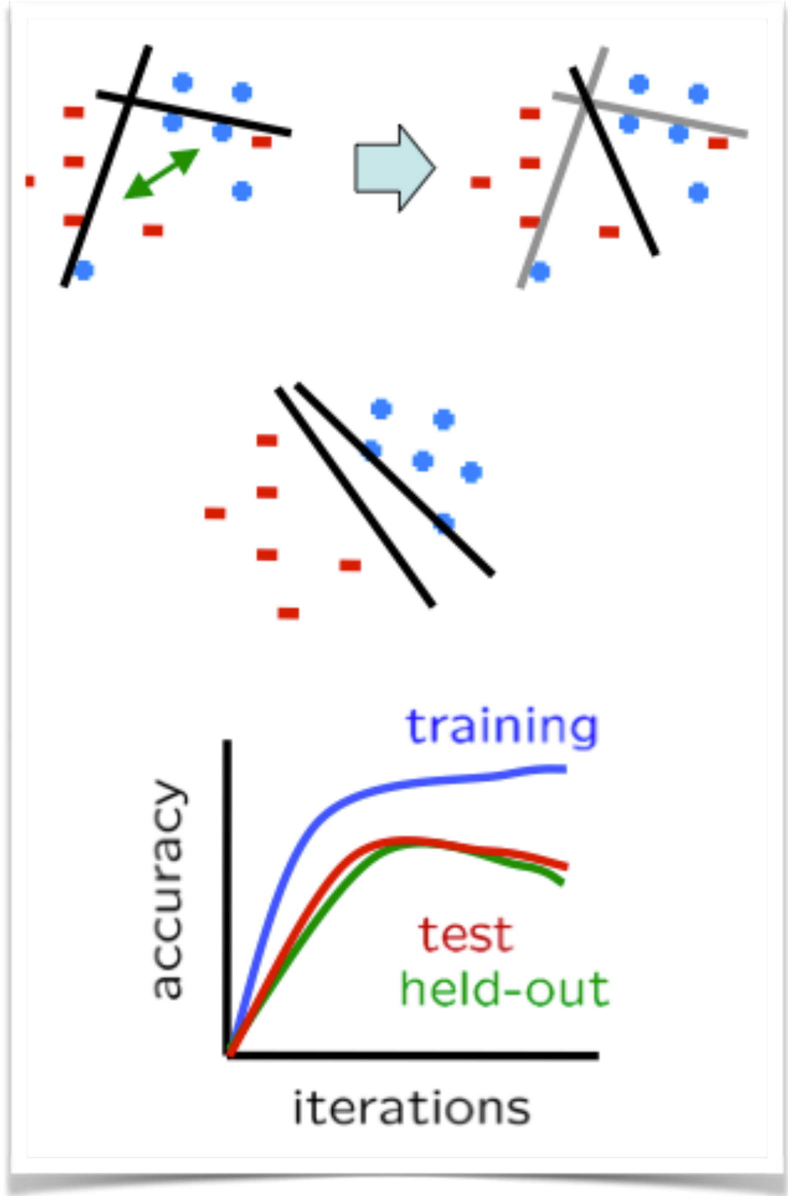
noise can cause this

data inherently not separable

Mediocre generalization: the algorithm finds a solution that "barely" separates the data

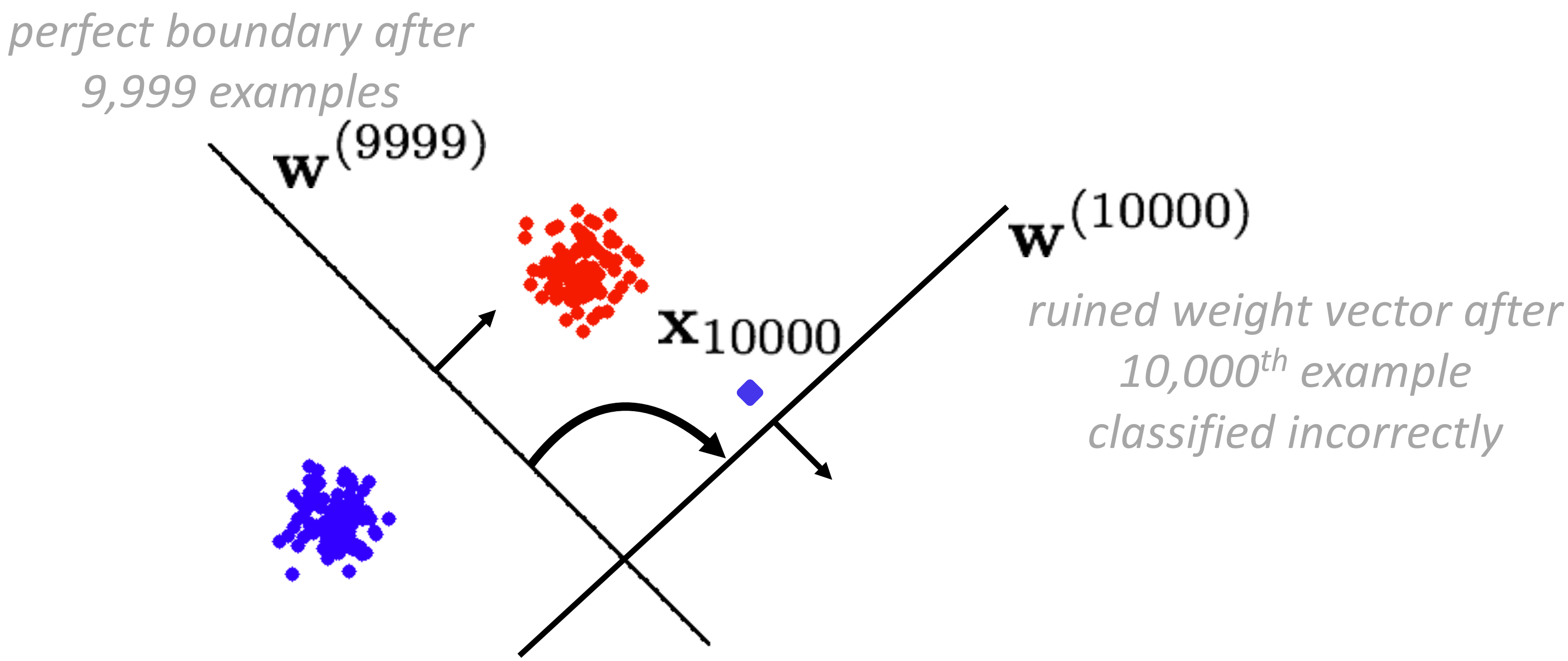
Overtraining: test/validation accuracy rises and then falls

Overly greedy updates: susceptible to most-recent inaccuracies



Slide courtesy Hamed Pirsiavash

Problem: Late Misclassifications



Solution 1: Voted Perceptron

Solution 2: Averaged Perceptron

Slide courtesy Hamed Pirsiavash

Voted perceptron

Key idea: remember how long each weight vector survives

$\mathbf{w}^{(1)}, \mathbf{w}^{(2)}, \ldots, \mathbf{w}^{(K)}$ *sequence* of weights

$c^{(1)}, c^{(2)}, \ldots, c^{(K)}$ "*survival*" times for each of these (# iterations since last update)

a weight that gets updated immediately gets *c* = 1

a weight that survives another round gets *c* = 2, etc.

$$\hat{y} = \text{sign}\left(\sum_{k=1}^{K} c^{(k)} \text{sign}\left(\mathbf{w}^{(k)T}\mathbf{x}\right)\right)$$

Q: What's potentially problematic here?

Slide courtesy Hamed Pirsiavash

Voted perceptron

Key idea: remember how long each weight vector survives

$\mathbf{w}^{(1)}, \mathbf{w}^{(2)}, \ldots, \mathbf{w}^{(K)}$ *sequence* of weights

$c^{(1)}, c^{(2)}, \ldots, c^{(K)}$ "*survival*" times for each of these (# iterations since last update)

a weight that gets updated immediately gets $c = 1$

a weight that survives another round gets $c = 2$, etc.

$$\hat{y} = \text{sign}\left(\sum_{k=1}^{K} c^{(k)} \text{sign}\left(\mathbf{w}^{(k)T}\mathbf{x}\right)\right)$$

Q: What's potentially problematic here?

A: Large memory requirements

Slide courtesy Hamed Pirsiavash

Averaged perceptron

$\mathbf{w}^{(1)}, \mathbf{w}^{(2)}, \ldots, \mathbf{w}^{(K)}$ *sequence* of weights

$c^{(1)}, c^{(2)}, \ldots, c^{(K)}$ "*survival*" times for each of these (# iterations since last update)

a weight that gets updated immediately gets $c = 1$

a weight that survives another round gets $c = 2$, etc.

voted $$\hat{y} = \text{sign}\left(\sum_{k=1}^{K} c^{(k)} \text{sign}\left(\mathbf{w}^{(k)T}\mathbf{x}\right)\right)$$

$$\downarrow$$

averaged $$\hat{y} = \text{sign}\left(\sum_{k=1}^{K} c^{(k)} \left(\mathbf{w}^{(k)T}\mathbf{x}\right)\right) = \text{sign}\left(\bar{\mathbf{w}}^{T}\mathbf{x}\right)$$

Slide courtesy Hamed Pirsiavash

Averaged Perceptron Training Algorithm

Initialize: $c = 0, \mathbf{w} = [0, \ldots, 0], \bar{\mathbf{w}} = [0, \ldots, 0]$

for iter = 1,…,T

for i = 1,..,n

predict according to the current model

$$\hat{y}_i = \begin{cases} +1 & \text{if } \mathbf{w}^T\mathbf{x}_i > 0 \\ -1 & \text{if } \mathbf{w}^T\mathbf{x}_i \leq 0 \end{cases}$$

if $y_i = \hat{y}_i$: $c = c + 1$

else:

$\bar{\mathbf{w}} \leftarrow \bar{\mathbf{w}} + c\mathbf{w}$

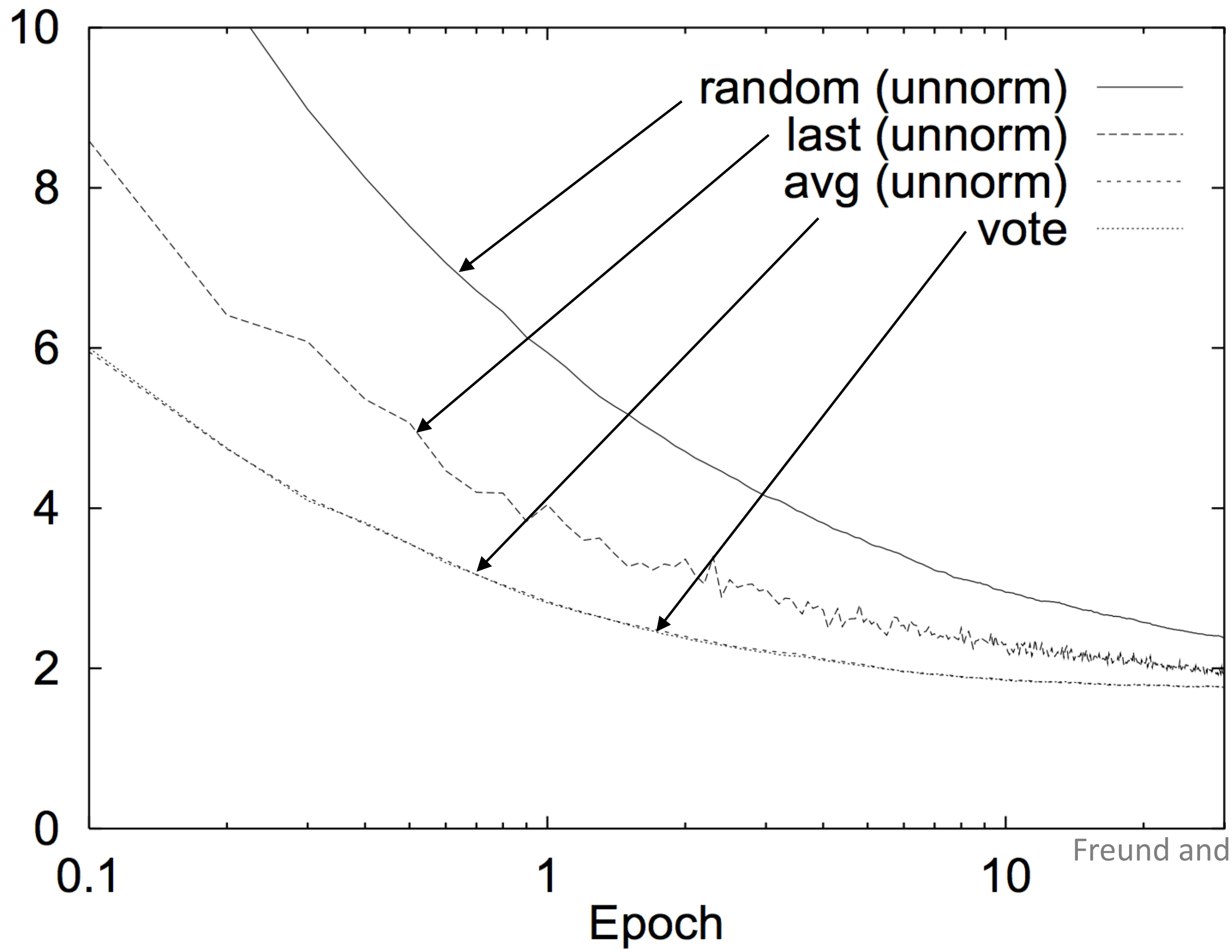
$\mathbf{w} \leftarrow \mathbf{w} + y_i\mathbf{x}_i$

$c = 1$

return $\bar{\mathbf{w}} + c\mathbf{w}$

Slide courtesy Hamed Pirsiavash

Comparison of perceptron variants: MNIST classification



Freund and Schapire (1999)

Slide courtesy Hamed Pirsiavash

Improving Perceptrons

Multilayer perceptrons: non-linear functions of the input (neural networks)

Feature-mapping: using kernels

Margin-based classifiers: improves generalization ability of the classifier (support vector machines, SVMs)

Adapted from Hamed Pirsiavash

Outline

Recap: Decision Theory and ERM

Gradient Optimization

Linear Models & Surrogate Loss

- Example 1: Linear Regression
- Example 2: Linear Classification Model
- Example 3: Perceptrons