CS57300
PURDUE UNIVERSITY

OCTOBER 6, 2021

DATA MINING

ANNOUNCEMENT

- Assignment 3 will be out today
  - Due time: October 24, 2021, 11:59pm
  - Start early!

SMOOTH OPTIMIZATION

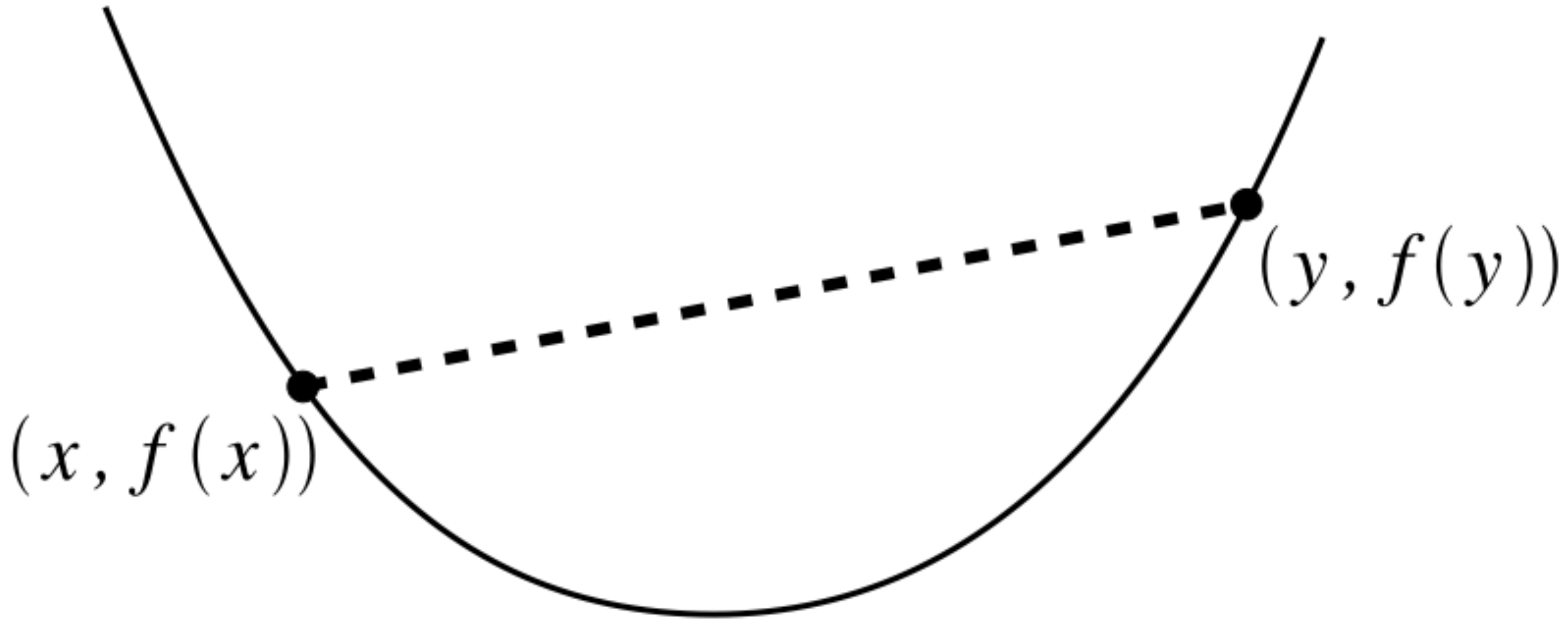
CONVEX OPTIMIZATION PROBLEMS

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in C \end{array}$$

- x is the optimization variable (*e.g., model parameters*)
  f (*e.g., score function*) is a **convex function**
  C is a **convex set** (*e.g., constraints on model parameters*)
- For convex optimization problems, all locally optimal points are globally optimal

CONVEX FUNCTIONS

- In graph of convex function *f*, the line connecting two points must lie above the function: $f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha) f(y)$ for all $0 \leq \alpha \leq 1$
- Practical test for convexity: a twice differentiable function *f* of a variable *x* is convex on an interval if an only if for any *x* in the interval: $f''(x) \geq 0$
  - Strictly convex if $f''(x) > 0$



- Sum of convex functions is convex; max of convex functions is convex

SOLVE CONVEX OPTIMIZATION PROBLEM

- Minimize a convex function without any constraints on the variables
  - If $f'(x)=0$ then x is a stationary point of f
  - If $f'(x)=0$ and $f''(x)$ is not negative then x is a local minimum of f (for convex function, this is also a global minimum)
  - If f is a strictly convex function, any stationary point of ***f*** is the unique global minimum of f
- What about minimizing a convex function with constraints?

USE LAGRANGE MULTIPLIERS TO SOLVE CONVEX OPTIMIZATION

- For a standard form of convex optimization problem (f_0 are f_i are convex, h_i is linear):

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \le 0, \quad \text{for } i = 1, \dots, m. \\ & h_i(x) = 0, \quad \text{for } i = 1, \dots, k. \end{array}$$

- The Lagrangian function of it is

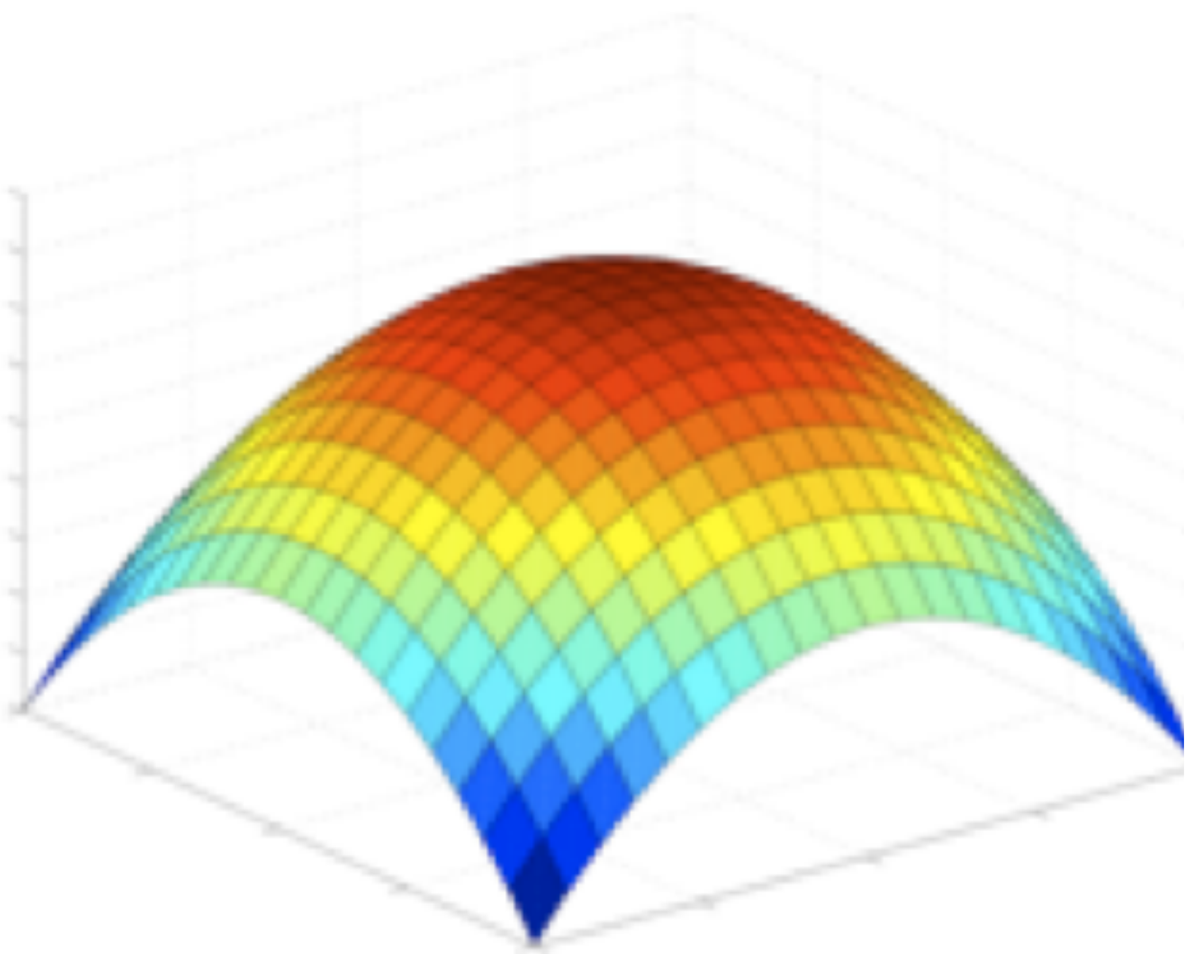
$$L(x, \lambda, v) = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{k} v_i h_i(x)$$

  - $\lambda_i \ge 0$ is the **Lagrange multiplier** for the *i*-th inequality constraint, v_i is the **Lagrange multiplier** for the *i*-th equality constraint
  - Solve the constrained optimization problem by finding the stationary point of the Lagrangian function

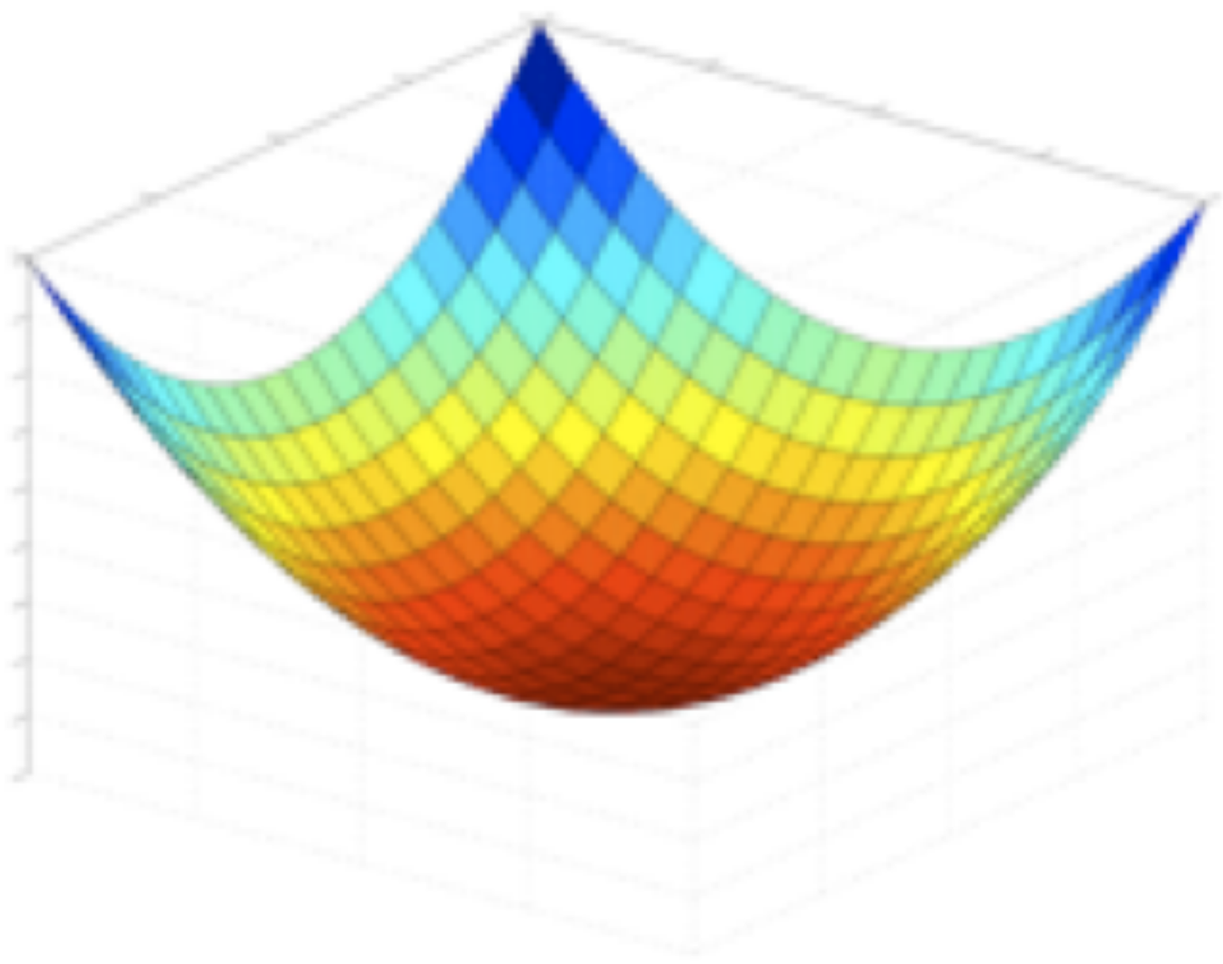
CONCAVE VS CONVEX

- Maximizing a concave function is equivalent to minimizing a convex function

concave



convex



NBC LEARNING REVISIT

- Maximize the log likelihood function
  - Likelihood: $L(\theta|D) = \prod_{i=1}^{n}\prod_{j=1}^{m} P(x_{ij}|c_i)P(c_i) = (\prod_{l=1}^{L} p_l^{N_l})(\prod_{l=1}^{L}\prod_{j=1}^{m}\prod_{k=1}^{K(j)} (q_l^{jk})^{N_l^{jk}})$
  - Log Likelihood: $logL(\theta|D) = \sum_{l=1}^{L} N_l log(p_l) + \sum_{l=1}^{L}\sum_{j=1}^{m}\sum_{k=1}^{K(j)} N_l^{jk} log q_l^{jk}$
  - Subject to constraints: $\sum_{l=1}^{L} p_l = 1, \sum_{k=1}^{K(j)} q_l^{jk} = 1$
- Lagrangian function:

$$L(p_l, q_l^{jk}, v_0, v_{lj}) = \sum_{l=1}^{L} N_l log(p_l) + \sum_{l=1}^{L}\sum_{j=1}^{m}\sum_{k=1}^{K(j)} N_l^{jk} log q_l^{jk} + v_0(\sum_{l=1}^{L} p_l - 1) + \sum_{l=1}^{L}\sum_{j=1}^{m} v_{lj}(\sum_{k=1}^{K(j)} q_l^{jk} - 1)$$

NBC LEARNING REVISIT

$$L(p_l, q_l^{jk}, v_0, v_{lj}) = \sum_{l=1}^{L} N_l log(p_l) + \sum_{l=1}^{L}\sum_{j=1}^{m}\sum_{k=1}^{K(j)} N_l^{jk} log q_l^{jk} + v_0(\sum_{l=1}^{L} p_l - 1) + \sum_{l=1}^{L}\sum_{j=1}^{m} v_{lj}(\sum_{k=1}^{K(j)} q_l^{jk} - 1)$$

$$\Downarrow$$

$$\frac{N_l}{p_l} + v_0 = 0, \frac{N_l^{jk}}{q_l^{jk}} + v_{lj} = 0$$

$$\Downarrow$$

$$p_l = -\frac{N_l}{v_0}, q_l^{jk} = -\frac{N_l^{jk}}{v_{lj}} \qquad + \quad \sum_{l=1}^{L} p_l = 1, \sum_{k=1}^{K(j)} q_l^{jk} = 1$$

$$\Downarrow$$

$$p_l = \frac{N_l}{N}, q_l^{jk} = \frac{N_l^{jk}}{N_l}$$

LOGISTIC REGRESSION LEARNING

- Logistic regression: $P(y=1|\mathbf{x}) = \dfrac{1}{1+e^{-(\mathbf{w}^T\mathbf{x}+w_0)}}$
  - Maximize (log) likelihood: $\mathsf{w} = (\mathsf{w}, w_0), \mathsf{x_i} = (\mathsf{x_i}, 1)$

$$
\begin{aligned}
logL(\mathsf{w}|D) &= \sum_{i=1}^{N} logp(y_i|\mathsf{w}) \\
&= \sum_{i=1}^{N} log[(\frac{1}{1+e^{-\mathsf{w}^T\mathsf{x_i}}})^{y_i}(\frac{e^{-\mathsf{w}^T\mathsf{x_i}}}{1+e^{-\mathsf{w}^T\mathsf{x_i}}})^{1-y_i}] \\
&= \sum_{i=1}^{N} (y_i\mathsf{w}^T\mathsf{x_i} - log(1+e^{\mathsf{w}^T\mathsf{x_i}}))
\end{aligned}
$$

  - Minimize: $\sum_{i=1}^{N} (-y_i\mathsf{w}^T\mathsf{x_i} + log(1+e^{\mathsf{w}^T\mathsf{x_i}}))$

LOGISTIC REGRESSION LEARNING

$$minimize \sum_{i=1}^{N} (-y_i \mathbf{w}^T \mathbf{x_i} + log(1 + e^{\mathbf{w}^T \mathbf{x_i}}))$$

$$\frac{dlogL(\mathbf{w} \mid D)}{dw_j} = \sum_{i=1}^{N} (-y_i x_{ij} + \frac{1}{1 + e^{\mathbf{w}^T \mathbf{x_i}}} e^{\mathbf{w}^T \mathbf{x_i}} x_{ij})$$

$$= \sum_{i=1}^{N} (-y_i + \frac{1}{1 + e^{\mathbf{w}^T \mathbf{x_i}}} e^{\mathbf{w}^T \mathbf{x_i}}) x_{ij}$$

$$= \sum_{i=1}^{N} (-y_i + P(y_i = 1 \mid \mathbf{w})) x_{ij}$$

Convex!

But no closed form solution!

GRADIENT DESCENT

- For some convex functions, we may be able to take the derivative, but it may be difficult to directly solve for parameter values
- Solution:
  - Start at some value of the parameters
  - Take derivative and use it to move the parameters in the direction of the negative gradient
  - Repeat until stopping criteria is met (e.g., gradient close to 0)

<u>Gradient Descent Rule:</u>

$$\underline{\mathbf{w}}_{\mathbf{new}} = \underline{\mathbf{w}}_{\mathbf{old}} - \eta \; \Delta(\underline{\mathbf{w}})$$

where

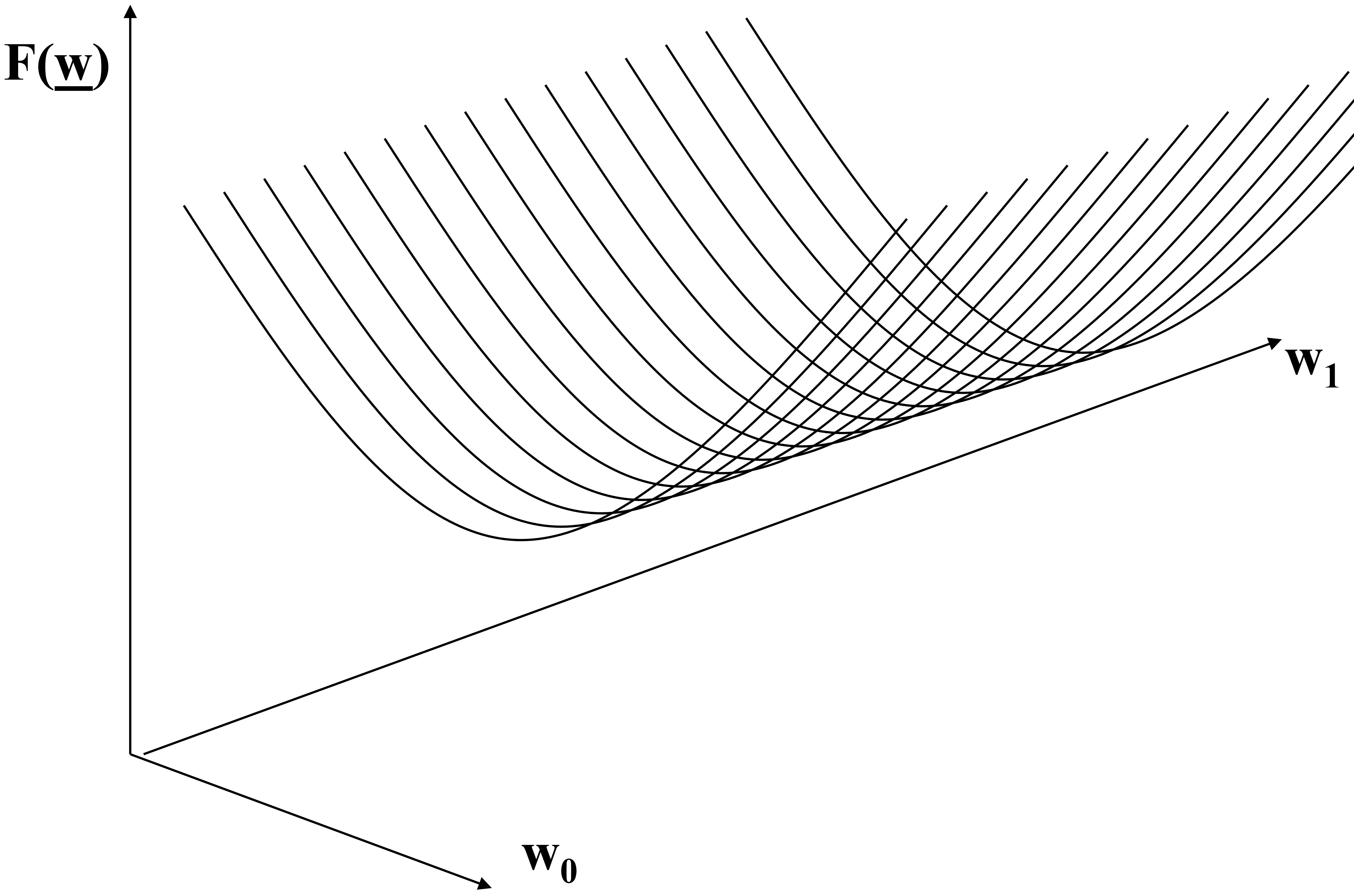
$\Delta(\underline{w})$ is the gradient and

η is the learning rate (small, positive)

Notes:

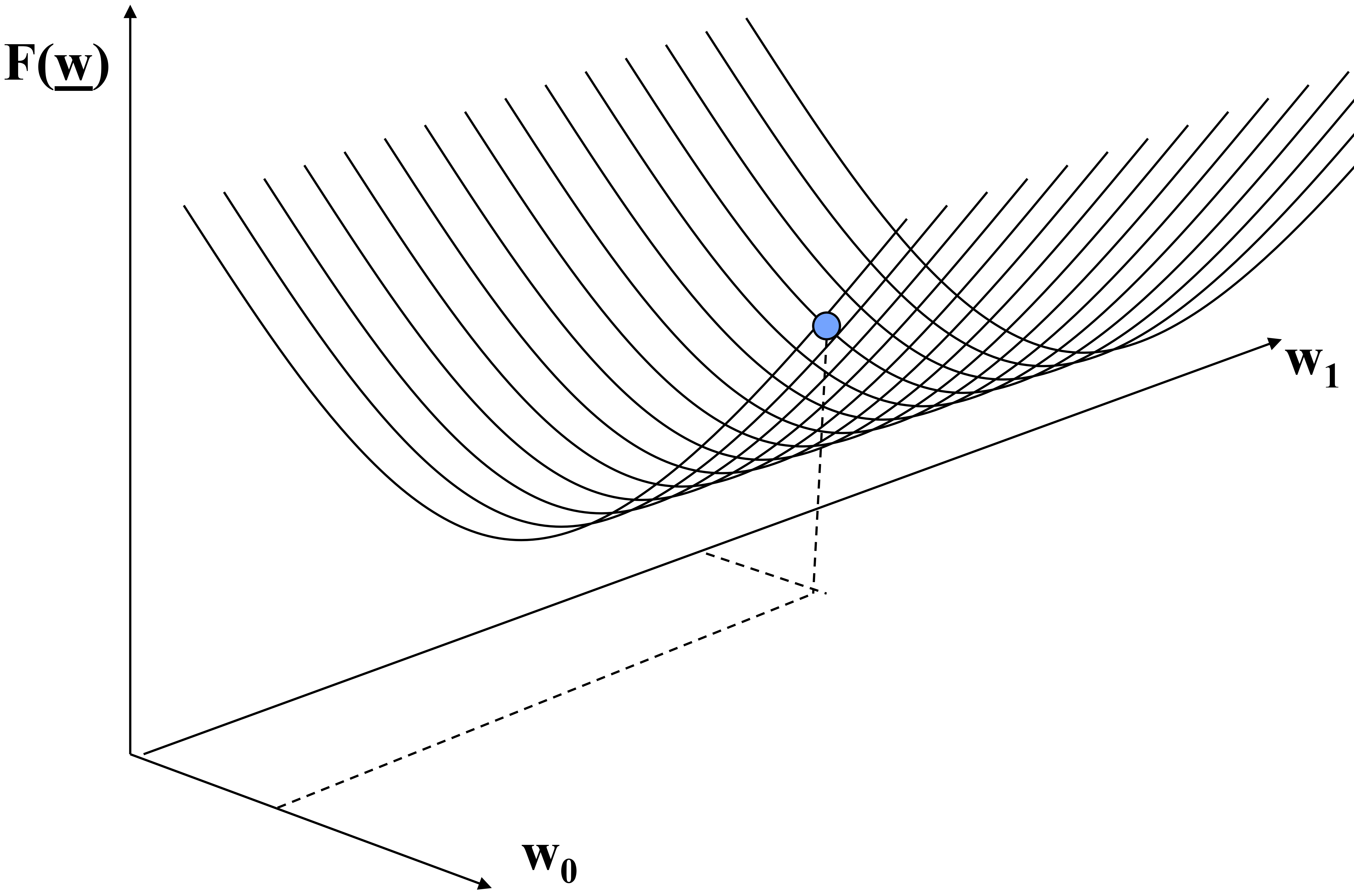
1. This moves us downhill in direction $\Delta(\underline{\mathbf{w}})$ (steepest downhill direction)
2. How far we go is determined by the value of $\boldsymbol{\eta}$

ILLUSTRATION OF GRADIENT DESCENT



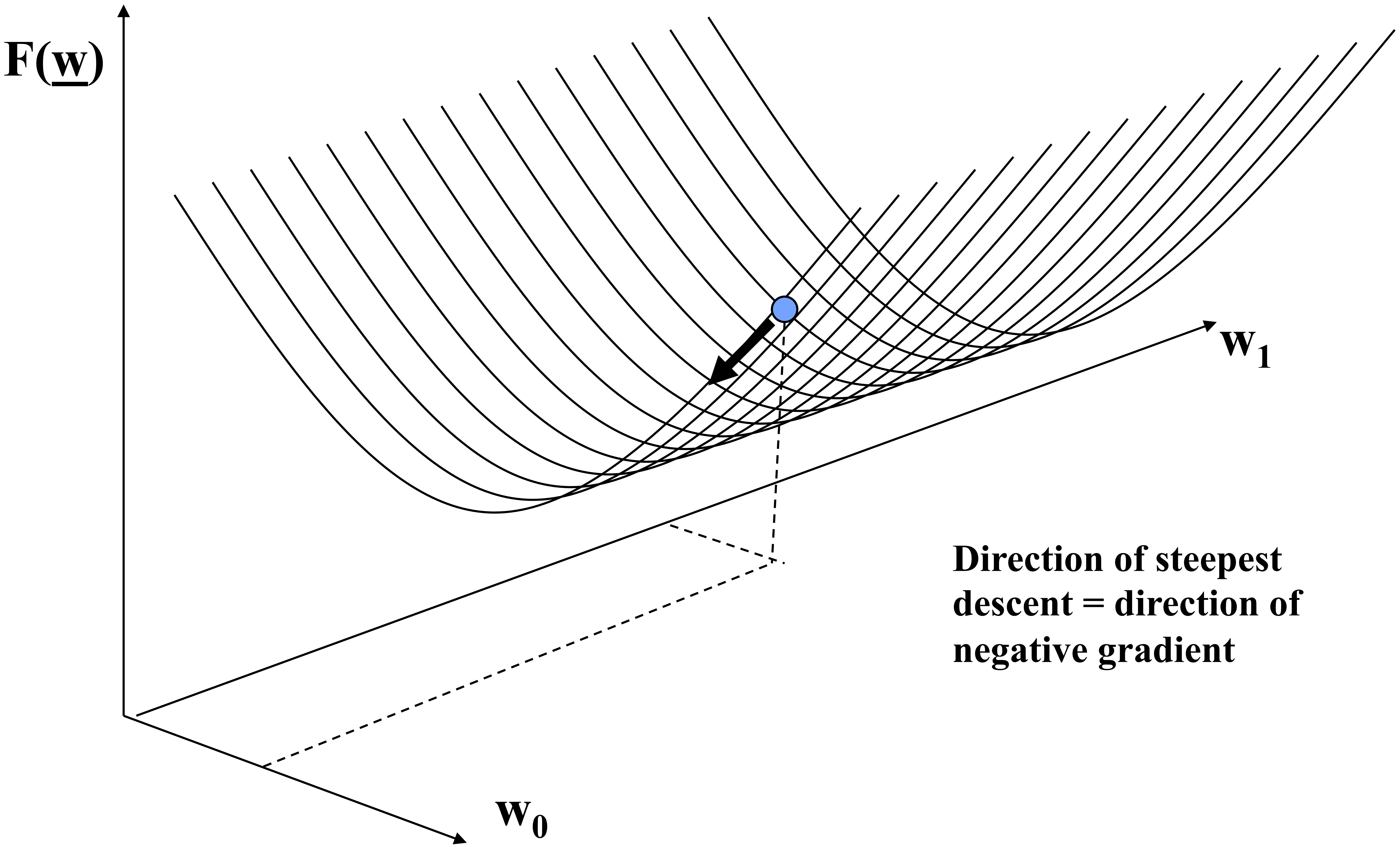
Slides adapted from CS175, UCIrvine, Padhraic Smyth

ILLUSTRATION OF GRADIENT DESCENT



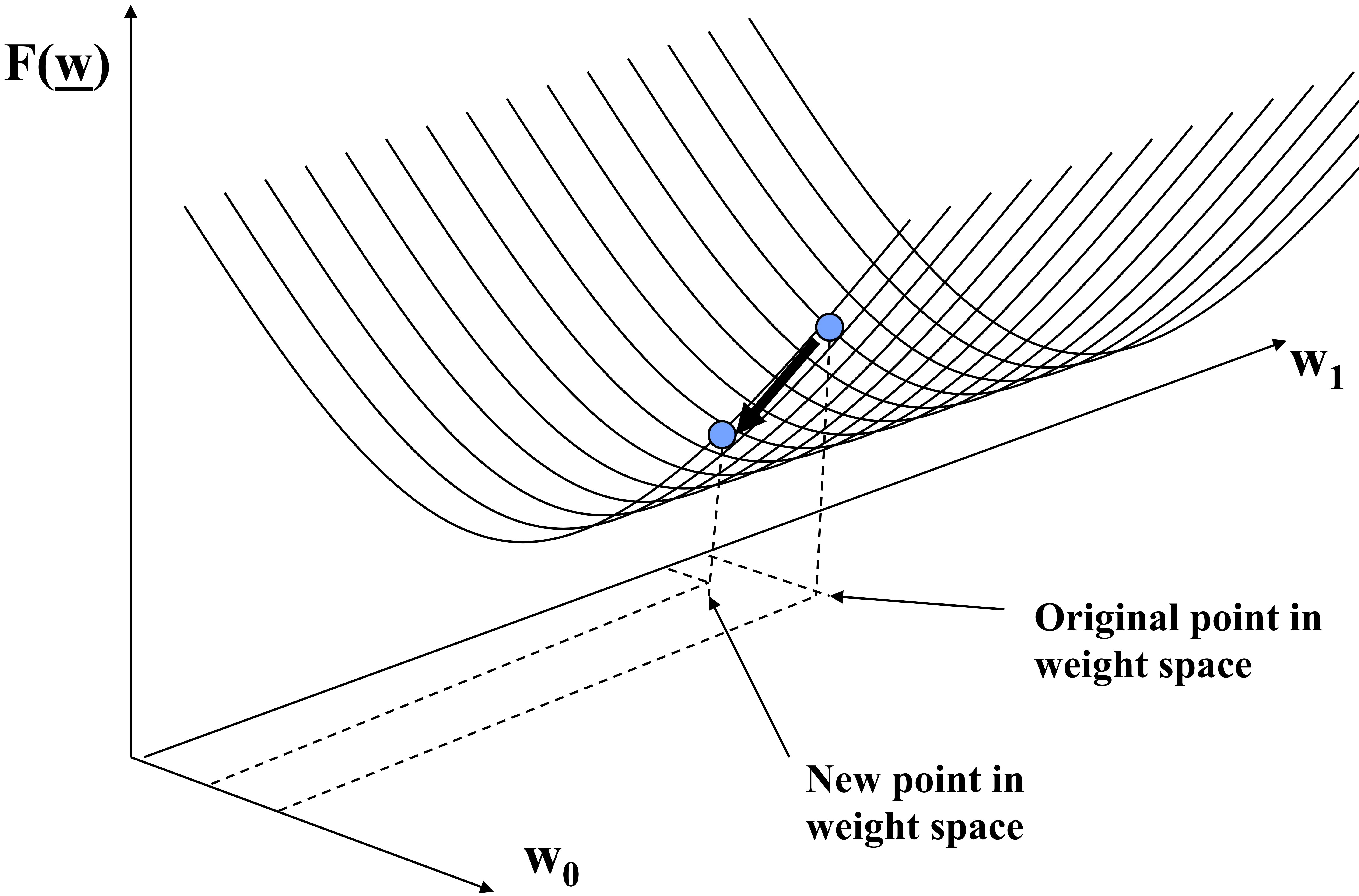
Slides adapted from CS175, UCIrvine, Padhraic Smyth

ILLUSTRATION OF GRADIENT DESCENT



Direction of steepest descent = direction of negative gradient

ILLUSTRATION OF GRADIENT DESCENT



For convex functions, when gradient descent converges, the solution is global minimum.

STOPPING CRITERIA FOR GRADIENT DESCENT

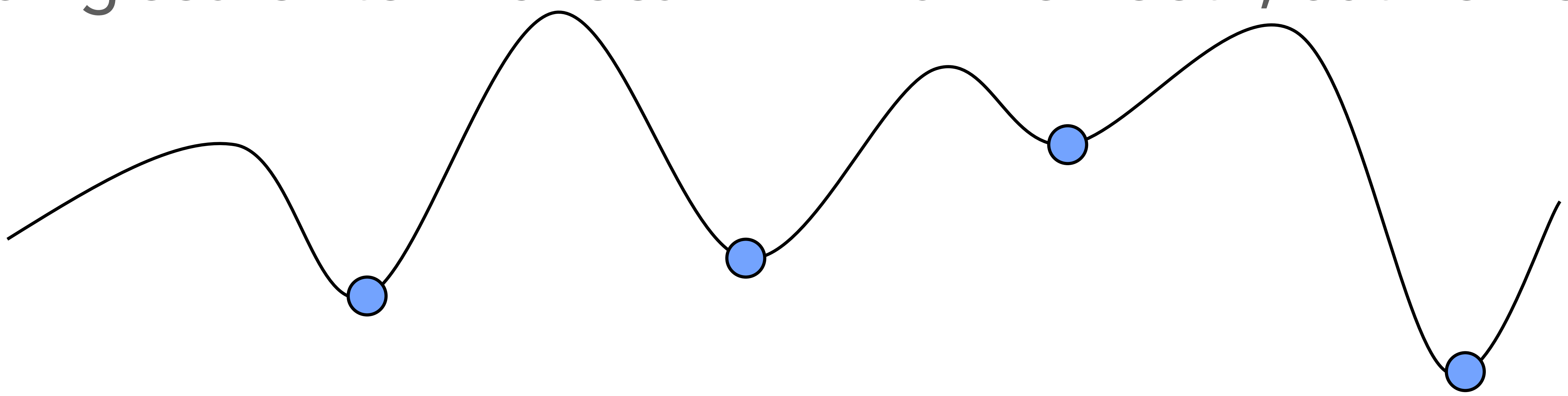
- Ideally, f′(x)=0…
- In practice…
  - $\|\nabla f(x)\| < \varepsilon$
  - $|f(x_{k+1}) - f(x_k)| < \varepsilon$
  - $\|x_{k+1} - x_k\| < \varepsilon$
  - Maximum number of iterations has been reached

GRADIENT ASCENT

- For concave functions that you want to *maximize*, take a step in direction of gradient (i.e., $w_{new} \leftarrow w_{old} + \eta\nabla(w)$)
- Otherwise same as gradient descent:
  - Start at some parameter values
  - Take derivative, move the parameters in the direction of gradient
  - Repeat until stopping criteria is met (e.g., gradient close to 0)

GRADIENT DESCENT FOR NON-CONVEX OPTIMIZATION

- Works on any objective function F(θ)
  - as long as we can evaluate the gradient Δ(θ)
  - this can be very useful for minimizing complex functions F
- Can be used in hill-climbing search to find local minima in smooth, but non-convex functions



- If function has multiple local minima, gradient descent goes to the closest local minimum:
  - solution: random restarts from multiple places in model space

LOGISTIC REGRESSION: RECAP

LOGISTIC REGRESSION

- Same parametric form as standard regression, but uses logistic function for binary classification

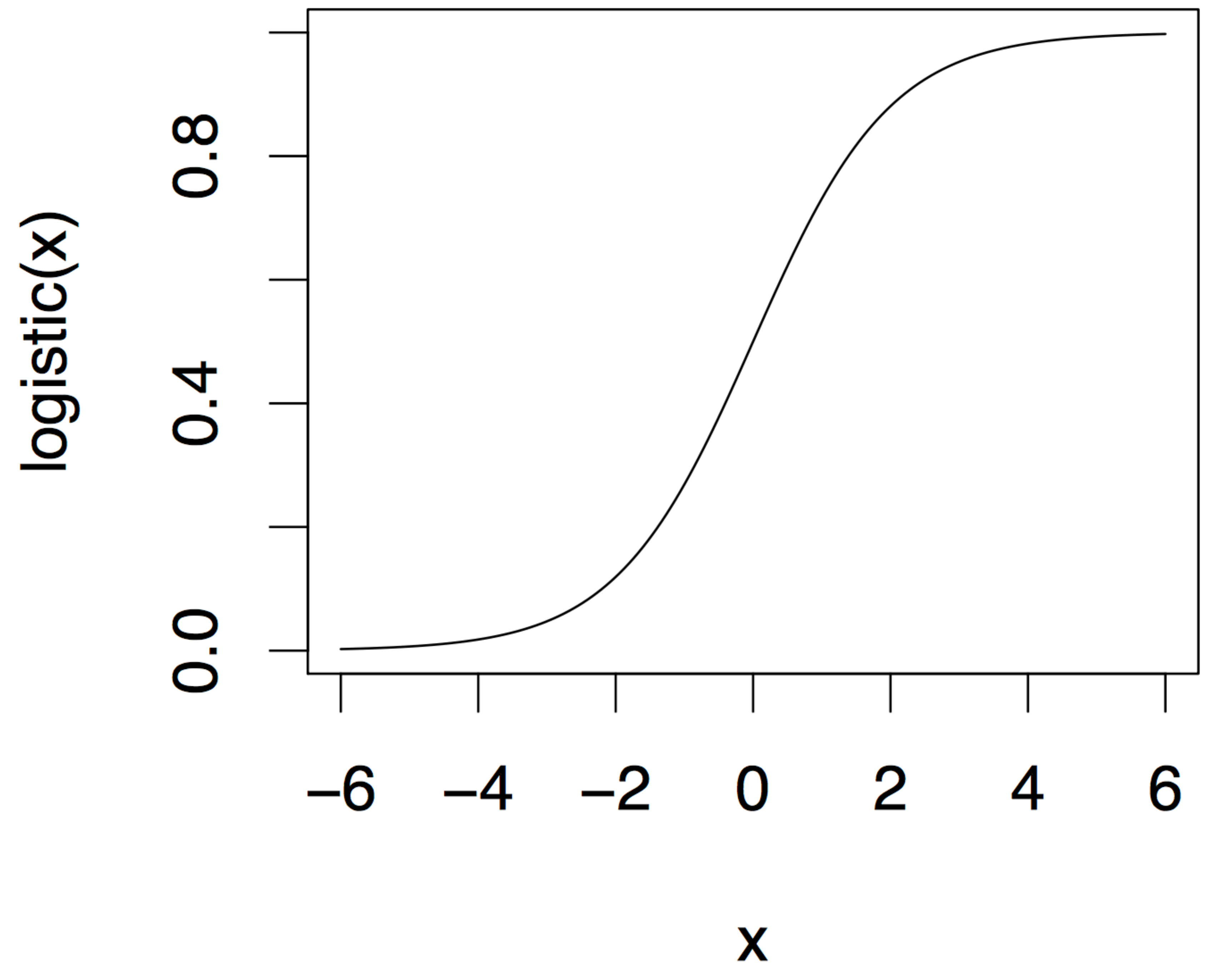
Logistic regression model:

$$P(y = 1|\mathbf{x}) = \frac{1}{1 + e^{-(\mathbf{w}^T\mathbf{x} + w_0)}}$$

- Output is the (positive) class probability rather than the binary prediction
- Logistic function transform ensures output is [0,1]

Logistic function:

$$\text{logistic}(x) := \frac{e^x}{1 + e^x} = \frac{1}{1 + e^{-x}}$$



LR EXAMPLE

$$P(BC=1|A,I,S,CR) = \frac{1}{1+e^{-(\mathbf{w}^T\mathbf{x})}}$$

Intercept	Age>40	Income=high	Student=yes	Credit=fair	BuysComp?
1	0	1	0	1	0
1	0	1	0	0	0
1	0	1	0	1	1
1	1	0	0	1	1
1	1	0	1	1	1
1	1	0	1	0	0
1	0	0	1	0	1
1	0	0	0	1	0
1	0	0	1	1	1
1	1	0	1	1	1
1	0	0	1	0	1
1	0	0	0	0	1
1	0	1	1	1	1
1	1	0	0	0	0

$$\mathbf{x} = [Int, A, I, S, CR]$$
$$\mathbf{w} = [w_0, w_A, w_I, w_S, w_{CR}]$$

LR parameters = w

- Score function: likelihood
- Estimate **w** with maximum likelihood estimation

LR LEARNING

- Score function: likelihood function

$$minimize \sum_{i=1}^{N} (-y_i \mathbf{w}^T \mathbf{x_i} + log(1 + e^{\mathbf{w}^T \mathbf{x_i}}))$$

- Estimate optimal **w** using gradient descent

Gradient descent:

Start at some **w**, e.g., **w**=[0,0,0,0,0]

Make predictions given current **w**: $\forall i \;\; \widehat{y}_i = P(y_i = 1|\mathbf{x}_i) = \frac{1}{1+e^{-\mathbf{w}^T\mathbf{x_i}}}$

Calculate gradient for each parameter: $\forall j \;\; \frac{d\, logL}{d\, w_j} = \left[\sum_{i=1}^{n}(-y_i + \hat{y}_i)x_{ij}\right] = \nabla_j$

Move parameters in direction of gradient: $\forall j \;\; w_j^{new} = w_j - \eta\nabla_j$

Repeat

LR PREDICTION

Intercept	Age>40	Income=high	Student=yes	Credit=fair	BuysComp?
1	0	1	0	1	0
1	0	1	0	0	0
1	0	1	0	1	1
1	1	0	0	1	1
1	1	0	1	1	1
1	1	0	1	0	0
1	0	0	1	0	1
1	0	0	0	1	0
1	0	0	1	1	1
1	1	0	1	1	1
1	0	0	1	0	1
1	0	0	0	0	1
1	0	1	1	1	1
1	1	0	0	0	0
1	**0**	**1**	**0**	**0**	**?**

- What is the probability that new person will buy a computer?

$$\mathbf{x} = [1, 0, 1, 0, 0]$$
$$\mathbf{w} = [-.5, 1.2, 3, -2, 0.7]$$

$$\mathbf{x}^T\mathbf{w} = 0.7$$

$$P(BC = 1|\mathbf{x}_i) = \frac{1}{1 + e^{-0.7}}$$
$$= 0.668$$

DEAL WITH OVERFITTING

- Simply finding the parameter values that lead to maximum likelihood function value in the training dataset may imply overfitting!
- Solution: add a **regularization term** in the scoring function to penalize complex models
  - e.g., L2 regularization term: $\frac{\lambda}{2}\|w\|^2$
  - λ is the regularization parameter; the larger the value, the more we are in favor of simple models

LR LEARNING WITH REGULARIZATION TERM

- Score function: likelihood with L2 regularization

$$minimize \sum_{i=1}^{N} (-y_i \mathsf{w}^\mathsf{T}\mathsf{x}_\mathsf{i} + log(1 + e^{\mathsf{w}^\mathsf{T}\mathsf{x}_\mathsf{i}})) + \frac{\lambda}{2}\|\mathsf{w}\|^2$$

- Estimate optimal **w** using gradient descent

Gradient descent:

Start at some **w**, e.g., **w**=[0,0,0,0,0]

Make predictions given current **w**: $\forall i \;\; \widehat{y}_i = P(y_i = 1|\mathbf{x}_i) = \frac{1}{1 + e^{-\mathbf{w}^T\mathsf{x}_\mathsf{i}}}$

Calculate gradient for each parameter: $\forall j \;\; \frac{d\,logL}{d\,w_j} = \left[\sum_{i=1}^{n}(-y_i + \hat{y}_i)x_{ij}\right] + \lambda w_j = \nabla_j$

Move parameters in direction of gradient: $\forall j \;\; w_j^{new} = w_j - \eta\nabla_j$

Repeat