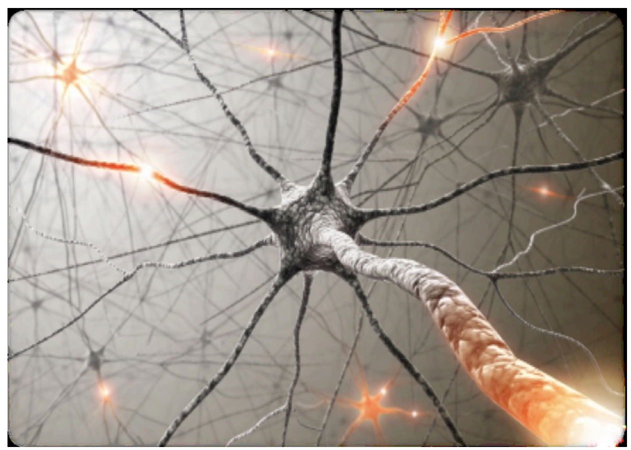
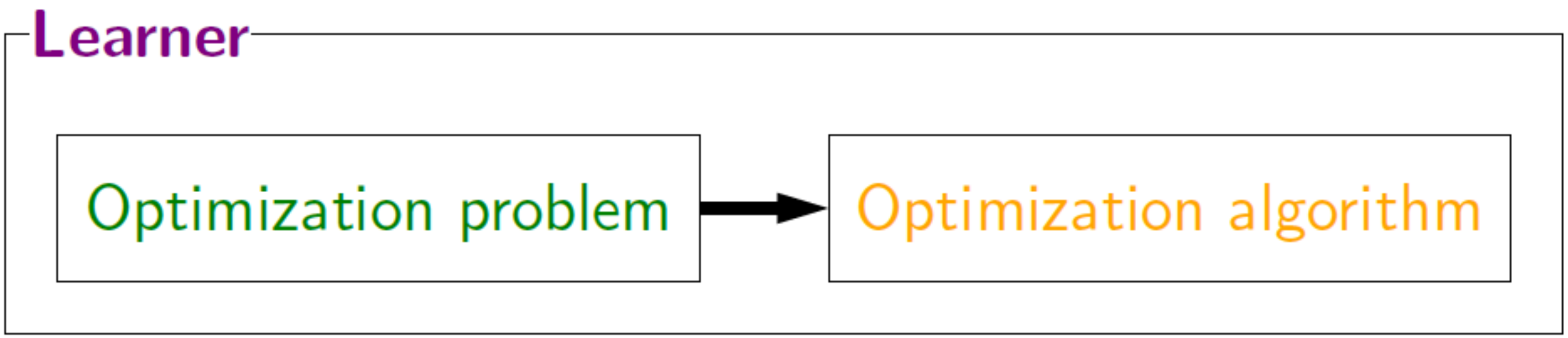
02 Linear Predictor



Roadmap

- Linear predictors
- Loss minimization
- Stochastic gradient descent

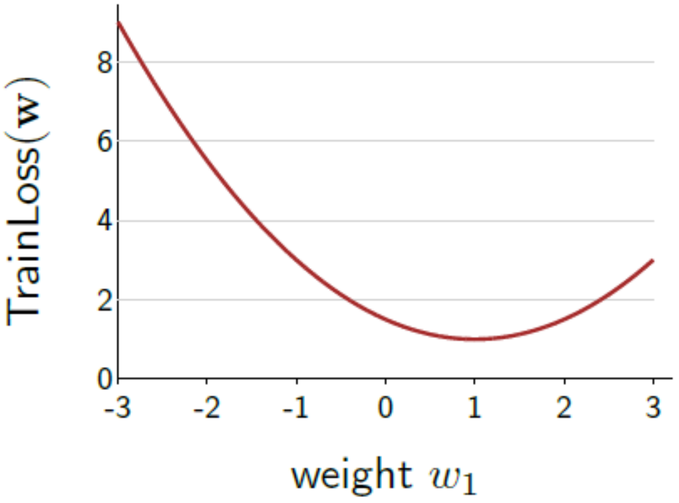
Learning as optimization



Optimization problem

Objective: $\min_{\mathbf{w} \in \mathbb{R}^d} \text{TrainLoss}(\mathbf{w})$

$\mathbf{w} \in \mathbb{R}$



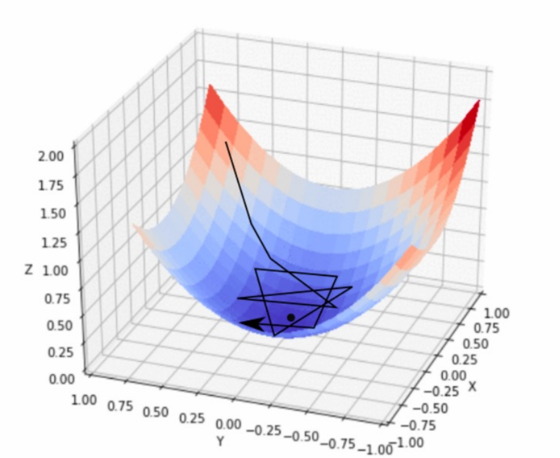
$\mathbf{w} \in \mathbb{R}^2$

[gradient plot]

How to optimize?

Definition: gradient

The gradient $\nabla_{\mathbf{w}} \mathrm{TrainLoss}(\mathbf{w})$ is the direction that increase the loss most.



Algorithm: gradient descent

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

For $t = 1, \ldots, T$:

$$\mathbf{w} \leftarrow \mathbf{w} - \underbrace{\eta}_{\text{step size}} \underbrace{\nabla_{\mathbf{w}} \mathrm{TrainLoss}(\mathbf{w})}_{\text{gradient}}$$

Least squares regression

- Objective function:

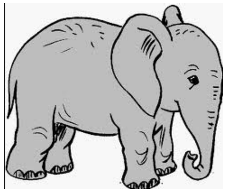
$$\text{TrainLoss}(\mathbf{w}) = \frac{1}{|\mathcal{D}_{\text{train}}|} \sum_{(x,y)\in\mathcal{D}_{\text{train}}} \left(\mathbf{w}\cdot\phi(x) - y\right)^2$$

- Gradient (use chain rule):

$$\nabla_{\mathbf{w}}\text{TrainLoss}(\mathbf{w}) = \frac{1}{|\mathcal{D}_{\text{train}}|} \sum_{(x,y)\in\mathcal{D}_{\text{train}}} 2\big(\underbrace{\mathbf{w}\cdot\phi(x) - y}_{\text{predict}-\text{target}}\big)\phi(x)$$

[live solution]

Gradient descent is slow



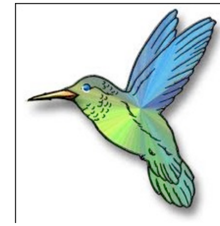
$$\text{TrainLoss}(\mathbf{w}) = \frac{1}{|\mathcal{D}_{\text{train}}|} \sum_{(x,y)\in\mathcal{D}_{\text{train}}} \text{Loss}(x, y, \mathbf{w})$$

Gradient descent:

$$\mathbf{w} \longleftarrow \mathbf{w} - \eta \nabla_{\mathbf{w}} \text{TrainLoss}(\mathbf{w})$$

Problem: each iteration requires going over all training examples—expensive when have lots of data!

Stochastic gradient descent



$$\text{TrainLoss}(\mathbf{w}) = \frac{1}{|\mathcal{D}_{\text{train}}|} \sum_{(x,y)\in\mathcal{D}_{\text{train}}} \text{Loss}(x, y, \mathbf{w})$$

Gradient descent (GD): $\mathbf{w} \longleftarrow \mathbf{w} - \eta \nabla_{\mathbf{w}} \text{TrainLoss}(\mathbf{w})$

Stochastic Gradient descent (SGD):

For each $(x, y) \in \mathcal{D}_{\text{train}}$:

$$\mathbf{w} \leftarrow \mathbf{w} - \eta \nabla_{\mathbf{w}} \text{Loss}(x, y, \mathbf{w})$$

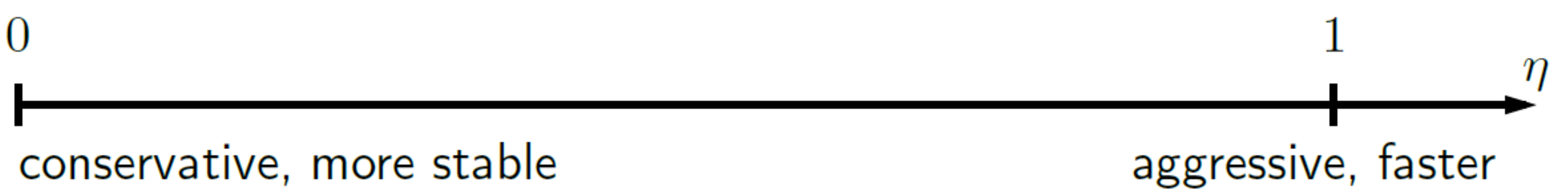
Key idea: stochastic updates

It's not about quality, it's about quantity.

Step size

$$\mathbf{w} \longleftarrow w - \underbrace{\eta}_{\text{step size}} \nabla_{\mathbf{w}} \text{Loss}(x, y, \mathbf{w})$$

Question: what should η be?



- **Strategies:**
  - Constant: $\eta = 0.1$
  - Decreasing: $\eta = 1/\sqrt{\#\text{ updates made so far}}$

Summary so far

- Linear predictors:

$f_{\mathbf{w}}(x)$ based on score $\mathbf{w} \cdot \phi(x)$

- Loss minimization: learning as optimization

$$\min_{\mathbf{w} \in \mathbb{R}^d} \mathrm{TrainLoss}(\mathbf{w})$$

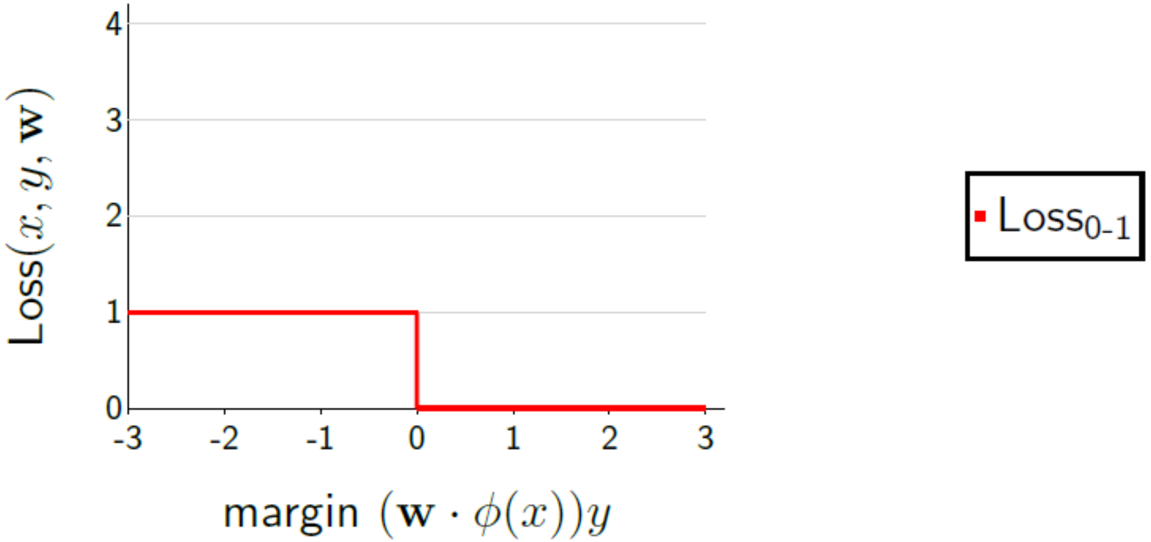
- Stochastic gradient descent: optimization algorithm

$$\mathbf{w} \longleftarrow w - \eta \nabla_{\mathbf{w}} \mathrm{Loss}(x, y, \mathbf{w})$$

- Done for linear regression; what about classification?

Zero-one loss

$$\text{Loss}_{0-1}(x, y, \mathbf{w}) = \mathbf{1}[(\mathbf{w} \cdot \phi(x))y \leq 0]$$

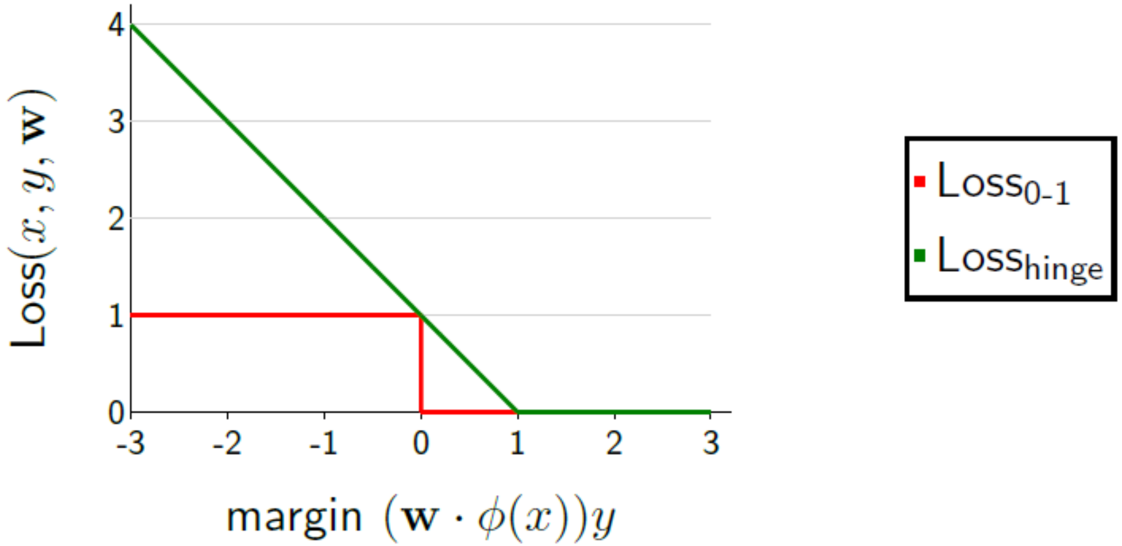


Problems:

- Gradient of Loss$_{0\text{-}1}$ is 0 everywhere, SGD not applicable
- Loss0-1 is insensitive to how badly model messed up

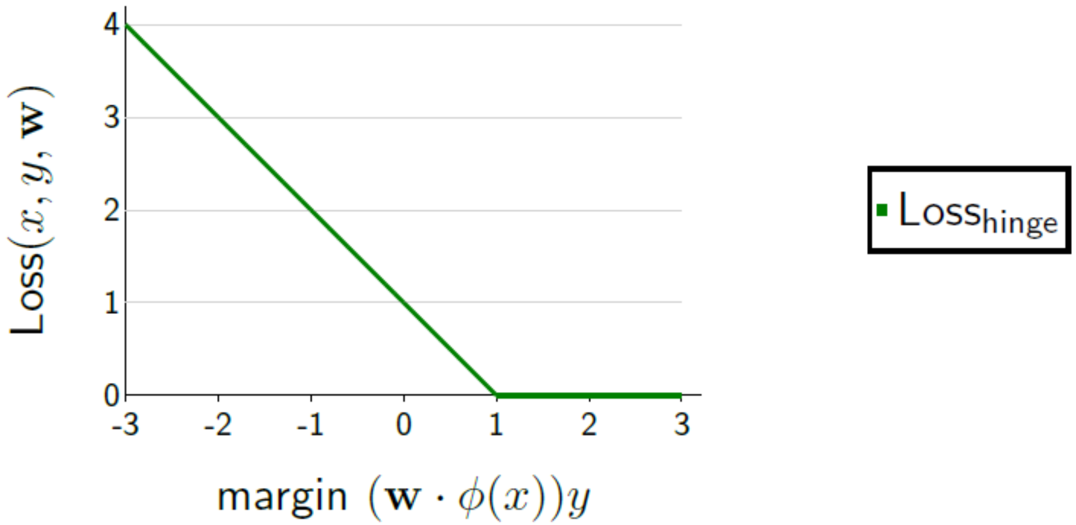
Support vector machines*

$$\text{Loss}_{\text{hinge}}(x, y, \mathbf{w}) = \max\{1 - (\mathbf{w} \cdot \phi(x))y, 0\}$$



- Intuition: hinge loss upper bounds 0-1 loss, has non-trivial gradient
- Try to increase margin if less than 1

A gradient exercise



Problem: Gradient of hinge loss

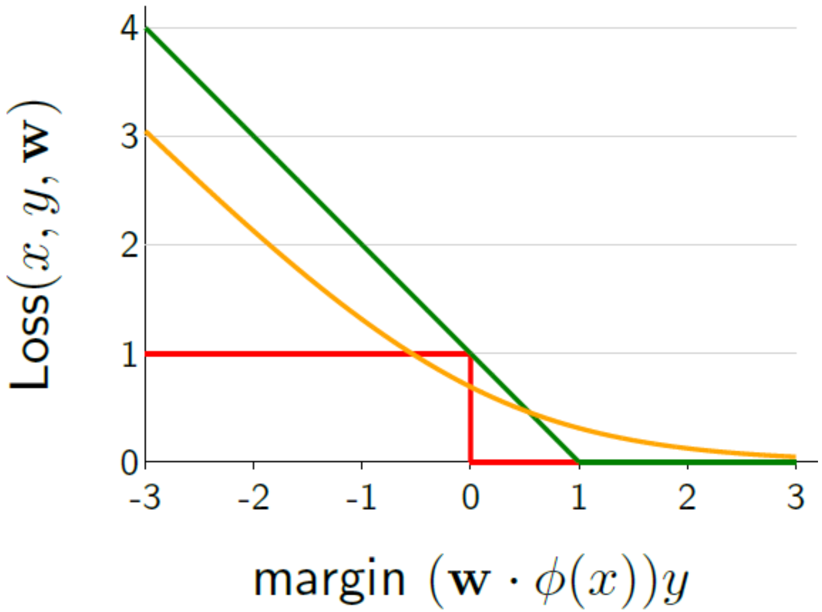
Compute the gradient of

$$\text{Loss}_{\text{hinge}}(x, y, \mathbf{w}) = \max\{1 - (\mathbf{w} \cdot \phi(x))y, 0\}$$

[Blackboard]

Logistic loss

$$\text{Loss}_{\text{logistic}}(x, y, \mathbf{w}) = \log(1 + e^{-(w \cdot \phi(x))y})$$



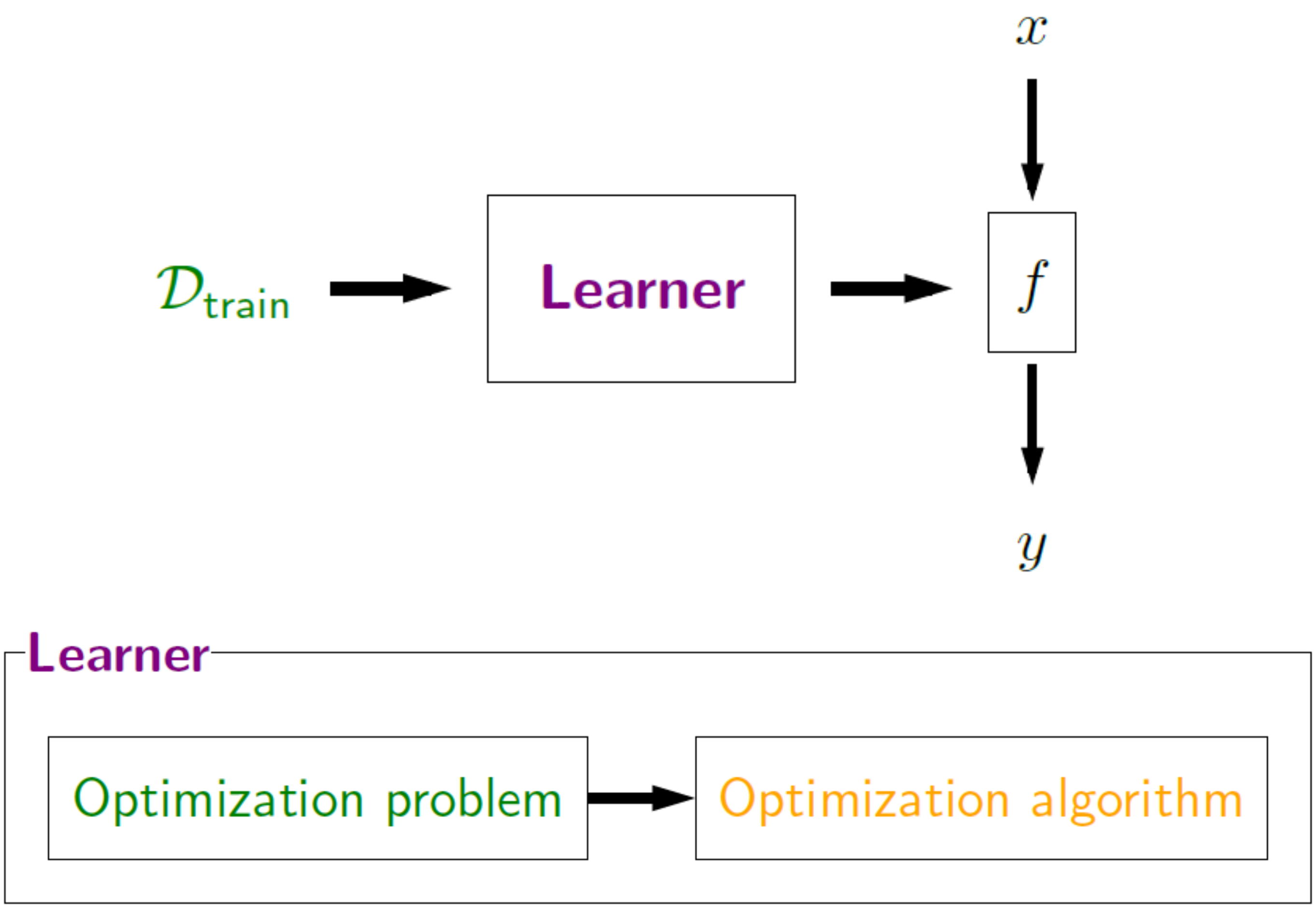
- Intuition: Try to increase margin even when it already exceeds 1

Summary so far

$$\underbrace{\mathbf{w} \cdot \phi(x)}_{\text{score}}$$

	Classification	Linear regression
Predictor $f_{\mathbf{w}}$	sign(score)	score
Relate to correct y	margin (score y)	residual (score $- y$)
Loss functions	zero-one hinge logistic	squared absolute deviation
Algorithm	SGD	SGD

Framework



x
$\mathcal{D}_{\text{train}}$
Learner
f
y
Learner
Optimization problem
Optimization algorithm

Next lecture

Linear predictors:

$f_{\mathbf{w}}(x)$ based on score $\mathbf{w} \cdot \phi(x)$

Which feature vector $\phi(x)$ to use?

Loss minimization:

$$\min_{\mathbf{w} \in \mathbb{R}^d} \mathrm{TrainLoss}(\mathbf{w})$$

How do we **generalize** beyond the training set?