Lecture 2: Machine Learning 1

Course plan

Search problems

Markov decision processes

Adversarial games

Constraint satisfaction problems

Markov networks

Bayesian networks

Reflex **States** **Variables** **Logic**

Low-level → High-level

Machine learning

Roadmap

Machine learning overview

Linear regression

Linear classification

Course plan

Search problems

Markov decision processes

Adversarial games

Constraint satisfaction problems

Markov networks

Bayesian networks

Reflex **States** **Variables** **Logic**

Low-level → High-level

Machine learning

Course plan

Search problems

Markov decision processes

Adversarial games

Constraint satisfaction problems

Markov networks

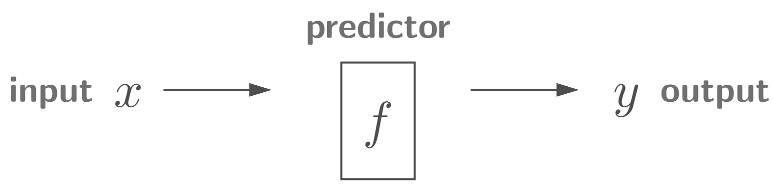
Bayesian networks

Reflex **States** **Variables** **Logic**

Low-level → High-level

Machine learning

Reflex-based models



Binary classification

classifier

$$x \longrightarrow \boxed{f} \longrightarrow y \in \{+1, -1\}$$ **label**

Fraud detection: credit card transaction $\rightarrow$ fraud or no fraud

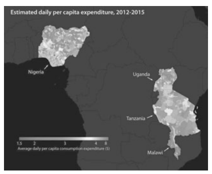
Toxic comments: online comment $\rightarrow$ toxic or not toxic

Higgs boson: measurements of event $\rightarrow$ decay event or background

Extension: multiclass classification: $y \in \{1, \ldots, K\}$

Regression

$$x \longrightarrow \boxed{f} \longrightarrow y \in \mathbb{R}$$ **response**



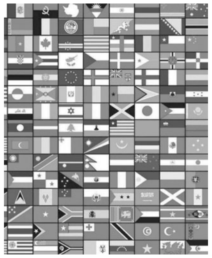
Poverty mapping: satellite image → asset wealth index

Housing: information about house → price

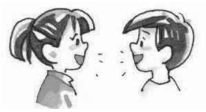
Arrival times: destination, weather, time → time of arrival

Structured prediction

$x \longrightarrow \boxed{f} \longrightarrow y$ is a complex object



Machine translation: English sentence $\rightarrow$ Japanese sentence



Dialogue: conversational history $\rightarrow$ next utterance

Image captioning: image $\rightarrow$ sentence describing image

Image segmentation: image $\rightarrow$ segmentation

Roadmap

Tasks

Linear regression

Linear classification

K-means

Optimization Algorithms

Gradient descent

Stochastic gradient descent

Backpropagation

Models

Non-linear features

Feature templates

Neural networks

Considerations

Generalization

Best practices

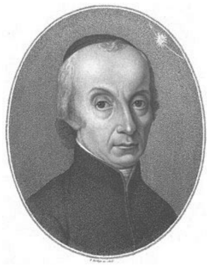
Roadmap

Machine learning overview

Linear regression

Linear classification

The discovery of Ceres



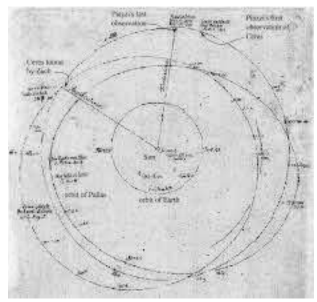
1801: astronomer Piazzi discovered Ceres, made 19 observations of location before it was obscured by the sun

Time	Right ascension	Declination
Jan 01, 20:43:17.8	50.91	15.24
Jan 02, 20:39:04.6	50.84	15.30
...	...	...
Feb 11, 18:11:58.2	53.51	18.43

When and where will Ceres be observed again?

Gauss's triumph

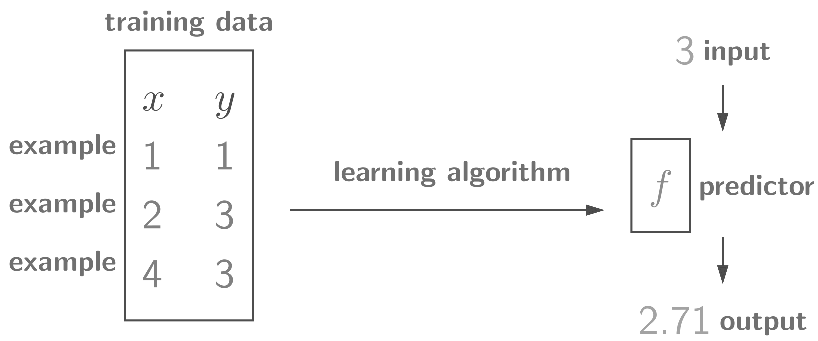
September 1801: Gauss took Piazzi's data and created a model of Ceres's orbit, makes prediction

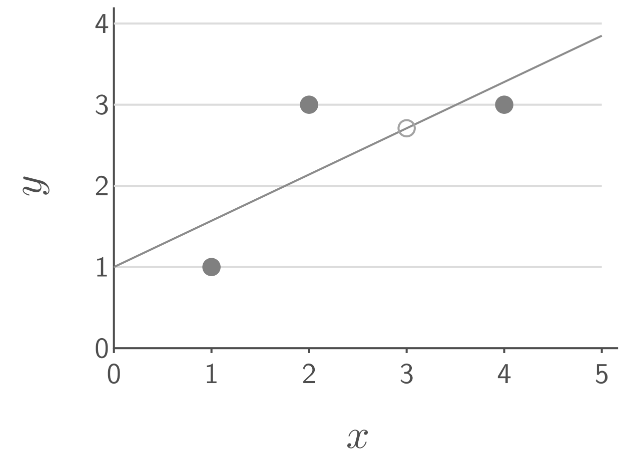


December 7, 1801: Ceres located within 1/2 degree of Gauss's prediction, much more accurate than other astronomers

Method: least squares linear regression

Linear regression framework





Design decisions:

Which predictors are possible? **hypothesis class**

How good is a predictor? **loss function**

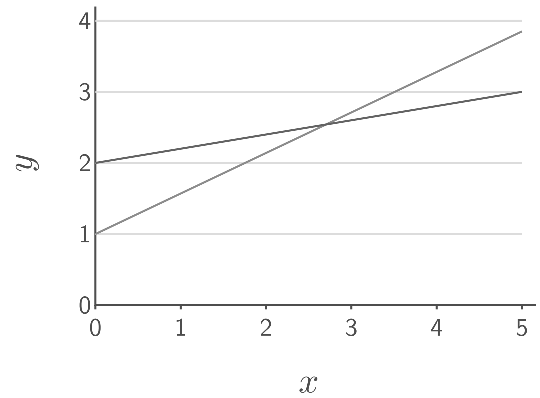
How do we compute the best predictor? **optimization algorithm**

Hypothesis class: which predictors?

$f(x) = 1 + 0.57x$

$f(x) = 2 + 0.2x$

$f(x) = w_1 + w_2 x$



Vector notation:

weight vector $\mathbf{w} = [w_1, w_2]$

feature extractor $\phi(x) = [1, x]$ **feature vector**

$$f_{\mathbf{w}}(x) = \mathbf{w} \cdot \phi(x) \text{ score}$$

$f_{\mathbf{w}}(3) = [1, 0.57] \cdot [1, 3] = 2.71$

Hypothesis class:

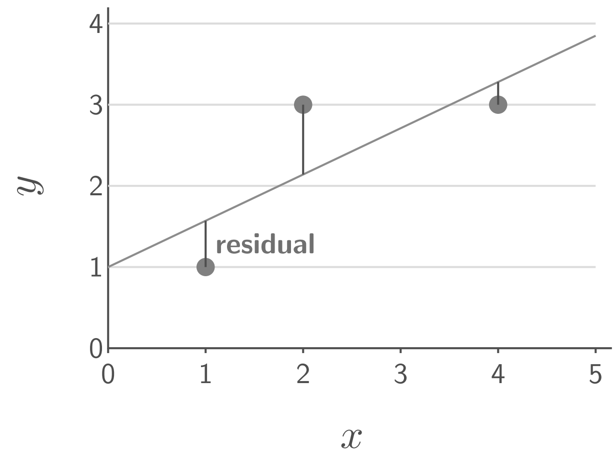
$$\mathcal{F} = \{f_{\mathbf{w}} : \mathbf{w} \in \mathbb{R}^2\}$$

Loss function: how good is a predictor?

$$f_{\mathbf{w}}(x) = \mathbf{w} \cdot \phi(x)$$
$$\mathbf{w} = [1, 0.57]$$
$$\phi(x) = [1, x]$$

training data $\mathcal{D}_{\text{train}}$

x	y
1	1
2	3
4	3



$$\text{Loss}(x, y, \mathbf{w}) = (f_{\mathbf{w}}(x) - y)^2$$ **squared loss**

$$\text{Loss}(1, 1, [1, 0.57]) = ([1, 0.57] \cdot [1, 1] - 1)^2 = 0.32$$
$$\text{Loss}(2, 3, [1, 0.57]) = ([1, 0.57] \cdot [1, 2] - 3)^2 = 0.74$$
$$\text{Loss}(4, 3, [1, 0.57]) = ([1, 0.57] \cdot [1, 4] - 3)^2 = 0.08$$

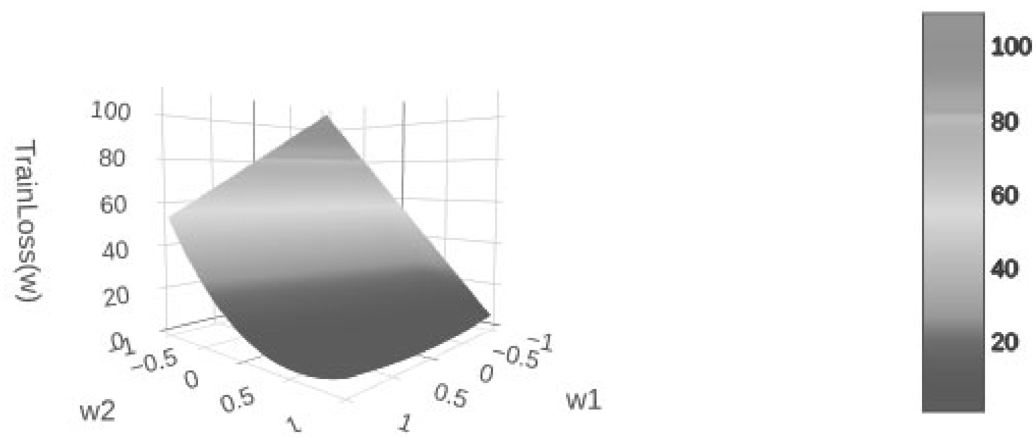
$$\text{TrainLoss}(\mathbf{w}) = \frac{1}{|\mathcal{D}_{\text{train}}|} \sum_{(x,y)\in\mathcal{D}_{\text{train}}} \text{Loss}(x, y, \mathbf{w})$$

$$\text{TrainLoss}([1, 0.57]) = 0.38$$

Loss function: visualization

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

$$\min_{\mathbf{w}} \text{TrainLoss}(\mathbf{w})$$

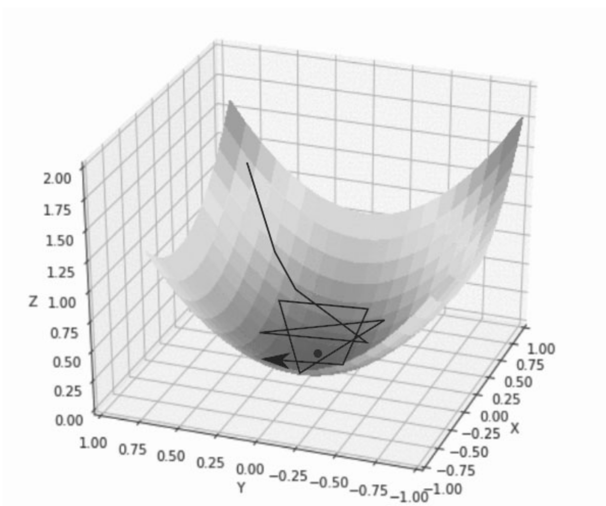


Optimization algorithm: how to compute best?

Goal: $\min_{\mathbf{w}} \text{TrainLoss}(\mathbf{w})$

Definition: gradient

The gradient $\nabla_{\mathbf{w}} \text{TrainLoss}(\mathbf{w})$ is the direction that increases the training loss the most.

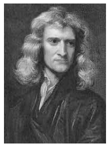


Algorithm: gradient descent

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

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

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



Computing the gradient

Objective function:

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

Gradient (use chain rule):

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

Gradient descent example

training data $\mathcal{D}_{\text{train}}$

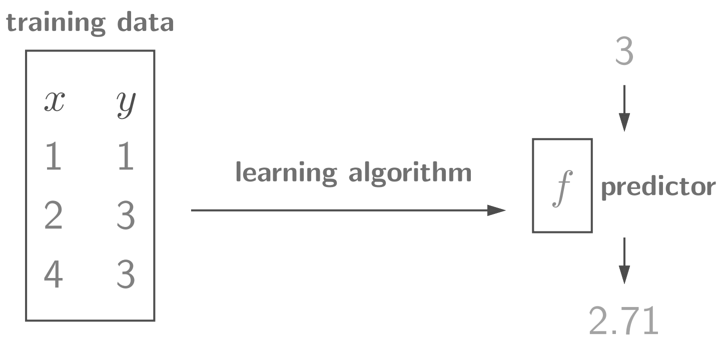
x	y
1	1
2	3
4	3

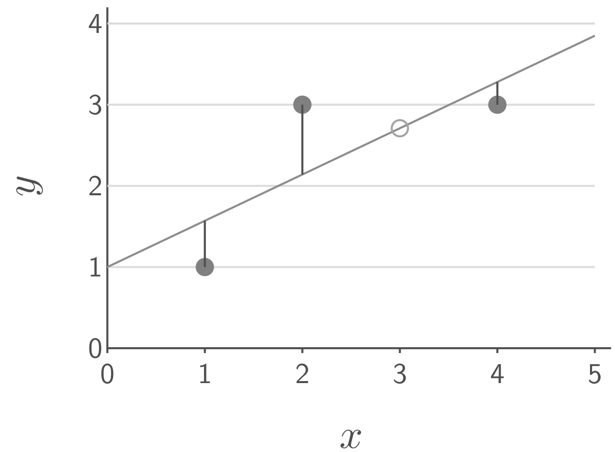
$\nabla_{\mathbf{w}} \text{TrainLoss}(\mathbf{w}) = \frac{1}{|\mathcal{D}_{\text{train}}|} \sum_{(x,y)\in\mathcal{D}_{\text{train}}} 2(\mathbf{w}\cdot\phi(x) - y)\phi(x)$

Gradient update: $\mathbf{w} \leftarrow \mathbf{w} - 0.1\nabla_{\mathbf{w}} \text{TrainLoss}(\mathbf{w})$

t	$\nabla_{\mathbf{w}} \text{TrainLoss}(\mathbf{w})$	$\mathbf{w}$
		$[0,0]$
1	$\underbrace{\frac{1}{3}(2([0,0]\cdot[1,1]-1)[1,1] + 2([0,0]\cdot[1,2]-3)[1,2] + 2([0,0]\cdot[1,4]-3)[1,4])}_{=[-4.67,-12.67]}$	$[0.47, 1.27]$
2	$\underbrace{\frac{1}{3}(2([0.47,1.27]\cdot[1,1]-1)[1,1] + 2([0.47,1.27]\cdot[1,2]-3)[1,2] + 2([0.47,1.27]\cdot[1,4]-3)[1,4])}_{=[2.18,7.24]}$	$[0.25, 0.54]$
...	...	...
200	$\underbrace{\frac{1}{3}(2([1,0.57]\cdot[1,1]-1)[1,1] + 2([1,0.57]\cdot[1,2]-3)[1,2] + 2([1,0.57]\cdot[1,4]-3)[1,4])}_{=[0,0]}$	$[1, 0.57]$

Summary





Which predictors are possible? **Hypothesis class**	Linear functions $\mathcal{F} = \{f_{\mathbf{w}}(x) = \mathbf{w} \cdot \phi(x)\}, \phi(x) = [1, x]$
How good is a predictor? **Loss function**	Squared loss $\text{Loss}(x, y, \mathbf{w}) = (f_{\mathbf{w}}(x) - y)^2$
How to compute best predictor? **Optimization algorithm**	Gradient descent $\mathbf{w} \leftarrow \mathbf{w} - \eta \nabla \textbf{TrainLoss}(\mathbf{w})$

Roadmap

Machine learning overview

Linear regression

Linear classification

Linear classification framework

training data

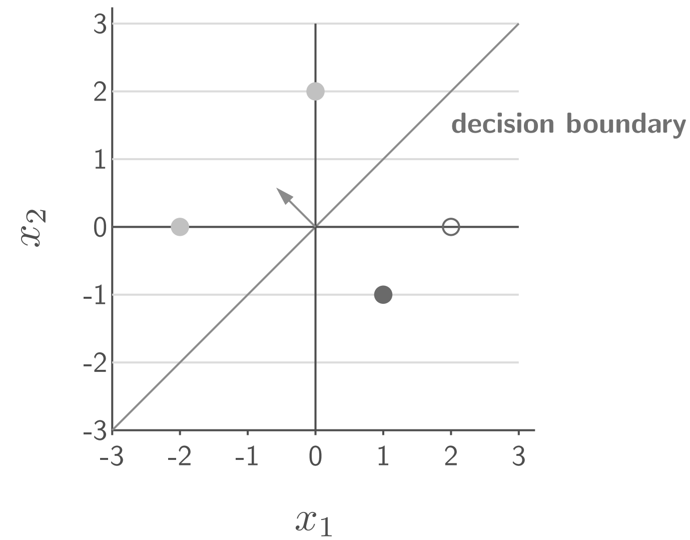
	x_1	x_2	y
example	0	2	1
example	-2	0	1
example	1	-1	-1

learning algorithm →

[2, 0] **input**

f **classifier**

-1 **label**



Design decisions:

Which classifiers are possible? **hypothesis class**

How good is a classifier? **loss function**

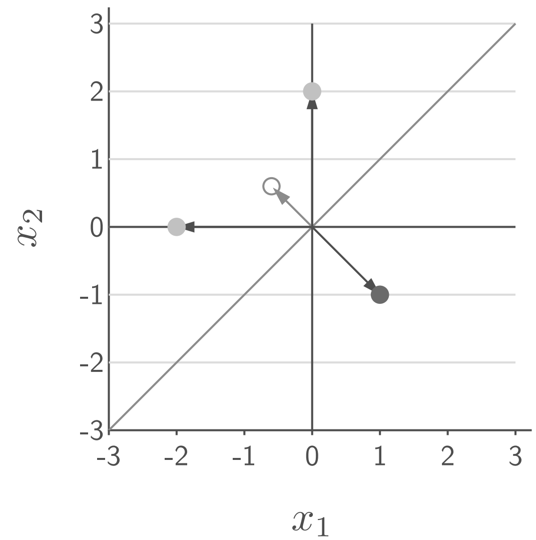
How do we compute the best classifier? **optimization algorithm**

An example linear classifier

$$f(x) = \text{sign}(\underbrace{[-0.6, 0.6]}_{\mathbf{w}} \cdot \underbrace{[x_1, x_2]}_{\phi(x)})$$

$$\text{sign}(z) = \begin{cases} +1 & \text{if } z > 0 \\ -1 & \text{if } z < 0 \\ 0 & \text{if } z = 0 \end{cases}$$

x_1	x_2	$f(x)$
0	2	1
-2	0	1
1	-1	-1



$$f([0,2]) = \text{sign}([-0.6, 0.6] \cdot [0, 2]) = \text{sign}(1.2) = 1$$

$$f([-2,0]) = \text{sign}([-0.6, 0.6] \cdot [-2, 0]) = \text{sign}(1.2) = 1$$

$$f([1,-1]) = \text{sign}([-0.6, 0.6] \cdot [1, -1]) = \text{sign}(-1.2) = -1$$

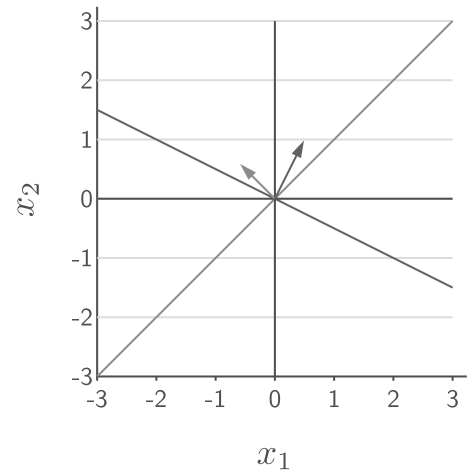
Decision boundary: x such that $\mathbf{w} \cdot \phi(x) = 0$

Hypothesis class: which classifiers?

$\phi(x) = [x_1, x_2]$

$f(x) = \text{sign}([-0.6, 0.6] \cdot \phi(x))$

$f(x) = \text{sign}([0.5, 1] \cdot \phi(x))$



General binary classifier:

$$f_{\mathbf{w}}(x) = \text{sign}(\mathbf{w} \cdot \phi(x))$$

Hypothesis class:

$$\mathcal{F} = \{f_{\mathbf{w}} : \mathbf{w} \in \mathbb{R}^2\}$$

Loss function: how good is a classifier?

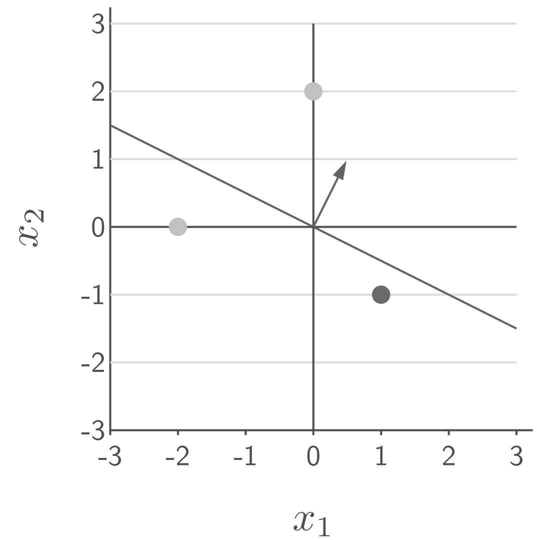
$f_{\mathbf{w}}(x) = \mathbf{w} \cdot \phi(x)$

$\mathbf{w} = [0.5, 1]$

$\phi(x) = [x_1, x_2]$

training data $\mathcal{D}_{\text{train}}$

x_1	x_2	y
0	2	1
-2	0	1
1	-1	-1



$\text{Loss}_{\text{0-1}}(x, y, \mathbf{w}) = \mathbf{1}[f_{\mathbf{w}}(x) \neq y]$ **zero-one loss**

$\text{Loss}([0, 2], 1, [0.5, 1]) = \mathbf{1}[\text{sign}([0.5, 1] \cdot [0, 2]) \neq 1] = 0$

$\text{Loss}([-2, 0], 1, [0.5, 1]) = \mathbf{1}[\text{sign}([0.5, 1] \cdot [-2, 0]) \neq 1] = 1$

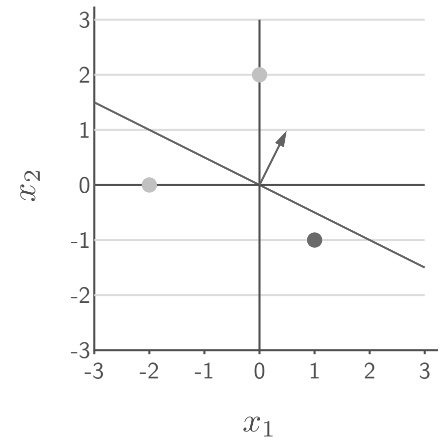
$\text{Loss}([1, -1], -1, [0.5, 1]) = \mathbf{1}[\text{sign}([0.5, 1] \cdot [1, -1]) \neq -1] = 0$

$\text{TrainLoss}([0.5, 1]) = 0.33$

Score and margin

Predicted label: $f_{\mathbf{w}}(x) = \text{sign}(\mathbf{w} \cdot \phi(x))$

Target label: y



Definition: score

The score on an example (x, y) is $\mathbf{w} \cdot \phi(x)$, how **confident** we are in predicting $+1$.

Definition: margin

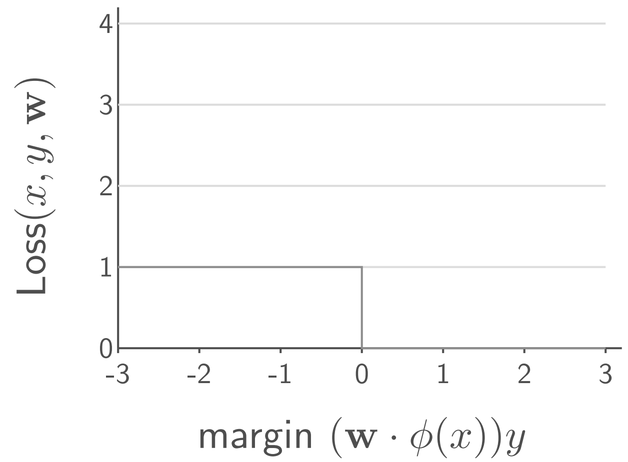
The margin on an example (x, y) is $(\mathbf{w} \cdot \phi(x))y$, how **correct** we are.

Zero-one loss rewritten

Definition: zero-one loss

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

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



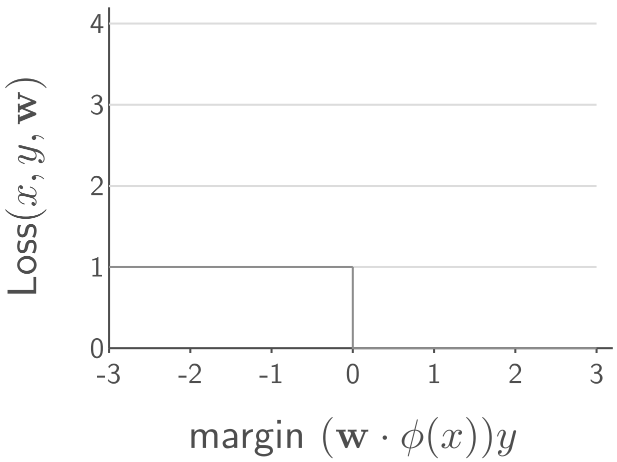
Optimization algorithm: how to compute best?

Goal: $\min_{\mathbf{w}} \text{TrainLoss}(\mathbf{w})$

To run gradient descent, compute the gradient:

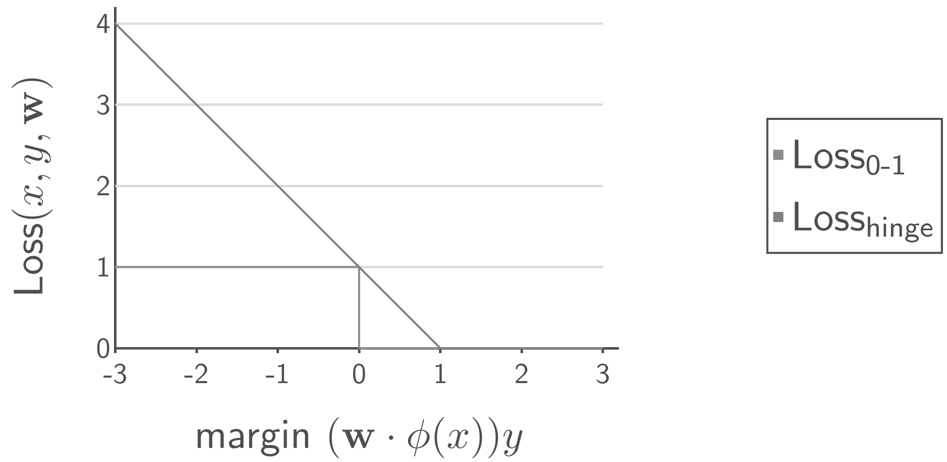
$$\nabla_{\mathbf{w}} \text{TrainLoss}(\mathbf{w}) = \frac{1}{|\mathcal{D}_{\text{train}}|} \sum_{(x,y)\in\mathcal{D}_{\text{train}}} \nabla \text{Loss}_{0\text{-}1}(x, y, \mathbf{w})$$

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



Gradient is zero almost everywhere!

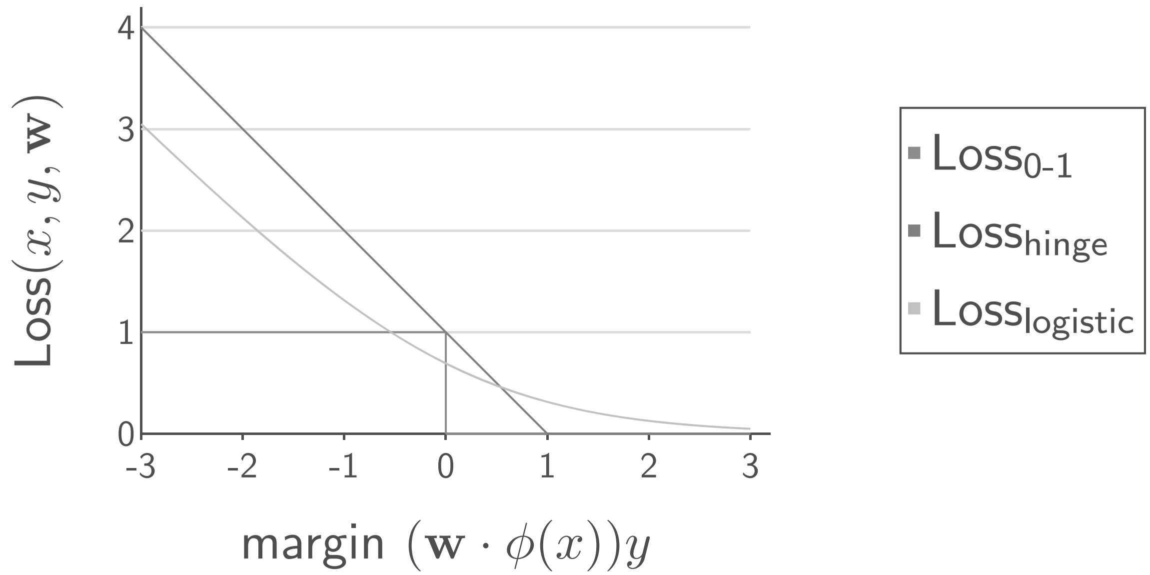
Hinge loss



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

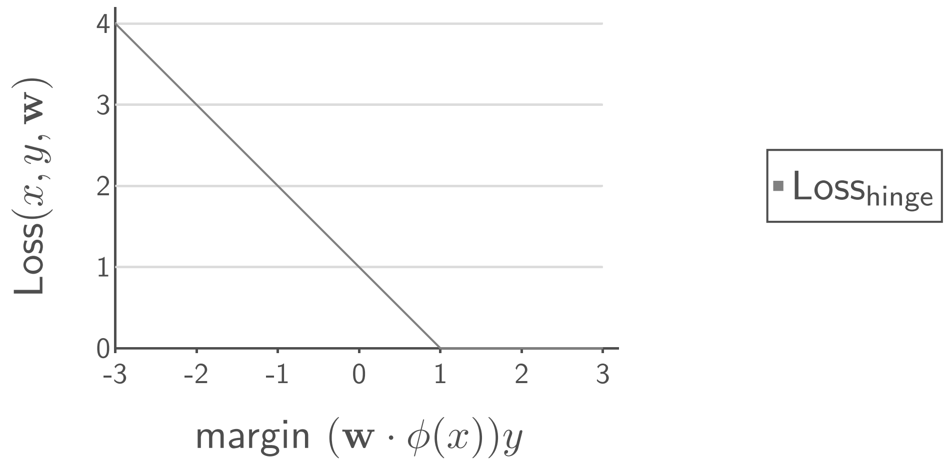
Digression: logistic regression

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



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

Gradient of the hinge loss



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

$$\nabla \mathsf{Loss}_{\mathsf{hinge}}(x, y, \mathbf{w}) = \begin{cases} -\phi(x)y & \text{if } \{1 - (\mathbf{w} \cdot \phi(x))y\} > \{0\} \\ 0 & \text{otherwise} \end{cases}$$

Hinge loss on training data

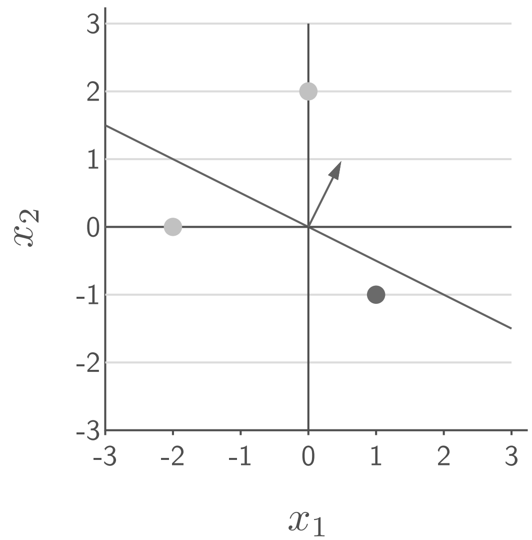
$f_{\mathbf{w}}(x) = \mathbf{w} \cdot \phi(x)$

$\mathbf{w} = [0.5, 1]$

$\phi(x) = [x_1, x_2]$

training data $\mathcal{D}_{\text{train}}$

x_1	x_2	y
0	2	1
-2	0	1
1	-1	-1



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

$\text{Loss}([0,2], 1, [0.5,1]) = \max\{1 - [0.5,1] \cdot [0,2](1), 0\} = 0$

$\text{Loss}([-2,0], 1, [0.5,1]) = \max\{1 - [0.5,1] \cdot [-2,0](1), 0\} = 2$

$\text{Loss}([1,-1], -1, [0.5,1]) = \max\{1 - [0.5,1] \cdot [1,-1](-1), 0\} = 0.5$

$\text{TrainLoss}([0.5,1]) = 0.83$

$\nabla\text{Loss}([0,2], 1, [0.5,1]) = [0,0]$

$\nabla\text{Loss}([-2,0], 1, [0.5,1]) = [2,0]$

$\nabla\text{Loss}([1,-1], -1, [0.5,1]) = [1,-1]$

$\nabla\text{TrainLoss}([0.5,1]) = [1,-0.33]$

Summary so far

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

	Regression	Classification
Prediction $f_{\mathbf{w}}(x)$	score	sign(score)
Relate to target y	residual (score $- y$)	margin (score y)
Loss functions	squared absolute deviation	zero-one hinge logistic
Algorithm	gradient descent	gradient descent

homework

due: next week

作业 1-周1-pytorch安装