PHYS 139/239: Machine Learning in Physics Lecture 2: Perceptron Learning Algorithm & (Stochastic) Gradient Descent

Javier Duarte – January 12, 2023



Recap: Bias-variance tradeoff

- If L is the squared loss, we can decompose the expected test error: $\mathbb{E}\left[L_P(f(x \mid w_S))\right] = \mathbb{E}_S \mathbb{E}_{(x,y) \sim P(x,y)} \left[L_{X,y}\right]$ $= \mathbb{E}_{(x,y) \sim P(x,y)} \left| \mathbb{E}_{S} \right|$
- where $F(x) = \mathbb{E}_S |f(x | w_S)|$ is the average prediction of our model over different possible training datasets
- Variance: difference in predictions when training on different datasets
- **Bias:** difference from ground truth

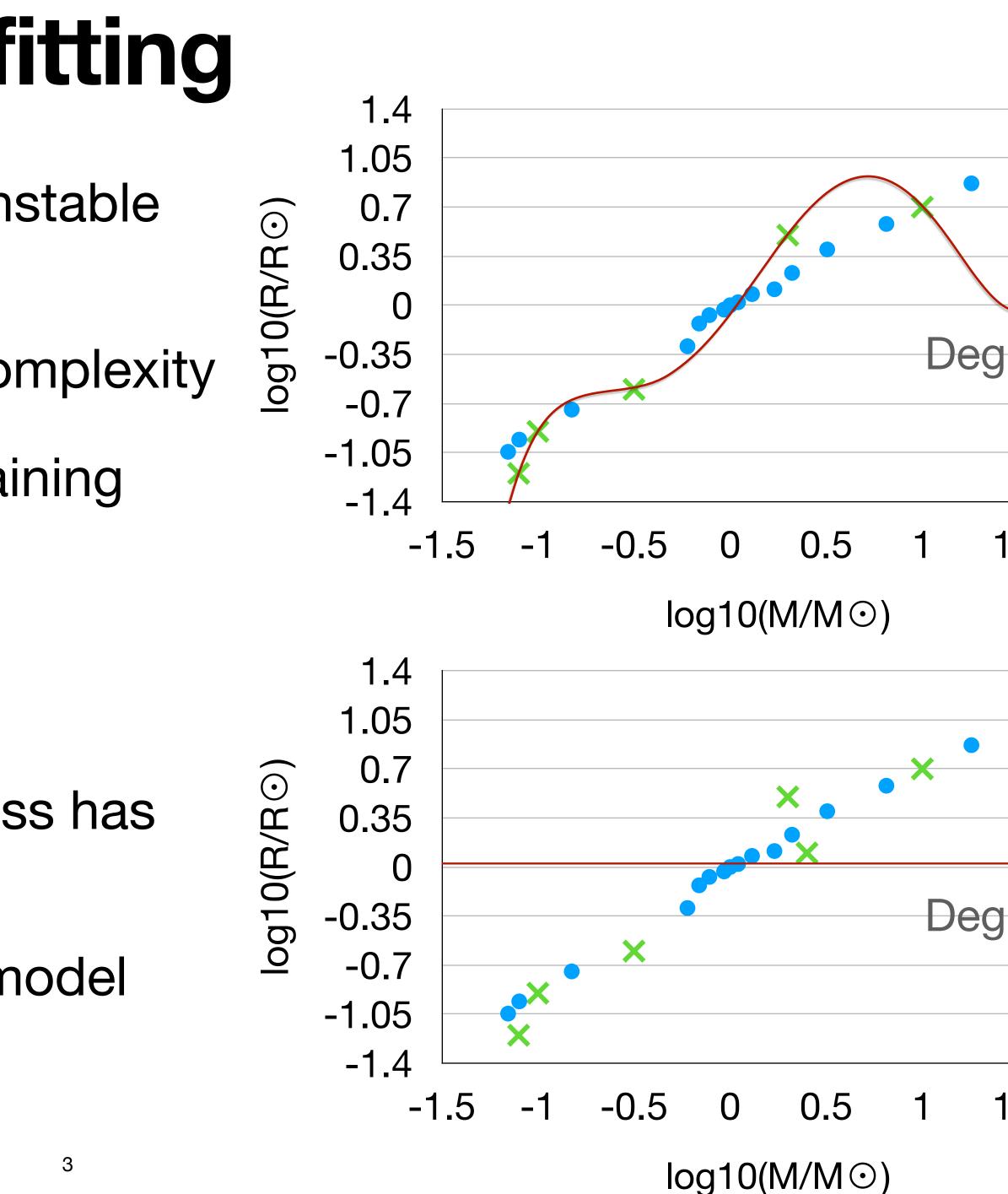
$$L(y, f(x | w_S))]$$

$$[(f(x | w_S) - F(x))^2] + (F(x) - y)^2]$$

Variance (Squared) bias

Overfitting vs. underfitting

- Overfitting implies high variance (unstable model class)
 - Variance increases with model complexity
 - Variance decreases with more training data
- Underfitting implies high bias
 - Even with no variance, model class has high error
 - Underfitting happens whenever model complexity is too low





Model selection

- We only have a finite training dataset
- We cannot measure the true test error
- Simple model classes underfit
- Complex model classes overfit

(but not so straightforward for deep neural networks!)

Goal: Select the model class with the lowest test error

Bias-variance tradeoff

Validation set

Original dataset

- Split the original dataset into a training and validation set
- Train model on the training set
- Evaluate on the validation set to estimate the test error
- Select the model class that gives the lowest estimated error
- Optionally, re-train the selected model class on the whole dataset (training + validation)
- **Issue:** we would like both training and validation sets to be as large as possible (so that the estimate is better), but they must not overlap!

k-fold cross-validation

- Split the original dataset into k equal parts (e.g, k = 5)
- Train on the k-1 parts and validate on the remaining one





• Advantage: use all data as validation to improve the estimate of the test error, at the cost of more computation (k trainings)

- Original dataset
- Repeat for every choice of the k-1 parts and average the validation errors

Recap: Supervised learning pipeline

- Training dataset: $S = \{(x_1, y_1), \dots, (x_N, y_N)\}$ where $x \in \mathbb{R}^D$ and $y \in \mathbb{R}$
- Model / hypothesis class: $f(x | w) = w^{\mathsf{T}} x$ (linear models)
- Loss function: $L(y, y') = (y y')^2$ (squared loss) or $\phi(x)$ instead of x
- Optimization algorithm to minimize the learning objective:

$$\underset{w}{\operatorname{arg\,min}} \sum_{i=1}^{N} L(y_i, f(x_i \mid w))$$

- Cross validation and model selection:
- Testing and deployment



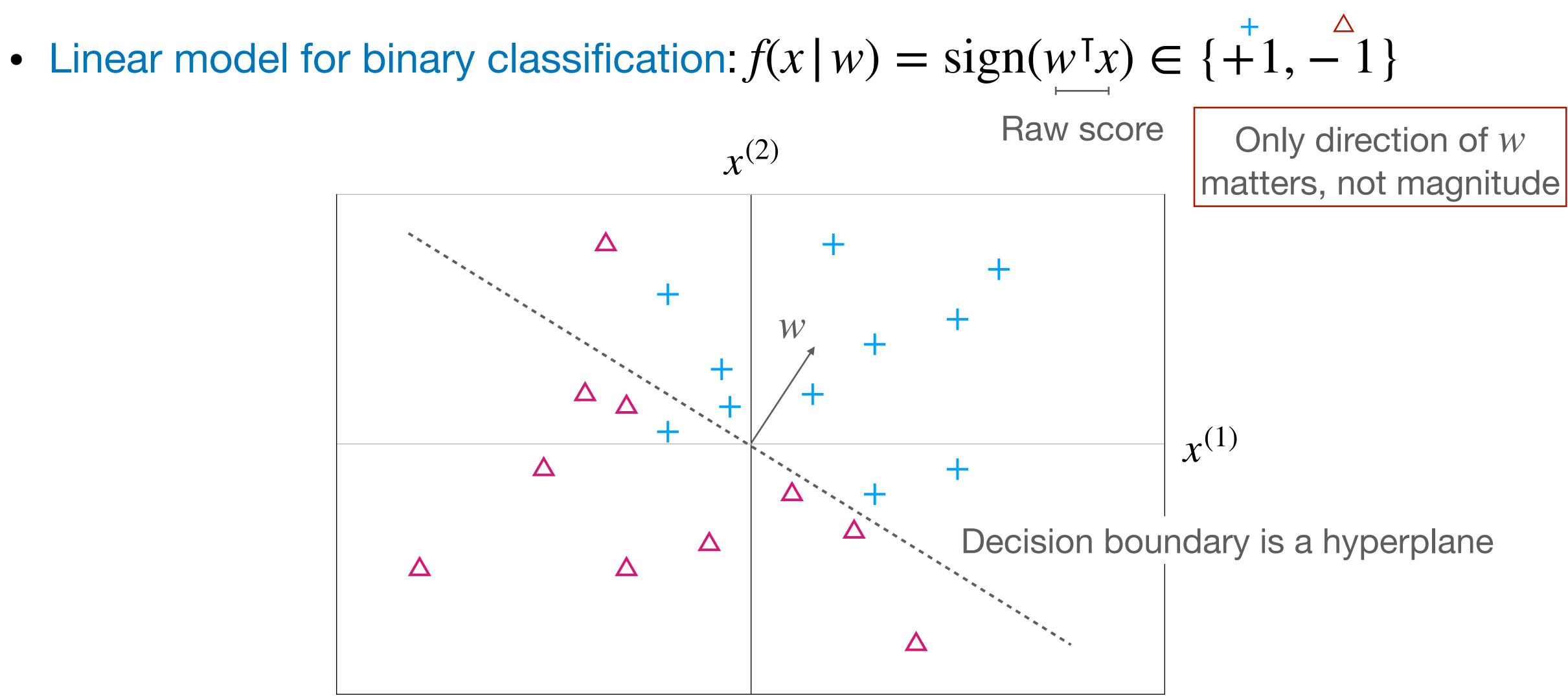
Important: if a testing set is available, never use it to make decisions on the model!

Today: Learning algorithms

- Perceptron learning algorithm
- (Stochastic) gradient descent
 - Solving the actual optimization problem in general
- How to view the perceptron learning algorithm as an example of stochastic gradient descent

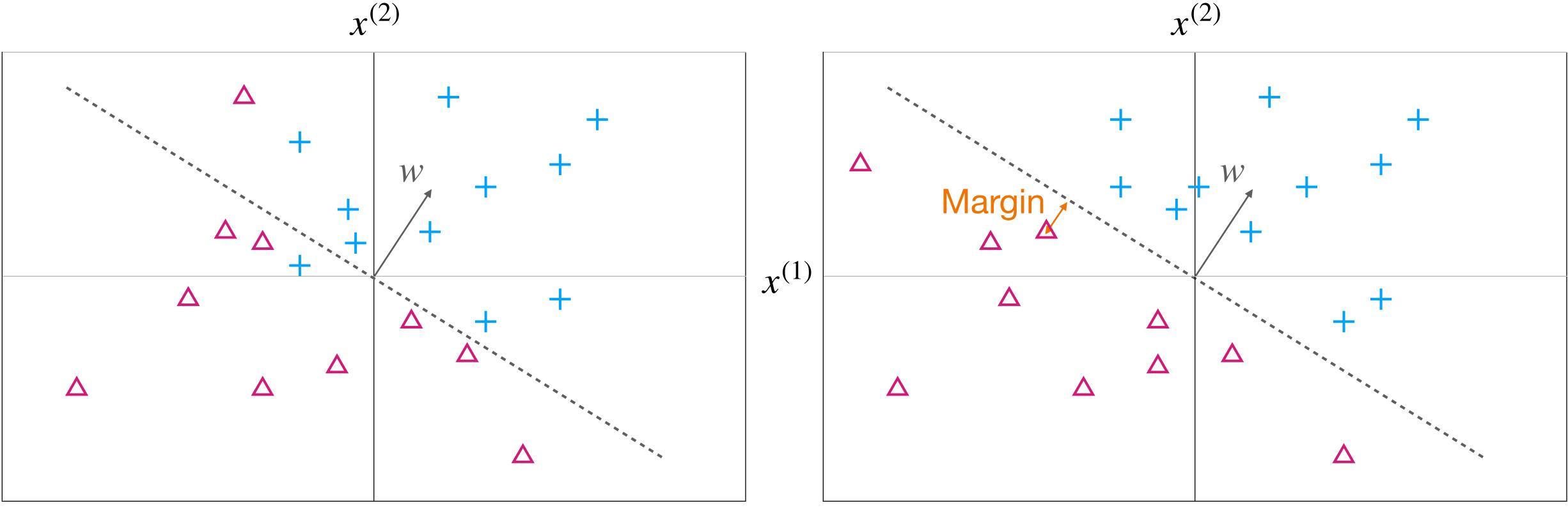
Linear models for binary classification

- Linear model for regression: $f(x \mid w) = w^{\mathsf{T}} x$



Linearly separable datasets

• Linear model for binary classification: $f(x | w) = sign(w^T x)$



Not linearly separable:

no hyperplane separates the classes perfectly

Linearly separable:

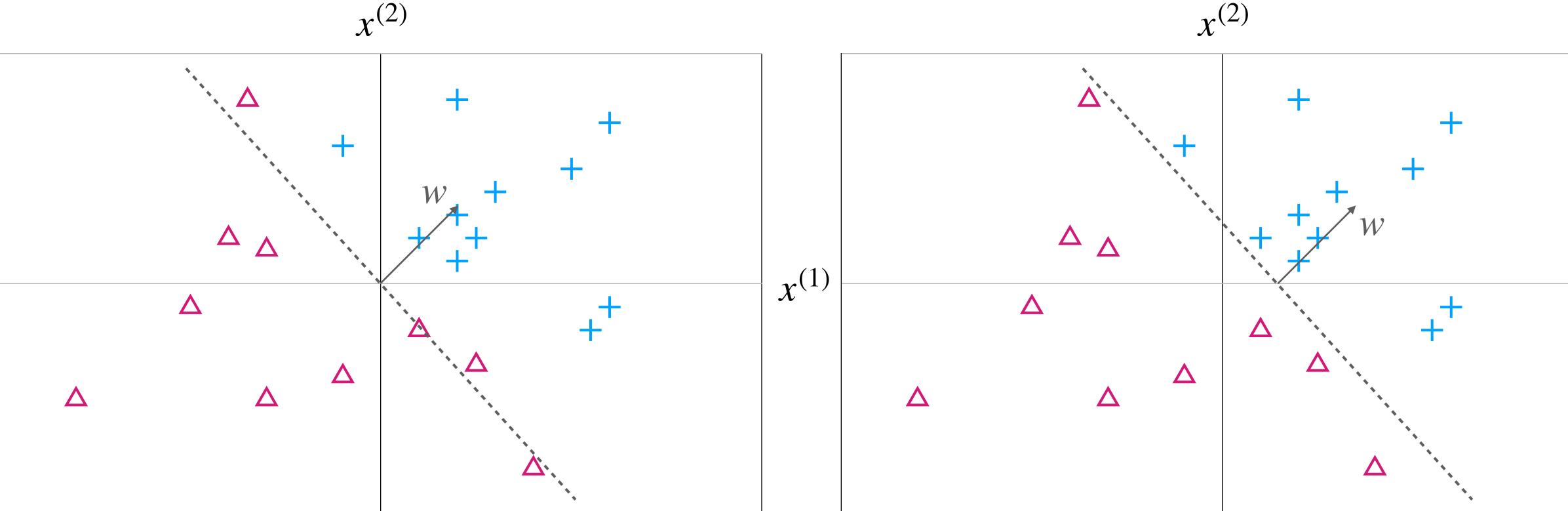
there is a hyperplane that separates the classes

Margin of the classifier = distance between the decision boundary and the closest data point





Linearly separable datasets



Not linearly separable without bias

• By adding a bias term, the decision boundary does not have to pass through the origin

Adding the bias (through dummy feature $x^{(0)} = 1$) makes it linearly separable





Perceptron

- One of the earliest learning algorithms
 - 1957 by Frank Rosenblatt [<u>10.1037/h0042519</u>]
- Still a great algorithm
 - Fast
 - Clean analysis
 - Precursor to neural networks

THE PERCEPTRON: A PROBABILISTIC MODEL FOR INFORMATION STORAGE AND ORGANIZATION IN THE BRAIN¹

F. ROSENBLATT

Cornell Aeronautical Laboratory

the capability of higher organisms for perceptual recognition, generalization, recall, and thinking, we must first have answers to three fundamental questions:

1. How is information about the physical world sensed, or detected, by the biological system?

2. In what form is information stored, or remembered?

3. How does information contained in storage, or in memory, influence recognition and behavior?

The first of these questions is in the province of sensory physiology, and is the only one for which appreciable understanding has been achieved. This article will be concerned primarily with the second and third questions, which are still subject to a vast amount of speculation, and where the few relevant facts currently supplied by neurophysiology have not yet been integrated into an acceptable theory.

With regard to the second question, two alternative positions have been maintained. The first suggests that storage of sensory information is in the form of coded representations or images, with some sort of one-to-one mapping between the sensory stimulus

¹ The development of this theory has been carried out at the Cornell Aeronautical Laboratory, Inc., under the sponsorship of the Office of Naval Research, Contract Nonr-2381 (00). This article is primarily an adaptation of material reported in Ref. 15, which constitutes the first full report on the program.

If we are eventually to understand and the stored pattern. According to this hypothesis, if one understood the code or "wiring diagram" of the nervous system, one should, in principle, be able to discover exactly what an organism remembers by reconstructing the original sensory patterns from the "memory traces" which they have left, much as we might develop a photographic negative, or translate the pattern of electrical charges in the "memory" of a digital computer. This hypothesis is appealing in its simplicity and ready intelligibility, and a large family of theoretical brain models has been developed around the idea of a coded, representational memory (2, 3, 9, 14). The alternative approach, which stems from the tradition of British empiricism, hazards the guess that the images of stimuli may never really be recorded at all, and that the central nervous system simply acts as an intricate switching network, where retention takes the form of new connections, or pathways, between centers of activity. In many of the more recent developments of this position (Hebb's "cell assembly," and Hull's "cortical anticipatory goal response," for example) the "responses" which are associated to stimuli may be entirely contained within the CNS itself. In this case the response represents an "idea" rather than an action. The important feature of this approach is that there is never any simple mapping of the stimulus into memory, according to some code which would permit its later reconstruction. Whatever in-

- Set w(t = 0) = 0
- At iteration t,
 - Receive example $(x, y) \checkmark$
 - If f(x | w(t)) = y (example is correctly classified)
 - w(t+1) = w(t) (no update)
 - Else
 - w(t+1) = w(t) + yx (update!)

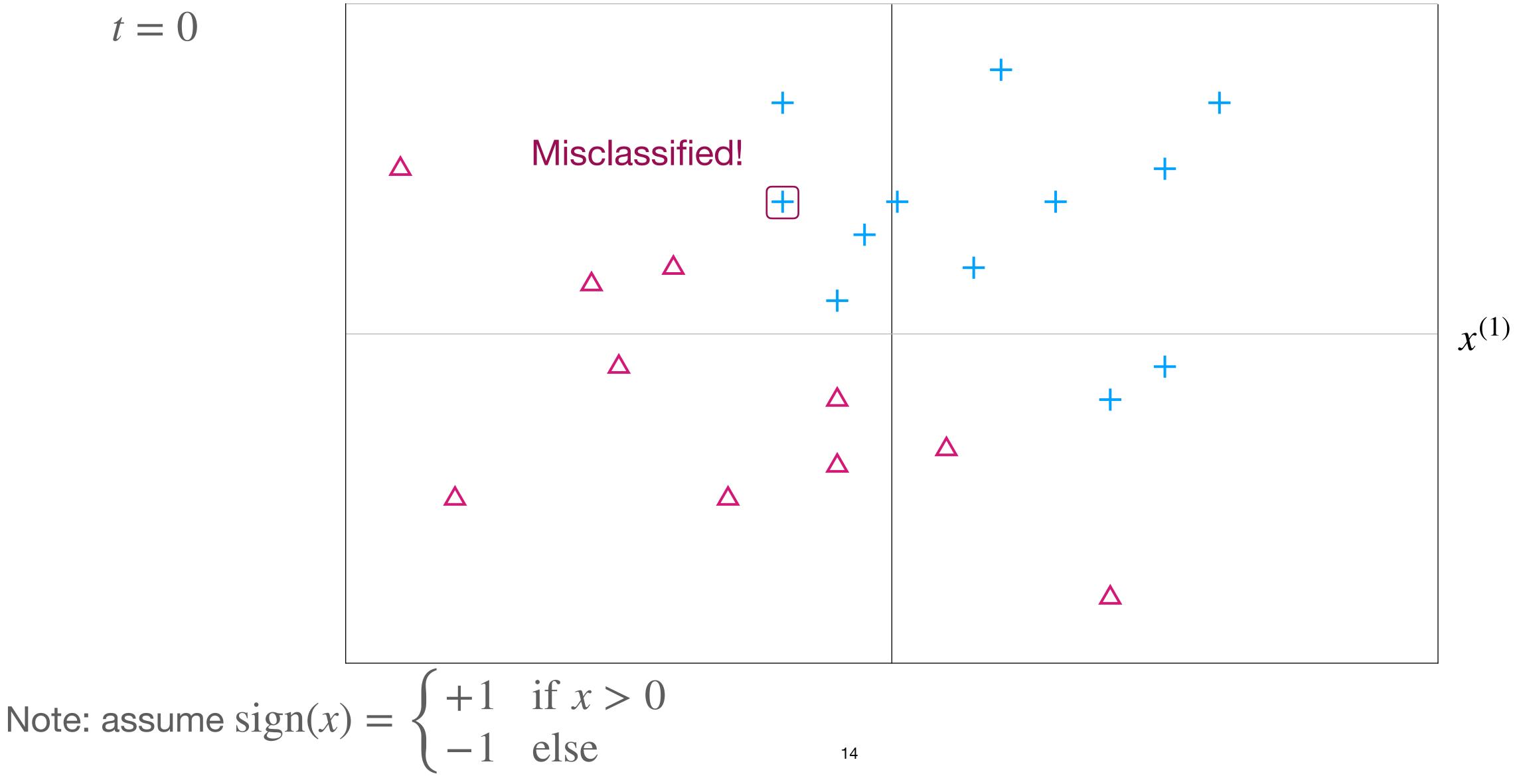
Model: $f(x \mid w) = \operatorname{sign}(w^{\mathsf{T}}x)$

Go through training set in an arbitrary order (e.g., randomly)

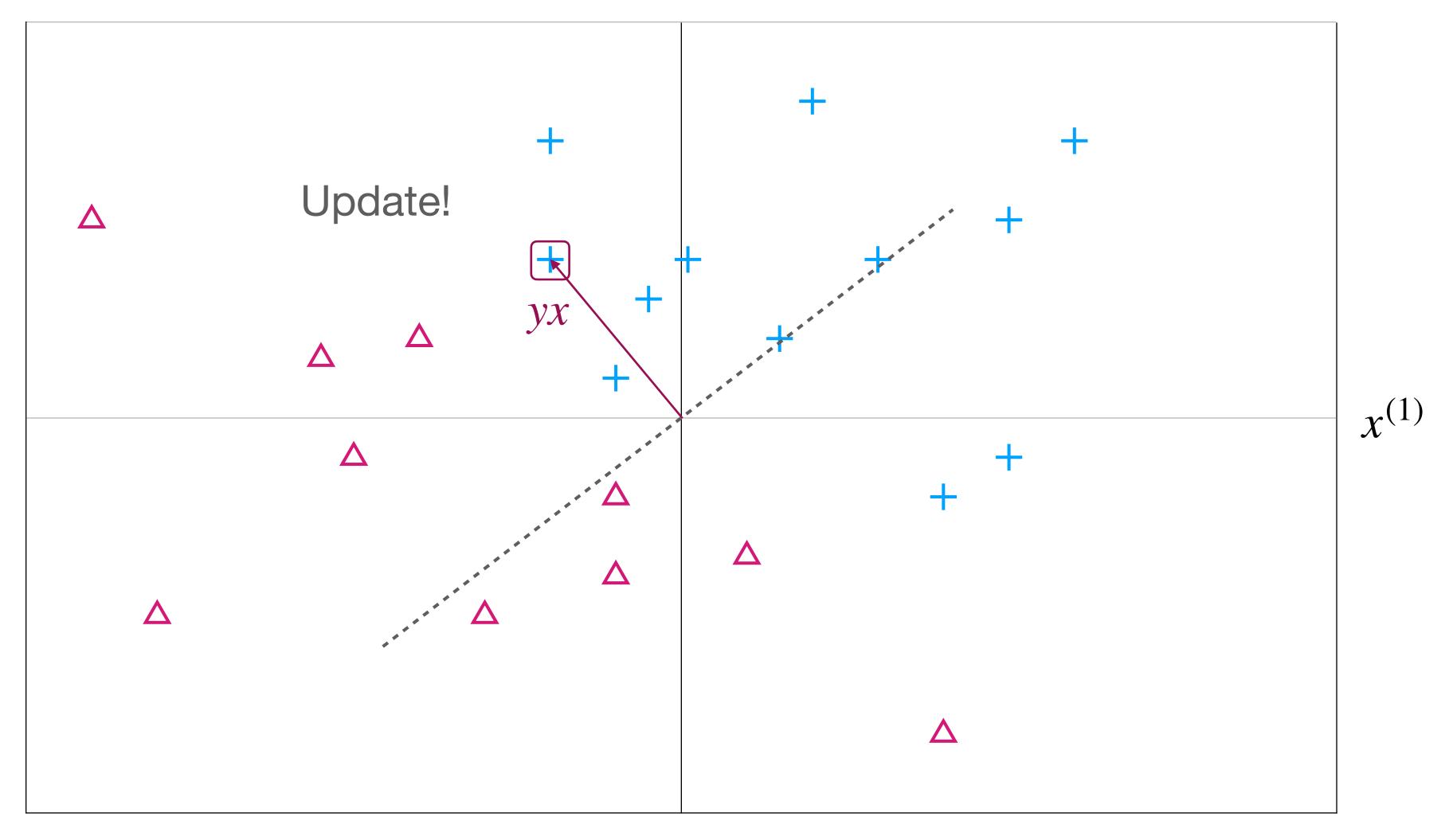
Training set: $S = \{(x_1, y_1), \dots, (x_N, y_N)\}$ $y \in \{+1, -1\}$

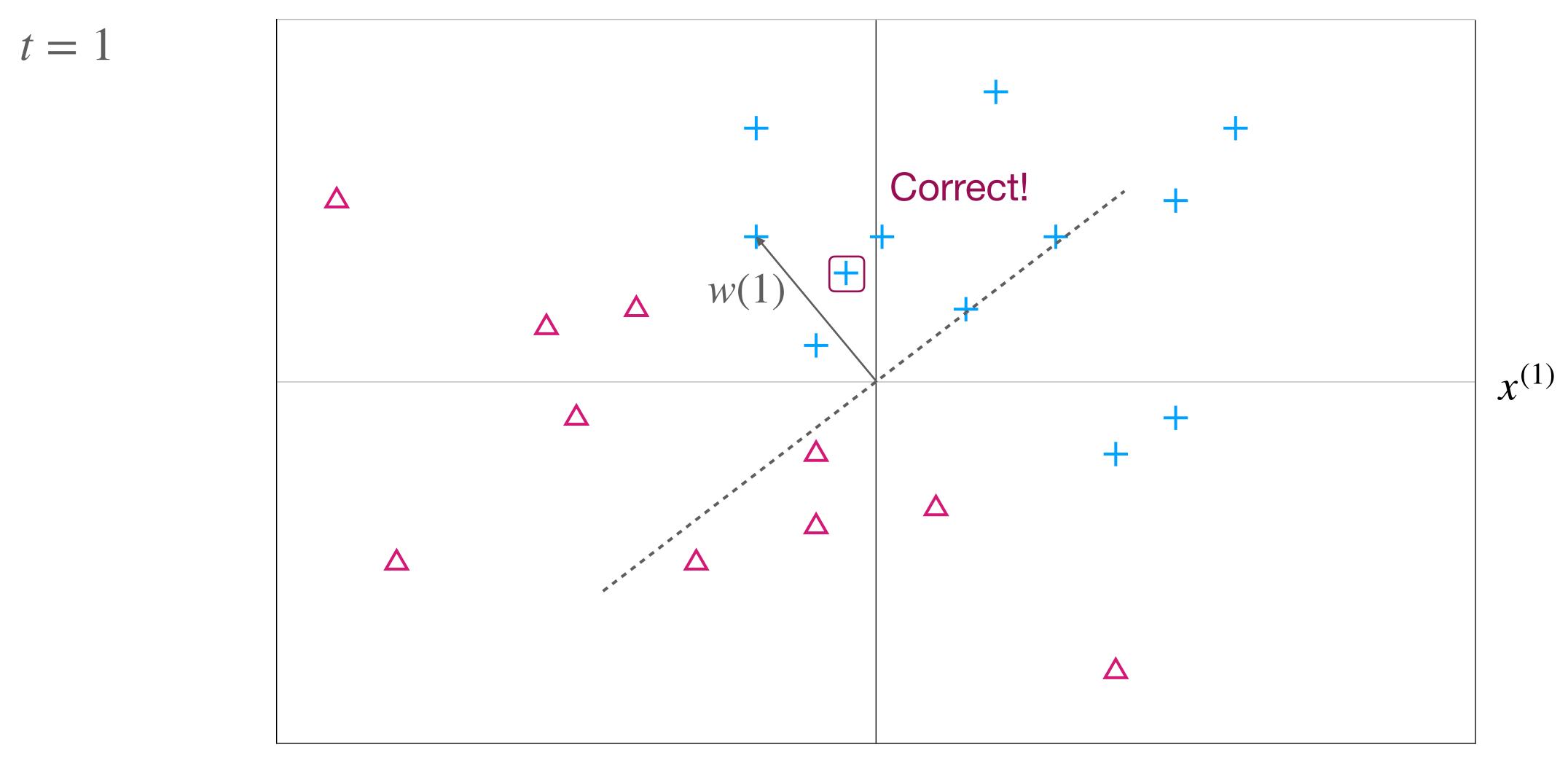




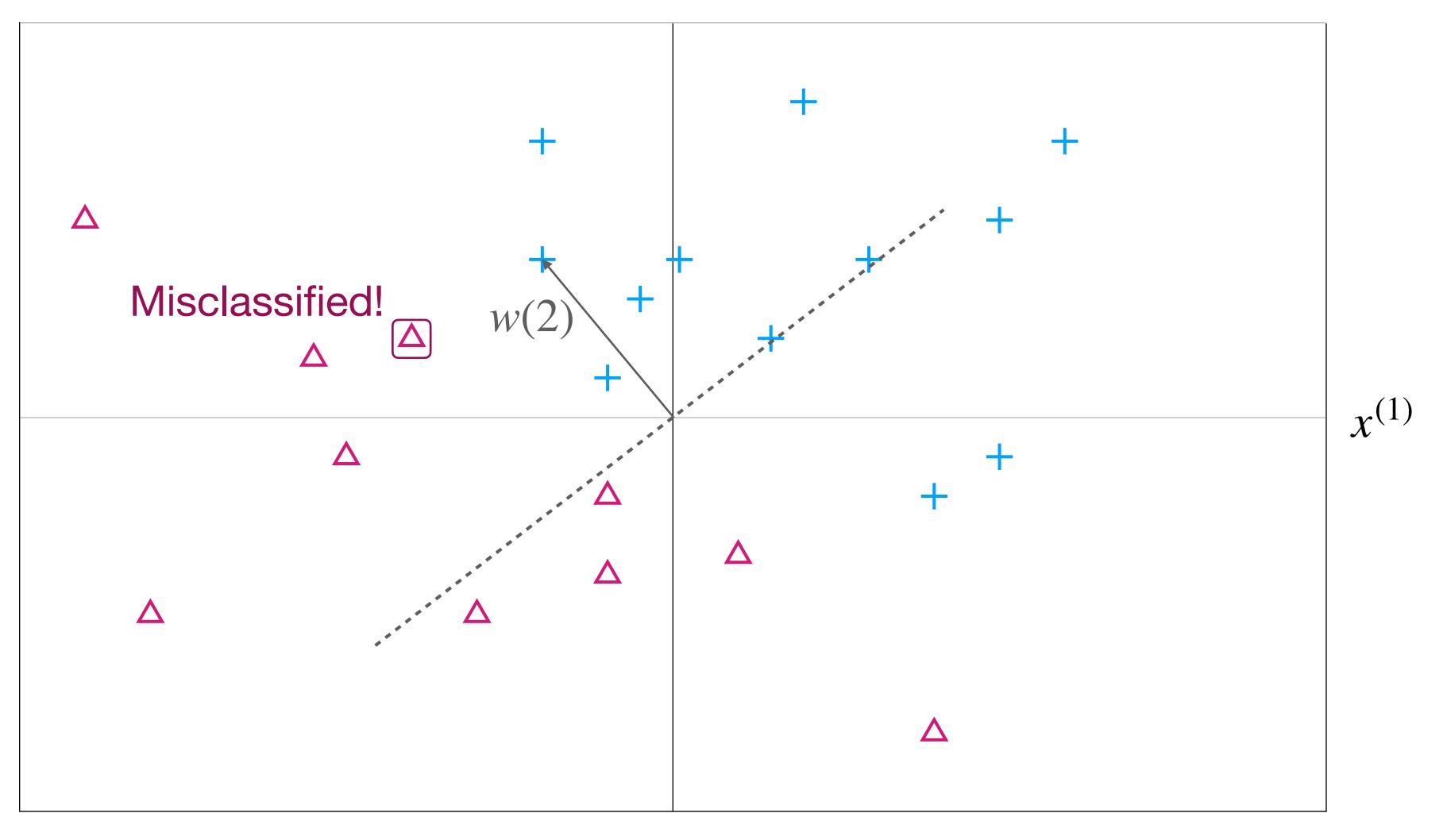


t = 0

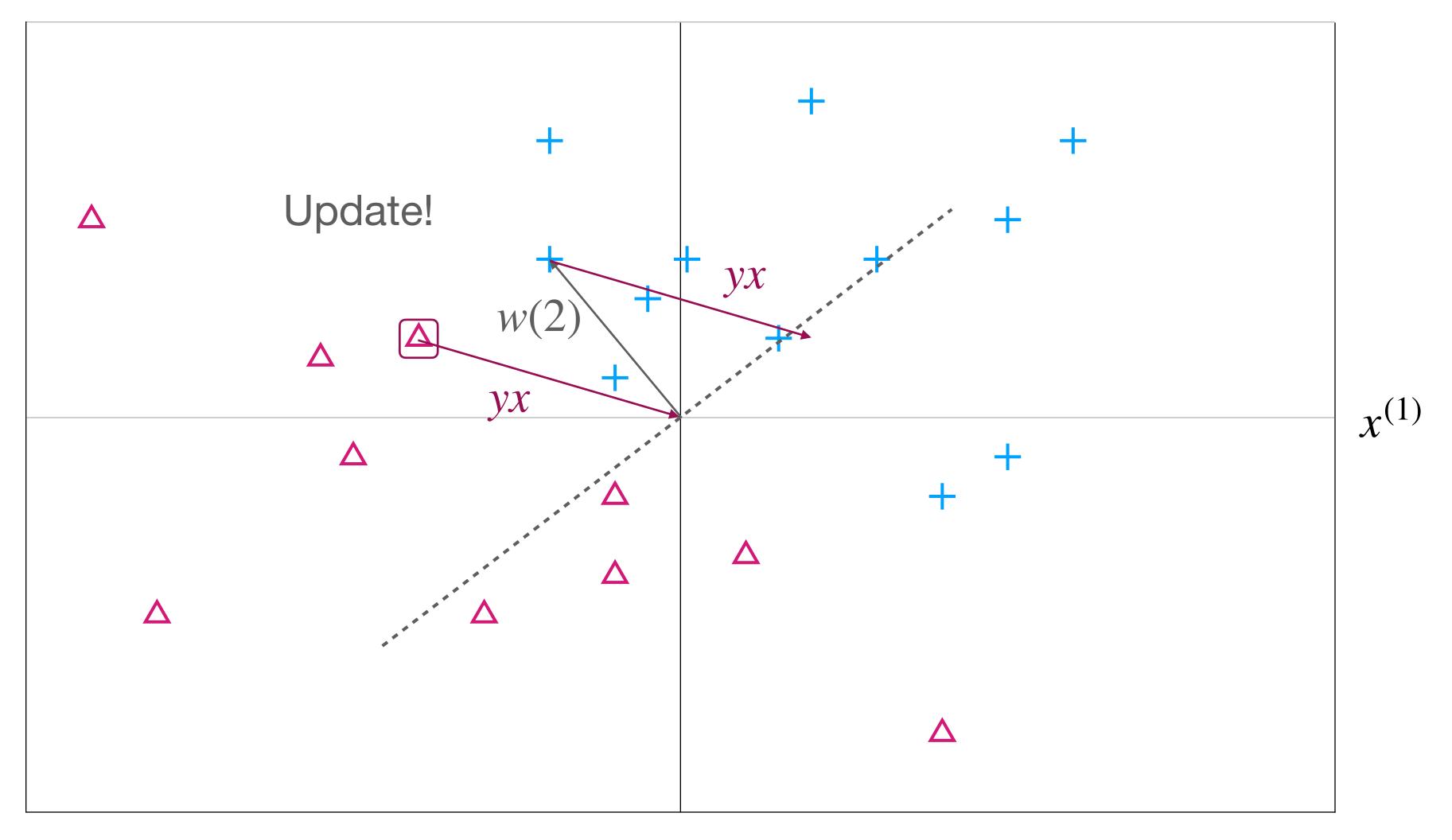




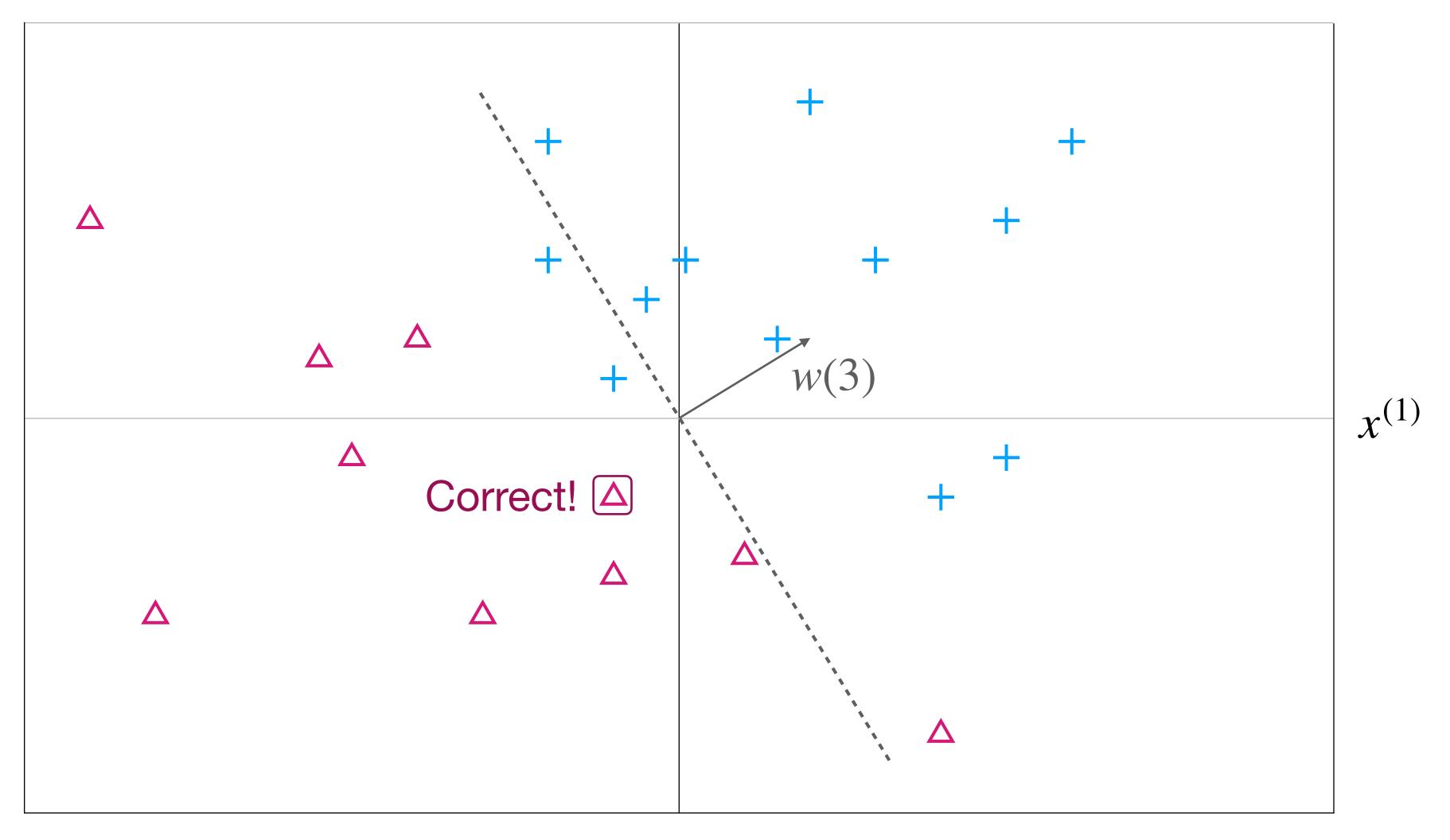




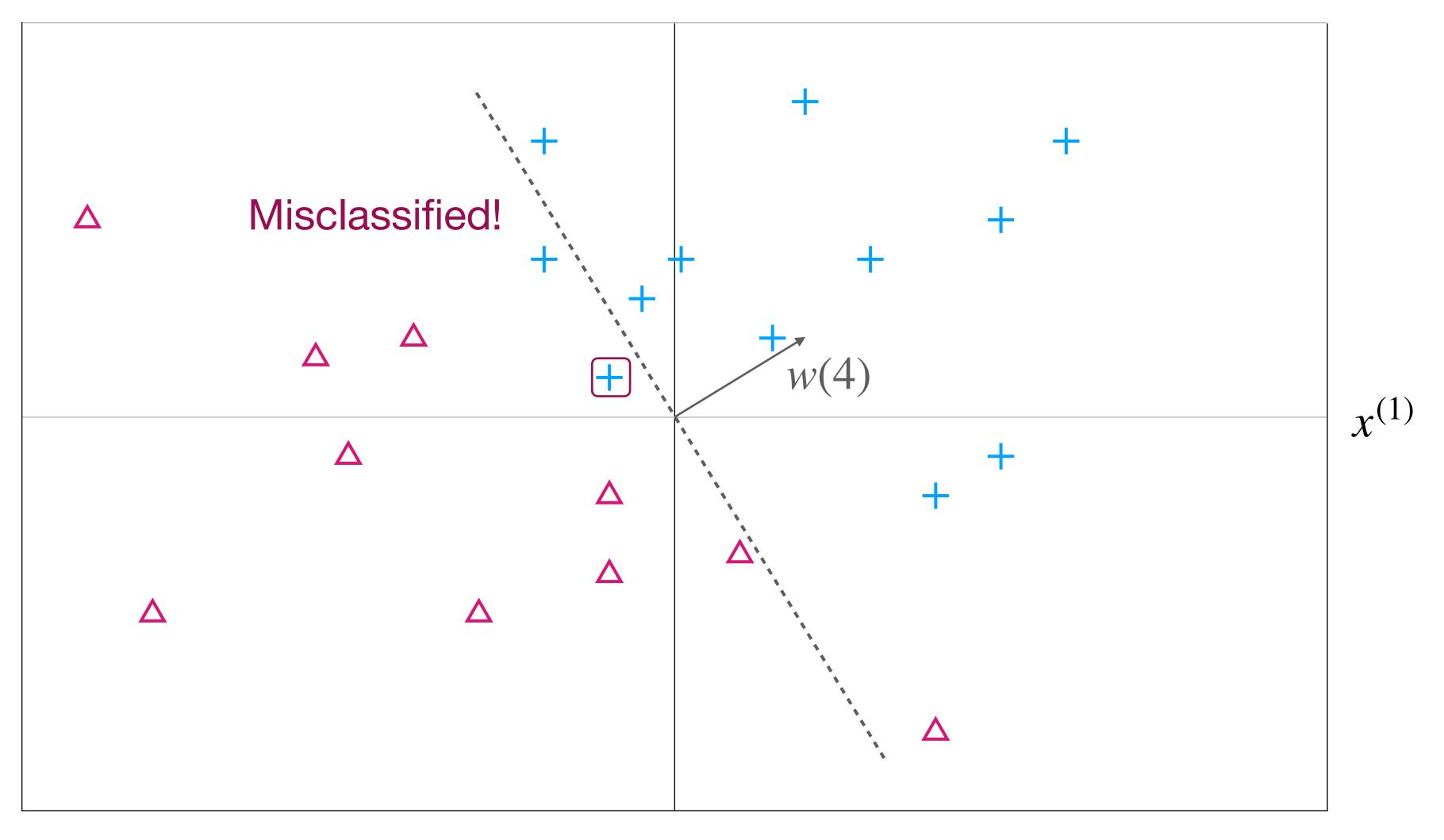
t = 2



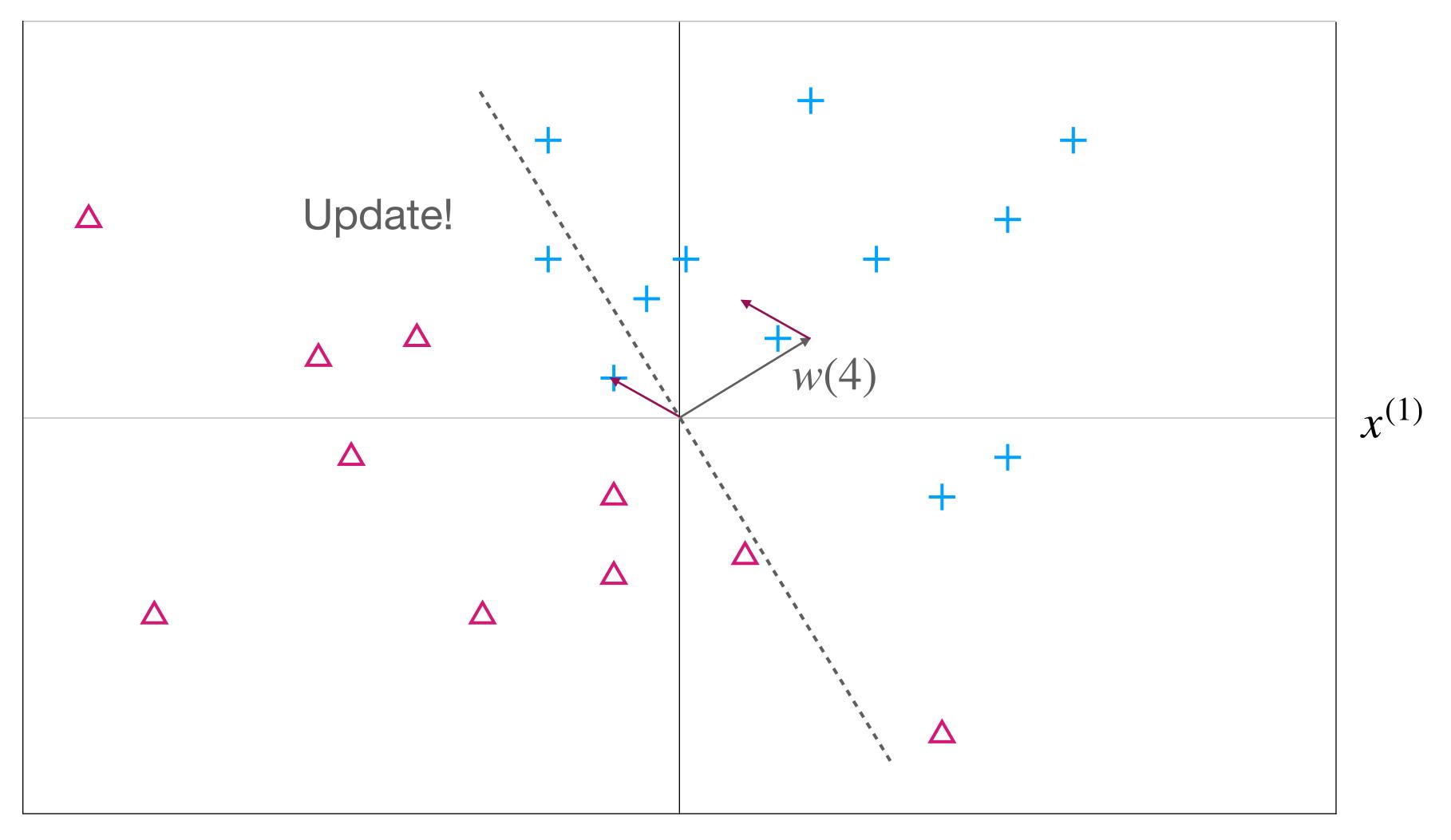
t = 3



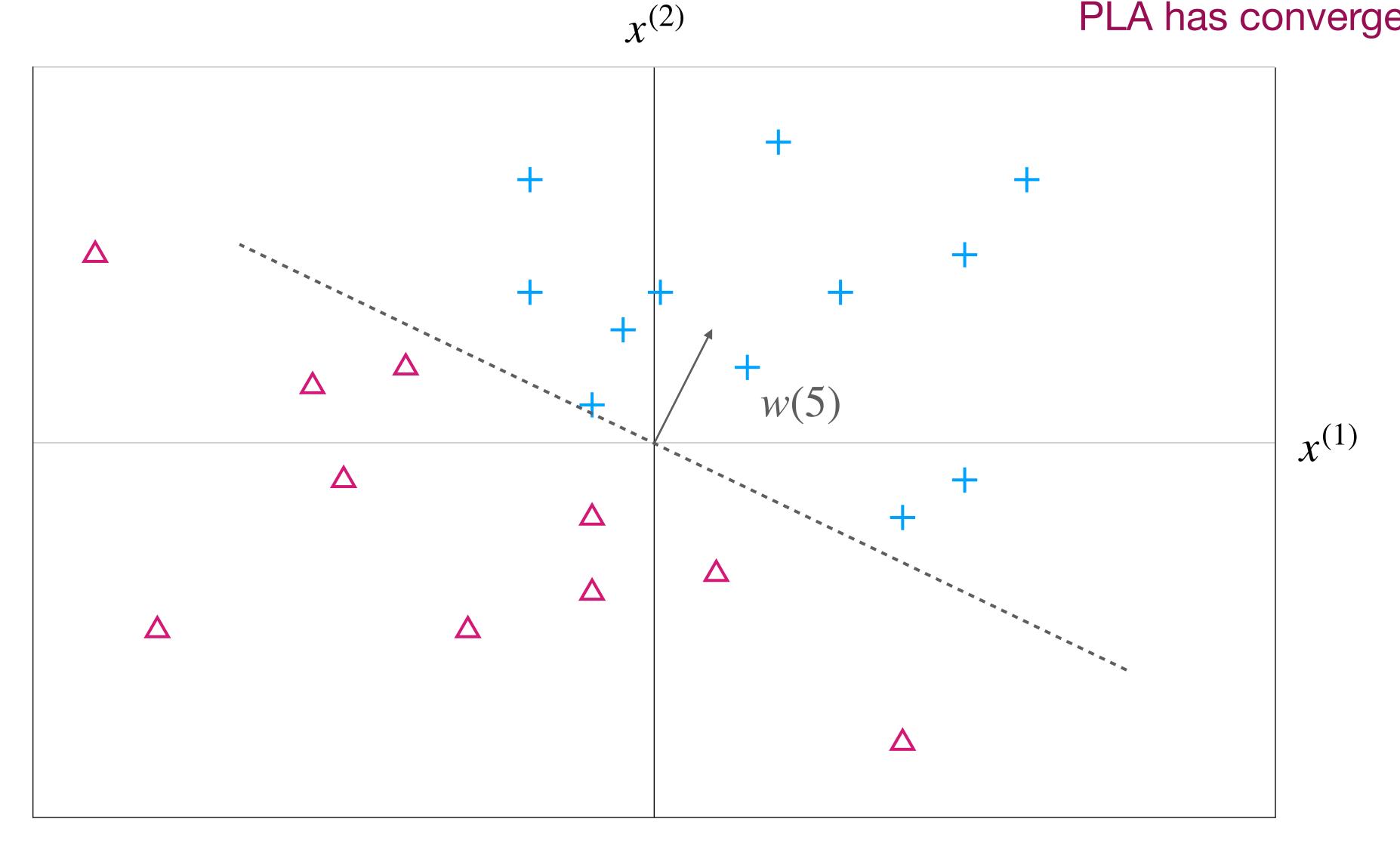
t = 4



t = 4



t = 5



All examples correctly classified! PLA has converged

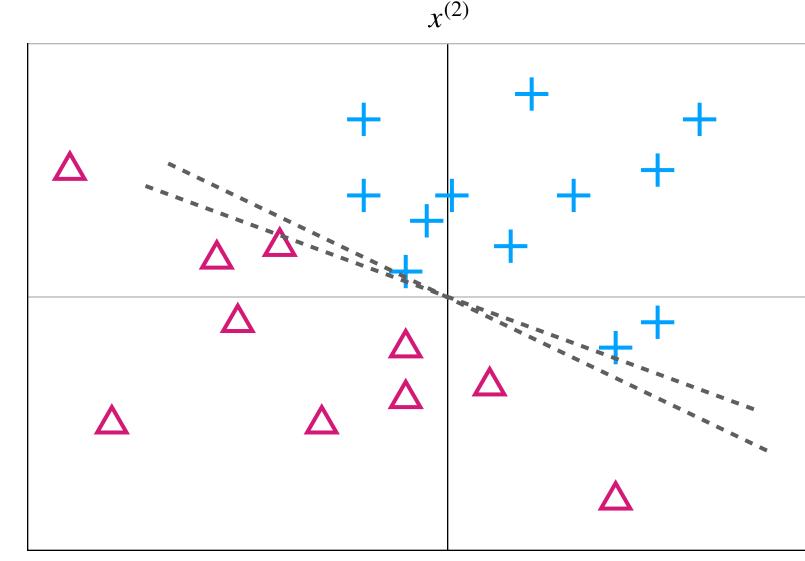


- Converges in a finite number of steps if the training dataset is linearly separable (but may take a long time)
- needed
- data points
 - And it may not be the "best" (maximum margin) model
- How can we recast PLA as an example of our supervised learning pipeline?

Two valid solutions: better one has a bigger margin, but PLA stops as soon as it finds any valid solution

• Does not converge if the dataset is not linearly separable, so early stopping is

The final model depends on the initialization and the order you traverse the







Optimizing the learning objective

In supervised learning, we want to optimize the objective \bullet

$$l(w) = \sum_{i=1}^{N} L(y_i, f(x_i | w))$$

- For linear regression, we had a closed-form solution, but in general?

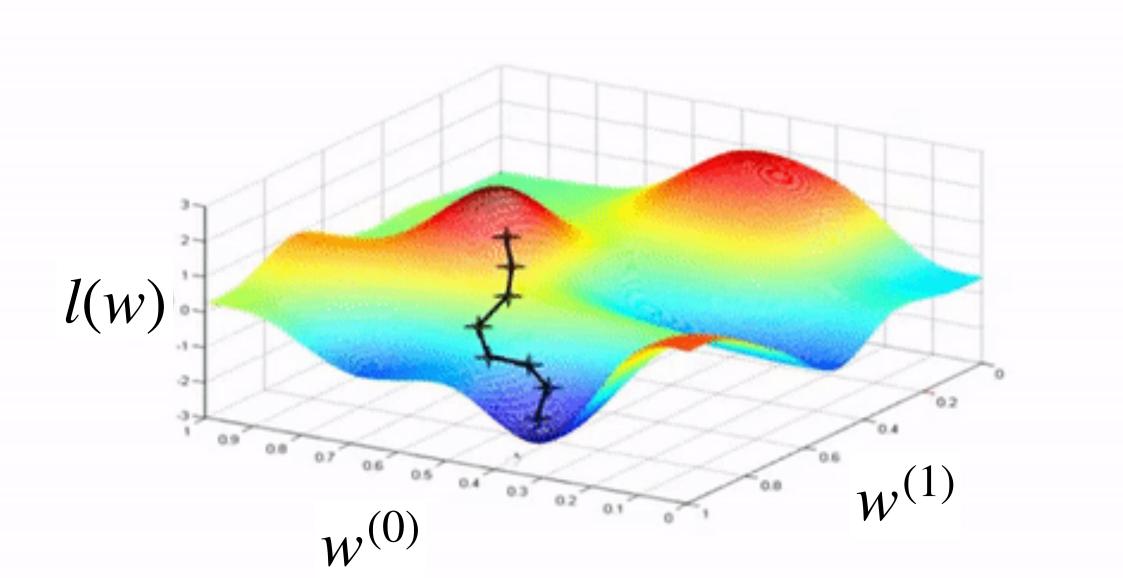
• We need an optimization algorithm to find the optimal (or just "good") w

Gradient descent

- Set w(t = 0) to some values (e.g., w(0) = 0 or some random value)
- At iteration *t*,
 - w(t)
 - Take a small step in the opposite direction: $w(t+1) = w(t) - \eta \nabla_w l(w(t))$

Step size / learning rate

• Compute the gradient $\nabla_w l(w(t))$: direction of steepest increase of l(w) at



Linear regression with gradient descent

• Learning objective:

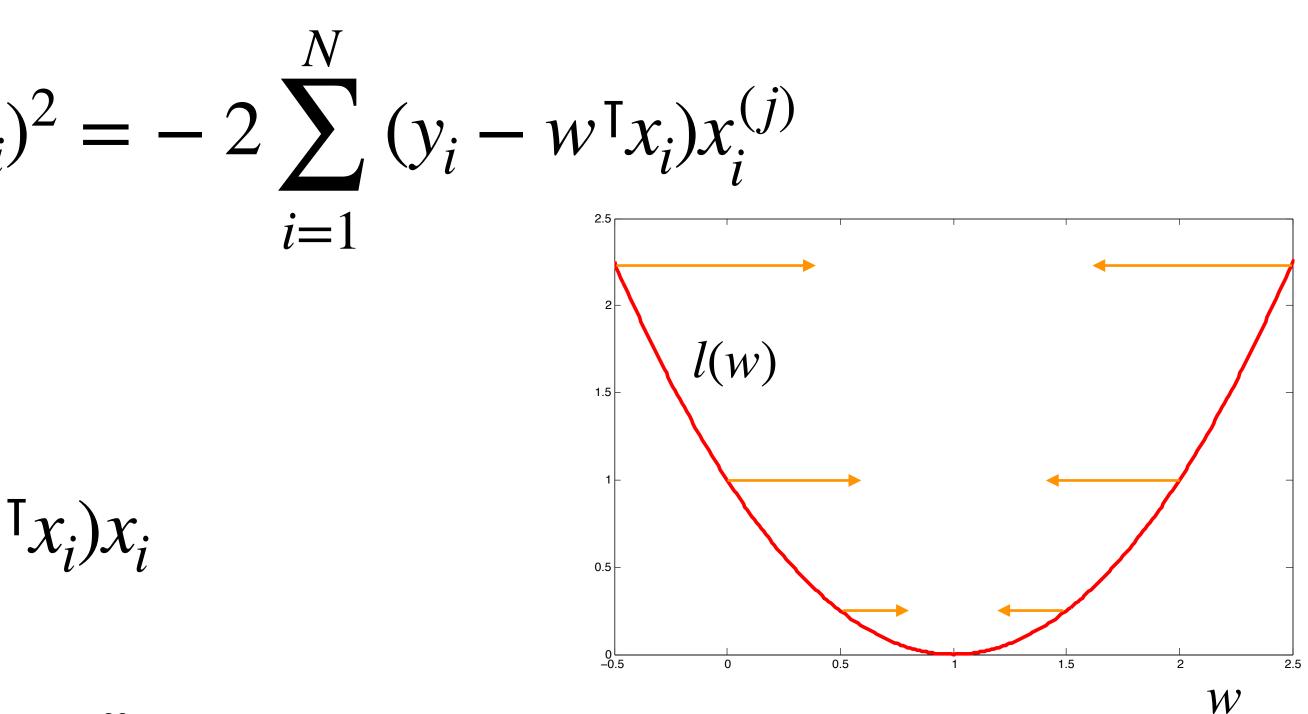
$$l(w) = \sum_{i=1}^{N} (y_i - w^{\mathsf{T}} x_i)^2$$

• Partial derivatives:

$$\frac{\partial}{\partial w^{(j)}} l(w) = \frac{\partial}{\partial w^{(j)}} \sum_{i=1}^{N} (y_i - w^{\mathsf{T}} x_i)$$

• Gradient descent update:

$$w(t+1) = w(t) + 2\eta \sum_{i=1}^{N} (y_i - w^{\mathsf{T}})$$



Limitation of gradient descent

• Requires a full pass over the training dataset at each iteration:

$$w(t+1) = w(t) - \eta \nabla_w \sum_{i=1}^N L(y_i, f(x_i))$$

$$l(w)$$

• Prohibitively expensive if the training dataset is large!

 $(x_i, |w))$

Stochastic gradient descent

• The learning objective decomposes additively:

$$l(w) = \frac{1}{N} \sum_{i=1}^{N} L(y_i, f(x_i, |w)) = \mathbb{E}_{(x, y_i)}$$

add normalization

• The total gradient is the expected gradient of the single-example losses:

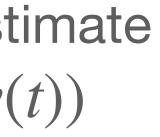
$$\nabla_{w} l(w) = \nabla_{w} \mathbb{E}_{(x,y) \in S} \left[L(y, f(x \mid w)) \right] = \mathbb{E}_{(x,y) \in S} \left[\nabla_{w} L(y, f(x \mid w)) \right]$$

Gradient descent update: $w(t+1) = w(t) - \eta \nabla_{w} l(w(t))$
SGD update: $w(t+1) = w(t) - \eta \nabla_{w} L(y, f(x \mid w(t)))$ Unbiased esonof $\nabla_{w} l(w(t))$

- (-
- \$

$y \in S \left[L(y, f(x \mid w)) \right]$

for a random $(x, y) \in S$



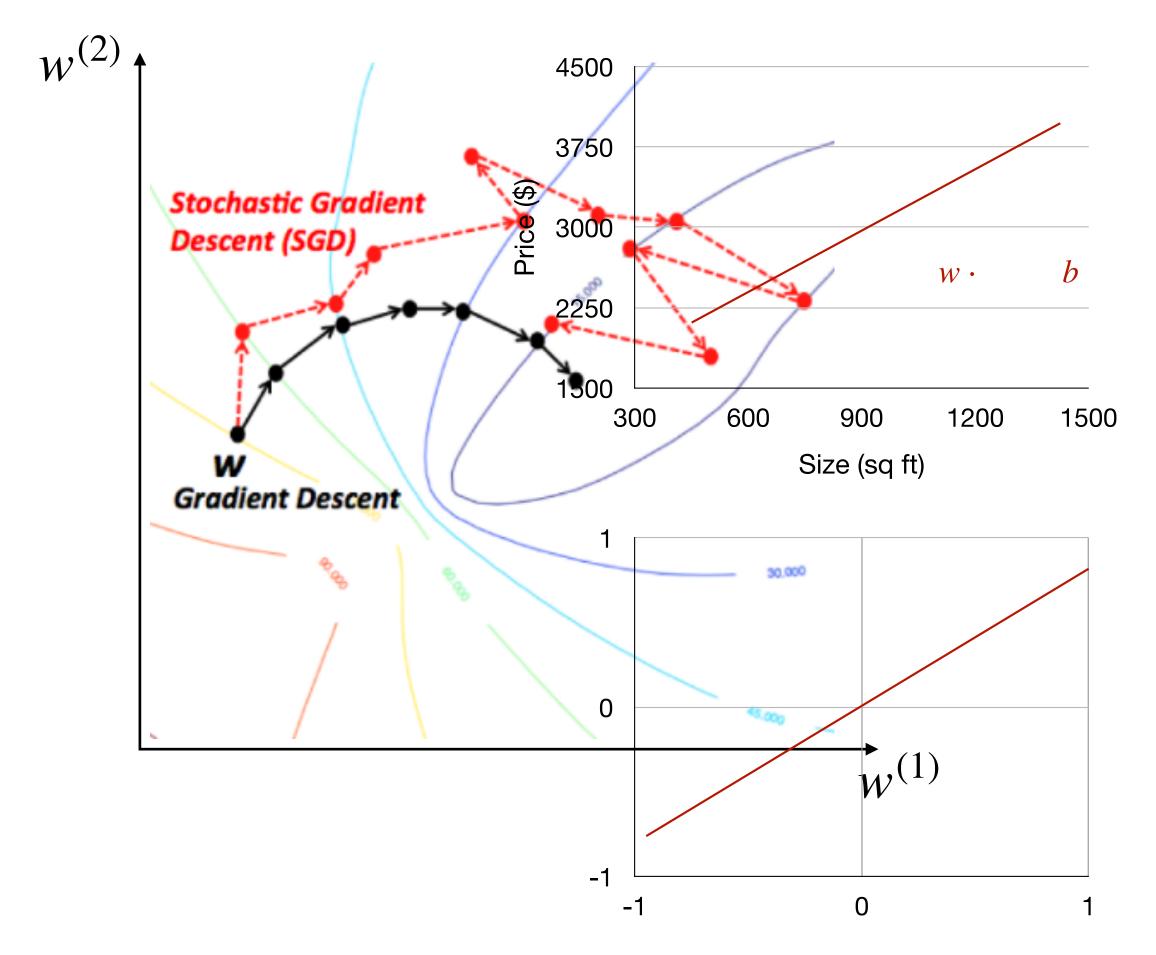
Stochastic gradient descent

- In practice, we use mini-batch SGD: compute gradient on a mini-batch of training example (e.g., 8, 16, or 32 examples)
 - Leverages fast vector operations (especially on GPU)
 - Decreases volatility of gradient updates (but there is still some **useful** noise)
 - Can be parallelized (e.g. different cores compute gradients on different minibatches)
 - Useful also for least-squares regression on large datasets
- Note: no need to check validation error at every iteration

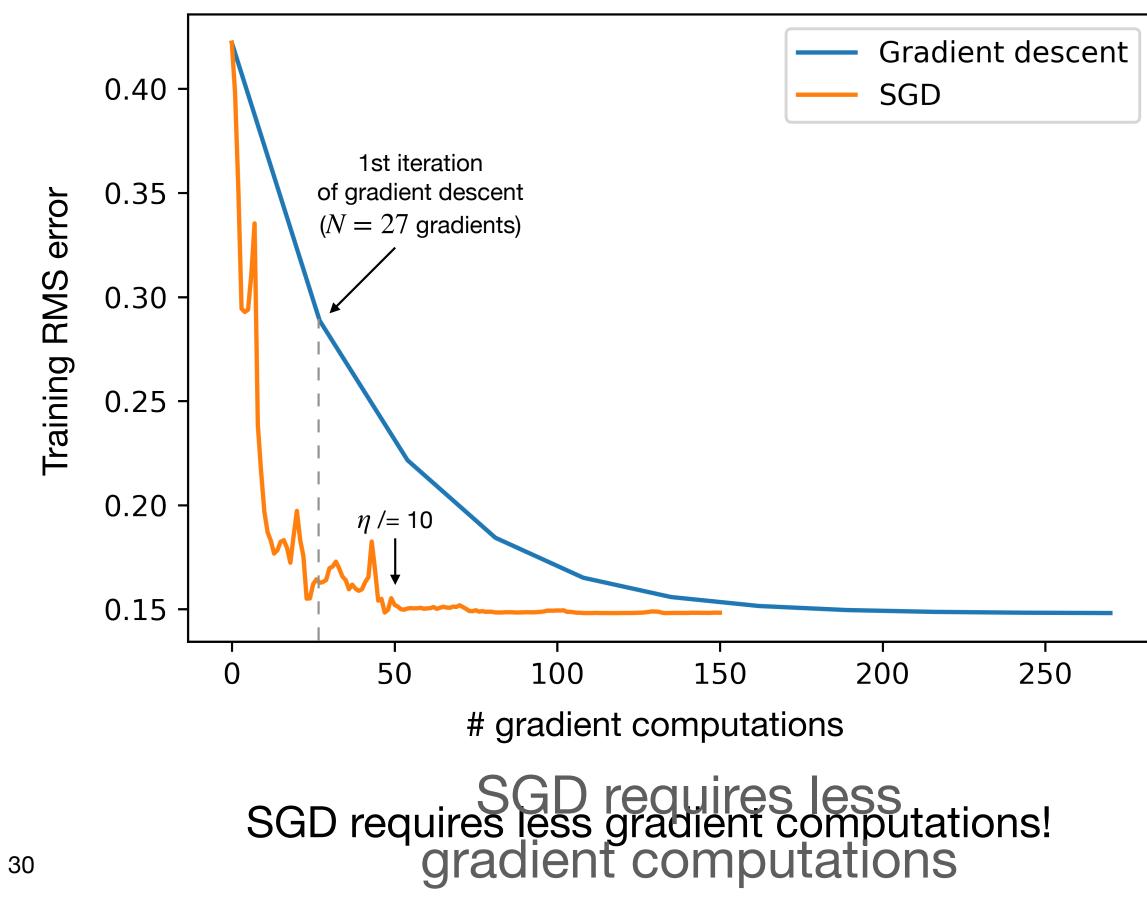
• SGD is an **online** optimization algorithm (only needs one example at a time)

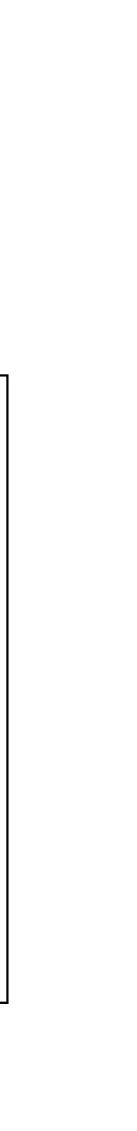
Gradient descent vs. SGD

- Gradient descent update: $w(t + 1) = w(t) \eta \nabla_w l(w(t))$
- SGD update:



$w(t+1) = w(t) - \eta \nabla_{w} L(y, f(x \mid w(t)))$ $(x, y) \in S$ (x, y) $\in S$





Updated: Supervised learning pipeline

- Training dataset: $S = \{(x_1, y_1), \dots, (x_N, y_N)\}$ where $x \in \mathbb{R}^D$ and $y \in \mathbb{R}$ • Model / hypothesis class: $f(x | w) = w^{\mathsf{T}} x$ (linear models) For regression • Loss function: $L(y, y') = (y - y')^2$ (squared loss) or $\phi(x)$ instead of x
- Optimization algorithm: SGD
- Cross validation and model selection:
- Testing and deployment

Important: if a testing set is available, never use it to make decisions on the model!



Optimizing the linear classification model

The most straightforward loss for classification is the 0/1 loss

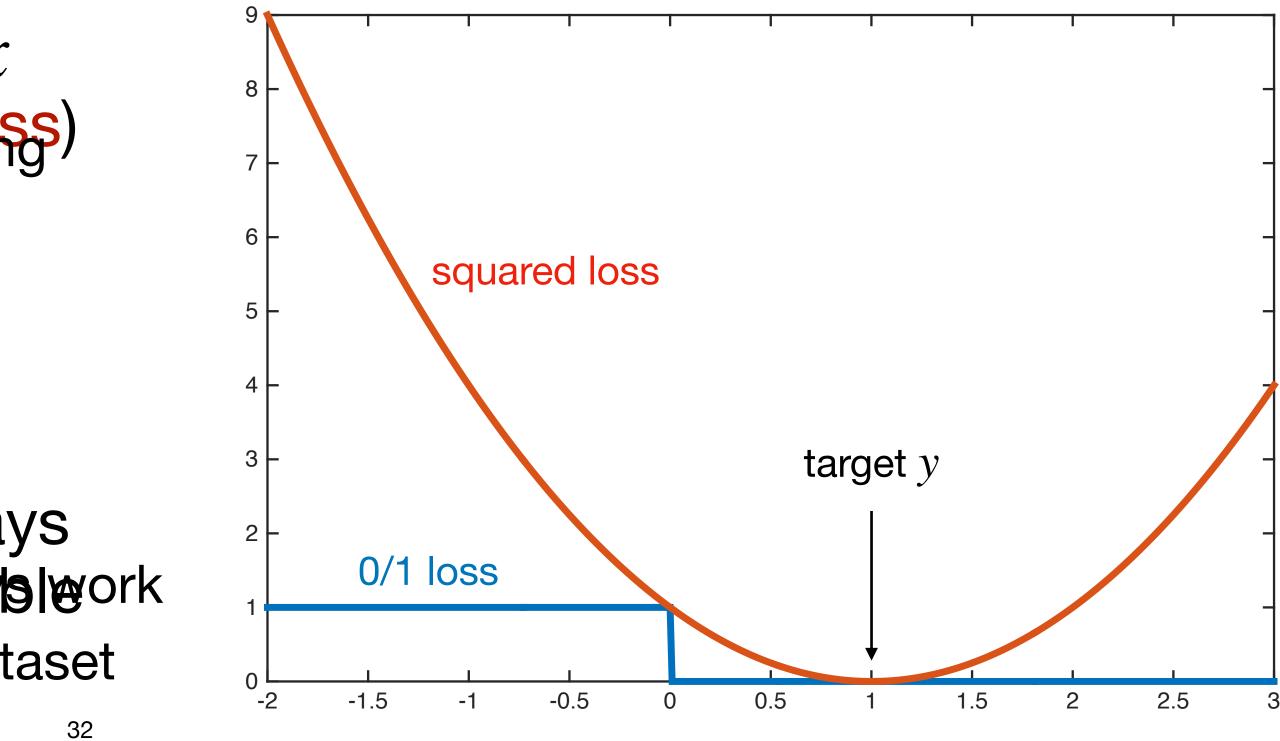
- We can optimize the raw score $w^{T}x$ using another loss (avgscore with dusing)

another loss (e.g., squared loss)

$$\arg\min_{w} \sum_{i=1}^{W} \frac{(y_i - w k_i)^2}{\arg\min_{w} \sum_{i=1}^{W} (y_i - w^t x_i)^2}$$

But the squared loss does not always work Bwethesepared asineers, needed and sineers, needed and sineer dataset ll, even on a linearly separable dataset

Good to evaluate the validation/test error, difficult to optimize (gradient is 0)



Perceptron algorithm revisited

An alternative is the perceptron loss

$$L(y, y'_{y'}) = \begin{cases} 0 & \text{sign}(y') = y \\ -yy' & \text{otherwise} \end{cases}$$

(0 if correct, -y' if y = +1, +y' if y = -1)

 Running SGD with this loss yields the period strong to the strong the strong the strong to t

perceptron algorithm:

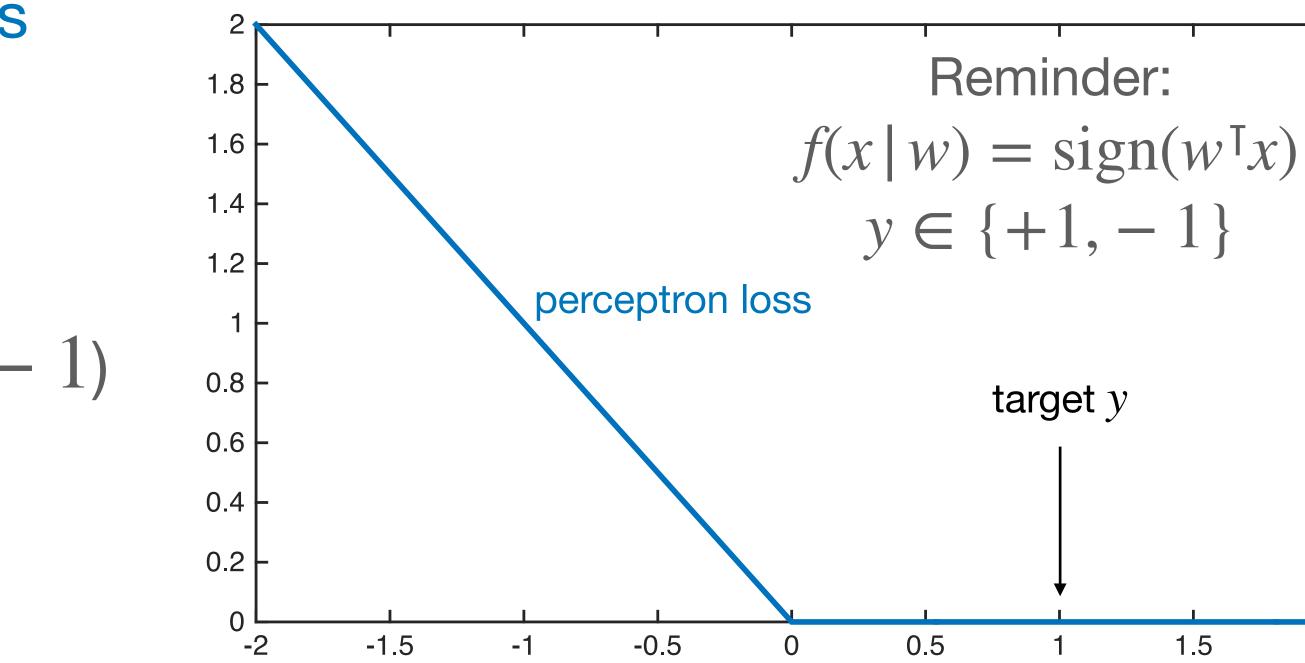
- Initialize w(t=0) = 0 $w_0 = 0$

$$w_{t+1} = w_t + \eta y x$$

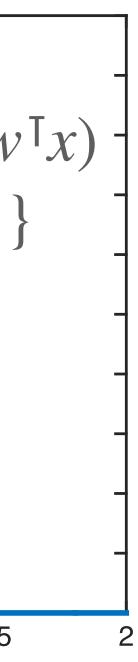
$$w_{t+1} = w_t + \eta yx \quad \text{(we can set } n = 1 \text{ because the magnitude of } w \text{ does not } m \text{ COrrect}$$

$$w(t+1) = w(t) - \eta \nabla_w L(y, f(x \mid w(t))) = \begin{cases} w(t) & \text{COrrect} \\ w(t) & \text{Correct} \end{cases}$$

(we can set $\eta = 1$ because the magnitude of *w* does not matter)



• For iteration, \mathcal{L} , pick a random example (x, y) and perform an update



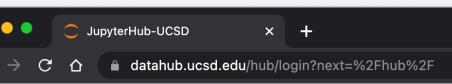


Next time

- More on (stochastic) gradient descent
- Different loss functions
 - Hinge loss (support vector machine)
 - Log loss / cross entropy loss (logistic regression)

DataHub

- We will use DataHub for inclass hands-on portions
 - Recommend to use it for homework, final project, etc.
- Address: datahub.ucsd.edu
- Similar to public, free services Google Colab, but with access to better CPUs and GPUs and run by UCSD
- Provides a "Jupyter notebook" interface (Python-based but interactive coding like MATLAB/Mathematica)

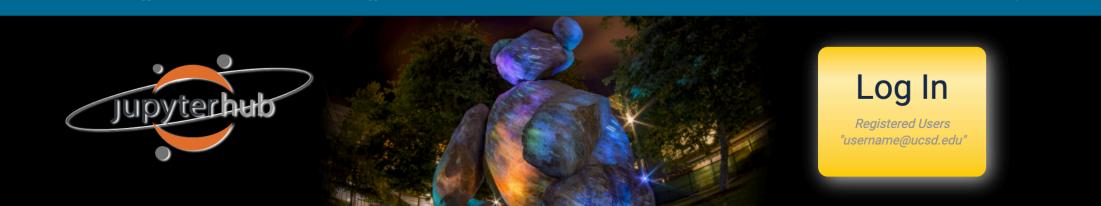


DATA SCIENCE / MACHINE LEARNING PLATFORM

UC San Diego

Help - FAQ

Information Technology Services - Academic Technology Services



UC San Diego Jupyterhub (Data Science) Platform

If you are unable to log in: Please try opening a private/incognito window in your browser | FAQ

Student Resources

- Datahub/DSMLP Cluster Status
- Independent Study Access Request
- Data Science Resources
- Datahub/DSMLP Knowledge Base
 - Launching Containers from the Command Line
 - Configuring Your Container Launch
 - Building Your Own Custom Image

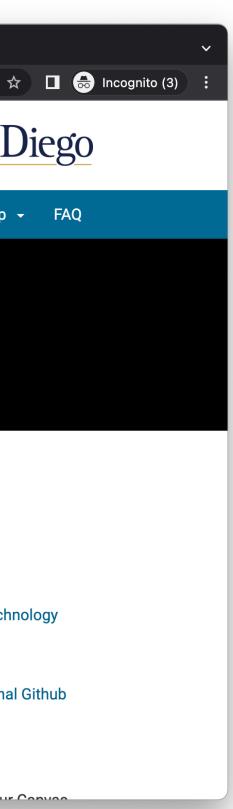
Transforring Eiles and Date

Instructor Resources

- Request Datahub/DSMLP Instructional Technology Request (CINFO)
- Instructor Guidance for Datahub/DSMLP
- Educational Technology Services Instructional Github

Suno applicamento and gradoo with your Conve

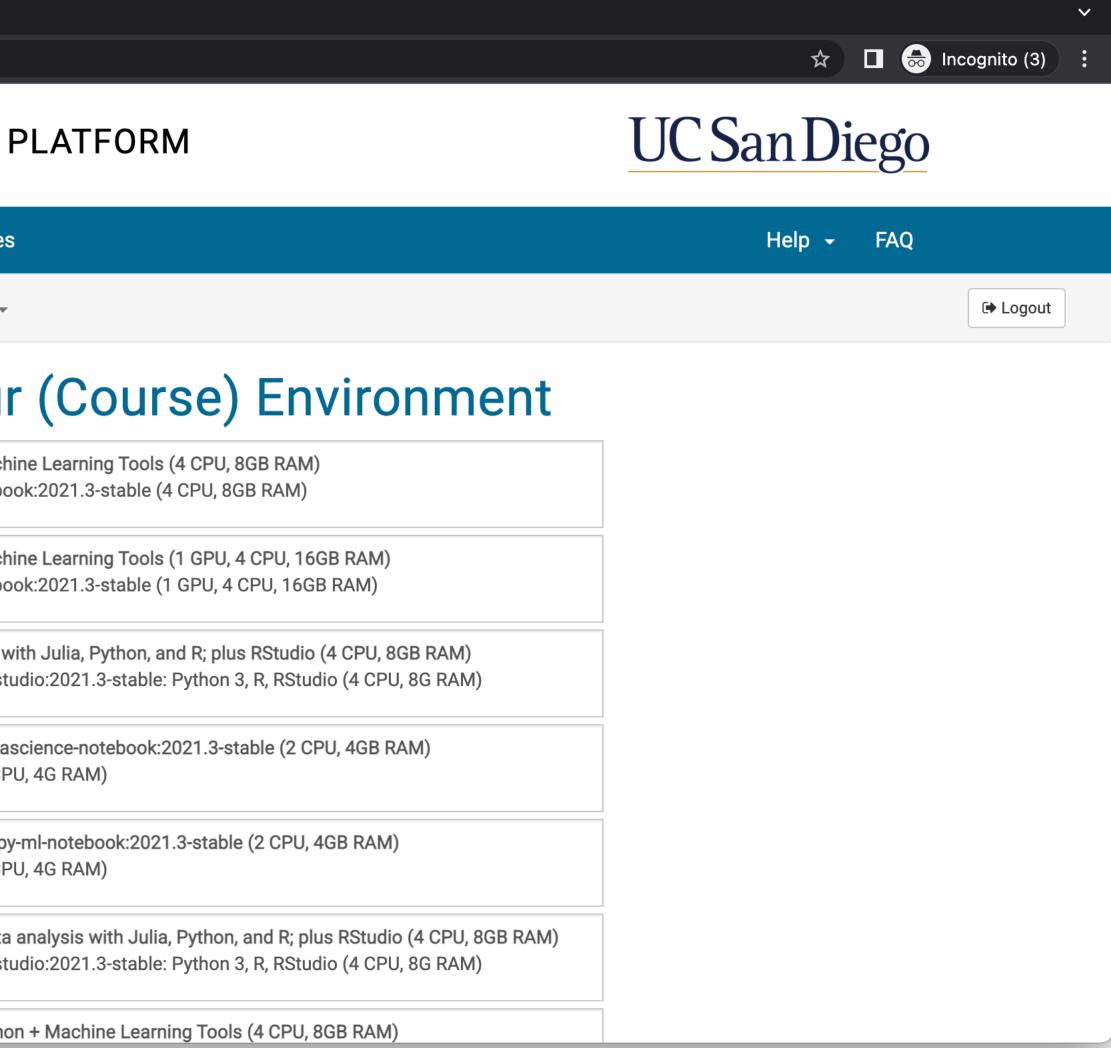
- Blink Documentation
- Datahub Grading Tools
 - nbgrader



Logging in

• After logging in, choose a course environment... PHYS 139 PHYS 239 - Special Topics - Machine Learning - Duarte [WI23], ucsdets/scipy-ml-notebook:2022.3-stable, (2 CPU, 4G RAM)

DATA	SCIENCE	E / MA	CHIN	E LEA	RNING I
Informat	ion Technolog	y Service:	s - Acader	nic Techno	logy Services
С Jupyterhub но	me Token	DSML	P Status	News	Services -
			S	elec	t Your
			0		Python + Mach scipy-ml-notebo
			0		Python + Machi scipy-ml-notebo
			0		Data analysis w latascience-rstu
			0		: ucsdets/datas nbgrader (2 CPI
			0		: ucsdets/scipy nbgrader (2 CP
			0		: RStudio: Data latascience-rstu
			0	scipy-ml:	Scientific Pytho



Jupyter interface

• Spawns a "JupyterHub" (let's go step by step through all the buttons)

File Running Custers Custers Assignments Upter# Time Running Custers Custers Time Provide the size Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Custers Image: Cus	Files Running Clusters Courses Announcements Assignments Select items to perform actions on them. Upload New • C O • • • / Capstone-particle-physics-domain 6 months ago	> C 合	ub.ucsd.edu/user/jduar	rte/tree?						☆ □ (👼 Incogi
Select items to perform actions on them. Image: Comparison of them Image: Comparison of them Image: Comparison of them Image: Com <th>Select items to perform actions on them. Upload New • 0 • • Name • Last Modified File size capstone-particle-physics-domain</th> <th>💭 Jul</th> <th>oyter<mark>hub</mark></th> <th></th> <th></th> <th></th> <th></th> <th></th> <th>Logout</th> <th>ontrol Panel</th> <th></th>	Select items to perform actions on them. Upload New • 0 • • Name • Last Modified File size capstone-particle-physics-domain	💭 Jul	oyter <mark>hub</mark>						Logout	ontrol Panel	
□ □ Name ↓ Last Modified File size □ capstone-particle-physics-domain 6 6	□ 0 - Name ↓ Last Modified □ capstone-particle-physics-domain 6 months ago	Files	Running Clusters	Courses A	nnouncements	Assignments					
C capstone-particle-physics-domain 6 months ago	C capstone-particle-physics-domain 6 months ago	Select iter	ns to perform actions on th	nem.					Upload	New - 2	;
		0	✓ ■ /					Name 🕹	Last Modified	File size	
D D phys141 3 months ago	phys141 3 months ago		capstone-particle-physic	s-domain					6 months ago)	
) phys141						3 months ago)	

Start coding!

Coding in Jupyter notebooks

Home Page - Select or create a X phys24	I_test - Jupyter Notebo 🗙 🖉 1.6 Appendix A. Complex numb 🗙 🏼 🄌 python - Does Numpy automati 🗙	cupy version - Google Search × + ∽
\leftarrow \rightarrow C \triangle \triangleq datahub.ucsd.edu/user/jduarte/note	books/phys241_test.ipynb	☆ 🗖 👼 Incognito (3) 🚦
Cjupyterhub phys241_te	est Last Checkpoint: a few seconds ago (unsaved changes)	Logout Control Panel
File Edit View Insert Cel	Kernel Widgets Help	Trusted Python 3 (clean) O
	un 📕 C 🕨 Code 🗸 🖾 Show Usage Validate 🛄	
In [1]: print("hello worl hello world	d")	
In [2]: import numpy as r	g	
[1,		
Out[3]: (4, 3)		
<pre>In [4]: b = np.array([[1]</pre>	,	
Out[4]: (3, 1)		
In [5]: a@b		
Out[5]: array([[8], [10], [5], [8]])		
In [6]: from scipy.specia from math import		

Plotting

• Plotting can be done easily with Matplotlib

🛛 🔍 🗢 🔁 Hor	ne Page - Select or c	create a 🗙 🧧 phys241_test - Jupyter Notebo 🗙 🥃 1.6
$\leftarrow \rightarrow \mathbf{C} \mathbf{\nabla}$	datahub.ucsd	.edu/user/jduarte/notebooks/phys241_test.ipynb
	💭 Jupyter	hub phys241_test Last Checkpoint: 2 minutes ago
	File Edit	View Insert Cell Kernel Widgets Help
		2
		<pre># Range of x determine by classical tunnin xmin, xmax = -np.sqrt(2 * E(VMAX)), np.sqn x = np.linspace(xmin, xmax, 10000) fig, ax = plt.subplots(figsize=(5,5)) for v in range(VMAX): # plot potential V(x) ax.plot(x, V(x), color='black') # plot psi squared which we shift up H ax.plot(x, psi(v,x) ** 2 + E(v), lw=2) # add lines and labels ax.axhline(E(v), color='gray', linesty ax.text(xmax, 1.2 * E(v), f"v={v}") ax.set_xlabel('x') ax.set_ylabel('\$\psi^2_n(x)\$') plt.show()</pre>

1.6 Appendix A. Complex numb 🗙 📔 🏄 python - Does Numpy automati 🗙 📔 🤇	cupy ve	rsion ·	- Google Search	×	+	~
			*		Incognito (3)	:
go (autosaved)	? [Logo	ut Control Pane	el (
0	Trusted	A	Python 3 (clean)	0		
Velidate Validate						
<i>ing points:</i> qrt(2 * E(VMAX))						
by values of energy 2)						
tyle='')						

Installing a missing library

- Not all libraries are preloaded, but it's easy to install a new one
- Note: restart your "kernel" after doing this

In [1]:	<pre>!pip install cupy-cuda112user</pre>
	Collecting cupy-cuda112 Using cached cupy_cuda112-10.3.1-cp39-cp39-manylinux1_x86_64 Collecting fastrlock>=0.5 Using cached fastrlock-0.8-cp39-cp39-manylinux_2_5_x86_64.ma _64.wh1 (49 kB) Requirement already satisfied: numpy<1.25,>=1.18 in /opt/conda 5) Installing collected packages: fastrlock, cupy-cuda112 Successfully installed cupy-cuda112-10.3.1 fastrlock-0.8

t's easy to install a new one ng this

64.whl (78.9 MB)

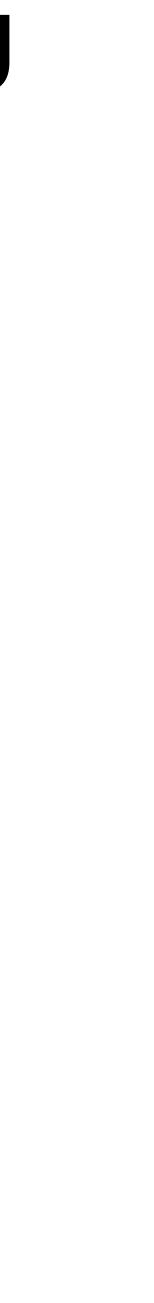
manylinux1_x86_64.manylinux_2_12_x86_64.manylinux2010_x86

da/lib/python3.9/site-packages (from cupy-cuda112) (1.19.

Speed up numerical calculations with the GPU

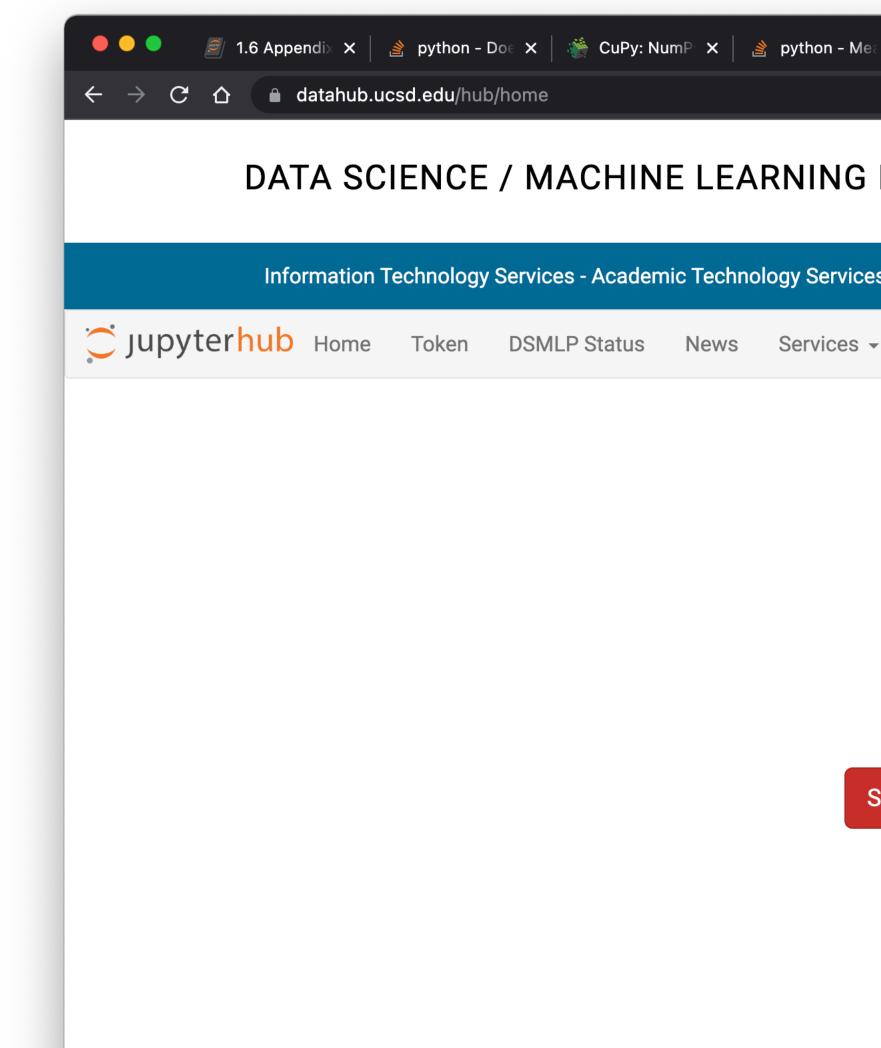
Many libraries to do this: TensorFlow, CuPy, PyTorch, …

	🔌 python - Does N 🗙 🛛 🐞 CuPy: NumPy & 🗙 📄 python - Measu 🗙 🔵 Home Page - Se 🗙 🥃 phys241_te d.edu/user/jduarte/notebooks/phys241_test.ipynb	☆ □ 👼 Incogr							
💭 jupyte	hub phys241_test Last Checkpoint: 16 minutes ago (autosaved)	Logout Control Panel							
File Edit	View Insert Cell Kernel Widgets Help	Not Trusted Python 3 (clean) O							
In [2]	: import cupy as cp								
In [18]	<pre>a = cp.ones((10000, 10000)) b = cp.ones((10000, 10000))</pre>								
In [19]	<pre>%%time a@b</pre>								
	CPU times: user 2.34 ms, sys: 1.21 ms, total: 3.55 ms Wall time: 2.1 ms								
Out[19]	<pre>array([[10000., 10000., 10000.,, 10000., 10000., 10000.], [10000., 10000., 10000.,, 10000., 10000., 10000.], [10000., 10000., 10000.,, 10000., 10000., 10000.], [10000., 10000., 10000.,, 10000., 10000., 10000.], [10000., 10000., 10000.,, 10000., 10000., 10000.]])</pre>								
In [16]	<pre>In [16]: import numpy as np a = np.ones((10000, 10000)) b = np.ones((10000, 10000))</pre>								
In [17]	<pre>17]: %%time a@b CPU times: user 2min 21s, sys: 1min 21s, total: 3min 43s Wall time: 33.1 s</pre>								
Out[17]	<pre>array([[10000., 10000., 10000.,, 10000., 10000., 10000.], [10000., 10000., 10000.,, 10000., 10000., 10000.], [10000., 10000., 10000.,, 10000., 10000., 10000.],</pre>								
	, [10000., 10000., 10000.,, 10000., 10000., 10000.],								



Exiting / shutting down server

• When you're done; hit control panel and "stop your server" When you start it again, all your data will still be there



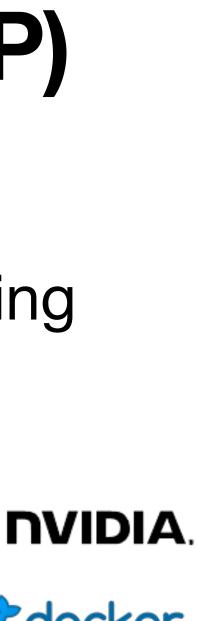
 X ⊂ JupyterHub- 	× C Home Pa	age - 🗙 🛛	<i>i</i> phys241_t	es x >_	https://datah	× │		+ ncognito (3)	~ :
PLATFORM					UC Sa	anDi	ego		
es						Help 🚽	FAQ		
.								C Logout	
Stop My Server	My Server								

Data Science & Machine Learning Platform (DSMLP)

- UCSD Data Science & Machine Learning Platform (DSMLP)
 - Based on docker and Kubernetes (K8s): open-source system for automating deployment, scaling, and management of containerized applications



- **Overview:**
 - You log in to a remote "login" node
 - From that login node, you can launch a "pod" running a container
 - The pod automatically starts a "JupyterHub" web server (only accessible from campus so you must VPN if off campus)
 - You are also automatically logged into that pod so you can run interactive terminal commands, etc.



More Resources

- UCSD DSMLP Cluster
 - Using the DSMLP server
 - <u>Customizing the DSMLP Containers</u>
- Terminal and Command-Line Interface
 - Bash Scripting Reference
 - <u>Git(Hub) Resources</u>