

CS124

Introduction to Lab 2:

Logistic Regression
and

Text Classification

Why logistic regression?

1. Important analytic tool in social and health sciences when we are studying something that is a discrete variable.
2. Baseline supervised machine learning tool for classification
3. Foundation of neural networks

The duality: social/health sciences vs ML

In social sciences

$$P(y = 1) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X_1 + \dots + \beta_k X_k)}}$$

- we care about weights (β) to tell us which factors matter
- don't care about the classification accuracy

In machine learning:

$$P(y = 1) = \frac{1}{1 + \exp(-(\mathbf{w} \cdot \mathbf{x} + b))}$$

- we care about classification accuracy
- we don't care about the weights $\mathbf{w}$

Neural networks and LLMs

Logistic regression is like a single neural unit

A neural net can be modeled as a set of logistic regression classifier chained together

Prediction in LLMs is based on **softmax**,
generalization of the sigmoid to more than 2
output classes

What can we classify? A lot!

En sus remotas páginas está escrito que los animales se dividen en:

- | | |
|--------------------------------|--|
| a. pertenecientes al Emperador | h. incluidos en esta clasificación |
| b. embalsamados | i. que se agitan como locos |
| c. amaestrados | j. innumerables |
| d. lechones | k. dibujados con un pincel finísimo de pelo de camello |
| e. sirenas | l. etcétera |
| f. fabulosos | m. que acaban de romper el jarrón |
| g. perros sueltos | n. que de lejos parecen moscas |

Borges (1964)

it is written that animals are divided into (a) those that belong to the Emperor, (b) embalmed ones, (c) those that are trained, (d) suckling pigs, (e) mermaids, (f) fabulous ones, (g) stray dogs, (h) those that are included in this classification, (i) those that tremble as if they were mad, (j) innumerable ones, (k) those drawn with a very fine camel's hair brush, (l) others, (m) those that have just broken a flower vase, (n) those that resemble flies from a distance.

Historical story:: Logistic regression and early LLMs

1970s

IBM: world's leading speech recognition group

Predicting transcript using recently declassified statistical models



1990: They switch to text!

Reinvented logistic regression
Applied it as a classifier for machine translation

**A Maximum Entropy Approach
to Natural Language Processing**
**The Mathematics of Machine
Translation:**

Adam L. Berger[†]
Columbia University

Stephen A. Della Pietra[‡]
Renaissance Technologies

Peter F. Brown^{*}

IBM T.J. Watson Research Center
Vincent J. Della Pietra[‡]
Renaissance Technologies

Vincent J. Della Pietra^{*}
IBM T.J. Watson Research Center

ROBERT L. MERCER^{*}
IBM T.J. Watson Research Center

1992 IBM team switches to language modeling

They realize that a language model is just classifying the context into possible next words

1993, predicting more than words

Realized predicting words was like predicting the stock market.

Peter Brown, Robert Mercer, Stephen and Vincent della Pietra leave to join a tiny new hedge fund called Renaissance

Predict tiny changes in the market based on context

Became most successful fund in history



Today's Lab

1. Get solid intuitions about the sigmoid
 - In addition to being the core of LR
 - Sigmoid is also the basis of softmax, which is how LLMs produce probabilities
2. Get solid intuitions about gradient descent
3. Think about the ethics of classification, and especially about thresholds

Reminder: Classification with sigmoid

Each feature x_i weight w_i tells us importance of x_i

Score: sum the weighted features and the bias

$$z = \left(\sum_{i=1}^n w_i x_i \right) + b$$

$$z = w \cdot x + b$$

If this sum is high, we say $y=1$; if low, then $y=0$

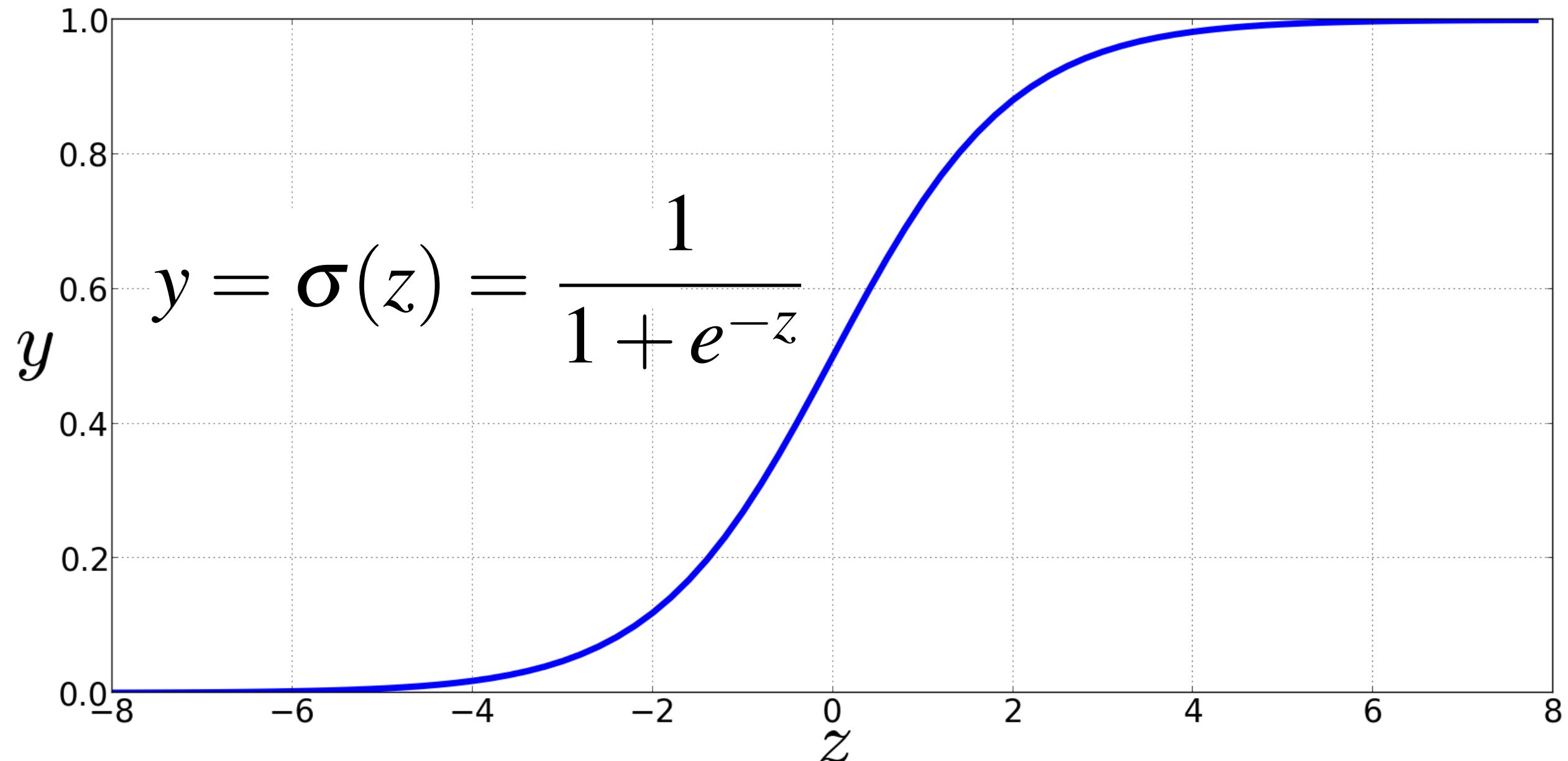
How to make a probability from z

$$z = w \cdot x + b$$

Use a function of z that goes from 0 to 1

$$y = \sigma(z) = \frac{1}{1 + e^{-z}} = \frac{1}{1 + \exp(-z)}$$

The very useful sigmoid or logistic function



Summary of logistic regression:

We compute $w \cdot x + b$

Pass it through sigmoid function:

$$\sigma(w \cdot x + b)$$

And we'll treat it as a probability

Turning a probability into a classifier

$$\text{decision}(x) = \begin{cases} 1 & \text{if } P(y = 1|x) > 0.5 \\ 0 & \text{otherwise} \end{cases}$$

0.5 here is called the **decision boundary**

Reminder: gradient descent for learning weights

Reminder: gradient descent for learning weights

Cross-entropy loss function: distance between $\hat{y}$ and y

We want to know how far is the classifier output:

$$\hat{y} = \sigma(w \cdot x + b)$$

from the true output:

$$y \quad [= \text{either } 0 \text{ or } 1]$$

We'll call this difference:

$$L(\hat{y}, y) = \text{how much } \hat{y} \text{ differs from the true } y$$

Goal of learning

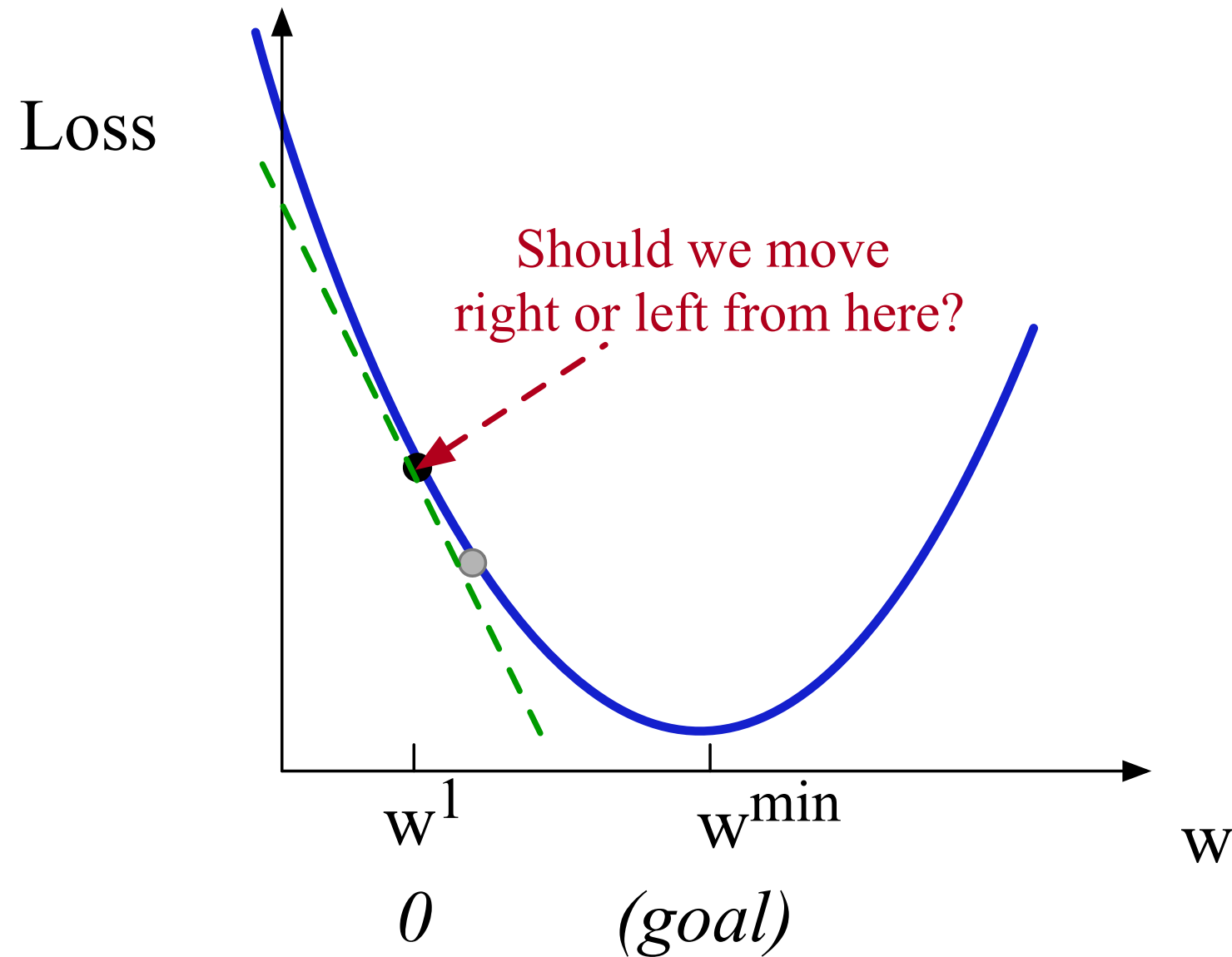
We want to move the weights

To minimize the loss function $L(\hat{y}, y)$.

Let's visualize for a single scalar w

Q: Given current w , should we make it bigger or smaller?

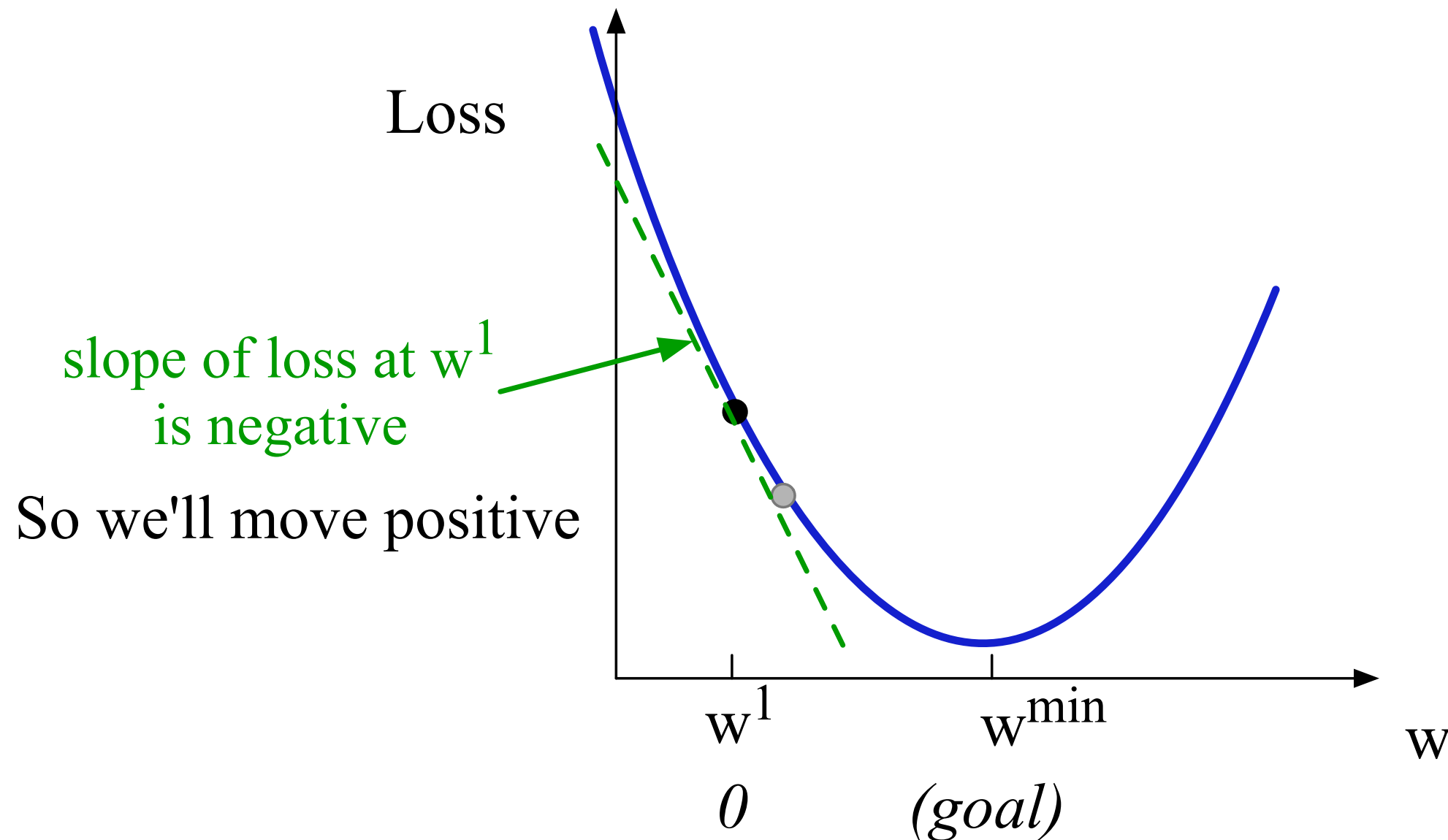
A: Move w in the reverse direction from the slope of the function



Let's visualize for a single scalar w

Q: Given current w , should we make it bigger or smaller?

A: Move w in the reverse direction from the slope of the function



Let's visualize for a single scalar w

Q: Given current w , should we make it bigger or smaller?

A: Move w in the reverse direction from the slope of the function

