CS 2002
Artificial Intelligence

Waheed Ahmed
Email : waheedahmed@nu.edu.pk

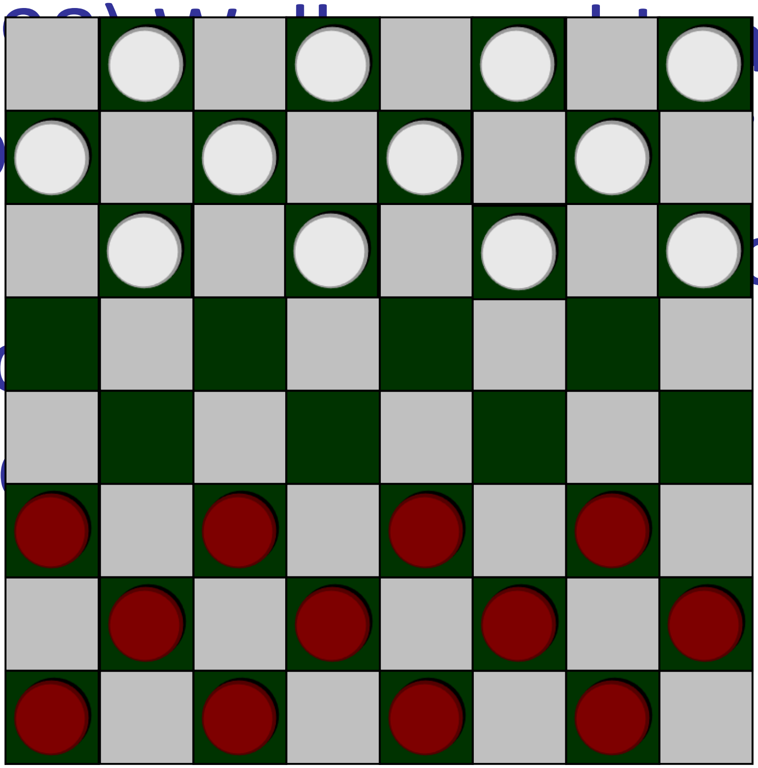
Week 10:
Supervised Learning (Learning from Examples)

Russell & Norvig, Chapter 18.

(Most of slides from **Wang Ling, Pieter Abbeel)**

Machine Learning definition

- Arthur Samuel (1959). Machine Learning: Field of study that gives computers the ability to learn without being explicitly programmed.
- Tom Mitchell (19 … rning Problem: A comp … d to *learn* from experience … ome task T and some performan … performance on T, as measure … ith experience E.



Machine Learning algorithms

Machine learning algorithms:
- Supervised learning
- Unsupervised learning

Others: Reinforcement learning, recommender systems.

Also talk about: Practical advice for applying learning algorithms.

Supervised Learning

Supervised learning describes a class of problem that involves using a model to learn a mapping between input examples and the target variable.

- *Applications in which the training data comprises examples of the input vectors along with their corresponding target vectors are known as supervised learning problems.*

Pattern Recognition and Machine Learning, 2006.

- There are two main types of supervised learning problems: **Classification**: Supervised learning problem that involves predicting a class label.
- **Regression**: Supervised learning problem that involves predicting a numerical label.

Numbers are our friends

Abby

4 

Variables are our friends

Abby

4 

Bert

5 

Variables are our friends

Abby

4 x

5 y

Bert

Operators are our friends

Bert

4 🍎

1 🍎

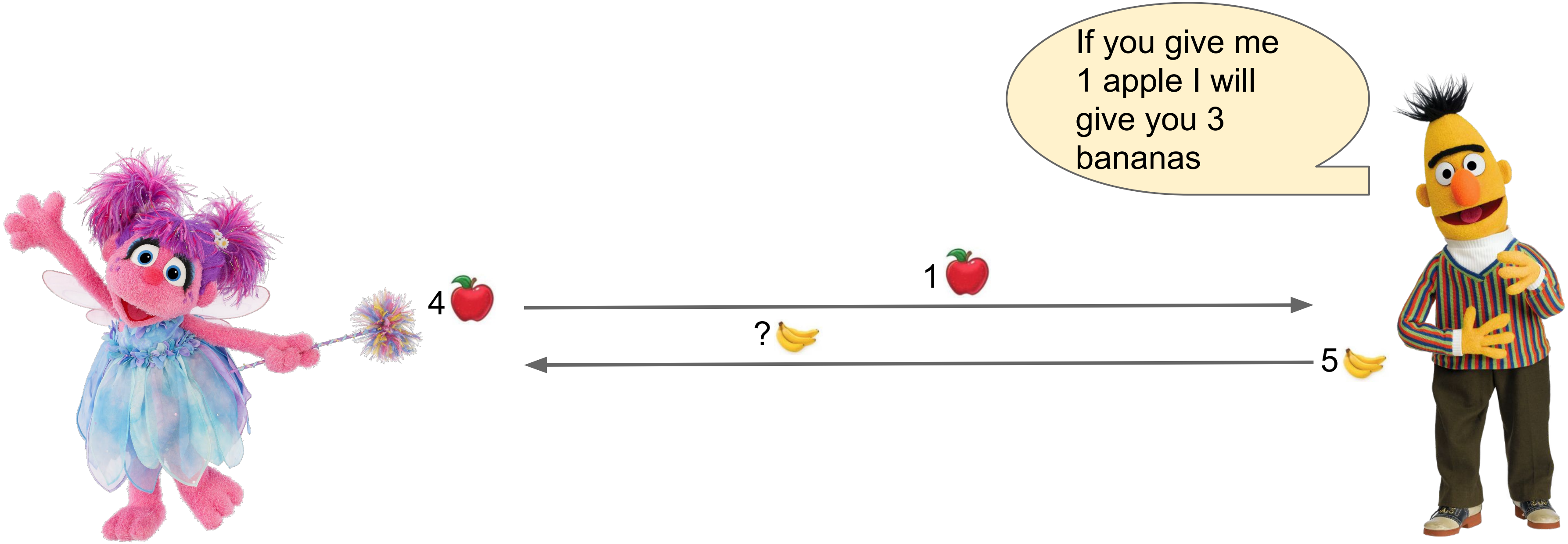
If Abby has 4 apples, and gives Bert 1 apple, how many apples will Abby have?

Operators are our friends

$$4\mathbf{x} - 1\mathbf{x} = 3\mathbf{x}$$

Bert

Functions are our friends



Functions are our friends

$y = 3x$

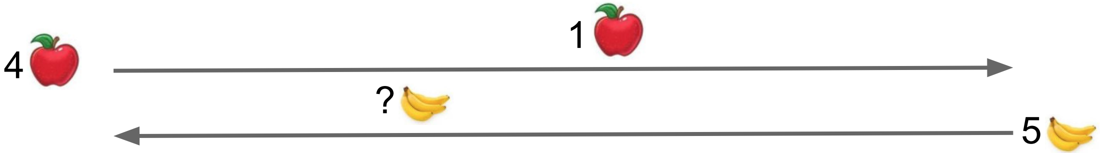
- Input, x - Number of Apples given by Abby

Functions are our friends

$y = 3x$

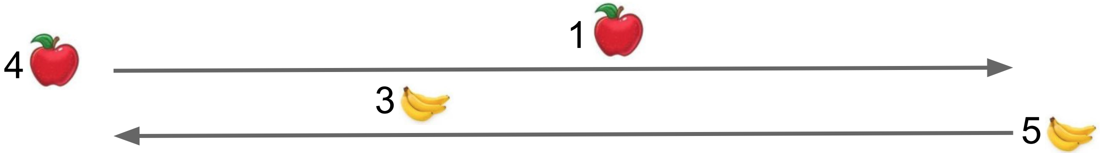
- Input, x - Number of Apples given by Abby
- Output, y - Number of Bananas received by Abby

Functions are our friends



$$y = 3x \;,\; x = 1$$

Functions are our friends



$y = 3x$, $x = 1$

$y = 3$

Functions are our friends

$y = 3x$

Functions are our friends

x : English Sentence

y : Spanish Sentence

Functions are our friends

x : Board

y : Move

Functions are our friends

x : Image

y : Category

Functions are our friends

x : Board

???????????????????????????

y : Move

Functions are our friends

$y = 3x$

Cookie Monster

Functions are our friends

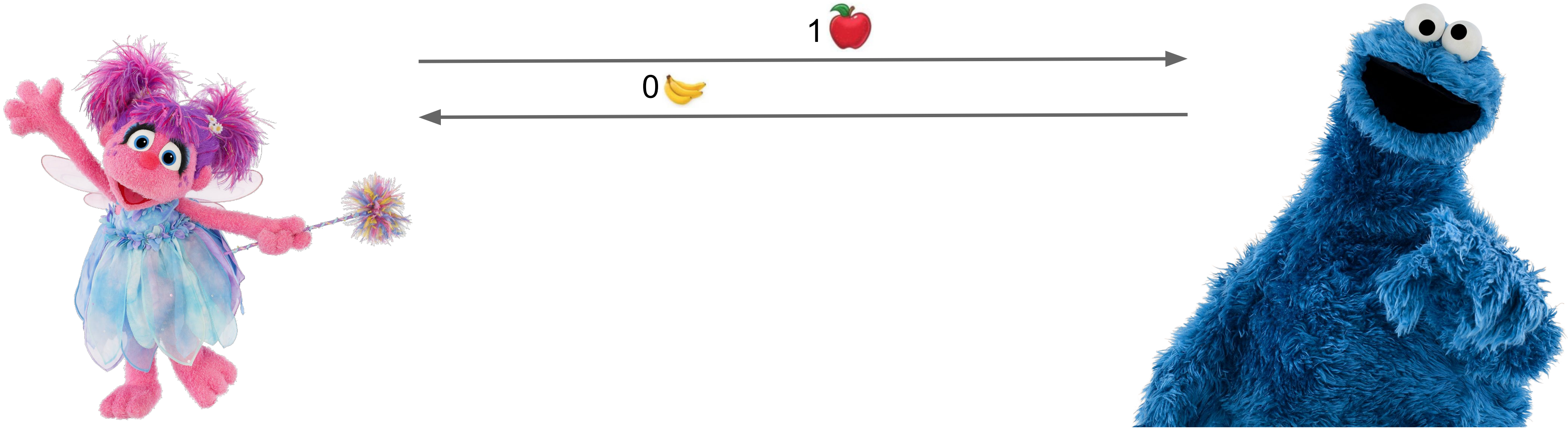
$y = ??$

$y = 3x$

Find it out for yourself

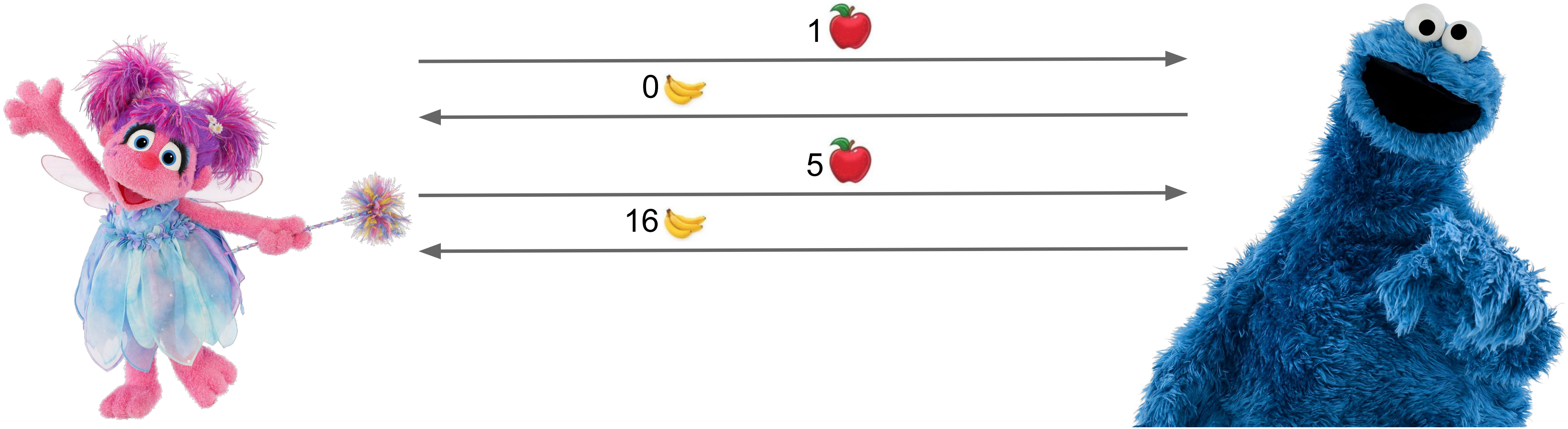
Functions are our friends

y = ??



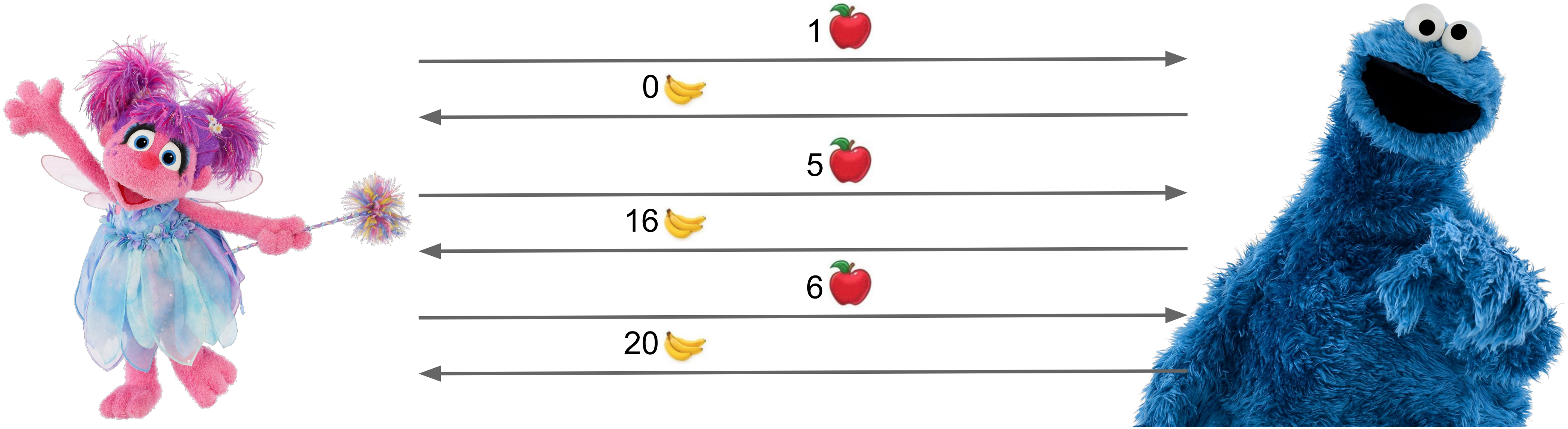
Functions are our friends

y = ??



Functions are our friends

y = ??



Functions are our friends

I want to know how many bananas I get,
but I ran out of apples....

y = ??

1 🍎 →

← 0 🍌

5 🍎 →

← 16 🍌

6 🍎 →

← 20 🍌

3 🍎 →

← ? 🍌

Parameters are our friends

$$y = 3x + 1$$

- Input
- Output

Parameters are our friends

Model

$$y = wx + b$$

- Input
- Output
- Parameters

Input - Fixed, comes from data
Parameters - Need to be estimated

Parameters are our friends

$y = wx + b$

1 🍎

0 🍌

5 🍎

16 🍌

6 🍎

20 🍌

3 🍎

? 🍌

Parameters are our friends

$$y = wx + b$$

Parameters are our friends

$$y = wx + b$$

Data	
x	$\hat{y}$
1	0
5	16
6	20

3

?

Parameters are our friends

Data

x	ŷ
1	0
5	16
6	20

Model

$$y = wx + b$$

Parameters are our friends

Data

x	ŷ
1	0
5	16
6	20

Model

$$y = wx + b$$

How to find the parameters w and b?

Parameters are our friends

Data

x	ŷ
1	0
5	16
6	20

Model

$$y = wx + b$$

Model
Candidate 1

$$y = 1x + 0$$

x	y
1	0
5	16
6	20

Parameters are our friends

Data

x	ŷ
1	0
5	16
6	20

Model

$$y = wx + b$$

Model Candidate 1

$$y = 1x + 0$$
$$1 = 1*1 + 0$$
$$5 = 1*5 + 0$$
$$6 = 1*6 + 0$$

x	ŷ	y
1	0	1
5	16	5
6	20	6

Parameters are our friends

Data

x	y
1	0
5	16
6	20

Model

$$y = wx + b$$

Model Candidate 1

$$y = 1x + 0$$

x	ŷ	y
1	0	1
5	16	5
6	20	6

Model Candidate 2

$$y = 2x + 2$$

x	ŷ	y
1	0	4
5	16	12
6	20	14

Parameters are our friends

Data

x	y
1	0
5	16
6	20

Model

$$y = wx + b$$

Model Candidate 1

$$y = 1x + 0$$

x	ŷ	y
1	0	1
5	16	5
6	20	6

Model Candidate 2

$$y = 2x + 2$$

x	ŷ	y
1	0	4
5	16	12
6	20	14

Which one is better ?

Parameters are our friends

Data

x	y
1	0
5	16
6	20

Model

$$y = wx + b$$

Model Candidate 1

$$y = 1x + 0$$

x	ŷ	y
1	0	1
5	16	5
6	20	6

Model Candidate 2

$$y = 2x + 2$$

x	ŷ	y
1	0	4
5	16	12
6	20	14

Cost functions are our friends

Data

n	x	y
0	1	0
1	5	16
2	6	20

Model

$$y_n = wx_n + b$$

Model Candidate 1

$$y = 1x + 0$$

x	ŷ	y
1	0	1
5	16	5
6	20	6

Model Candidate 2

$$y = 2x + 2$$

x	ŷ	y
1	0	4
5	16	12
6	20	14

Cost functions are our friends

Data

n	x	y
0	1	0
1	5	16
2	6	20

Model

$$y_n = wx_n + b$$

Cost

$$C(w,b)$$

Model Candidate 1

$$y = 1x + 0$$

x	ŷ	y
1	0	1
5	16	5
6	20	6

Model Candidate 2

$$y = 2x + 2$$

x	ŷ	y
1	0	4
5	16	12
6	20	14

Cost functions are our friends

Data

n	x	y
0	1	0
1	5	16
2	6	20

Model

$$y_n = wx_n + b$$

Cost

Square Loss

$$C(w,b) = \sum_{n\in\{0,1,2\}} (y_n - \hat{y}_n)^2$$

Model Candidate 1

$$y = 1x + 0$$

x	ŷ	y
1	0	1
5	16	5
6	20	6

Model Candidate 2

$$y = 2x + 2$$

x	ŷ	y
1	0	4
5	16	12
6	20	14

Cost functions are our friends

Data

n	x	y
0	1	0
1	5	16
2	6	20

Model

$$y_n = wx_n + b$$

Cost

$$C(w,b) = \sum_{n \in \{0,1,2\}} (y_n - \hat{y}_n)^2$$

Model Candidate 1

$$y = 1x + 0$$

n	x	ŷ	y	$(y-\hat{y})^2$
0	1	0	1	
1	5	16	5	
2	6	20	6	

Model Candidate 2

$$y = 2x + 2$$

x	ŷ	y
1	0	4
5	16	12
6	20	14

Cost functions are our friends

Data

n	x	y
0	1	0
1	5	16
2	6	20

Model

$$y_n = wx_n + b$$

Cost

$$C(w,b) = \sum_{n\in\{0,1,2\}} (y_n - \hat{y}_n)^2$$

Model Candidate 1

$$y = 1x + 0$$

n	x	ŷ	y	$(y-\hat{y})^2$
0	1	0	1	1
1	5	16	5	
2	6	20	6	

Model Candidate 2

$$y = 2x + 2$$

x	ŷ	y
1	0	4
5	16	12
6	20	14

Cost functions are our friends

Data

n	x	y
0	1	0
1	5	16
2	6	20

Model

$$y_n = w x_n + b$$

Cost

$$C(w,b) = \sum_{n \in \{0,1,2\}} (y_n - \hat{y}_n)^2$$

Model Candidate 1

$$y = 1x + 0$$

n	x	ŷ	y	$(y-\hat{y})^2$
0	1	0	1	1
1	5	16	5	121
2	6	20	6	

Model Candidate 2

$$y = 2x + 2$$

x	ŷ	y
1	0	4
5	16	12
6	20	14

Cost functions are our friends

Data

n	x	y
0	1	0
1	5	16
2	6	20

Model

$$y_n = w x_n + b$$

Cost

$$C(w,b) = \sum_{n \in \{0,1,2\}} (y_n - \hat{y}_n)^2$$

Model Candidate 1

$$y = 1x + 0$$

n	x	ŷ	y	$(y-\hat{y})^2$
0	1	0	1	1
1	5	16	5	121
2	6	20	6	196

Model Candidate 2

$$y = 2x + 2$$

x	ŷ	y
1	0	4
5	16	12
6	20	14

Cost functions are our friends

Data

n	x	y
0	1	0
1	5	16
2	6	20

Model

$$y_n = wx_n + b$$

Cost

$$C(w,b) = \sum_{n \in \{0,1,2\}} (y_n - \hat{y}_n)^2$$

Model Candidate 1

$$y = 1x + 0$$

n	x	ŷ	y	$(y-\hat{y})^2$
0	1	0	1	1
1	5	16	5	121
2	6	20	6	196
			$C(1,0)$	318

Model Candidate 2

$$y = 2x + 2$$

x	ŷ	y
1	0	4
5	16	12
6	20	14

Cost functions are our friends

Data

n	x	y
0	1	0
1	5	16
2	6	20

Model

$$y_n = wx_n + b$$

Cost

$$C(w,b) = \sum_{n \in \{0,1,2\}} (y_n - \hat{y}_n)^2$$

Model Candidate 1

$$y = 1x + 0$$

n	x	ŷ	y	$(y-\hat{y})^2$
0	1	0	1	1
1	5	16	5	121
2	6	20	6	196
			C(1,0)	318

Model Candidate 2

$$y = 2x + 2$$

n	x	ŷ	y	$(y-\hat{y})^2$
0	1	0	4	16
1	5	16	12	16
2	6	20	14	36
			C(2,2)	68

Cost functions are our friends

Data

n	x	y
0	1	0
1	5	16
2	6	20

Model

$$y_n = wx_n + b$$

Cost

$$C(w,b) = \sum_{n \in \{0,1,2\}} (y_n - \hat{y}_n)^2$$

Model Candidate 1

$$y = 1x + 0$$

$C(1,0)$ 318

Model Candidate 2

$$y = 2x + 2$$

$C(2,2)$ 68

Cost functions are our friends

Data

n	x	y
0	1	0
1	5	16
2	6	20

Model

$$y_n = w x_n + b$$

How to find the parameters w and b?

Cost

$$C(w,b) = \sum_{n \in \{0,1,2\}} (y_n - \hat{y}_n)^2$$

Optimizers are our friends

Data

n	x	y
0	1	0
1	5	16
2	6	20

Model

$$y_n = wx_n + b$$

Cost

$$C(w,b) = \sum_{n \in \{0,1,2\}} (y_n - \hat{y}_n)^2$$

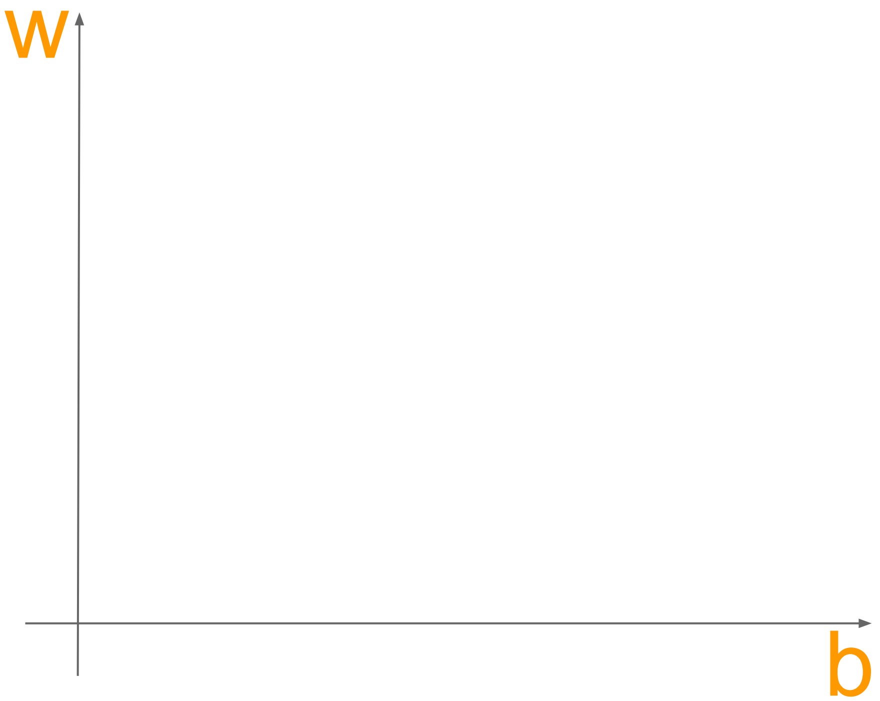
Optimizer

$$\underset{w,b \in [-\infty,\infty]}{\arg\min}\, C(w,b)$$

Optimizers are our friends

Optimizer

$$\underset{w,b\in[-\infty,\infty]}{\arg\min}\ C(w,b)$$



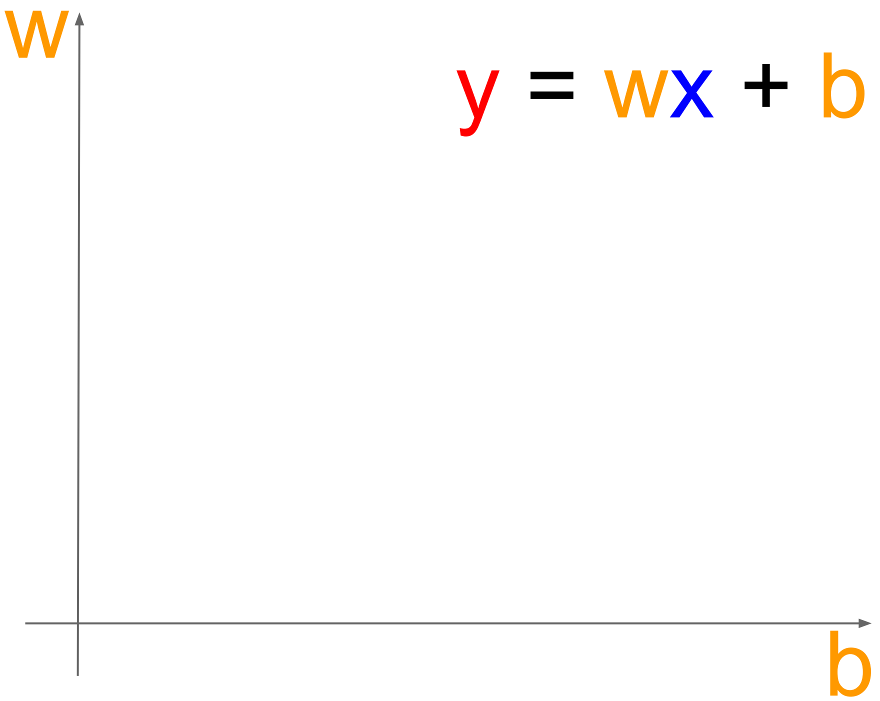
Optimizers are our friends

Optimizer

$$\underset{w,b \in [-\infty,\infty]}{\arg\min}\ C(w,b)$$

$w_0, b_0 = 2,2 : C(w_0, b_0) = 68$

$y = wx + b$



Optimizers are our friends

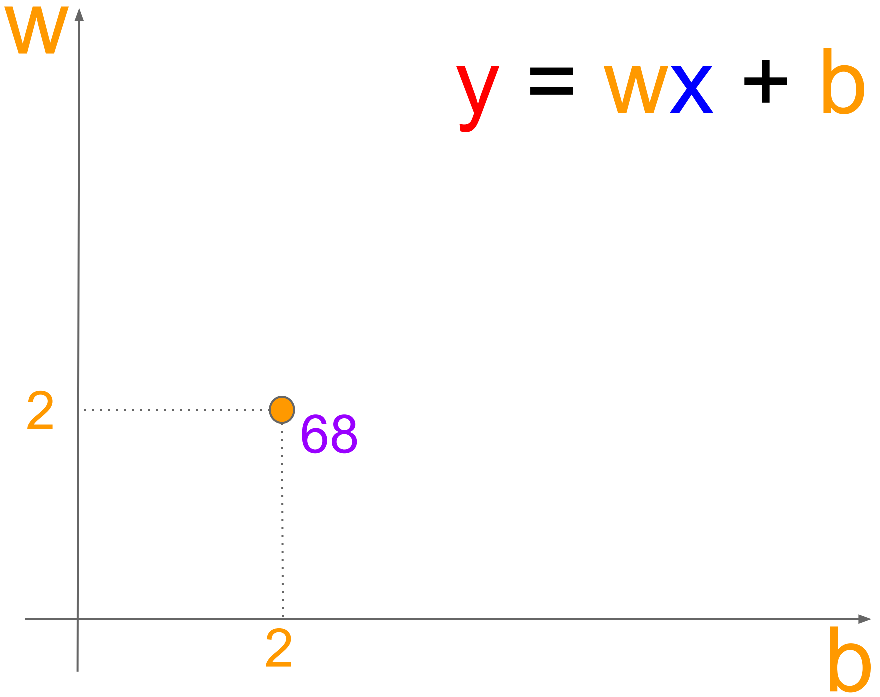
Optimizer

$$\arg\min C(w,b)$$

$$w,b \in [-\infty,\infty]$$

$$w_0,b_0 = 2,2 : C(w_0,b_0) = 68$$

$$y = wx + b$$



Optimizers are our friends

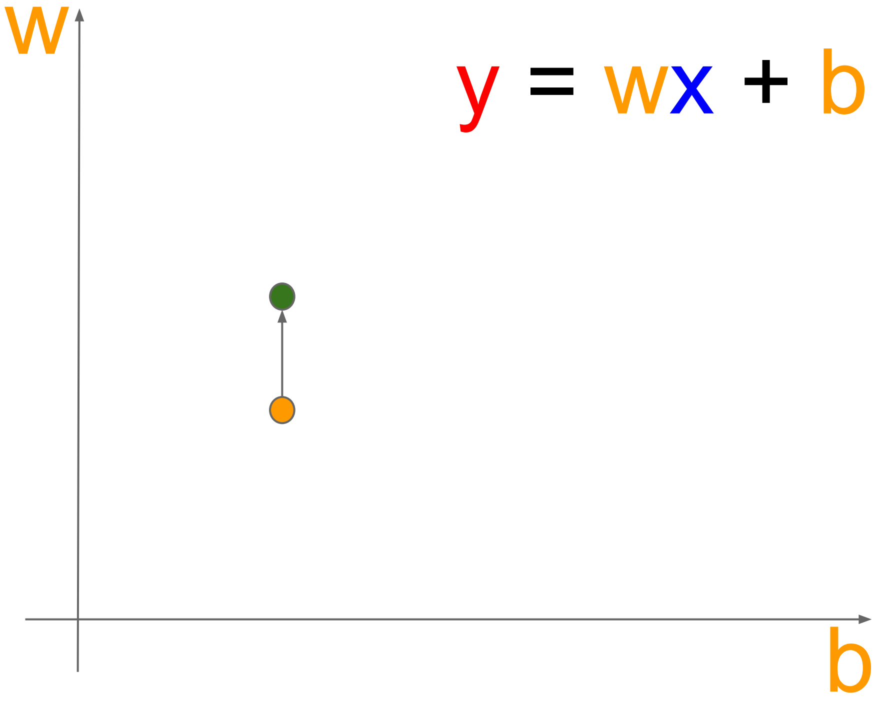
Optimizer

$$\underset{w,b \in [-\infty, \infty]}{\arg\min}\ C(w,b)$$

$w_0, b_0 = 2,2 : C(w_0, b_0) = 68$

$w_1, b_1 = 3,2 : C(w_1, b_1) = ?$

$y = wx + b$



Optimizers are our friends

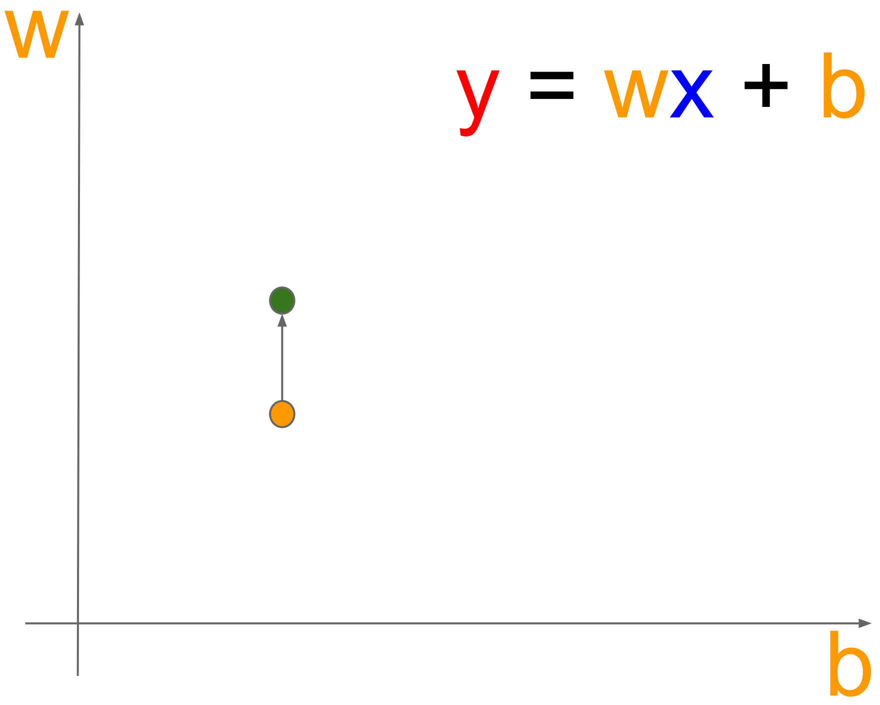
Optimizer

$$\arg\min_{w,b\in[-\infty,\infty]} C(w,b)$$

$w_0,b_0 = 2,2 : C(w_0,b_0) = 68$

$w_1,b_1 = 3,2 : C(w_1,b_1) = 26$

n	x	ŷ	y	$(y-\hat{y})^2$
0	1	0	5	25
1	5	16	17	1
2	6	20	20	0
			C(3,2)	26



Optimizers are our friends

Optimizer

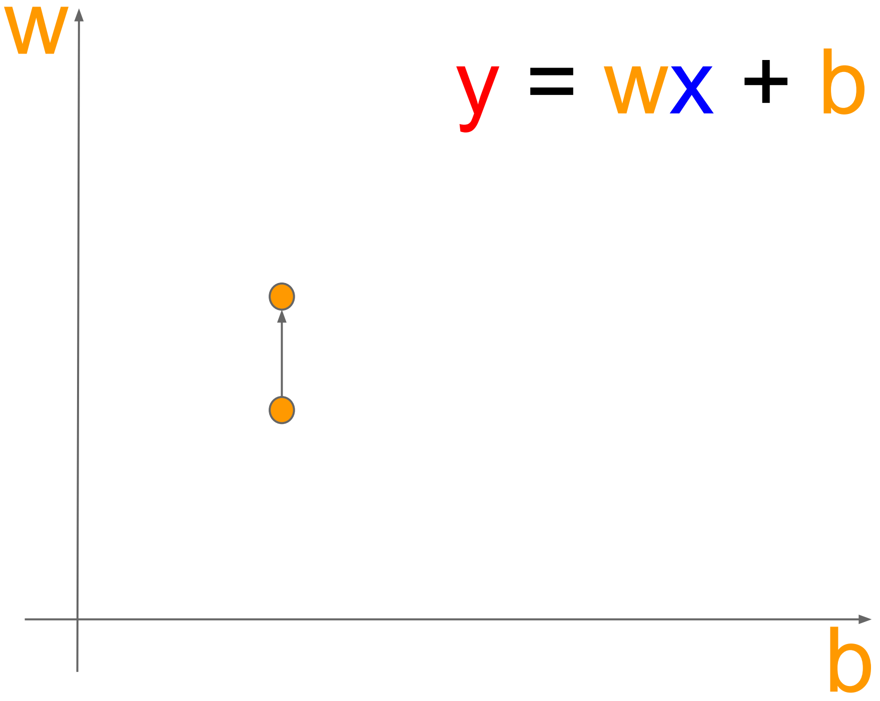
$$\underset{w,b \in [-\infty,\infty]}{\arg\min}\ C(w,b)$$

$w_0, b_0 = 2,2 : C(w_0, b_0) = 68$

$w_1, b_1 = 3,2 : C(w_1, b_1) = 26$

n	x	$\hat{y}$	y	$(y-\hat{y})^2$
0	1	0	5	25
1	5	16	17	1
2	6	20	20	0
			C(3,2)	26

$y = wx + b$



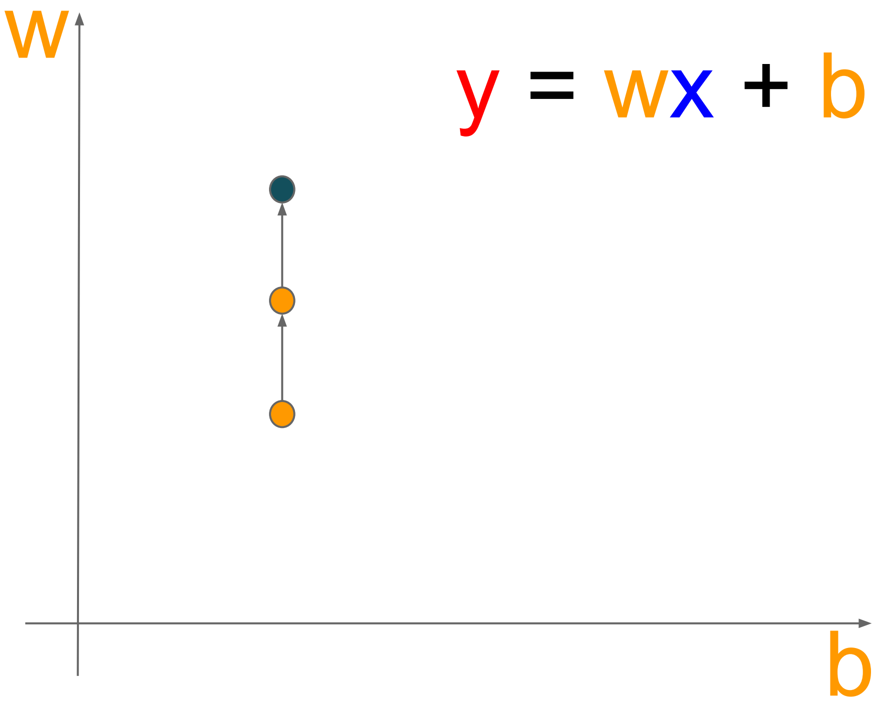
Optimizers are our friends

Optimizer

$$\arg\min C(w,b)$$
$$w,b \in [-\infty,\infty]$$

$w_1,b_1 = 3,2 : C(w_1,b_1) = 26$

$w_2,b_2 = 4,2 : C(w_2,b_2) = ??$



Optimizers are our friends

Optimizer

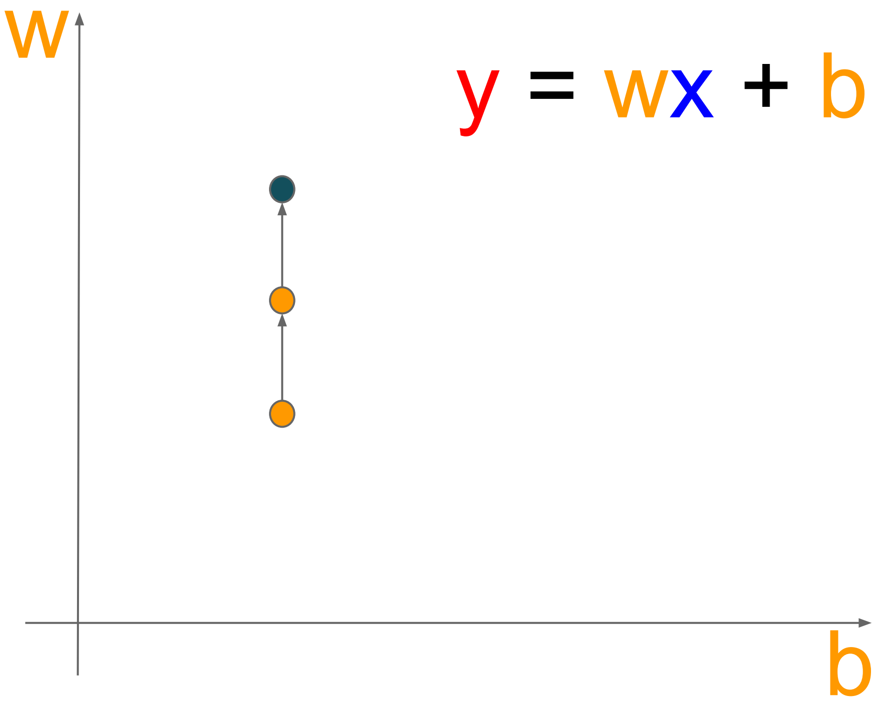
$$\underset{w,b\in[-\infty,\infty]}{\arg\min}\; C(w,b)$$

$w_1,b_1 = 3,2 : C(w_1,b_1) = 26$

$w_2,b_2 = 4,2 : C(w_2,b_2) = 136$

n	x	ŷ	y	$(y-\hat{y})^2$
0	1	0	6	36
1	5	16	22	64
2	6	20	26	36
			C(4,2)	136

$y = wx + b$



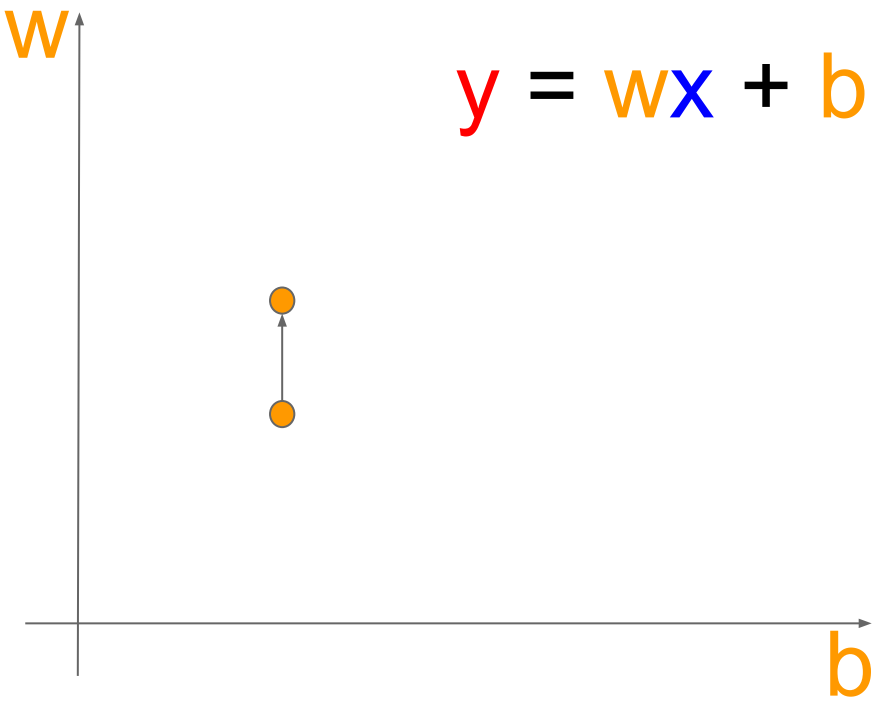
Optimizers are our friends

Optimizer

$$\underset{w,b \in [-\infty,\infty]}{\arg\min}\; C(w,b)$$

$w_1, b_1 = 3,2 : C(w_1, b_1) = 26$

$y = wx + b$



Optimizers are our friends

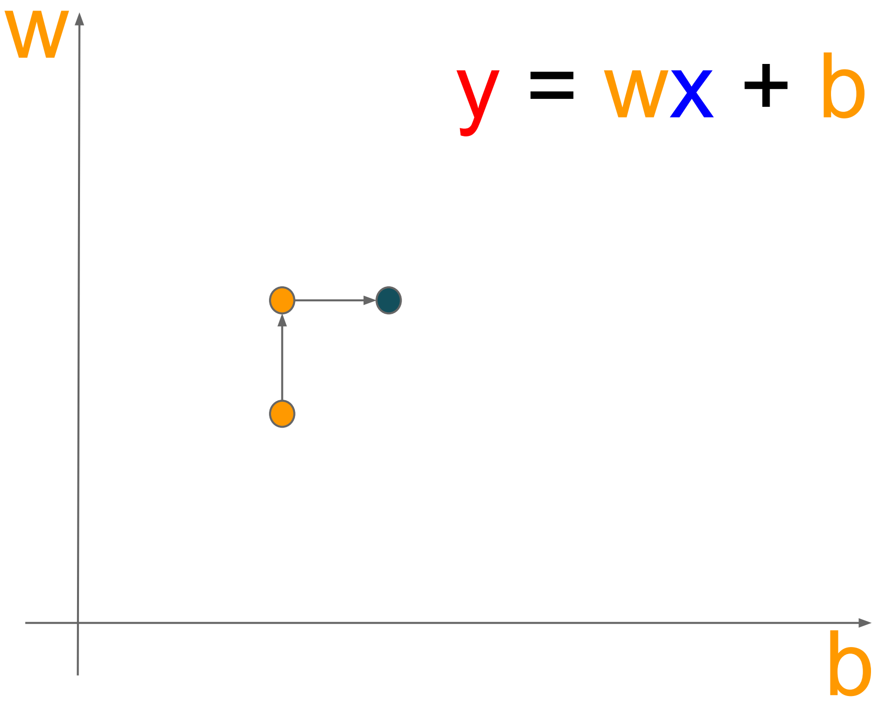
Optimizer

$$\arg\min C(w,b)$$
$$w,b \in [-\infty, \infty]$$

$w_1,b_1 = 3,2 : C(w_1,b_1) = 26$

$w_2,b_2 = 3,3 : C(w_2,b_2) = 41$

n	x	ŷ	y	$(y-\hat{y})^2$
0	1	0	6	36
1	5	16	18	4
2	6	20	21	1
			C(3,3)	41



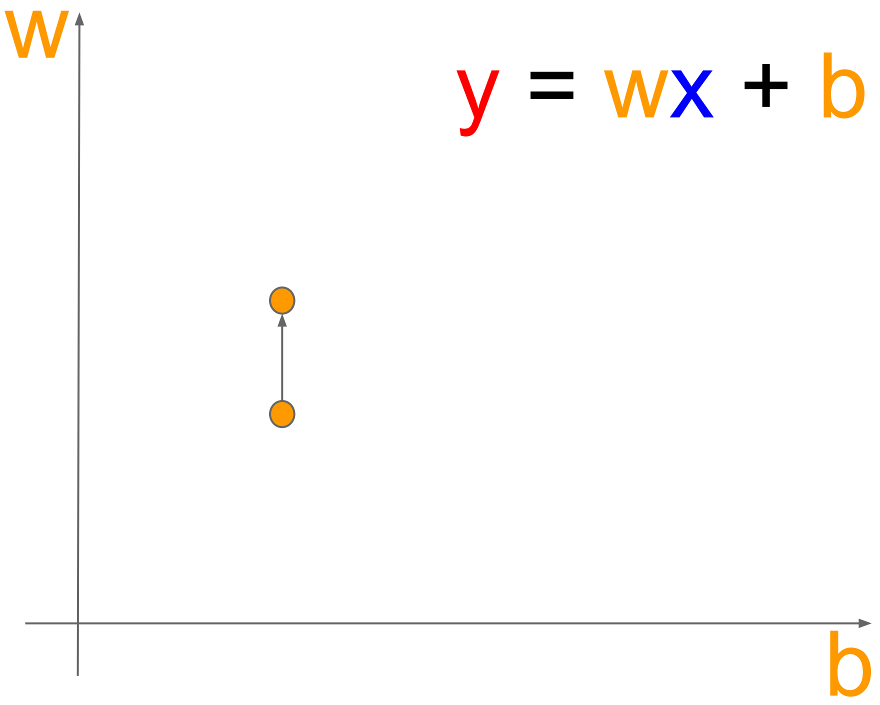
Optimizers are our friends

Optimizer

$$\arg\min_{w,b \in [-\infty,\infty]} C(w,b)$$

$w_1, b_1 = 3,2 : C(w_1, b_1) = 26$

$y = wx + b$



Optimizers are our friends

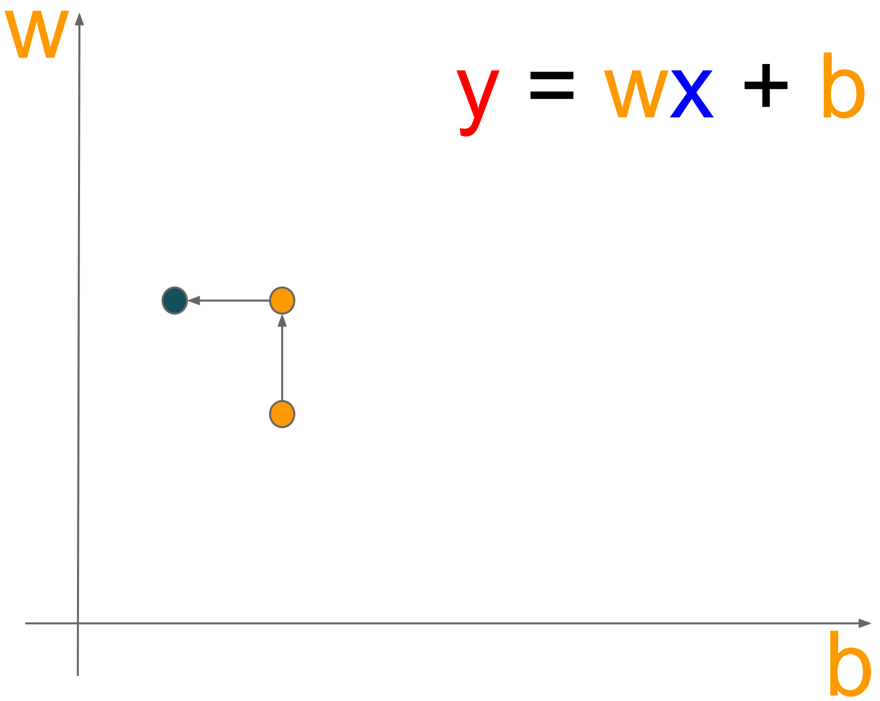
Optimizer

$$\arg\min_{w,b \in [-\infty,\infty]} C(w,b)$$

$w_1,b_1 = 3,2 : C(w_1,b_1) = 26$

$w_2,b_2 = 3,1 : C(w_2,b_2) = 17$

n	x	ŷ	y	$(y-\hat{y})^2$
0	1	0	4	16
1	5	16	16	0
2	6	20	19	1
			C(3,1)	17



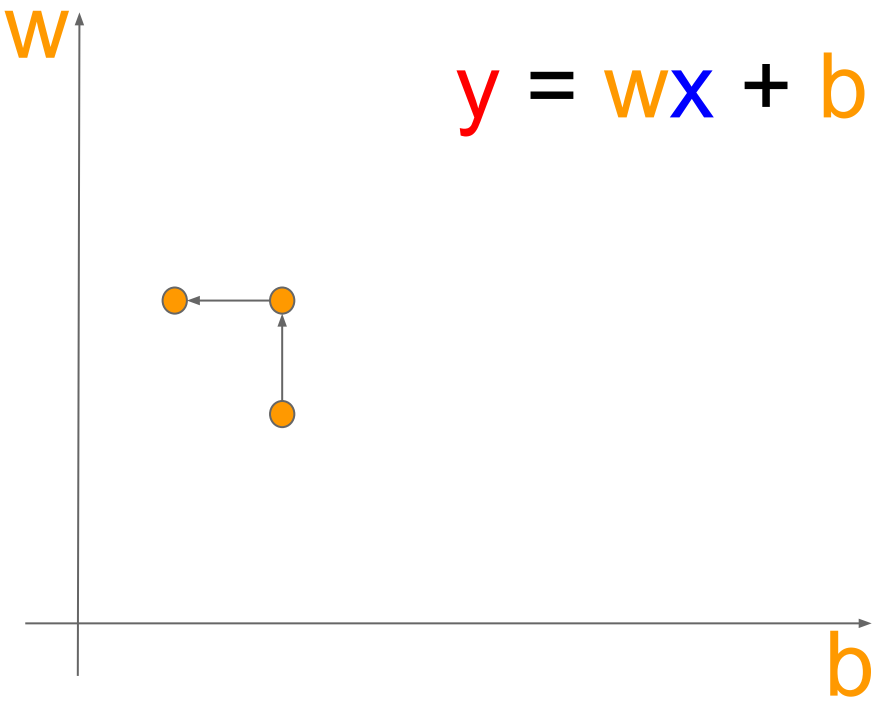
Optimizers are our friends

Optimizer

$$\underset{w,b \in [-\infty, \infty]}{\arg\min}\ C(w,b)$$

$w_2,b_2 = 3,1 : C(w_2,b_2) = 17$

$y = wx + b$



Optimizers are our friends

Optimizer

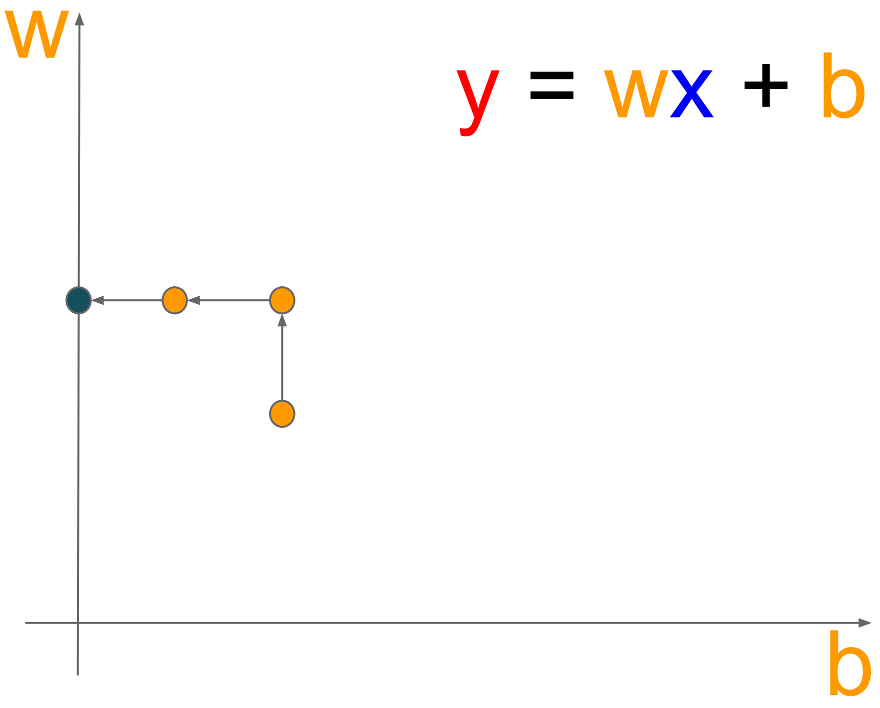
$$\arg\min C(w,b)$$

$$w,b \in [-\infty,\infty]$$

$w_2,b_2 = 3,1 : C(w_2,b_2) = 17$

$w_3,b_3 = 3,0 : C(w_3,b_3) = 13$

n	x	ŷ	y	$(y-\hat{y})^2$
0	1	0	3	9
1	5	16	15	1
2	6	20	18	4
			C(3,0)	13



$$y = wx + b$$

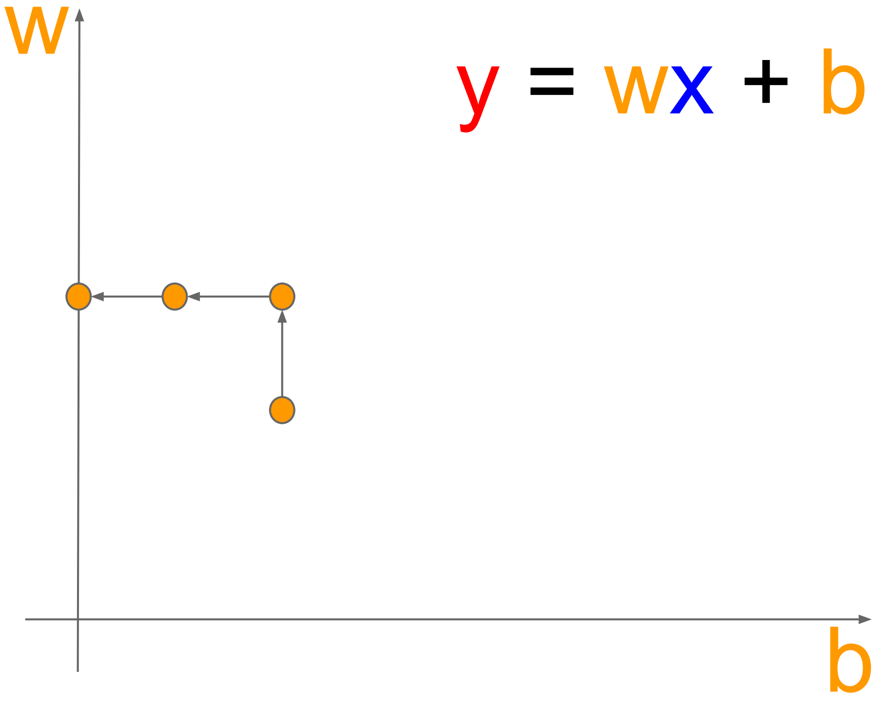
Optimizers are our friends

Optimizer

$$\arg\min C(w,b)$$

$w,b \in [-\infty,\infty]$

$w_3,b_3 = 3,0 : C(w_3,b_3) = 13$



$$y = wx + b$$

Optimizers are our friends

Optimizer

$$\arg\min C(w,b)$$

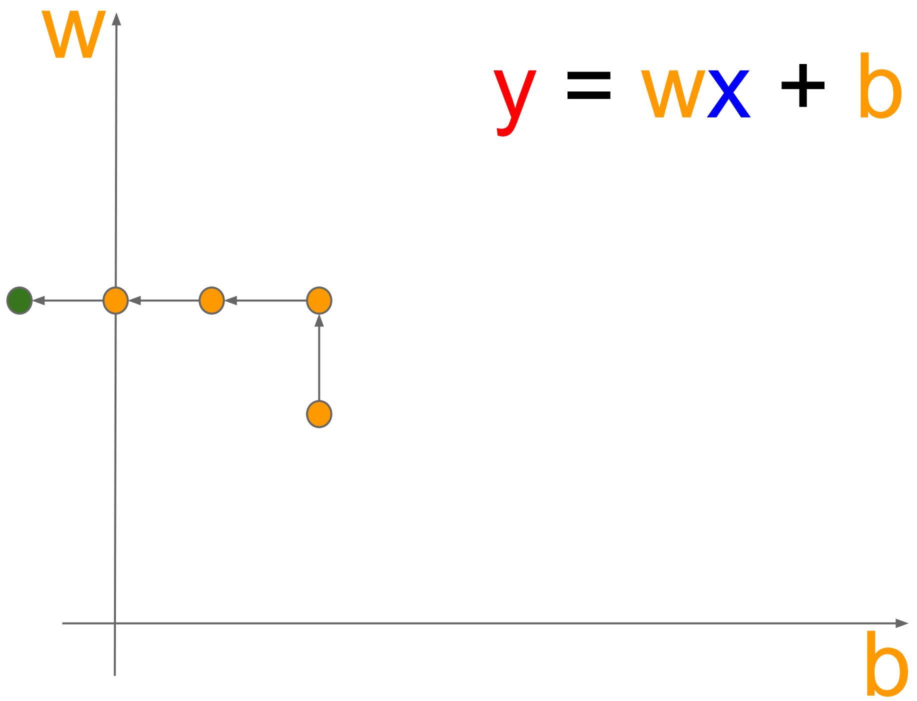
$w,b \in [-\infty,\infty]$

$w_3,b_3 = 3,0 : C(w_3,b_3) = 13$

$w_4,b_4 = 3,-1 : C(w_4,b_4) = 17$

n	x	$\hat{y}$	y	$(y-\hat{y})^2$
0	1	0	2	4
1	5	16	14	4
2	6	20	17	9
			C(3,-1)	17

$$y = wx + b$$



Optimizers are our friends

Optimizer

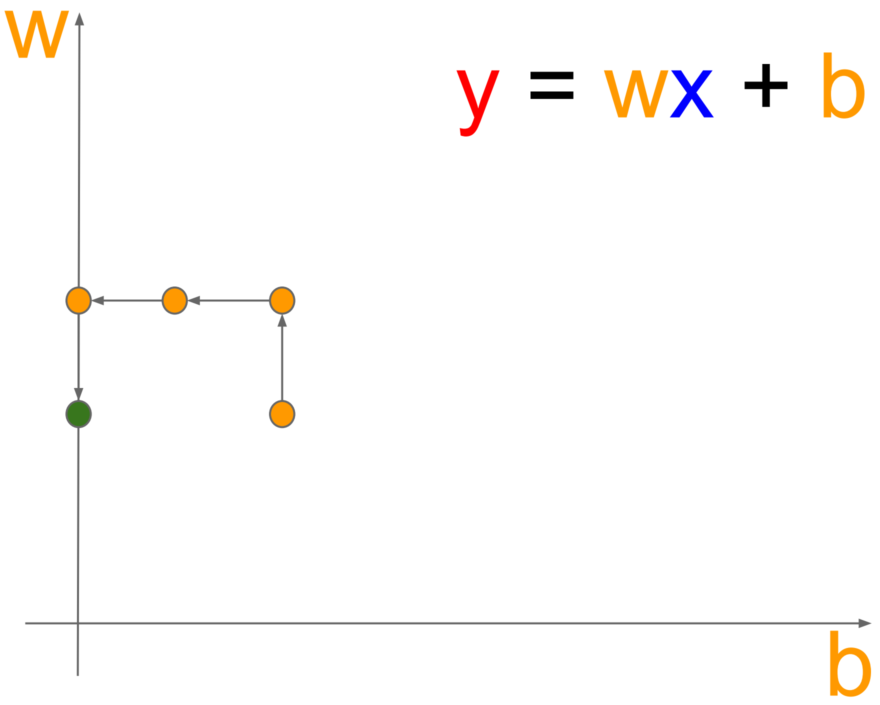
$$\arg\min C(w,b)$$

$w,b \in [-\infty,\infty]$

$w_3,b_3 = 3,0 : C(w_3,b_3) = 13$

$w_4,b_4 = 2,0 : C(w_4,b_4) = 104$

n	x	ŷ	y	$(y-\hat{y})^2$
0	1	0	2	4
1	5	16	10	36
2	6	20	12	64
			C(2,0)	104



$$y = wx + b$$

Optimizers are our friends

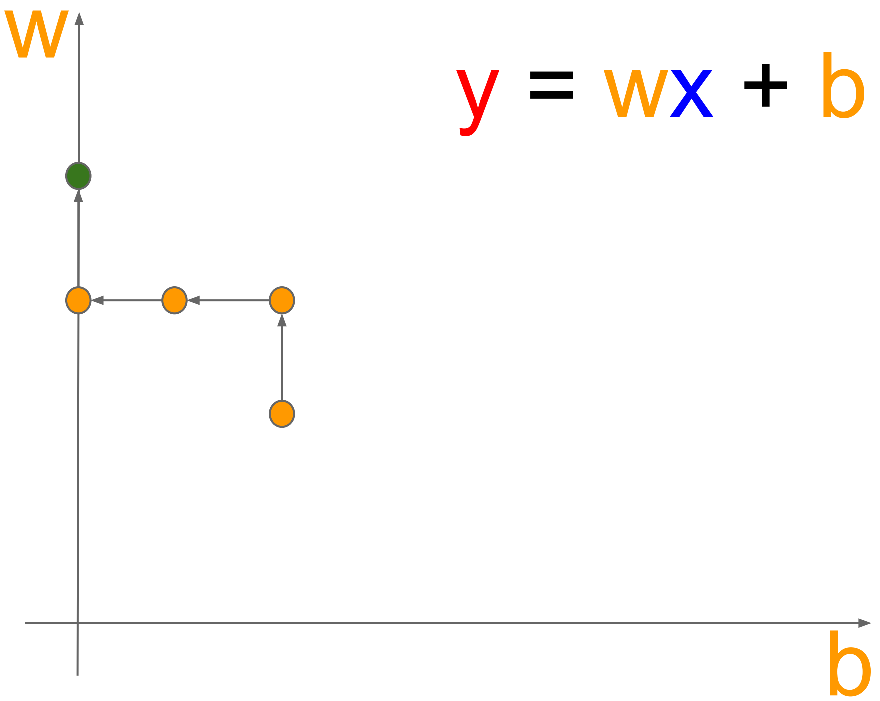
Optimizer

$$\underset{w,b\in[-\infty,\infty]}{\arg\min}\ C(w,b)$$

$w_3,b_3 = 3,0 : C(w_3,b_3) = 13$

$w_4,b_4 = 4,0 : C(w_4,b_4) = 104$

n	x	ŷ	y	$(y-\hat{y})^2$
0	1	0	4	16
1	5	16	20	16
2	6	20	24	16
			C(2,0)	54



$y = wx + b$

Optimizers are our friends

Optimizer

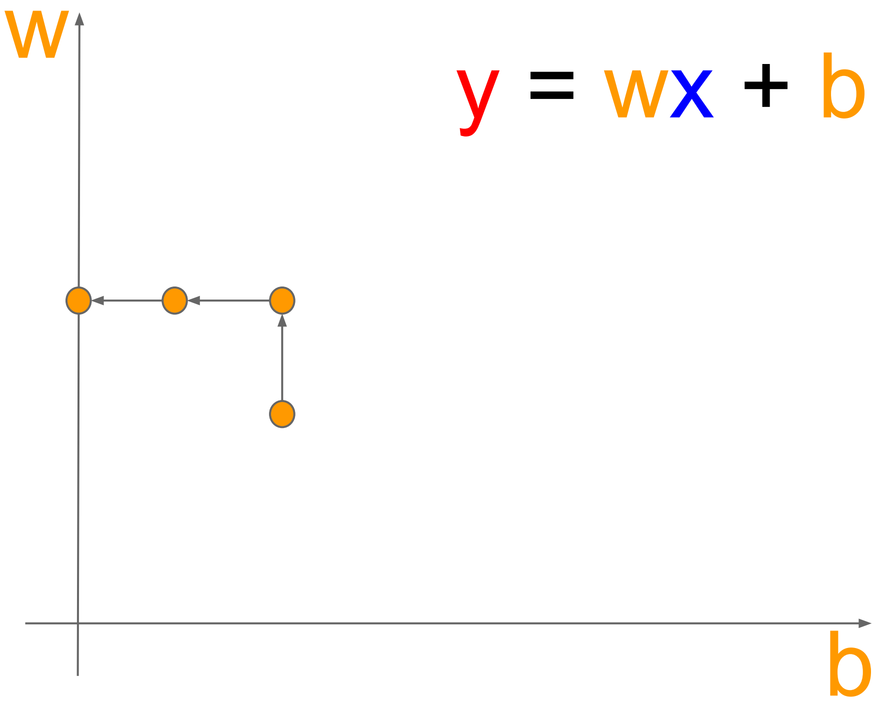
$$\arg\min C(w,b)$$

$w,b \in [-\infty,\infty]$

$w_3, b_3 = 3,0 : C(w_3, b_3) = 13$

The End?

$y = wx + b$

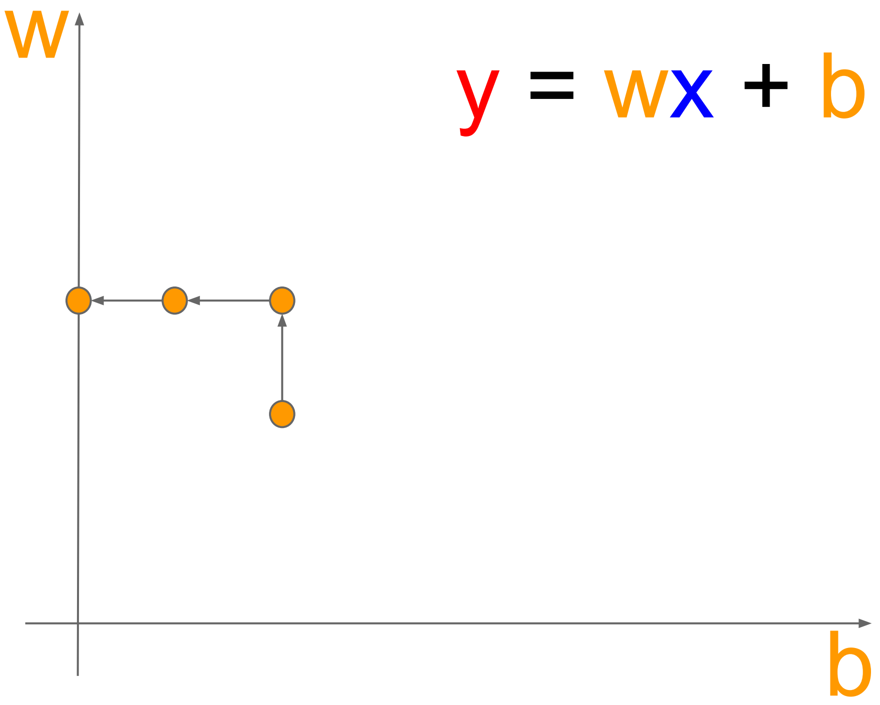


Optimizers are our friends

Optimizer

$$\underset{w,b \in [-\infty,\infty]}{\arg\min}\ C(w,b)$$

$w_?, b_? = 4, -2 : C(w_?, b_?) = ??$



$y = wx + b$

Optimizers are our friends

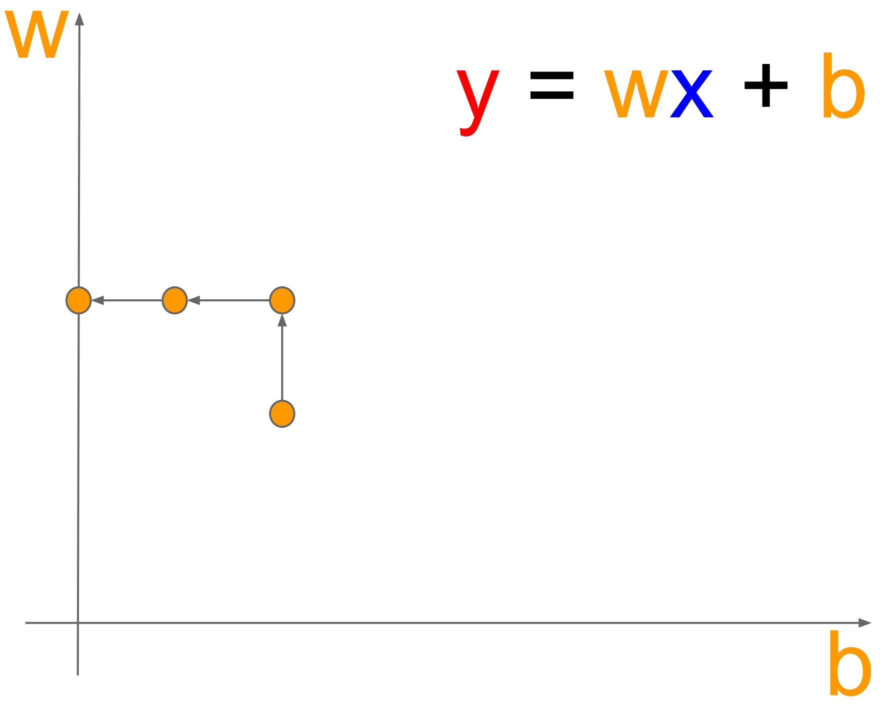
Optimizer

$\arg\min C(w,b)$

$w,b \in [-\infty,\infty]$

w?,b? = 4,-2 : C(w?,b?) = 12

$y = wx + b$



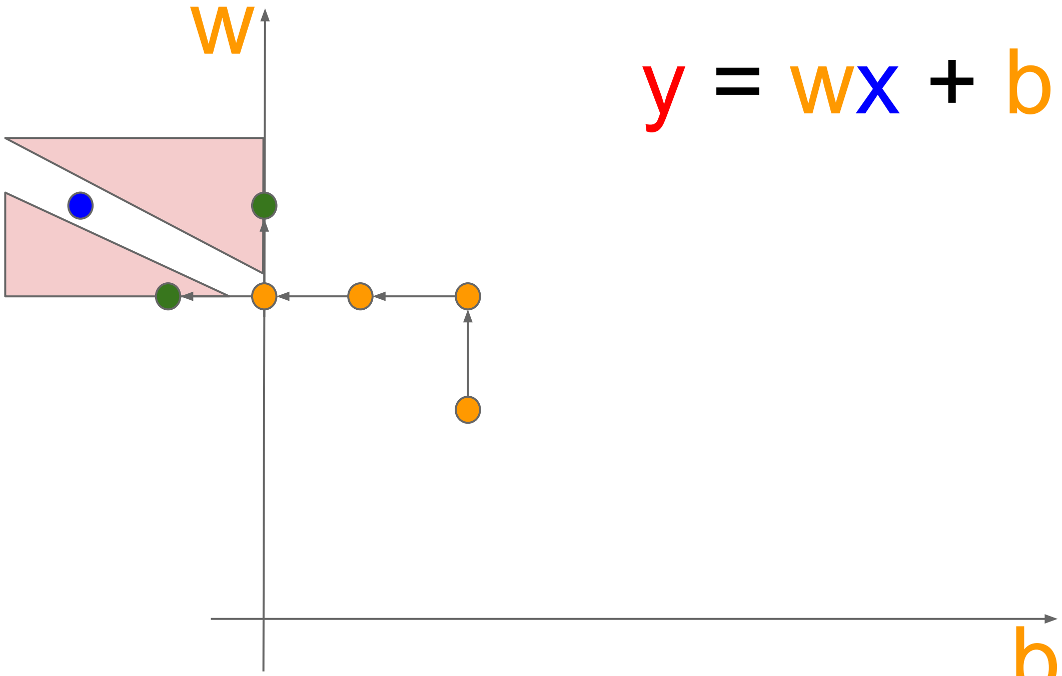
n	x	ŷ	y	$(y-\hat{y})^2$
0	1	0	2	4
1	5	16	18	4
2	6	20	22	4
			C(4,-2)	12

Optimizers are our friends

Optimizer

$$\underset{w,b \in [-\infty, \infty]}{\arg\min}\ C(w,b)$$

$w_3, b_3 = 3,0 : C(w_3, b_3) = 13$



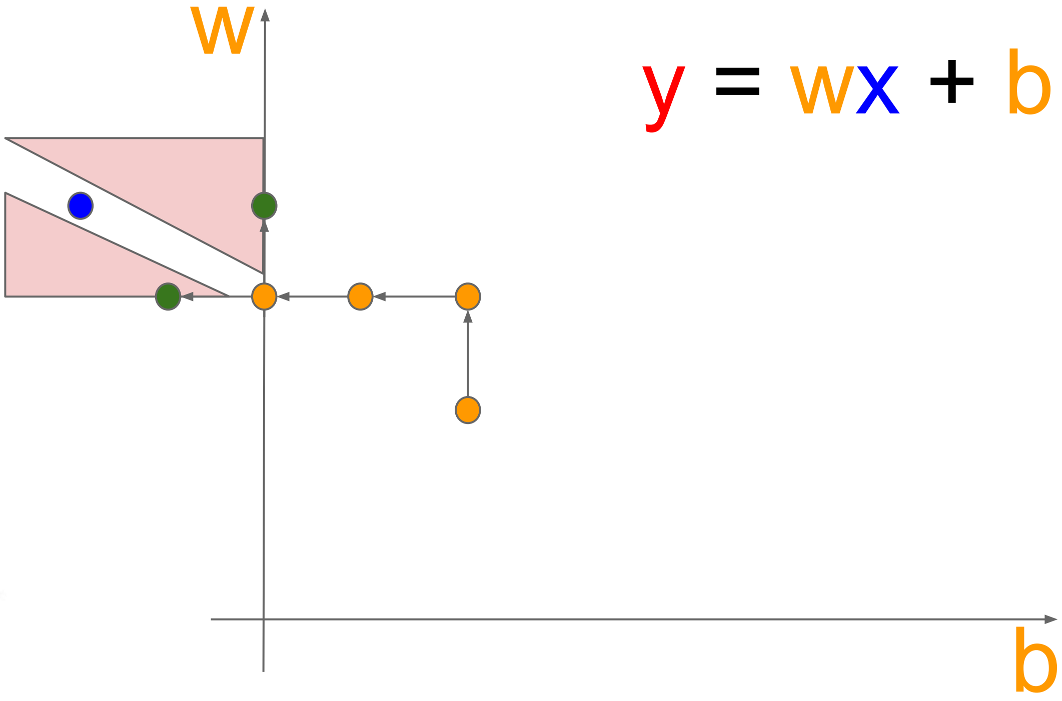
$y = wx + b$

Optimizers are our friends

Optimizer

$$\underset{w,b \in [-\infty,\infty]}{\arg\min}\; C(w,b)$$

$w_3,b_3 = 3,0 : C(w_3,b_3) = 13$



$y = wx + b$

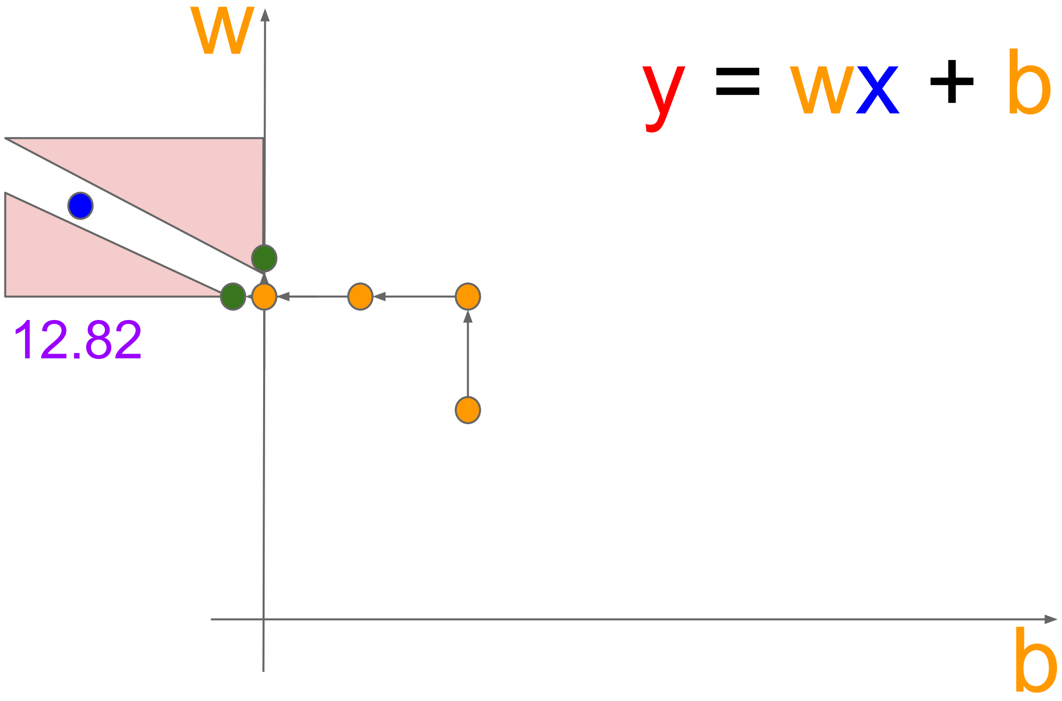
Optimizers are our friends

Optimizer

$$\arg\min_{w,b \in [-\infty,\infty]} C(w,b)$$

$w_3,b_3 = 3,0 : C(w_3,b_3) = 13$

$w_4,b_4 = 3.01,0 : C(w_4,b_4) = 12.82$



$y = wx + b$

n	x	$\hat{y}$	y	$(y-\hat{y})^2$
0	1	0	3.01	9.06
1	5	16	15.01	0.98
2	6	20	18.01	3.96

C(3.01,0) 12.82

Optimizers are our friends

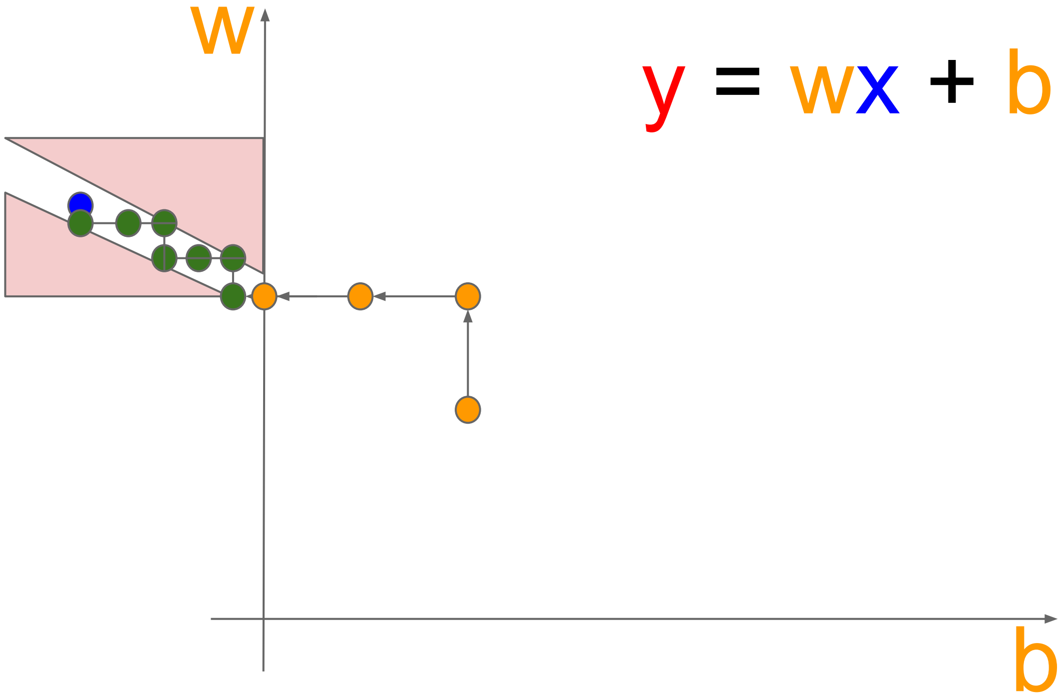
Optimizer

$$y = wx + b$$

$$\arg\min C(w,b)$$

$$w,b \in [-\infty, \infty]$$

$$w_*, b_* = 4,-2 : C(w_*, b_*) = 12$$



Optimizers are our friends

Large Step Size

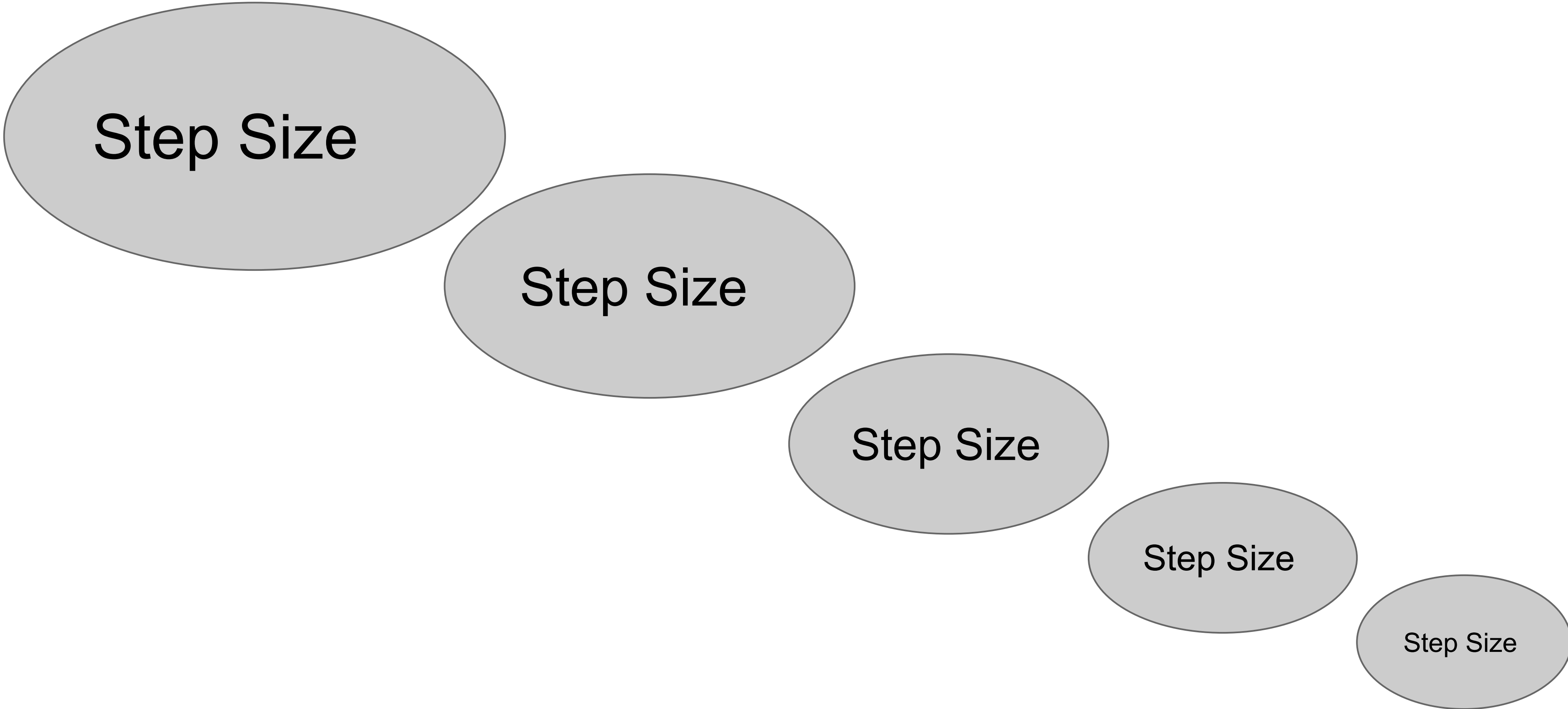
-Worse minimum
-But gets there faster

Vs

Small Step Size

-Better Minimum
-But gets there slowly

Optimizers are our friends

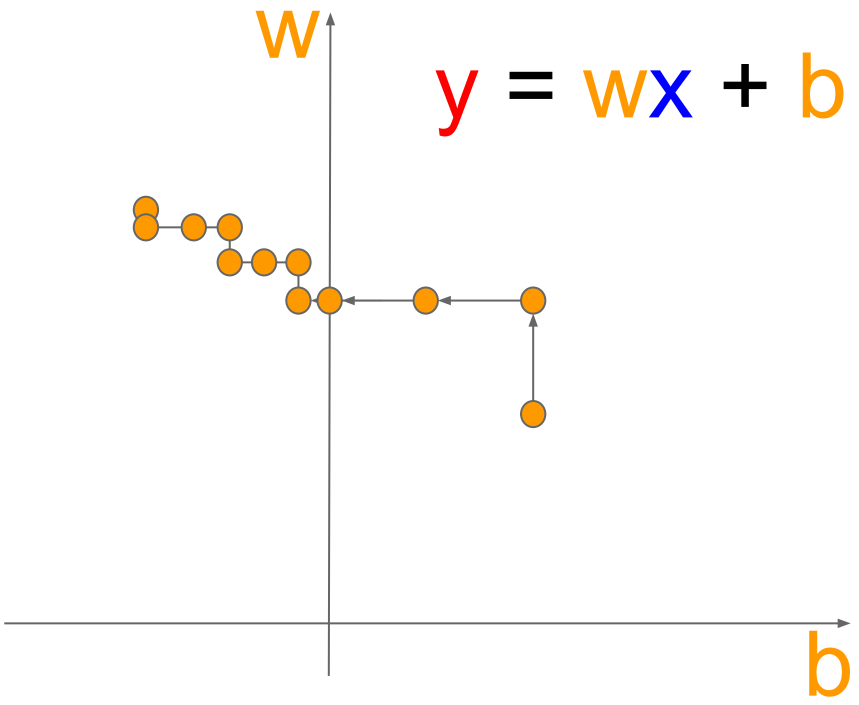


Optimizers are our friends

Optimizer

$$\arg\min_{w,b \in [-\infty,\infty]} C(w,b)$$

$w_*, b_* = 4,-2 : C(w_*, b_*) = 12$

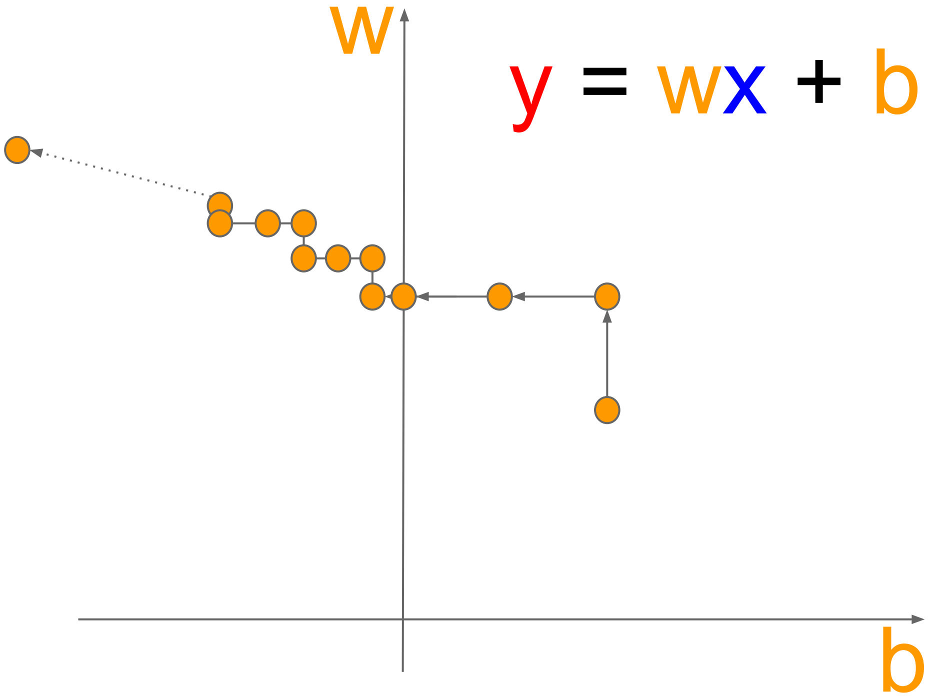


Optimizers are our friends

Optimizer

$$\arg\min_{w,b \in [-\infty,\infty]} C(w,b)$$

$w_*, b_* = 4,-4 : C(w_*, b_*) = 0$

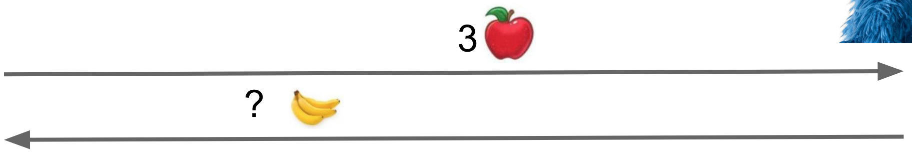


Optimizers are our friends

$y = wx + b$

Data

x	$\hat{y}$
1	0
5	16
6	20



Optimizers are our friends

$y = 4x - 4$

Data

x	$\hat{y}$
1	0
5	16
6	20

3 🍎

? 🍌

Optimizers are our friends

$y = 4x - 4$

Data

x	ŷ
1	0
5	16
6	20

3 🍎

8 🍌

Functions are our friends

$$y = wx + b$$

x : Image

y : Is this a cat

Functions are our friends

pixel (1,1)

pixel(1,3)

High
if cat

$y = w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b$

x : Image

y : Is this a cat

Functions are our friends

pixel (1,1)

pixel(1,3)

High
if cat

$$y = w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b$$

x : Image

y : Is this a cat

Millions of parameters

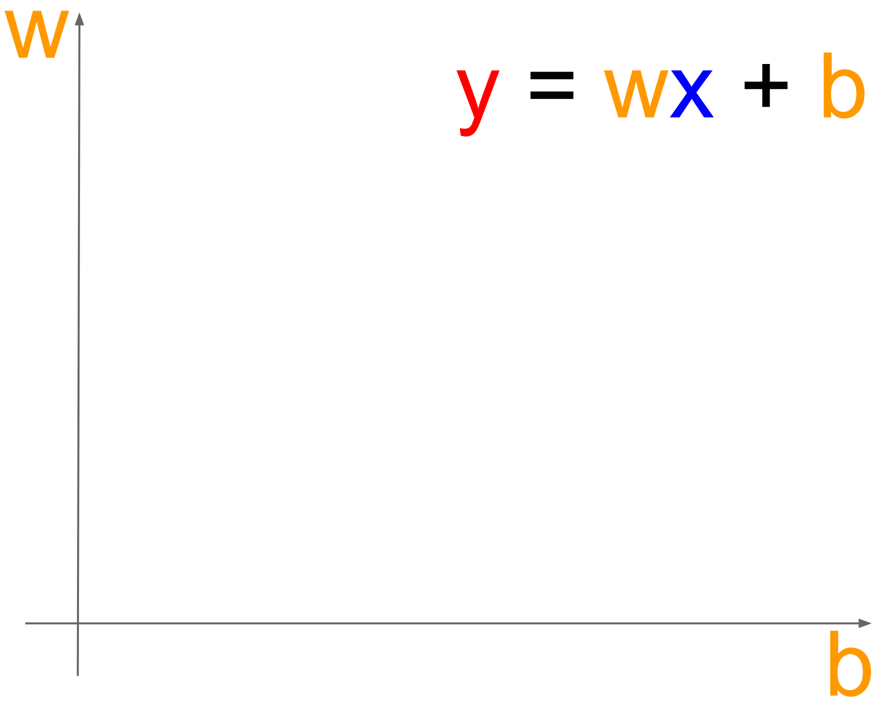
Millions of samples

Gradients are our friends

Optimizer

$$\underset{w,b \in [-\infty, \infty]}{\arg\min}\ C(w,b)$$

Very expensive
to compute
(hours or days)



$$y = wx + b$$

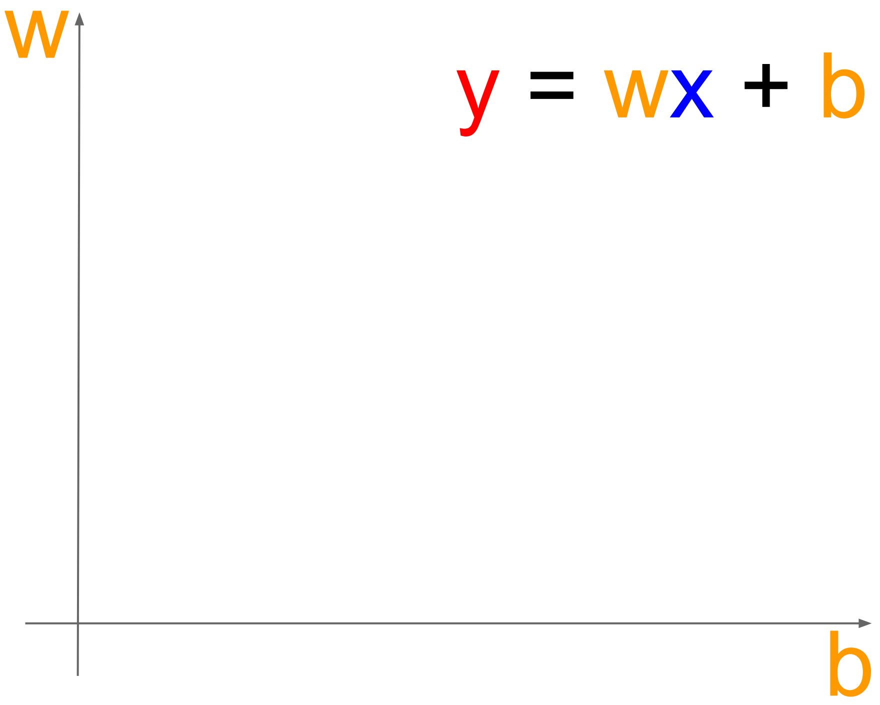
Gradients are our friends

Optimizer

$$\arg\min_{w,b \in [-\infty,\infty]} C(w,b)$$

Should be used sparingly

$$y = wx + b$$



Gradients are our friends

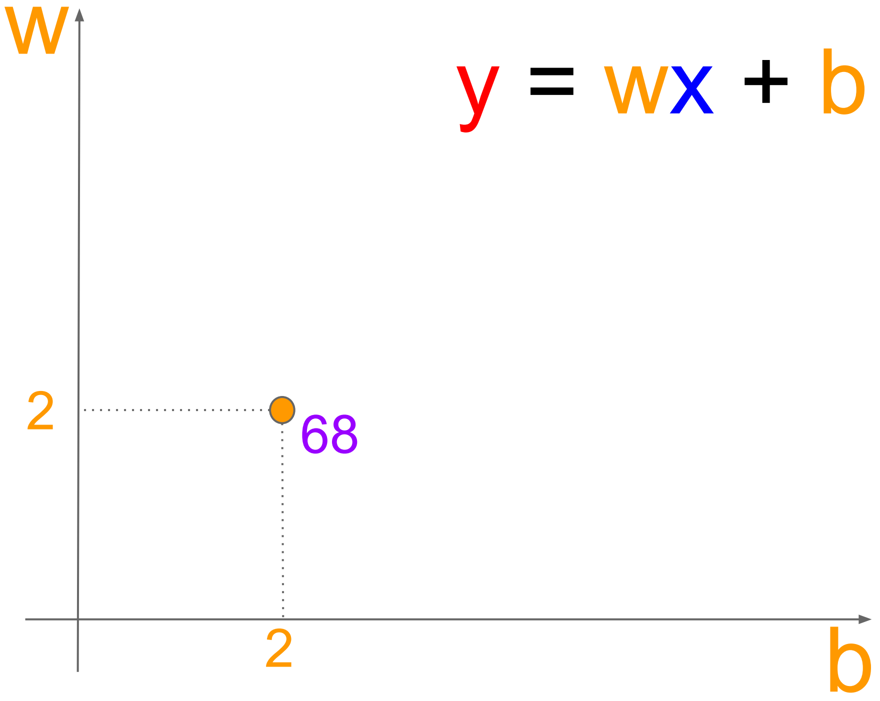
Optimizer

$$\arg\min C(w,b)$$

$w,b \in [-\infty,\infty]$

$w_0,b_0 = 2,2 : C(w_0,b_0) = 68$

$$y = wx + b$$



Gradients are our friends

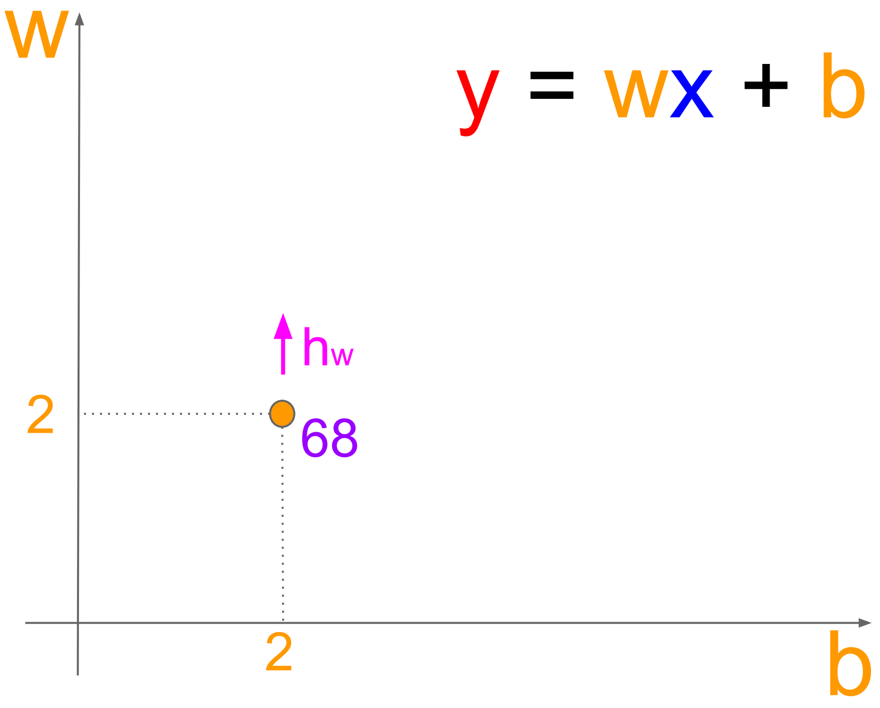
Optimizer

$$\arg\min_{w,b \in [-\infty,\infty]} C(w,b)$$

$w_0, b_0 = 2,2 : C(w_0, b_0) = 68$

$h_w = 1$

$y = wx + b$



Gradients are our friends

Optimizer

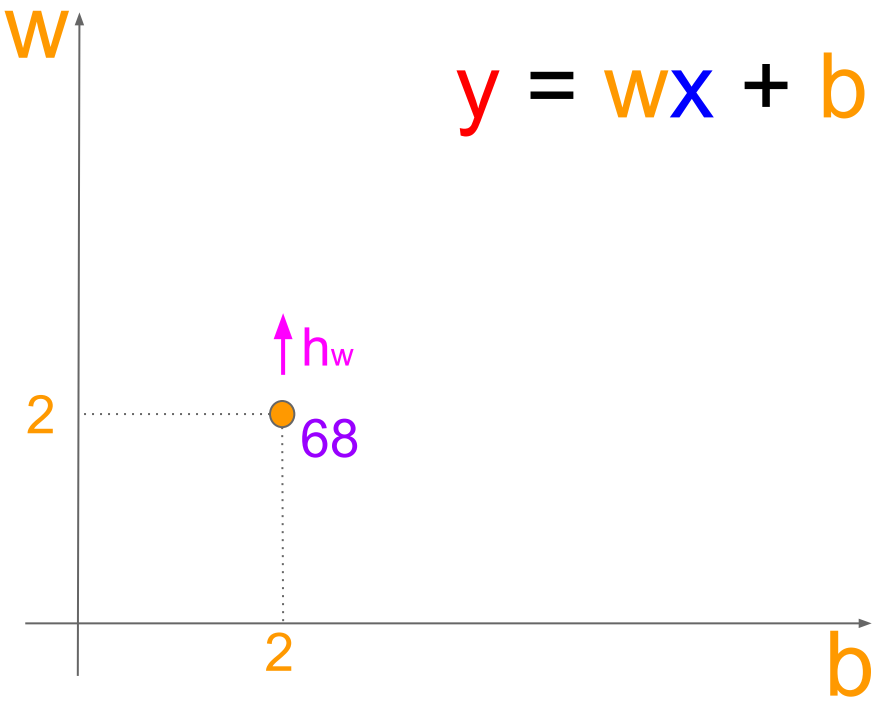
$$\arg\min_{w,b \in [-\infty,\infty]} C(w,b)$$

$w_0, b_0 = 2,2 : C(w_0, b_0) = 68$

$h_w = 1$

$C(w_0 + h_w, b_0) = C(3,2) = 26$

$y = wx + b$



Gradients are our friends

Optimizer

$$\arg\min C(w,b)$$

$$w,b \in [-\infty,\infty]$$

$$w_0,b_0 = 2,2 : C(w_0,b_0) = 68$$

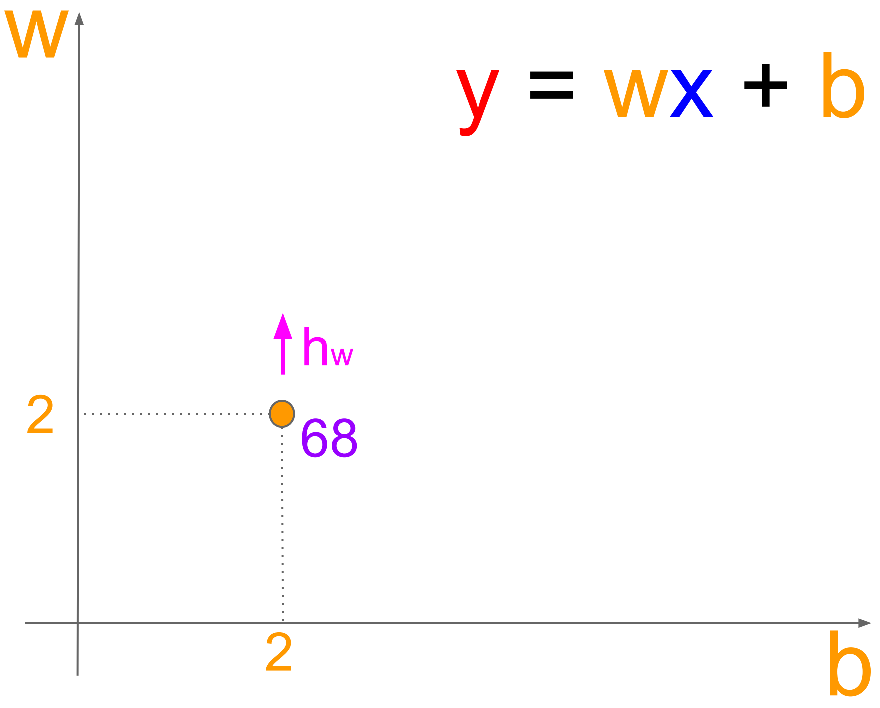
$$h_w = 1$$

$$C(w_0+h_w,b_0) = C(3,2) = 26$$

$$r = \frac{(C(w_0+1,b_0)-C(w_0,b_0))}{1}$$

$$r = \frac{(C(3,2)-C(2,2))=-42}{1}$$

$$y = wx + b$$



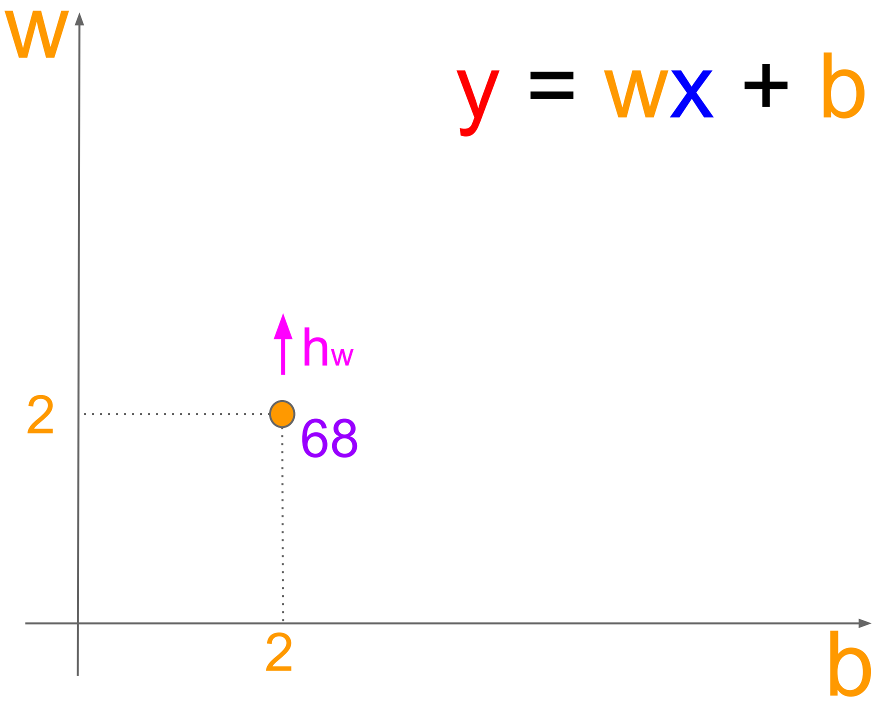
Gradients are our friends

Optimizer

$$\arg\min C(w,b)$$
$$w,b \in [-\infty, \infty]$$

$w_0, b_0 = 2,2 : C(w_0, b_0) = 68$

$h_w = 1$, $r = -42$
$h_w = 0.1$, $r = -98$
$h_w = 0.01$, $r = -104$
$h_w = 0.001$, $r = -104$



$$y = wx + b$$

Gradients are our friends

Optimizer

$$\arg\min C(w,b)$$

$w,b \in [-\infty,\infty]$

$w_0, b_0 = 2,2 : C(w_0,b_0) = 68$

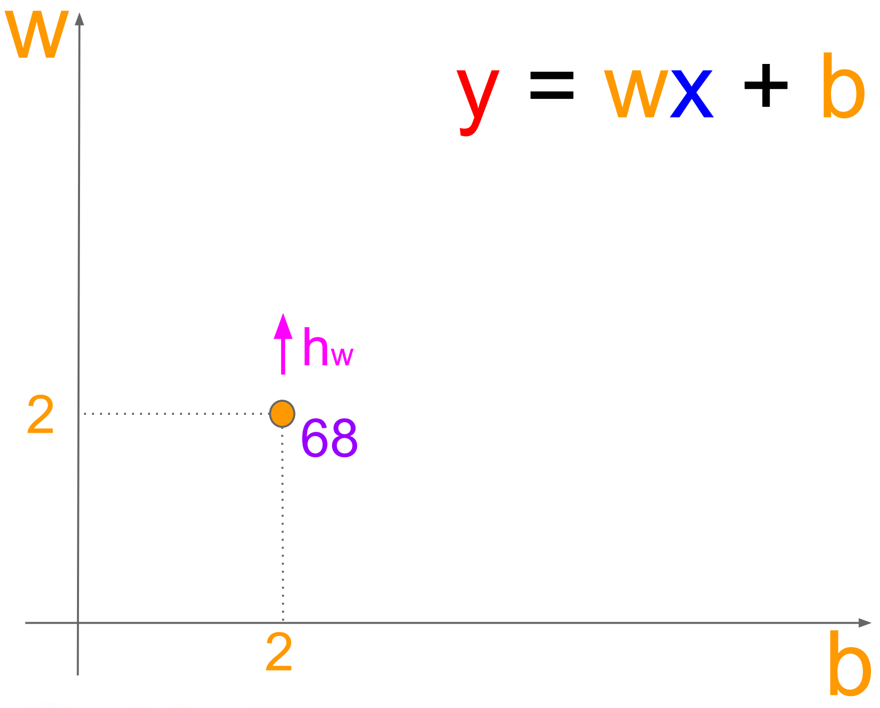
$h_w = 1$, $r = -42$

$h_w = 0.1$, $r = -98$

$h_w = 0.01$, $r = -104$

$h_w = 0.001$, $r = -104$

$h_w \rightarrow 0$, $r = \frac{\partial C}{\partial w}(w_0,b_0)$



$y = wx + b$

$$D_{\mathbf{u}} f(\mathbf{a}) = \lim_{h \to 0} \frac{f(\mathbf{a} + h\mathbf{u}) - \mathbf{f}(\mathbf{a})}{h}$$

Gradients are our friends

Optimizer

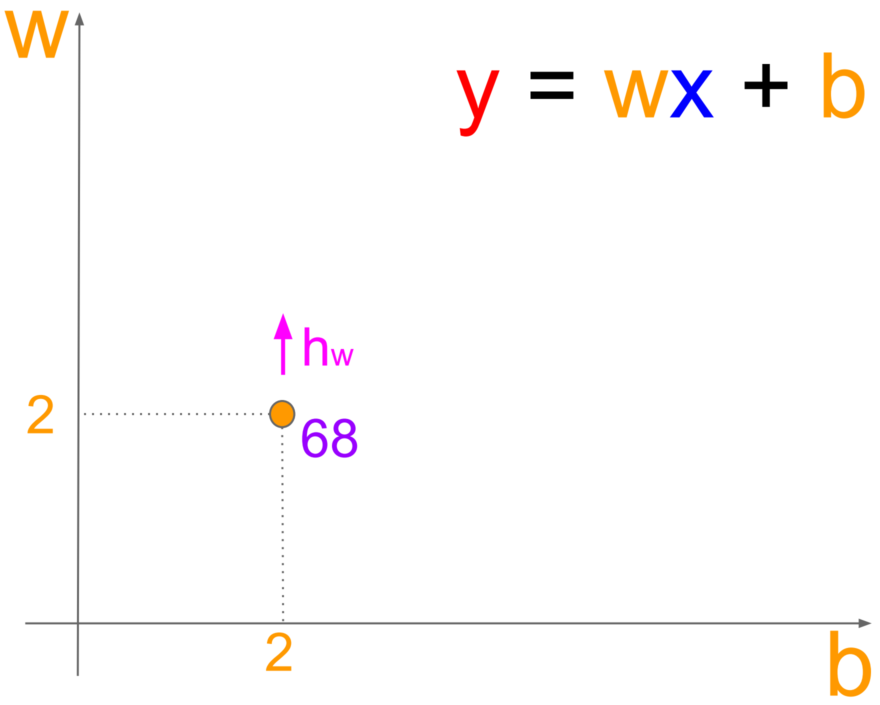
$$\arg\min C(w,b)$$

$w,b \in [-\infty,\infty]$

$w_0,b_0 = 2,2 : C(w_0,b_0) = 68$

$$\frac{\partial C}{\partial w} = \frac{\partial \sum_n (y_n - \hat{y}_n)^2}{\partial w}$$

$y = wx + b$



Gradients are our friends

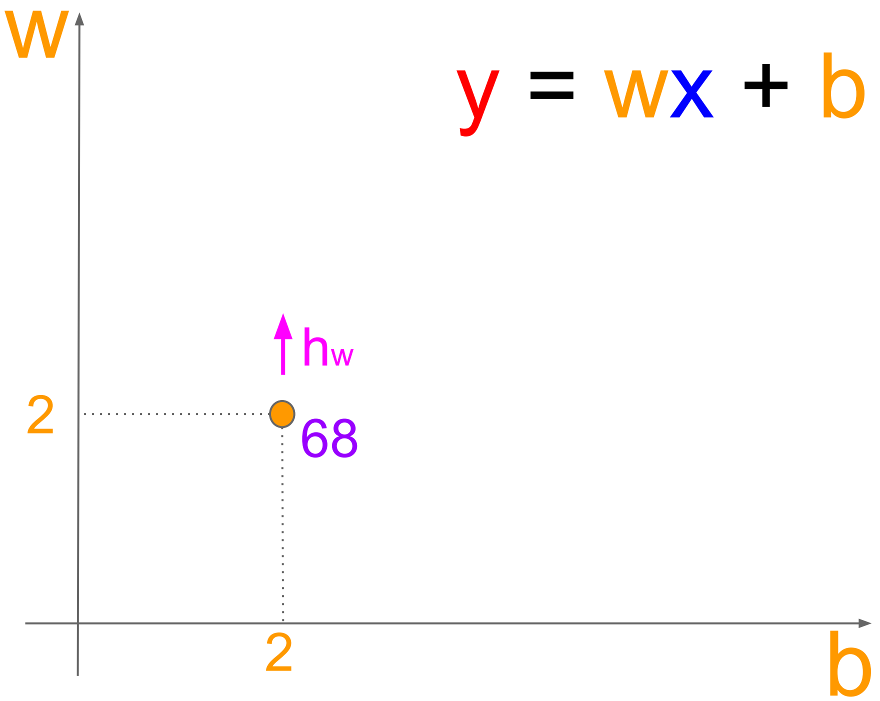
Optimizer

$$\underset{w,b\in[-\infty,\infty]}{\arg\min}\ C(w,b)$$

$$w_0,b_0 = 2,2 : C(w_0,b_0) = 68$$

$$\frac{\partial C}{\partial w} = \frac{\partial \sum_n (y_n-\hat{y}_n)^2}{\partial w} = \sum_n 2(y_n-\hat{y}_n)x_n$$

$$y = wx + b$$



Gradients are our friends

Optimizer

$$\underset{w,b \in [-\infty,\infty]}{\arg\min}\ C(w,b)$$

$w_0, b_0 = 2,2 : C(w_0,b_0) = 68$

$$\frac{\partial C}{\partial w} = \frac{\partial \sum_n (y_n - \hat{y}_n)^2}{\partial w} = \sum_n 2(y_n - \hat{y}_n)x_n$$

$$h_w \rightarrow 0,\ r = \frac{\partial C}{\partial w}(w_0,b_0) = -104$$

n	x	ŷ	y	(y-ŷ)	2(y-ŷ)x
0	1	0	4	4	8
1	5	16	12	-4	-40
2	6	20	14	-6	-72

Gradients are our friends

Optimizer

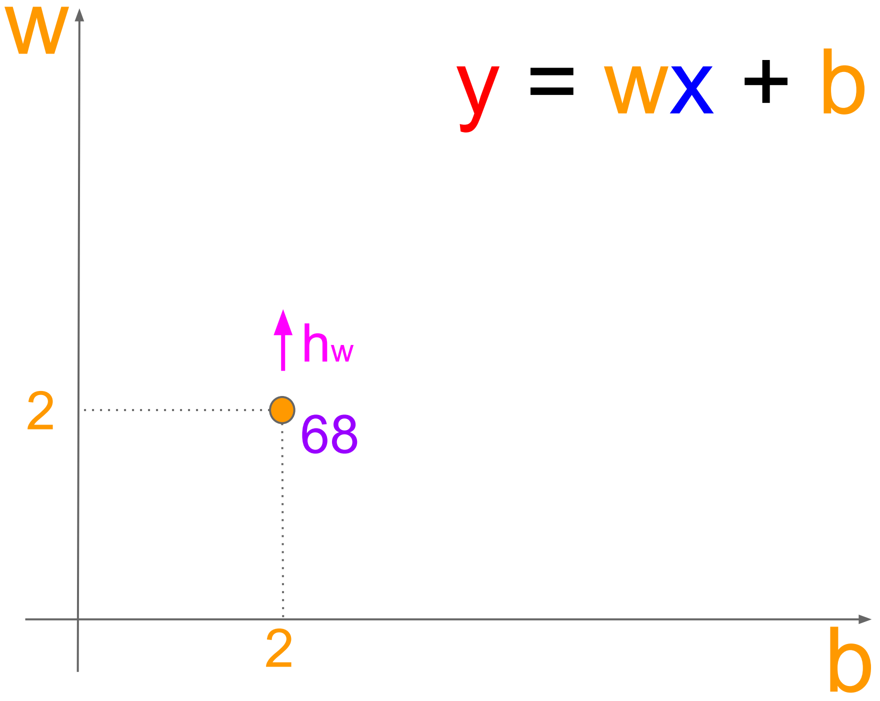
$$\arg\min C(w,b)$$
$$w,b \in [-\infty,\infty]$$

$$w_0,b_0 = 2,2 : C(w_0,b_0) = 68$$

$$\frac{\partial C}{\partial w} = \frac{\partial \sum_n (y_n-\hat{y}_n)^2}{\partial w} = \sum_n 2(y_n-\hat{y}_n)x_n$$

$$\frac{\partial C}{\partial b} = \frac{\partial \sum_n (y_n-\hat{y}_n)^2}{\partial b} = \sum_n 2(y_n-\hat{y}_n)$$

$$y = wx + b$$



Gradients are our friends

Optimizer

$$\underset{w,b \in [-\infty,\infty]}{\arg\min}\, C(w,b)$$

$$w_0, b_0 = 2,2 : C(w_0,b_0) = 68$$

$$h_w \rightarrow 0,\ r_w = \frac{\partial C}{\partial w}(w_0,b_0) = -104$$

$$h_b \rightarrow 0,\ r_b = \frac{\partial C}{\partial w}(w_0,b_0) = -12$$

n	x	ŷ	y	(y-ŷ)	2(y-ŷ)
0	1	0	4	4	8
1	5	16	12	-4	-8
2	6	20	14	-6	-12

Gradients are our friends

Optimizer

$$\underset{w,b \in [-\infty,\infty]}{\arg\min}\ C(w,b)$$

$w_0, b_0 = 2,2 : C(w_0,b_0) = 68$

$h_w \rightarrow 0,\ r_w = \frac{\partial C}{\partial w}(w_0,b_0) = -104$

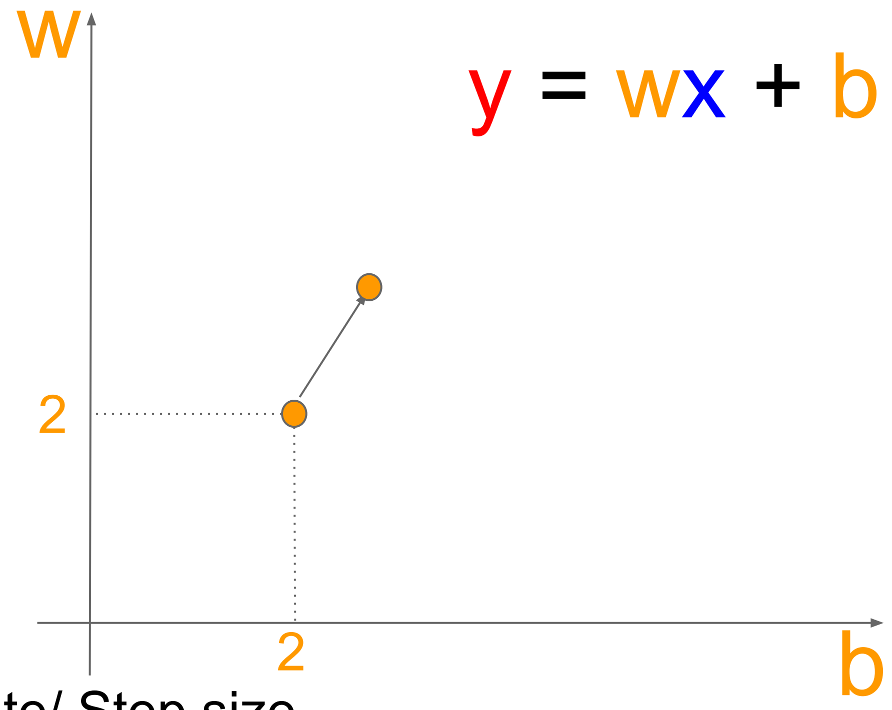
$h_b \rightarrow 0,\ r_b = \frac{\partial C}{\partial w}(w_0,b_0) = -12$

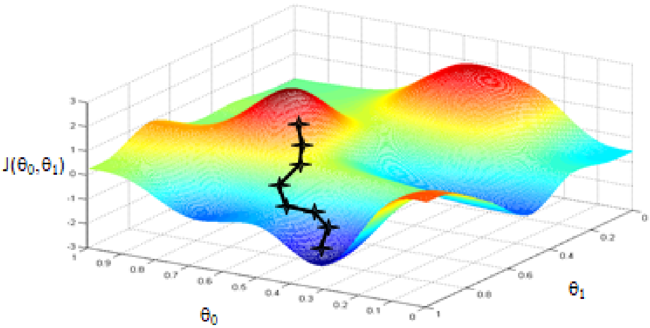
$w_1 = w_0 - r_w \alpha$

$b_1 = b_0 - r_b \alpha$

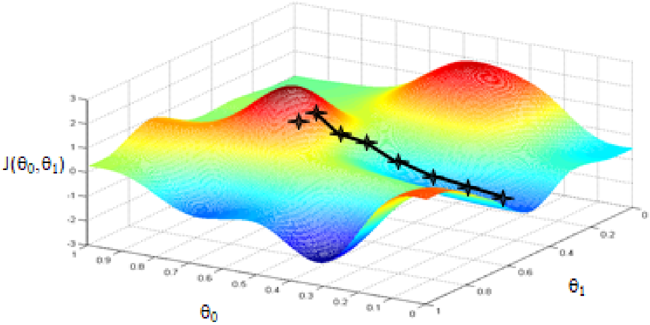
$\alpha \rightarrow$ Learning Rate/ Step size

$y = wx + b$





$J(\theta_0,\theta_1)$
3
2
1
0
-1
-2
-3
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
θ_0
0
0.2
0.4
0.6
0.8
1
θ_1



Andrew Ng

Summary

Data

n	x	ŷ
0	1	0
1	5	16
2	6	20

Model

$$y_n = wx_n + b$$

Cost

$$C(w,b) = \sum_{n \in \{0,1,2\}} (y_n - \hat{y}_n)^2$$

Optimizer

$$\underset{w,b \in [-\infty,\infty]}{\arg\min}\ C(w,b)$$

Summary

Data

n	x	$\hat{y}$
0	1	0
1	5	16
2	6	20

Model

$$y_n = wx_n + b$$

System

$$y = 4x - 4$$

Cost

$$C(w,b) = \sum_{n \in \{0,1,2\}} (y_n - \hat{y}_n)^2$$

Optimizer

$$\underset{w,b \in [-\infty,\infty]}{\arg\min}\ C(w,b)$$

This section

- Linear regression
  - Univariate case
  - Gradient descent algorithm

Regression

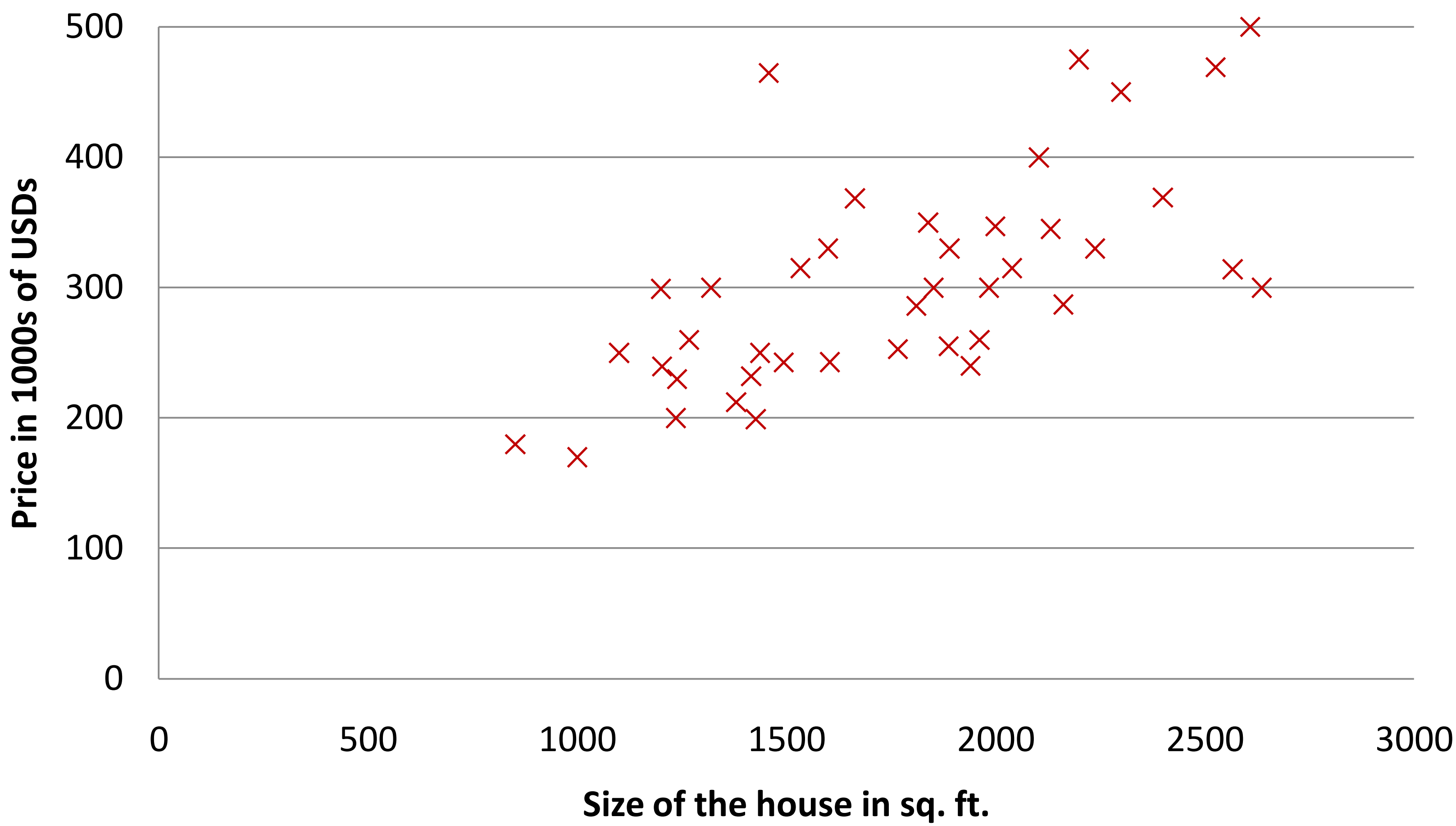
- Predicting a continuous outcome variable
  - Predicting the value of a company's future stock price using its pat and existing financial info
  - Predicting the amount of rainfall
  - Predicting …
- Key difference from classification
  - We measure prediction errors differently
  - This leads us to quite different learning models and algorithms

E.g., Predicting the sale price of a house

Which **features** to use? size, no. of rooms, neighborhood, annual taxes, requires renovation, etc..

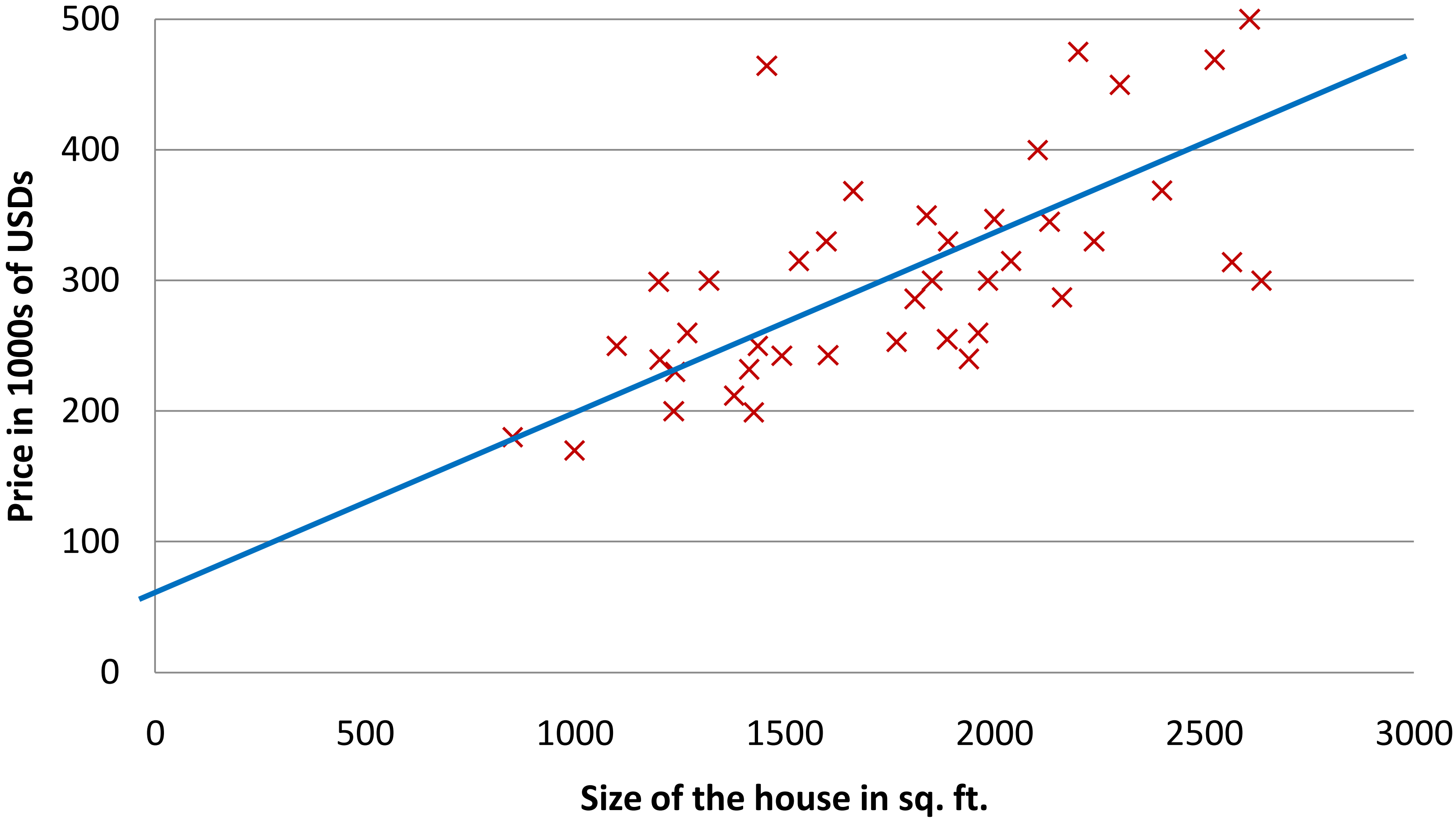
Let’s look at the relationship between price and size of the house

- Data for house sale prices in Portland, Oregon, USA



Possible linear relationship

Sale price ≈ price_per_sqft x square_footage + fixed_expense

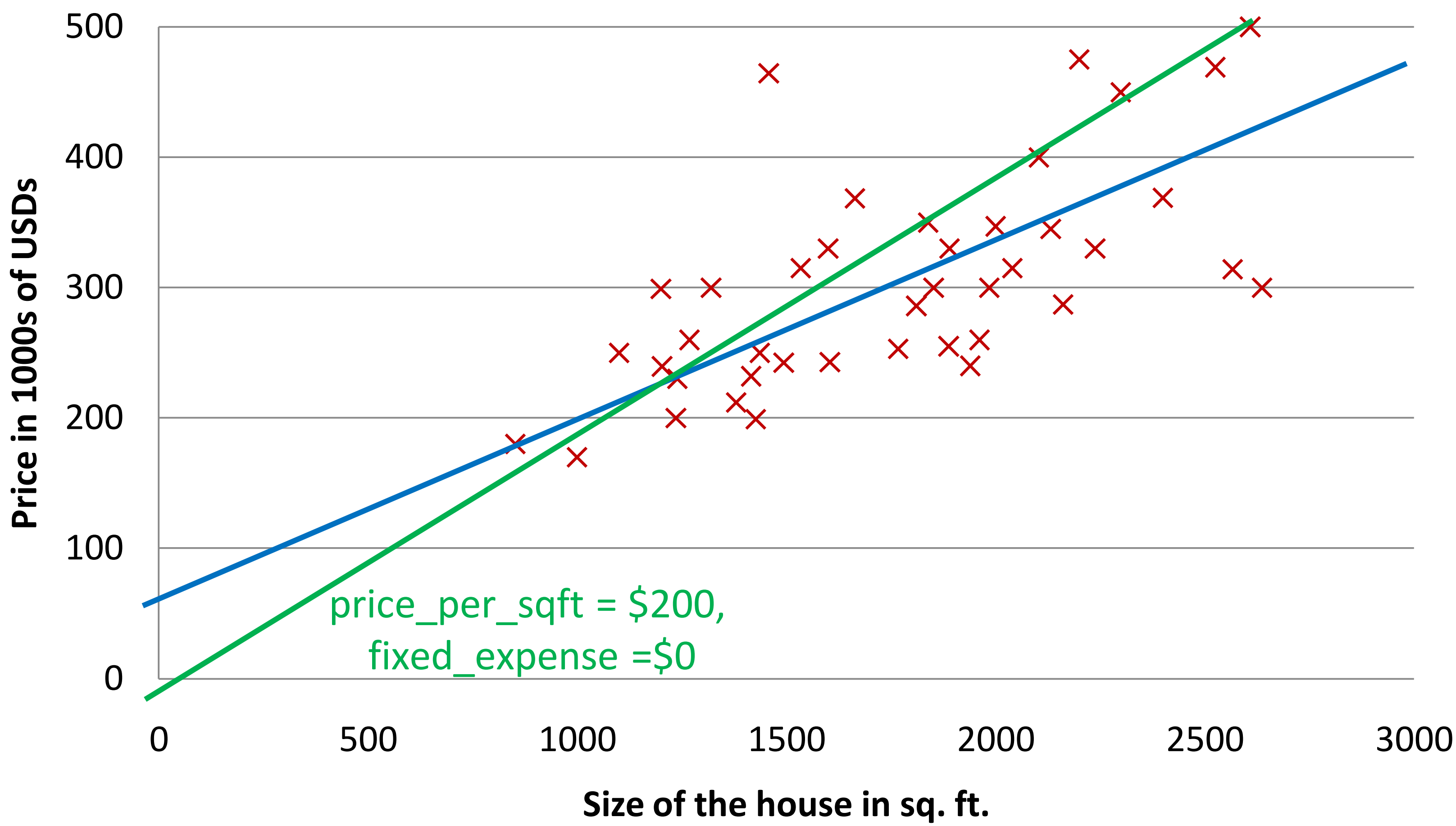


How to learn the parameters?

Size in feet^2 (x)	**Price ($) in 1000's (y)**
2104	460
1416	232
1534	315
852	178
…	…

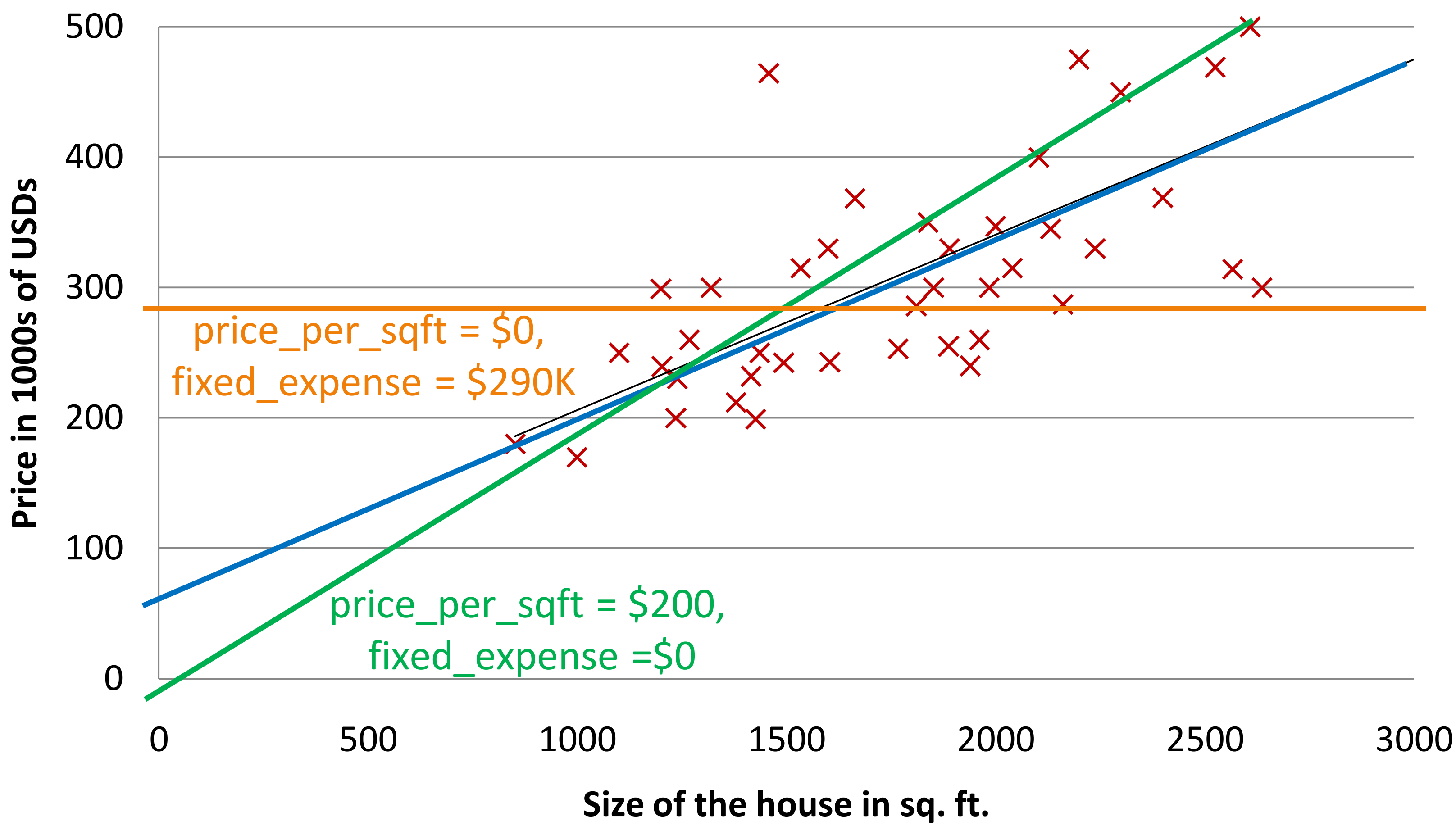
Which linear relationship

Sale price ≈ price_per_sqft x square_footage + fixed_expense



Which linear relationship

Sale price ≈ price_per_sqft x square_footage + fixed_expense



Definitions

- Let's denote the parameters: **price_per_sqft** as w_1 and **fixed_expense** as w_0

- The parameters, w_0 and w_1,are often represented together as a vector, **w**:

$$\mathbf{w} = [w_0 \; w_1]$$

- We can then make predictions using the function $f_{\mathbf{w}}$ with parameters **w** as follows (where, x = square footage of the house):

$$f_{\mathbf{w}}(x) = w_0 + w_1 x$$

*prediction functions are often called **hypothesis** in ML community*

- The function that computes the prediction error of the model with parameters **w** on the training set is called the **cost function** or the **error function**, $J(\mathbf{w})$

- **Goal**: Find w that minimizes the prediction error as much as possible

$$\arg\min_{\mathbf{w}} J(\mathbf{w})$$

How do we define errors?

- The classification error (hit or miss) is not appropriate for continuous outcomes

- We can look at the **absolute** difference:

$$|\text{prediction} - \text{sale price}|$$

- However, for simplicity we would look at the **squared** error:

$$(\text{prediction} - \text{sale price})^2$$

Residual sum of squares

- Define: $\mathrm{J}(\mathbf{w}) = \mathrm{RSS}(\mathbf{w})$

- $\mathrm{RSS}(\mathbf{w})$ is called **residual sum of squares**, defined as follows:

$$RSS(\mathbf{w}) = RSS(w_0, w_1) = \sum_n [y_n - f_{\mathbf{w}}(x_n)]^2 = \sum_n [y_n - (w_0 + w_1 x_n)]^2$$

- Other definitions of errors also exist.
- We will look into few examples as we go along.

Some intuition about RSS

Simplified

Hypothesis:

$$f_{\mathbf{w}}(x) = w_0 + w_1 x$$

$$f_{\mathbf{w}}(x) = w_1 x$$

Parameters:

$$\mathbf{w} = [w_0 w_1]$$

$$\mathbf{w} = [0 \quad w_1]$$

Cost Function:

$$J(\mathbf{w}) = \sum_n [y_n - (w_0 + w_1 x_n)]^2$$

$$J(\mathbf{w}) = \sum_n [y_n - w_1 x_n]^2$$

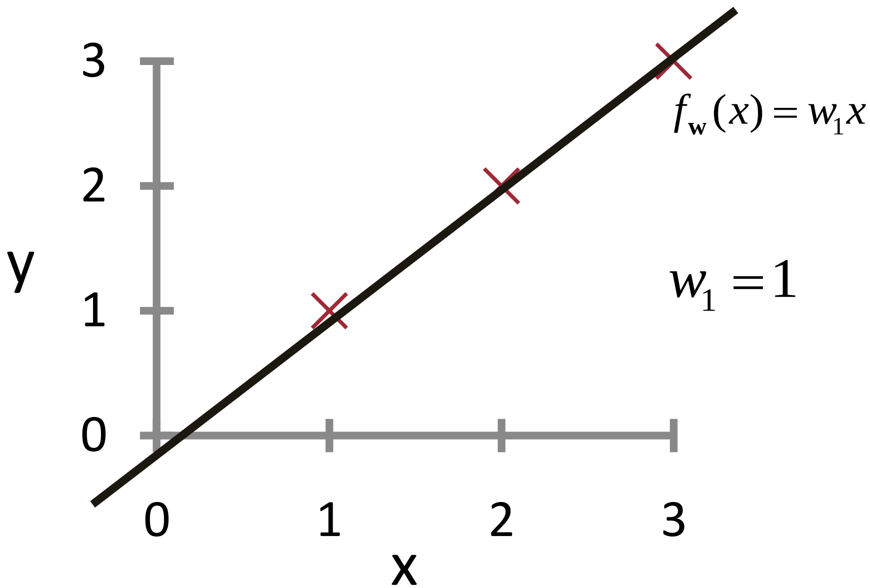
Goal:

$$\arg\min_{\mathbf{w}} J(\mathbf{w}) = \arg\min_{w_0, w_1} J(w_0, w_1)$$

$$\arg\min_{\mathbf{w}} J(\mathbf{w}) = \arg\min_{w_1} J(w_1)$$

$$f_{\mathbf{w}}(x)$$

(for fixed **w**, this is a function of x)

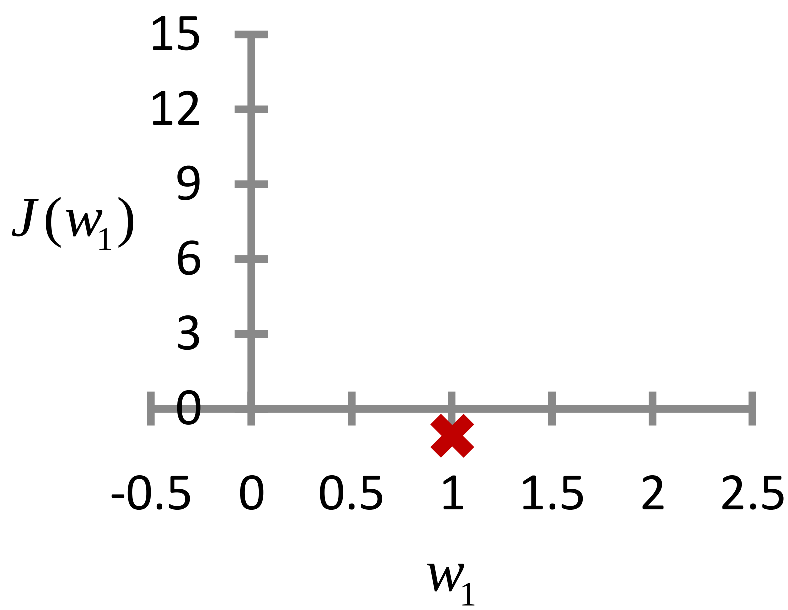


$$J(w_1) = (1-1)^2 + (2-2)^2 + (3-3)^2$$

$$J(w_1) = 0$$

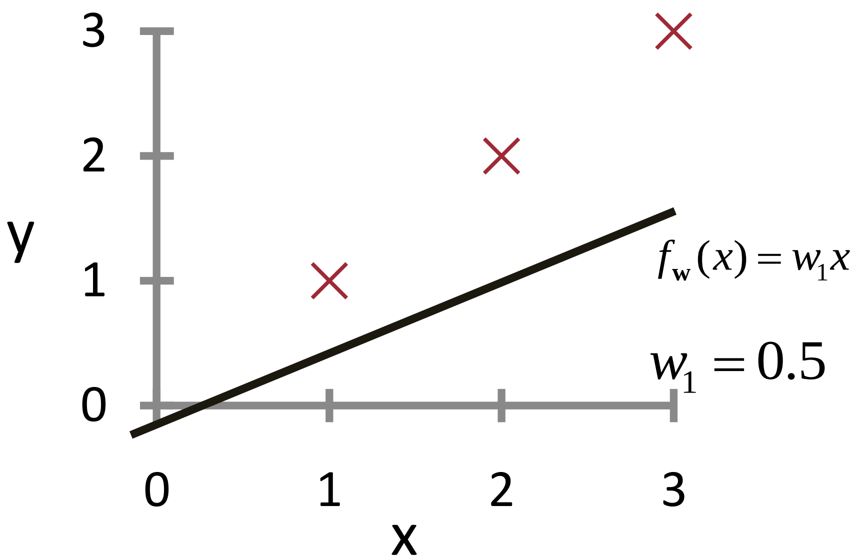
$$J(w_1)$$

(function of the parameter w_1)



$f_{\mathbf{w}}(x)$

(for fixed **w**, this is a function of x)

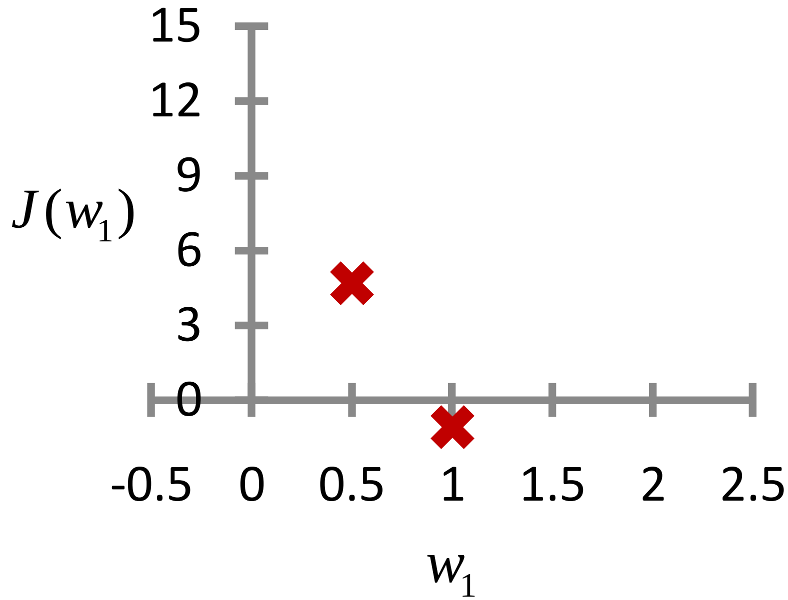


$$J(w_1) = (1 - 0.5)^2 + (2 - 1)^2 + (3 - 1.5)^2$$

$$J(w_1) = 3.5$$

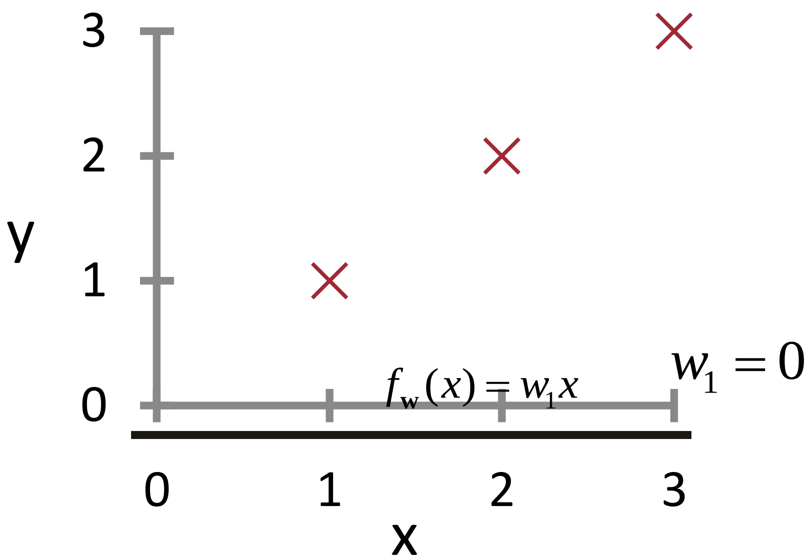
$J(w_1)$

(function of the parameter w_1)



$f_{\mathbf{w}}(x)$

(for fixed **w**, this is a function of x)

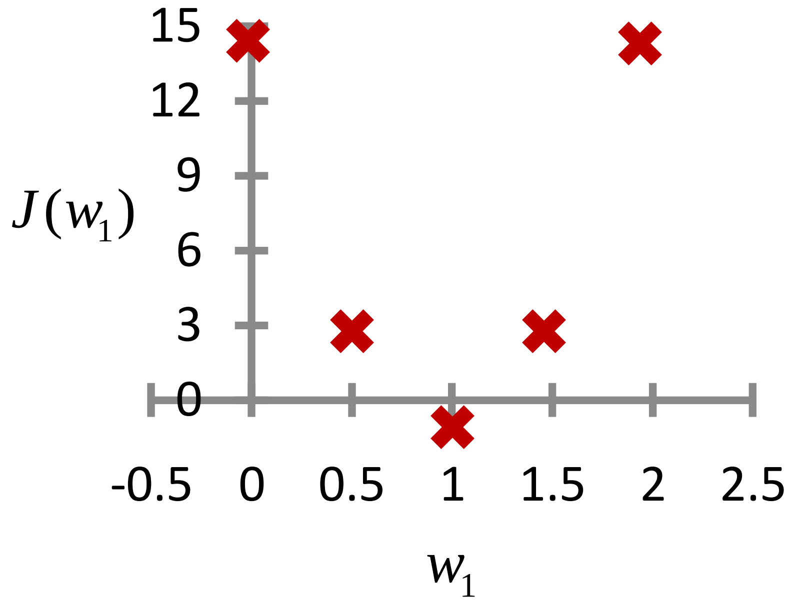


$$J(w_1) = (1-0)^2 + (2-0)^2 + (3-0)^2$$

$$J(w_1) = 14$$

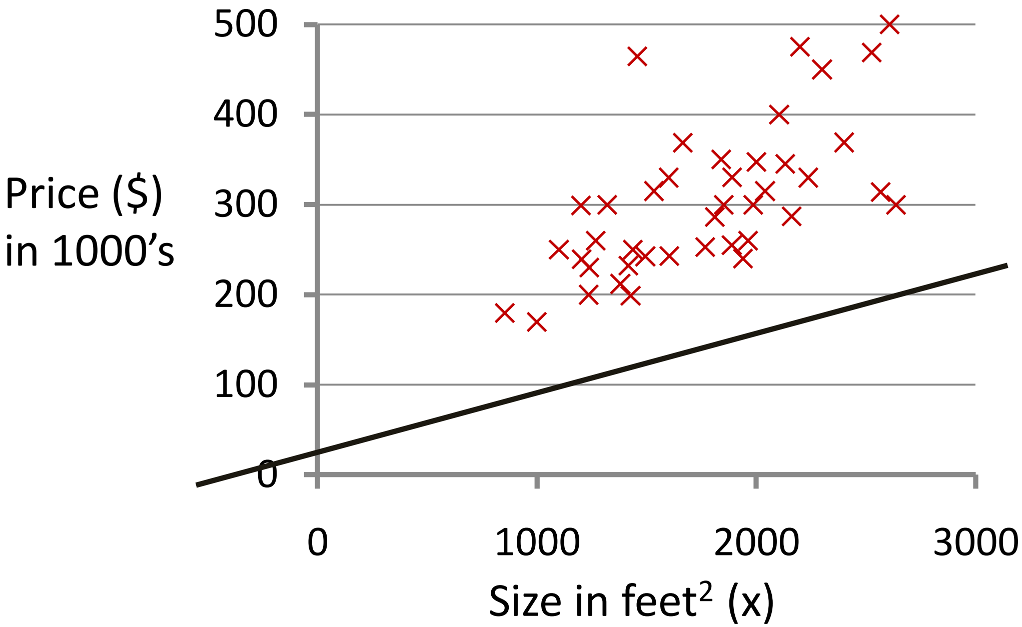
$J(w_1)$

(function of the parameter w_1)



$f_{\mathbf{w}}(x)$

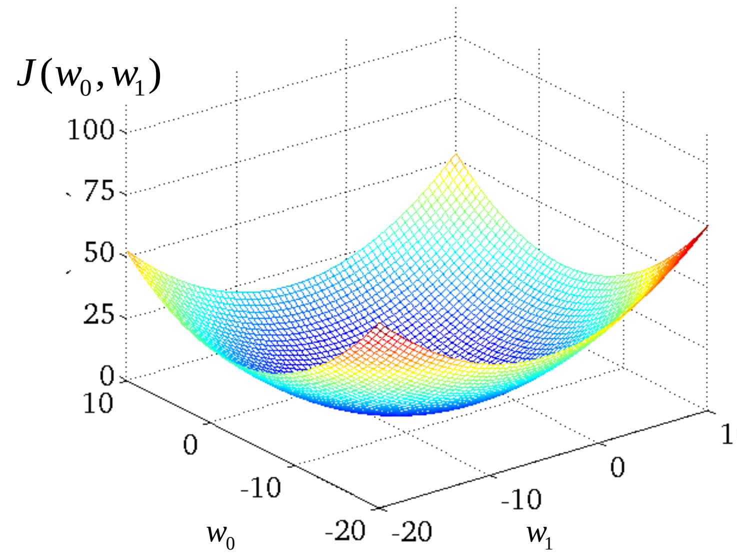
(for fixed **w**, this is a function of x)



$$f_{\mathbf{w}}(x) = 50 + 0.06x$$

$J(w_0, w_1)$

(function of parameters w_0, w_1)

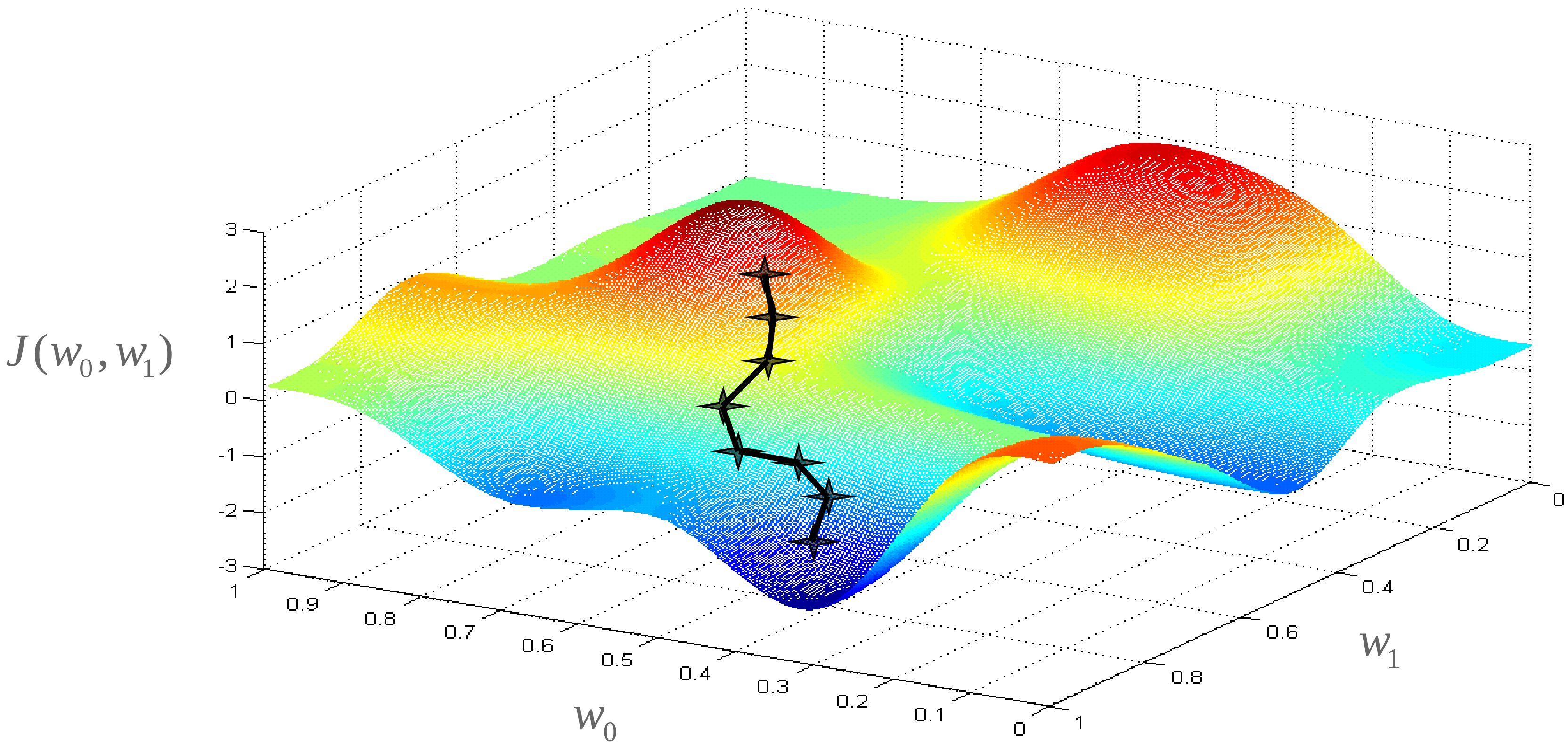


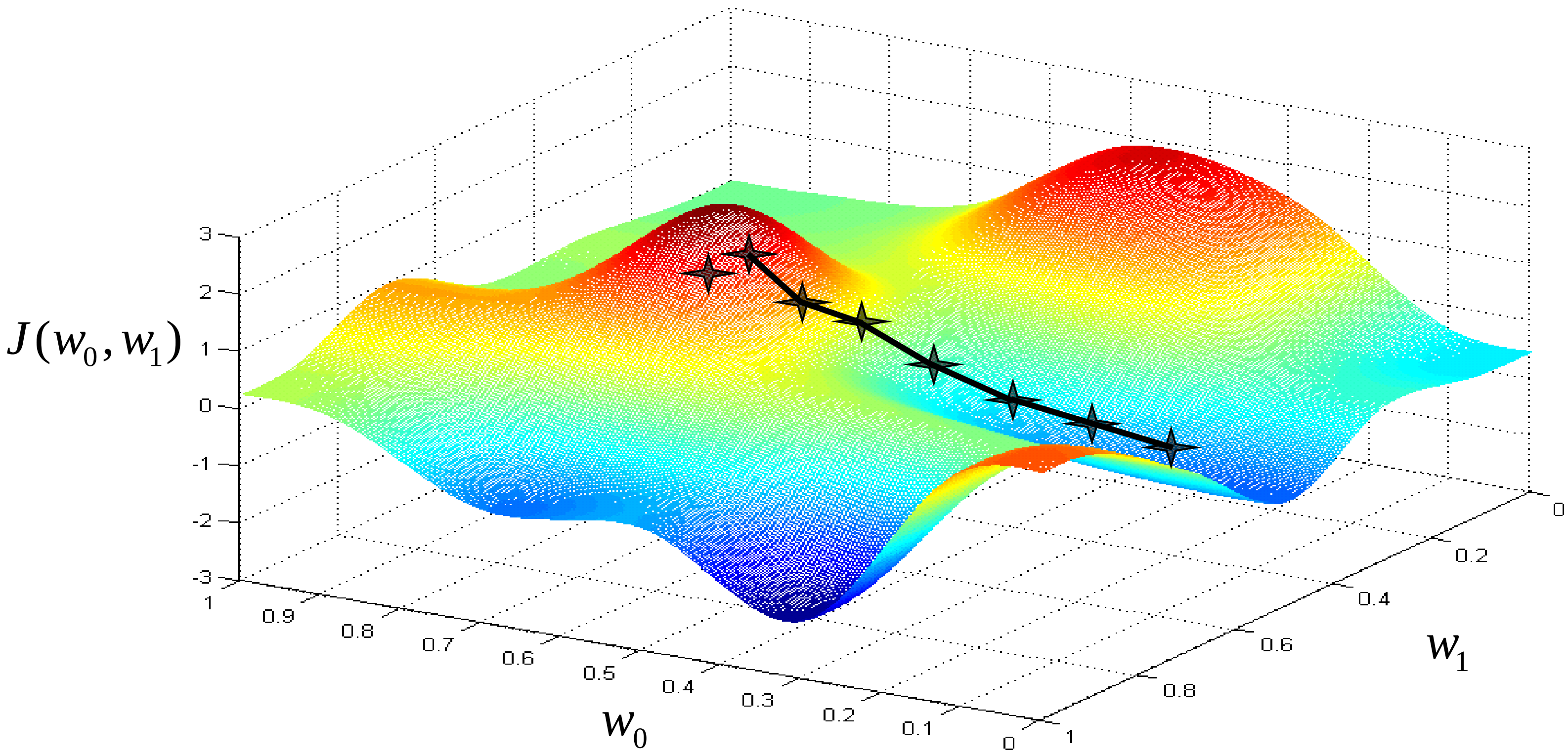
Gradient Descent Algorithm

Gradient Descent algorithm

- Have some function $J(\mathbf{w}) = J(w_0, w_1)$
- Want $\arg\min_{\mathbf{w}} J(\mathbf{w}) = \arg\min_{w_0, w_1} J(w_0, w_1)$

- Outline
  - Start with some w_0, w_1
  - Keep changing w_0, w_1 to reduce $J(w_0, w_1)$ until we hopefully end at a minimum


$J(w_0, w_1)$
3
2
1
0
-1
-2
-3
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
w_0
0
0.2
0.4
0.6
0.8
1
w_1



$J(w_0, w_1)$
3
2
1
0
-1
-2
-3
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
w_0
0
0.2
0.4
0.6
0.8
1
w_1

Gradient descent algorithm

Repeat until convergence {

$$w_i := w_i - \alpha \frac{\partial}{\partial w_i} J(w_0, w_1) \quad \text{for } i = 0 \text{ and } i = 1$$

}

Learning rate **Partial derivative**

Correct: Simultaneous update

$$temp0 := w_0 - \alpha \frac{\partial}{\partial w_0} J(w_0, w_1)$$

$$temp1 := w_1 - \alpha \frac{\partial}{\partial w_1} J(w_0, w_1)$$

$$w_0 := temp0$$

$$w_1 := temp1$$

Incorrect:

$$temp0 := w_0 - \alpha \frac{\partial}{\partial w_0} J(w_0, w_1)$$

$$w_0 := temp0$$

$$temp1 := w_1 - \alpha \frac{\partial}{\partial w_1} J(w_0, w_1)$$

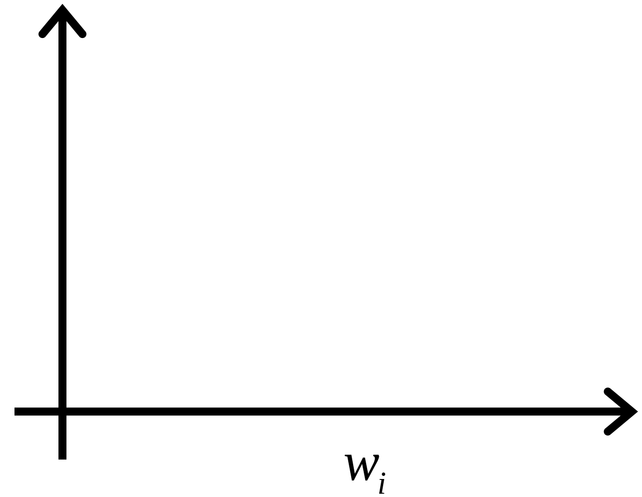
$$w_1 := temp1$$

Relating maths with intuition

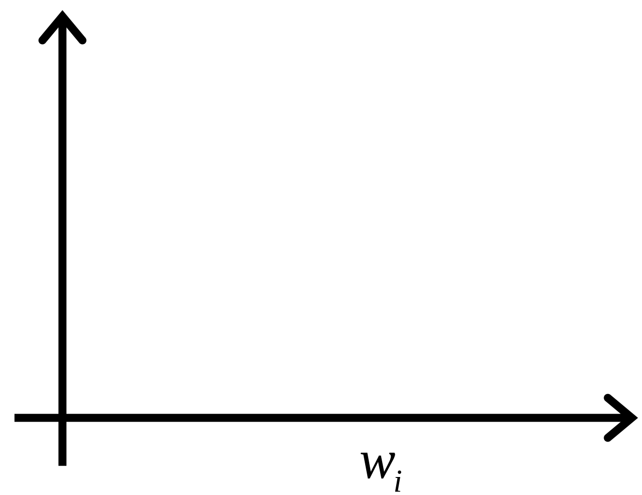
Effect of learning rate

$$w_i := w_i - \alpha \frac{\partial}{\partial w_i} J(w_0, w_1)$$

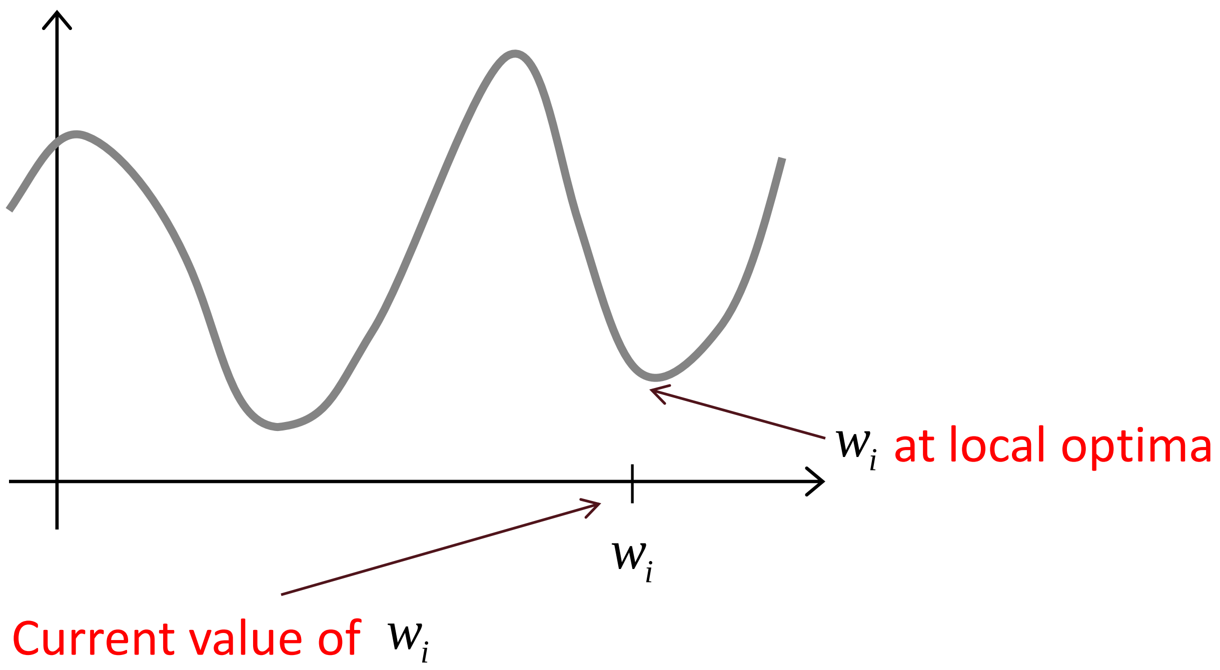
If α is too small, gradient descent can be slow.



If α is too large, gradient descent can overshoot the minimum. It may fail to converge, or even diverge.



What if you initialize w_i at a minimum



Gradient descent for Linear regression

Gradient descent algorithm for Linear regression

- Gradient descent
  - Repeat until convergence {

$$w_i := w_i - \alpha \frac{\partial}{\partial w_i} J(w_0, w_1) \quad \text{for } i = 0 \text{ and } i = 1$$

}

- For linear regression

$$J(\mathbf{w}) = \sum_n [y_n - (w_0 + w_1 x_n)]^2$$

- For $w_0 : \frac{\partial}{\partial w_0} \sum_n [y_n - (w_0 + w_1 x_n)]^2 = 2 \sum_n [y_n - (w_0 + w_1 x_n)]$

- For $w_1 : \frac{\partial}{\partial w_1} \sum_n [y_n - (w_0 + w_1 x_n)]^2 = 2 \sum_n [y_n - (w_0 + w_1 x_n)].x_n$

Gradient descent algorithm for Linear regression

Repeat until convergence {

$$w_0 := w_0 - \alpha \sum_n [y_n - (w_0 + w_1 x_n)]$$

$$w_1 := w_1 - \alpha 2 \sum_n [y_n - (w_0 + w_1 x_n)].x_n$$

}

This particular version is called "**Batch**" gradient descent

Are there other methods to find optimal **w**

- Closed form solution exists using Linear Algebra

$$\mathbf{y} = \mathbf{w}^T X - \mathbf{b}$$

- The method is called **Least squares** method