AI VIETNAM
All-in-One Course
(TA Session)

Simple Linear Regression

Extra

AI VIET NAM
@aivietnam.edu.vn

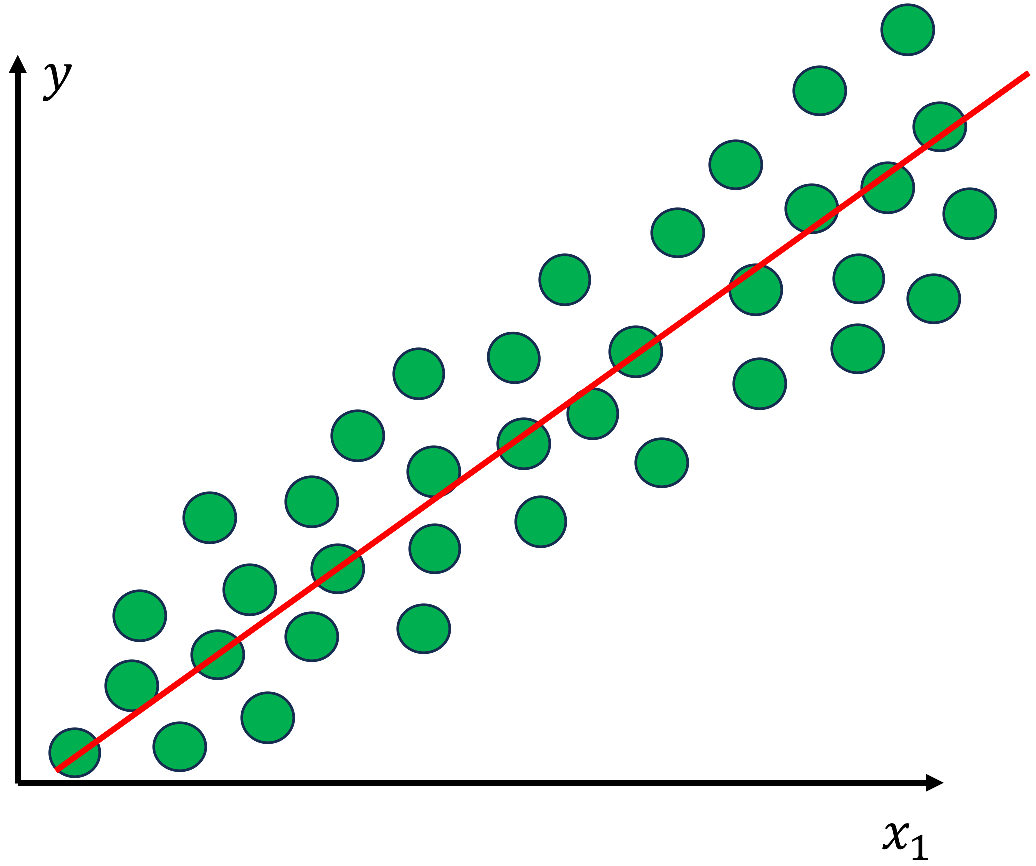
Dinh-Thang Duong – TA

Anh-Khoi Nguyen – STA

Year 2024

Getting Started

❖ Objectives



Our objectives:

- Introduction to Linear Regression.
- Discuss the idea of how Linear Regression learn to solve a Regression problem.
- Discuss about Gradient Descent.
- Implement a simple version of Linear Regression using Gradient Descent.

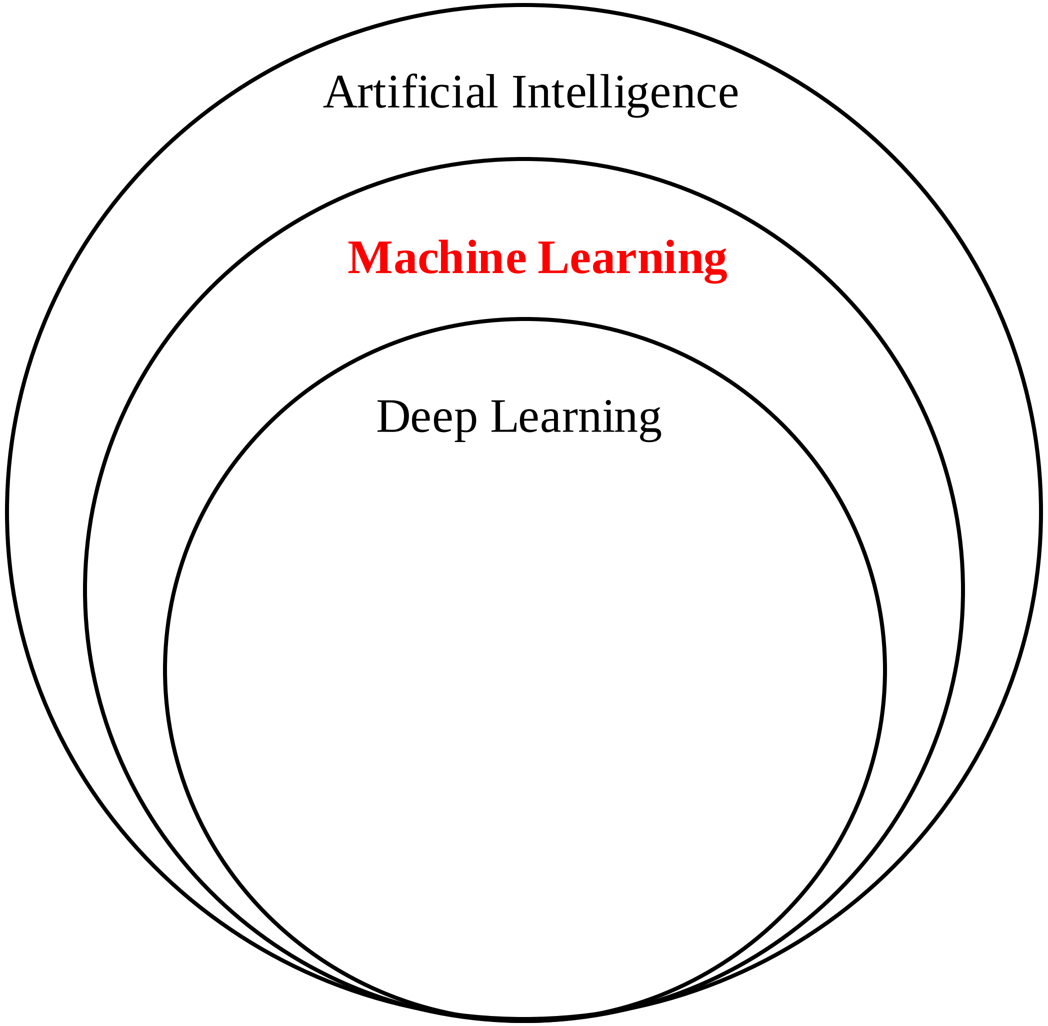
Outline

- Introduction
- Simple Linear Regression
- Code Implementation
- Question

Introduction

Introduction

❖ Getting Started



Machine Learning (ML): A branch of AI and Computer Science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy.

https://www.ibm.com/topics/machine-learning

Introduction

❖ Getting Started

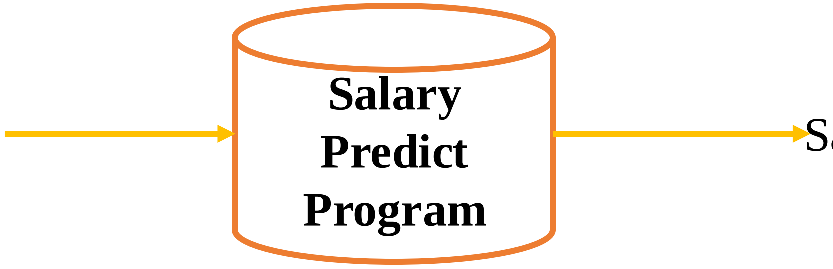
Suppose you got some dataset:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99.0	SF	25.0	6-6	235.0	Marquette	6796117.0
2	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	Boston University	NaN
3	R.J. Hunter	Boston Celtics	28.0	SG	22.0	6-5	185.0	Georgia State	1148640.0
4	Jonas Jerebko	Boston Celtics	8.0	PF	29.0	6-10	231.0	NaN	5000000.0
5	Amir Johnson	Boston Celtics	90.0	PF	29.0	6-9	240.0	NaN	12000000.0
6	Jordan Mickey	Boston Celtics	55.0	PF	21.0	6-8	235.0	LSU	1170960.0
7	Kelly Olynyk	Boston Celtics	41.0	C	25.0	7-0	238.0	Gonzaga	2165160.0
8	Terry Rozier	Boston Celtics	12.0	PG	22.0	6-2	190.0	Louisville	1824360.0
9	Marcus Smart	Boston Celtics	36.0	PG	22.0	6-4	220.0	Oklahoma State	3431040.0
10	Jared Sullinger	Boston Celtics	7.0	C	24.0	6-9	260.0	Ohio State	2569260.0
11	Isaiah Thomas	Boston Celtics	4.0	PG	27.0	5-9	185.0	Washington	6912869.0
12	Evan Turner	Boston Celtics	11.0	SG	27.0	6-7	220.0	Ohio State	3425510.0
13	James Young	Boston Celtics	13.0	SG	20.0	6-6	215.0	Kentucky	1749840.0

And you want to make a program to automatically predict value of 1 column based on others.

Input

Name,
Team,
Number,
Position,
Age,
Height,
Weight,
College



Output

Salary

X(Name, Team, Number, Position, Age, Height, Weight, College) -> Y(Salary)

Introduction

❖ Getting Started

Input X　　　　Output Y

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99.0	SF	25.0	6-6	235.0	Marquette	6796117.0
2	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	Boston University	NaN
3	R.J. Hunter	Boston Celtics	28.0	SG	22.0	6-5	185.0	Georgia State	1148640.0
4	Jonas Jerebko	Boston Celtics	8.0	PF	29.0	6-10	231.0	NaN	5000000.0
5	Amir Johnson	Boston Celtics	90.0	PF	29.0	6-9	240.0	NaN	12000000.0
6	Jordan Mickey	Boston Celtics	55.0	PF	21.0	6-8	235.0	LSU	1170960.0
7	Kelly Olynyk	Boston Celtics	41.0	C	25.0	7-0	238.0	Gonzaga	2165160.0
8	Terry Rozier	Boston Celtics	12.0	PG	22.0	6-2	190.0	Louisville	1824360.0
9	Marcus Smart	Boston Celtics	36.0	PG	22.0	6-4	220.0	Oklahoma State	3431040.0
10	Jared Sullinger	Boston Celtics	7.0	C	24.0	6-9	260.0	Ohio State	2569260.0
11	Isaiah Thomas	Boston Celtics	4.0	PG	27.0	5-9	185.0	Washington	6912869.0
12	Evan Turner	Boston Celtics	11.0	SG	27.0	6-7	220.0	Ohio State	3425510.0
13	James Young	Boston Celtics	13.0	SG	20.0	6-6	215.0	Kentucky	1749840.0

Using this data to "train" an ML Model

ML Model

Input　　　　**Output**

Use the trained model to predict Salary based on any given input

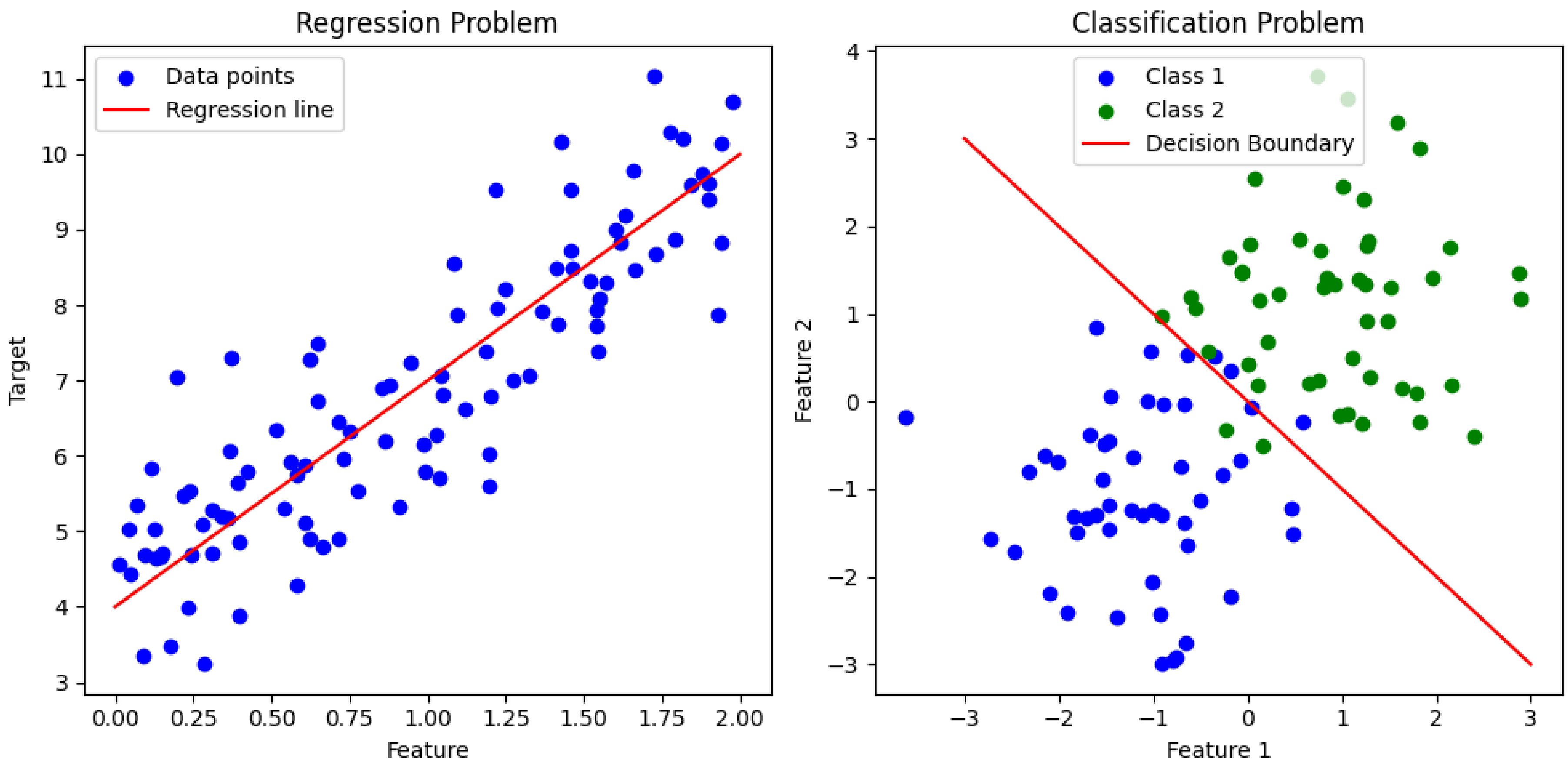
Name, Team, Number, Position, Age, Height, Weight, College → **Trained ML Model** → Salary

Since we **use a labeled dataset to train** ML Model.

=> This is called **Supervised-learning**.

Introduction

❖ Supervised Learning

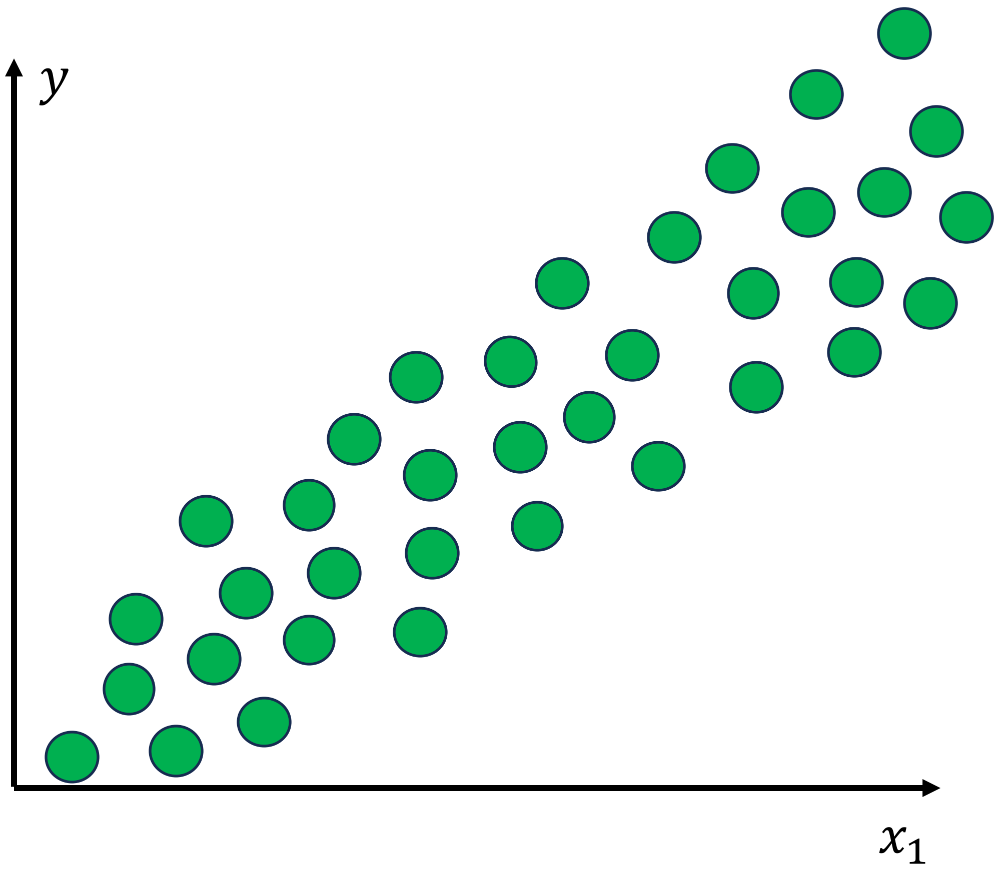


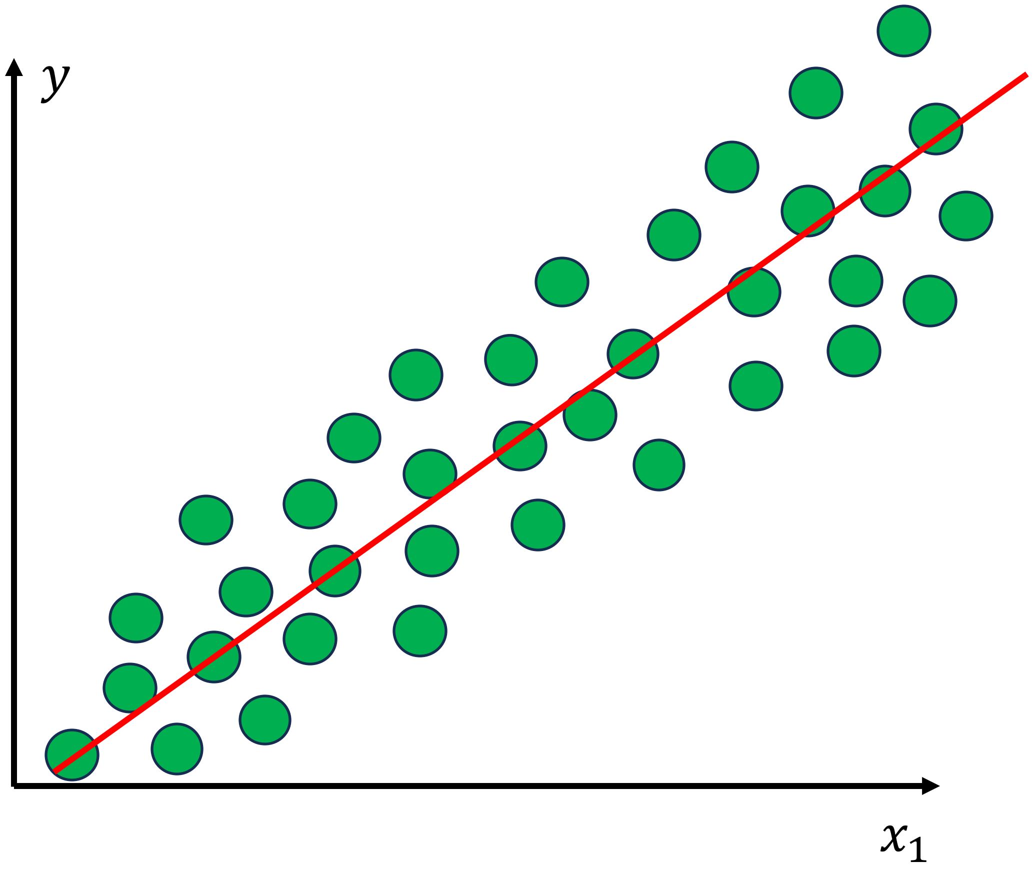
In ML Superivsed-learning algorithms, we often deal with Regression and Classification

Introduction

❖ Supervised Learning: Regression

Regression: A task involving predicting a **continuous value** based on given inputs.



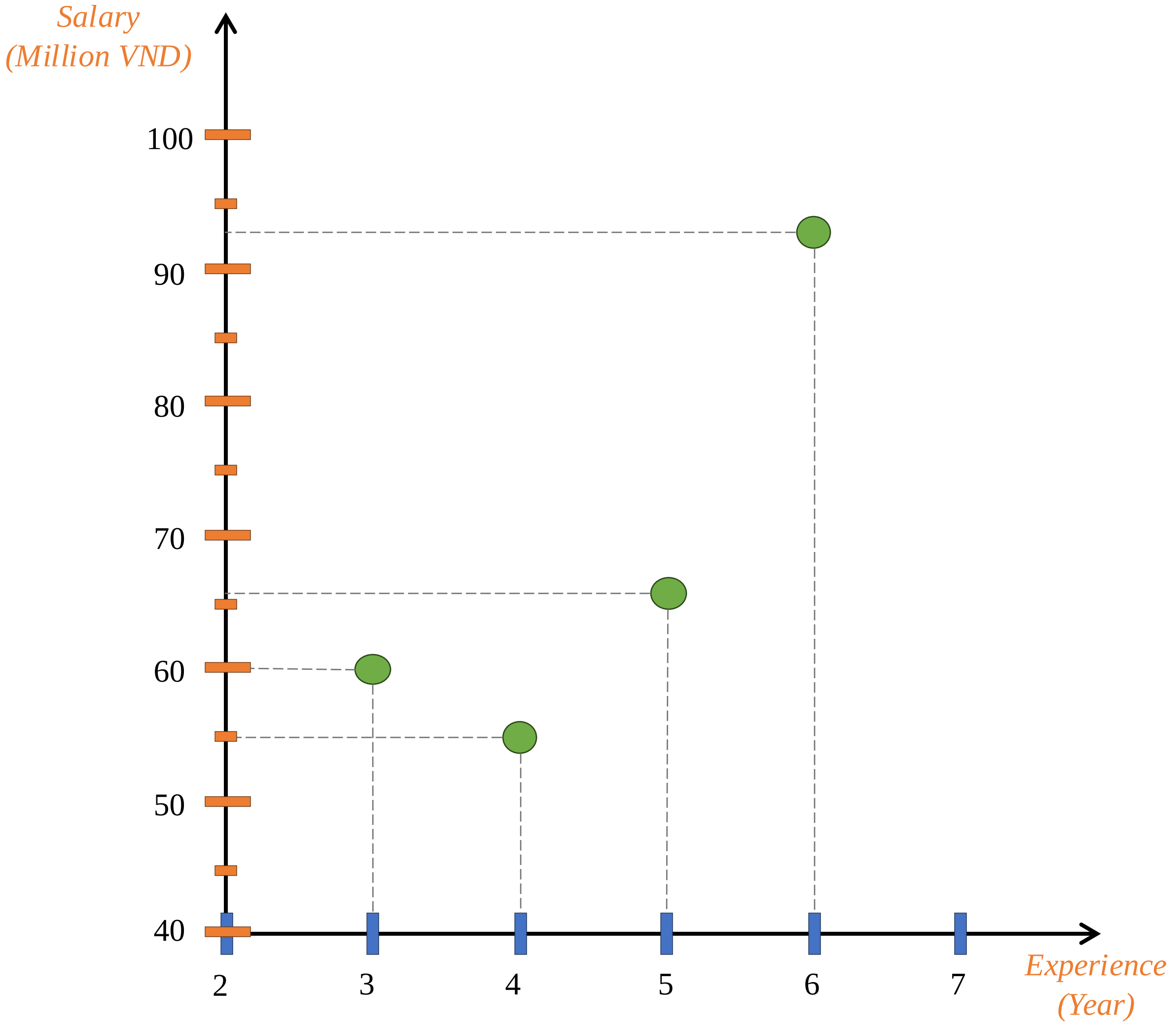


In general, we want to find the line that best fit the data distribution.

Simple Linear Regression

Simple Linear Regression

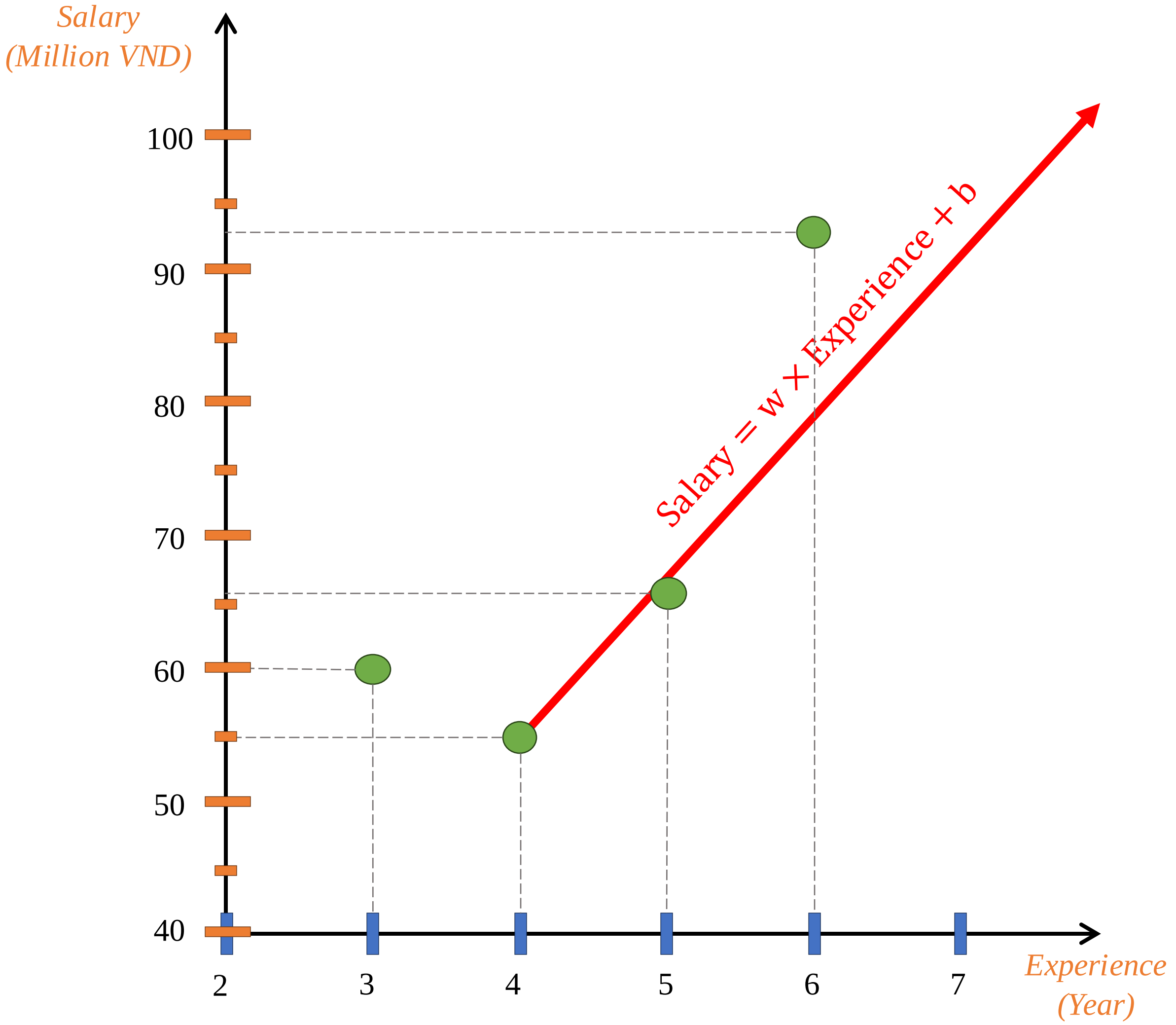
❖ Intro to Linear Regression



Experience	Salary
3	60
4	55
5	66
6	93
7	?

Simple Linear Regression

❖ Intro to Linear Regression

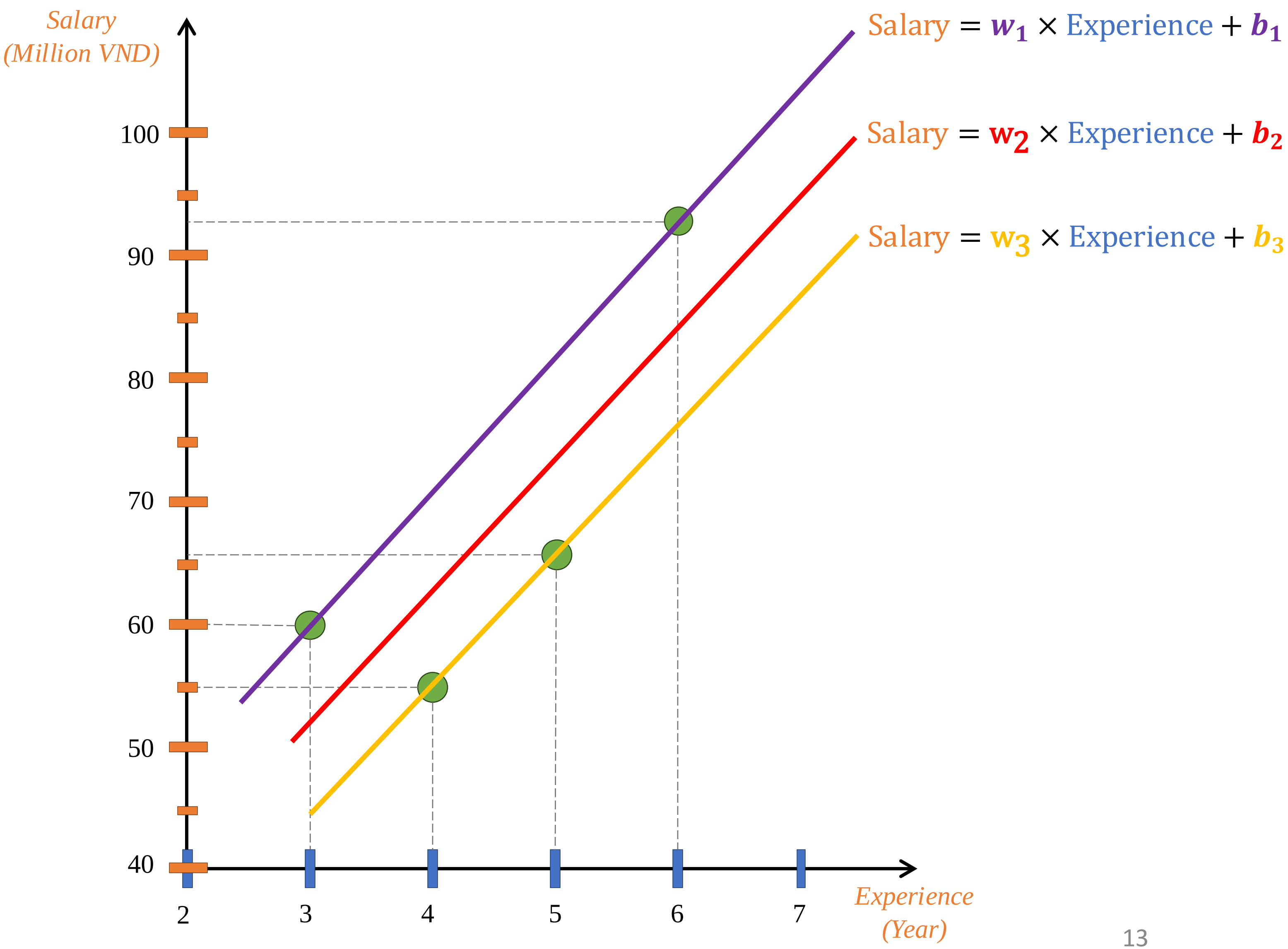


Experience	Salary
3	60
4	55
5	66
6	93
7	?

Simple Linear Regression

❖ Intro to Linear Regression

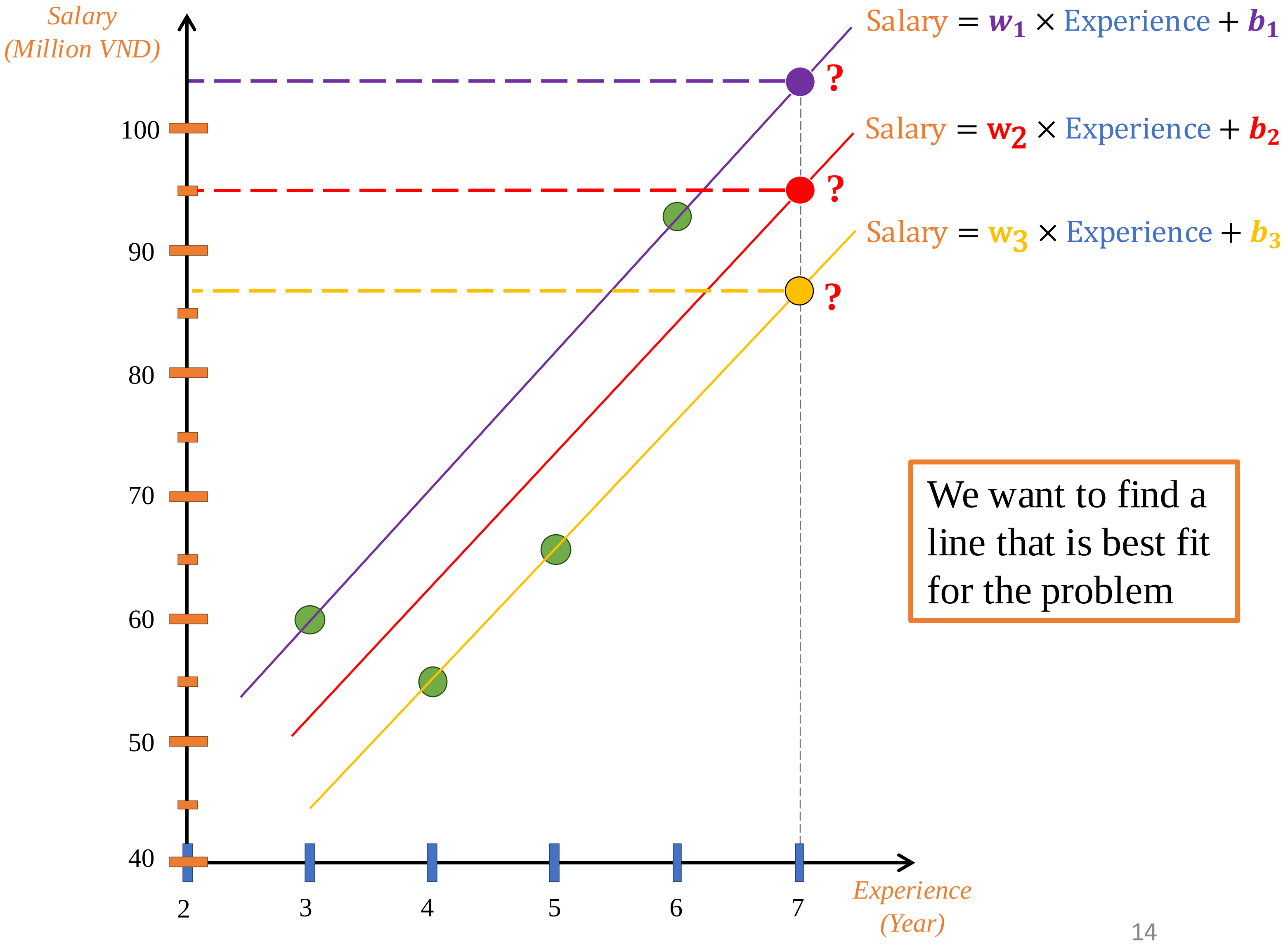
Experience	Salary
3	60
4	55
5	66
6	93
7	?



Simple Linear Regression

❖ Intro to Linear Regression

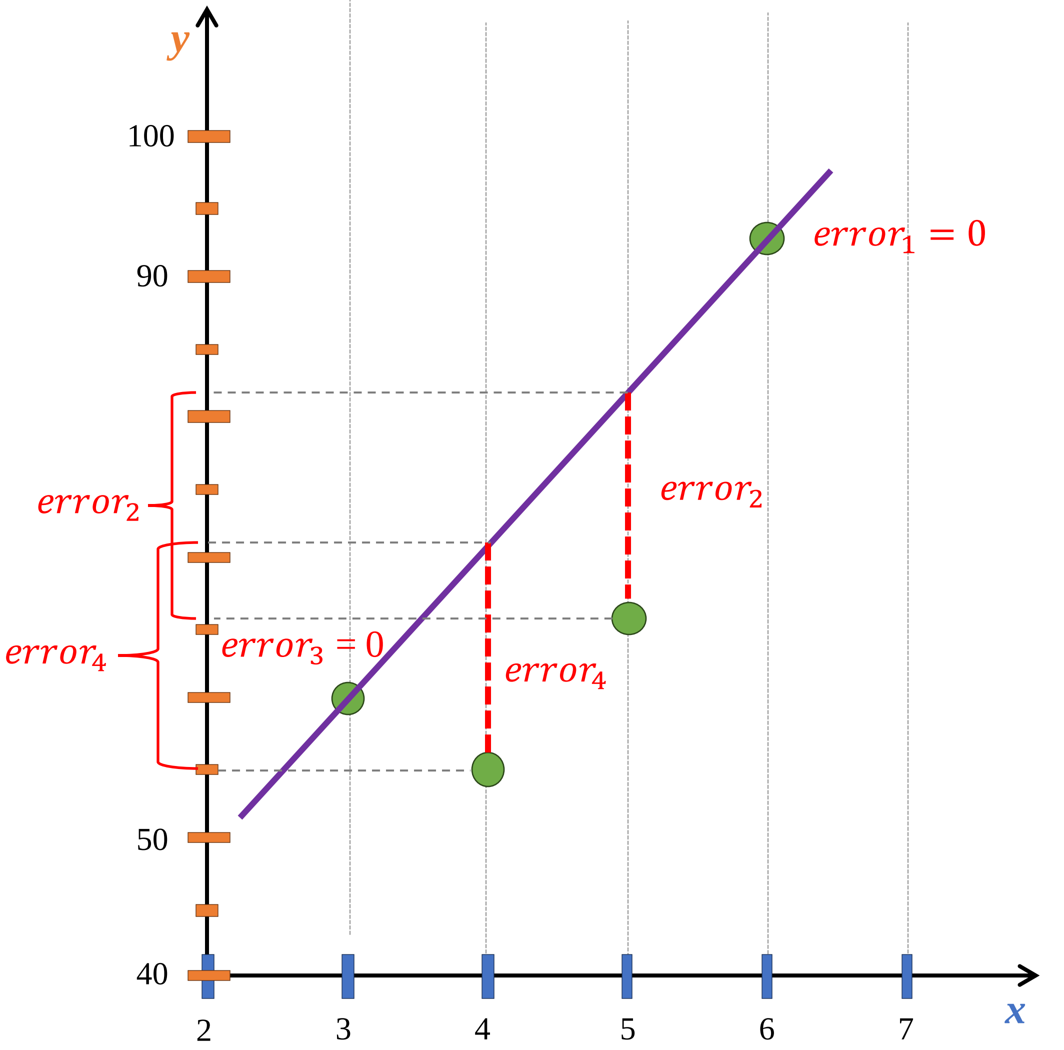
Experience	Salary
3	60
4	55
5	66
6	93
7	?



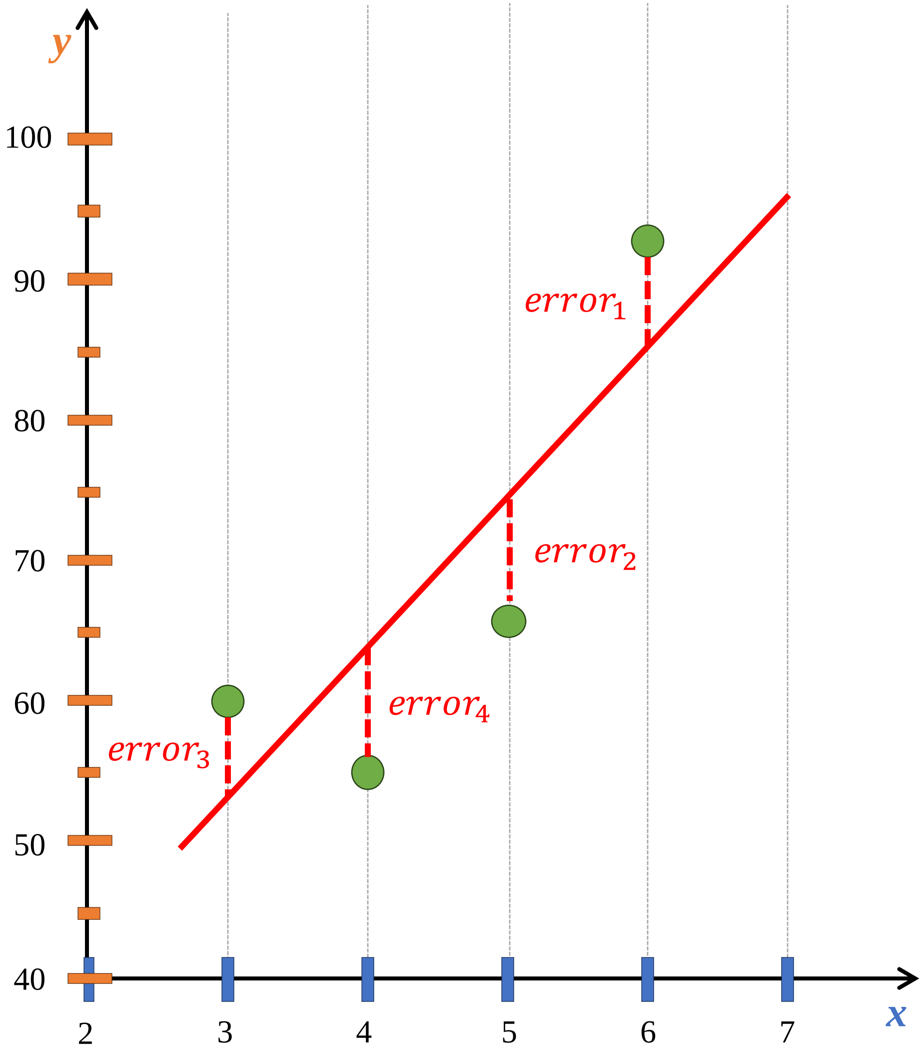
We want to find a line that is best fit for the problem

Simple Linear Regression

❖ Intro to Linear Regression

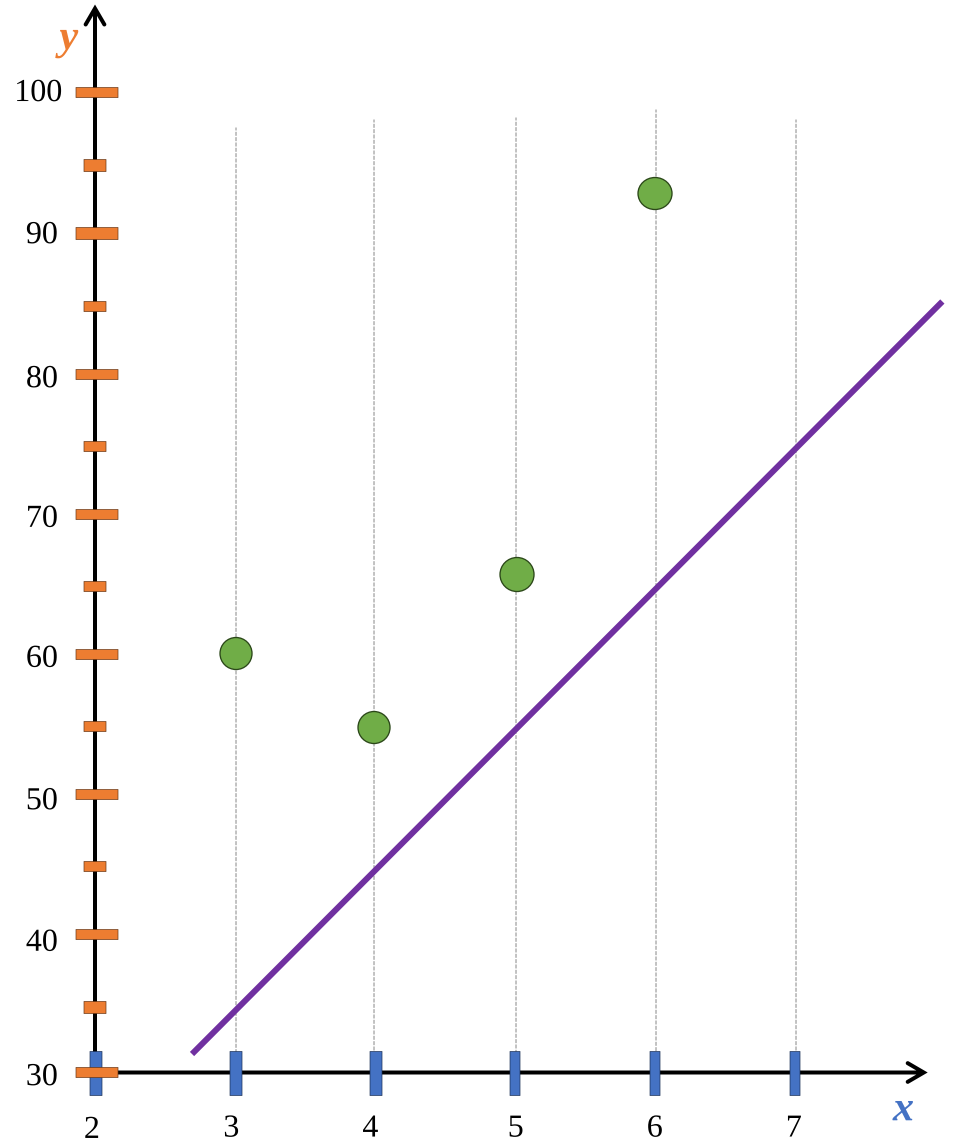


$\hat{y} = w_1 \times x + b_1$



$\hat{y} = w_2 \times x + b_2$

Simple Linear Regression



$\hat{y} = w \times x + b$

$$\text{predicted\_salary} = w \times \text{Experience} + b$$

$$error = (\text{predicted\_salary} - \text{real\_salary})^2$$

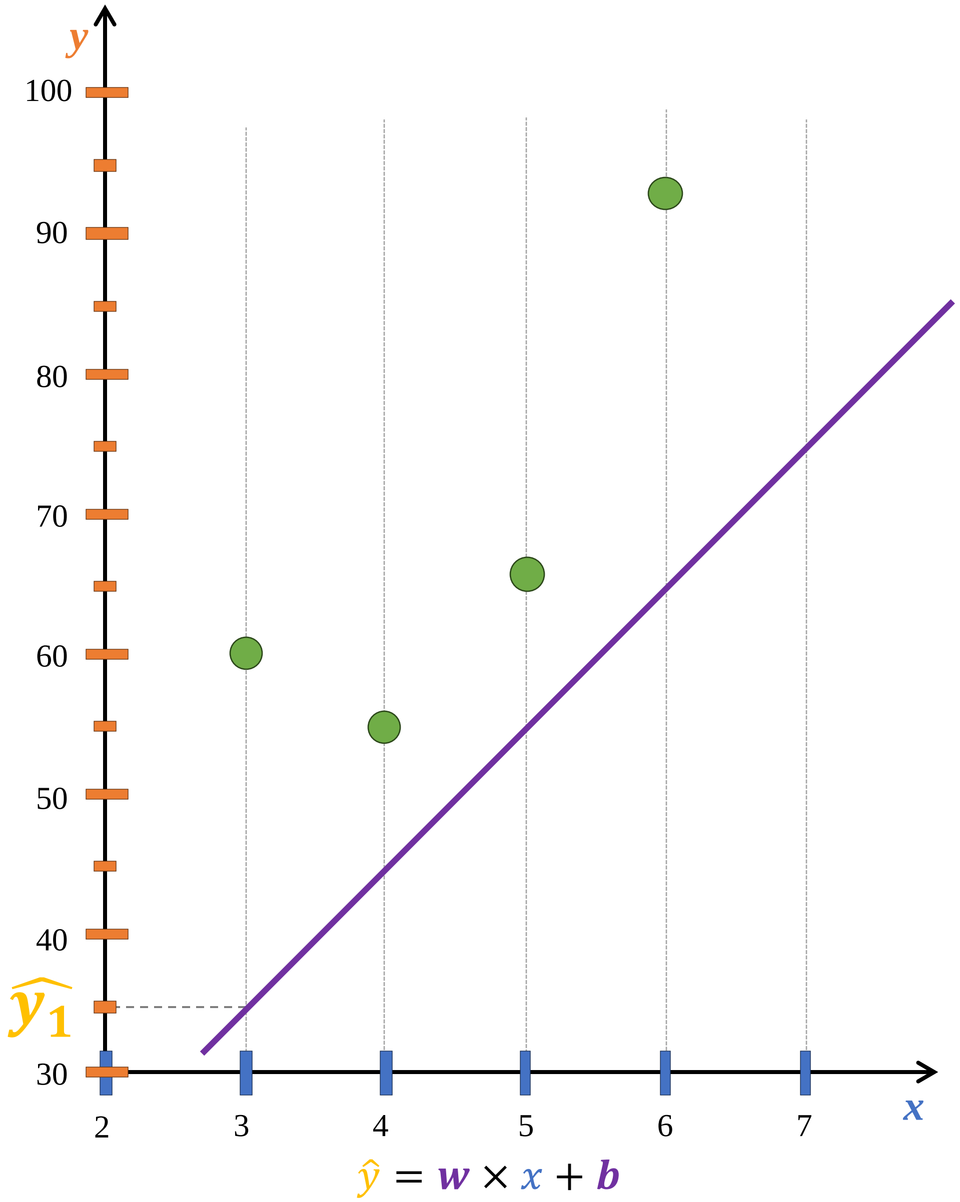
$$\hat{y}_i = wx_i + b$$

$$error(\hat{y}_i, y_i) = (\hat{y}_i - y_i)^2$$

$$w = 10$$
$$b = 5$$

	Experience	Salary	Predicted	Error
1	3	60	$wx_i + b$	$(\hat{y}_i - y_i)^2$
2	4	55	$wx_i + b$	$(\hat{y}_i - y_i)^2$
3	5	66	$wx_i + b$	$(\hat{y}_i - y_i)^2$
4	6	93	$wx_i + b$	$(\hat{y}_i - y_i)^2$

Simple Linear Regression



$$\text{predicted\_salary} = w \times \text{Experience} + b$$

$$error = (\text{predicted\_salary} - \text{real\_salary})^2$$

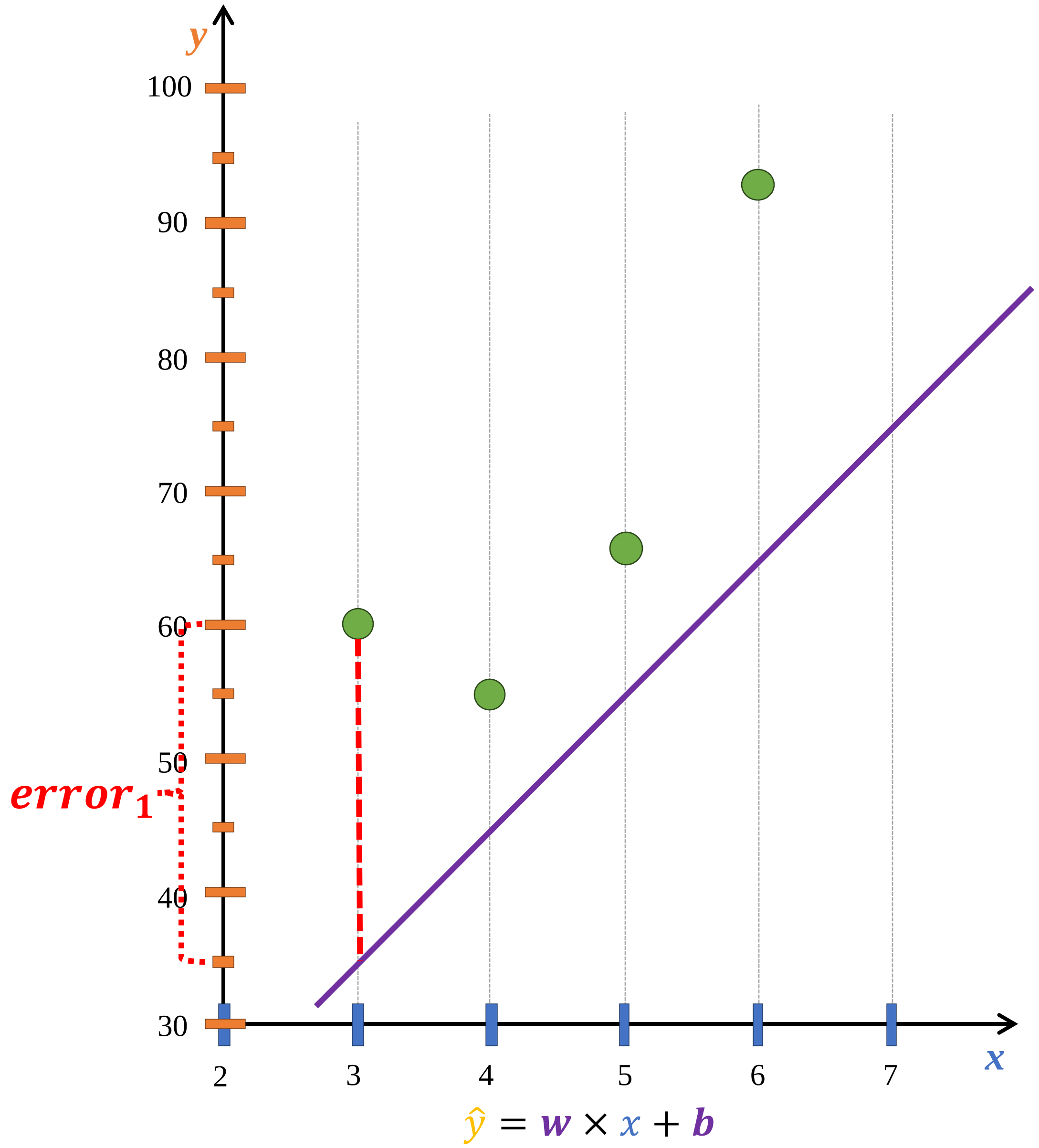
$$\hat{y}_i = wx_i + b$$

$$error(\hat{y}_i, y_i) = (\hat{y}_i - y_i)^2$$

$$w = 10$$
$$b = 5$$

	Experience	Salary	Predicted	Error
1	3	60	$10 \cdot 3 + 5 = 35$	
2	4	55		
3	5	66		
4	6	93		

Simple Linear Regression



$$\text{predicted\_salary} = w \times \text{Experience} + b$$

$$error = (\text{predicted\_salary} - \text{real\_salary})^2$$

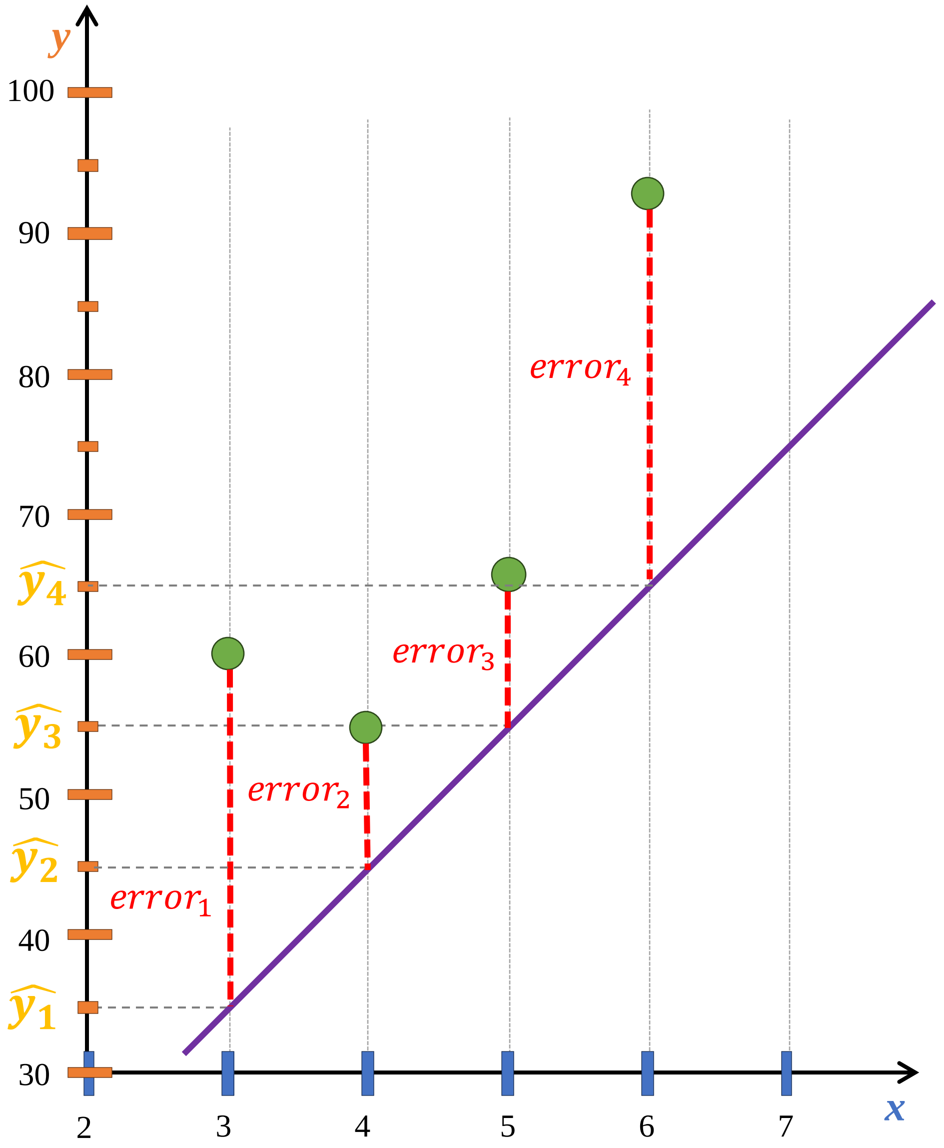
$$\hat{y}_i = wx_i + b$$

$$error(\hat{y}_i, y_i) = (\hat{y}_i - y_i)^2$$

$$w = 10$$
$$b = 5$$

	Experience	Salary	Predicted	Error
1	3	60	35	$(60 - 35)^2 = 625$
2	4	55		
3	5	66		
4	6	93		

Simple Linear Regression



$\hat{y} = w \times x + b$

$$\text{predicted\_salary} = w \times \text{Experience} + b$$

$$error = (\text{predicted\_salary} - \text{real\_salary})^2$$

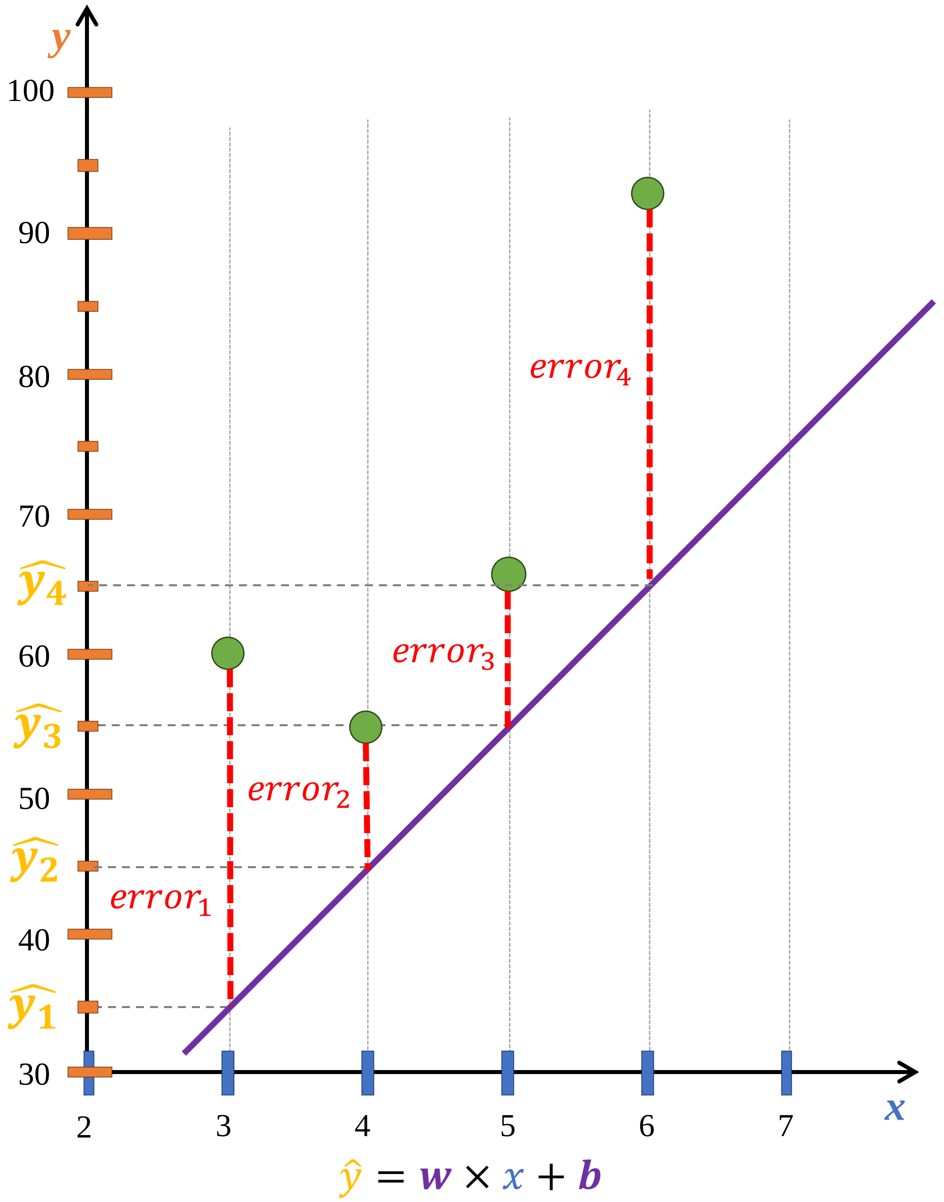
$$\widehat{y_i} = wx_i + b$$

$$error(\widehat{y_i}, y_i) = (\widehat{y_i} - y_i)^2$$

$$w = 10$$
$$b = 5$$

	Experience	Salary	Predicted	Error
1	3	60	35	625
2	4	55	$10 \cdot 4 + 5 = 45$	$(55 - 45)^2 = 100$
3	5	66	$10 \cdot 5 + 5 = 55$	$(66 - 55)^2 = 121$
4	6	93	$10 \cdot 6 + 5 = 65$	$(93 - 65)^2 = 784$

Simple Linear Regression



$$\hat{y_i} = w x_i + b$$

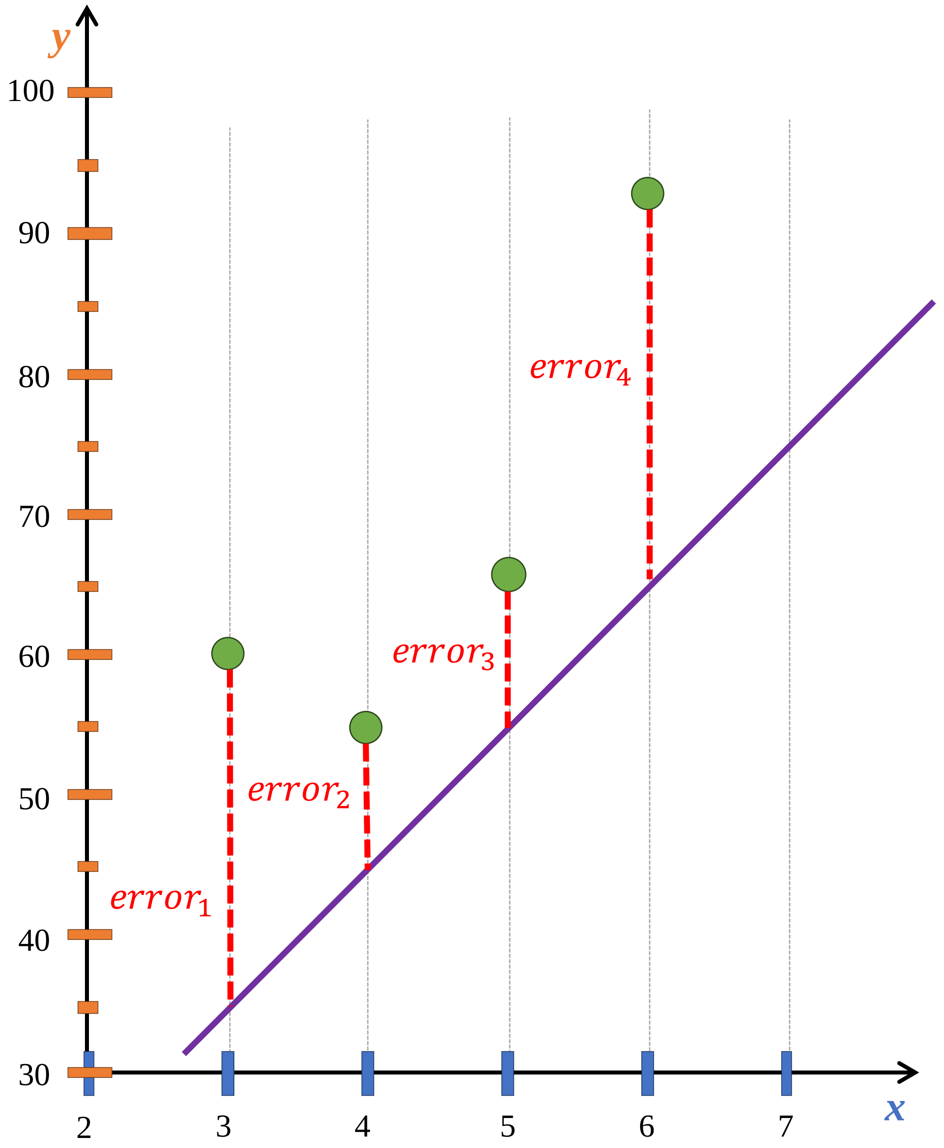
$$error(\hat{y_i}, y_i) = (\hat{y_i} - y_i)^2$$

$$w = 10$$
$$b = 5$$

$$Loss = \sum_i (\hat{y_i} - y_i)^2 = \sum_i error_i$$

	Experience	Salary	Predicted	Error	Loss
1	3	60	35	625	
2	4	55	45	100	
3	5	66	55	121	
4	6	93	65	784	

Simple Linear Regression



$\hat{y} = w \times x + b$

$$\hat{y}_i = wx_i + b$$

$$error(\hat{y}_i, y_i) = (\hat{y}_i - y_i)^2$$

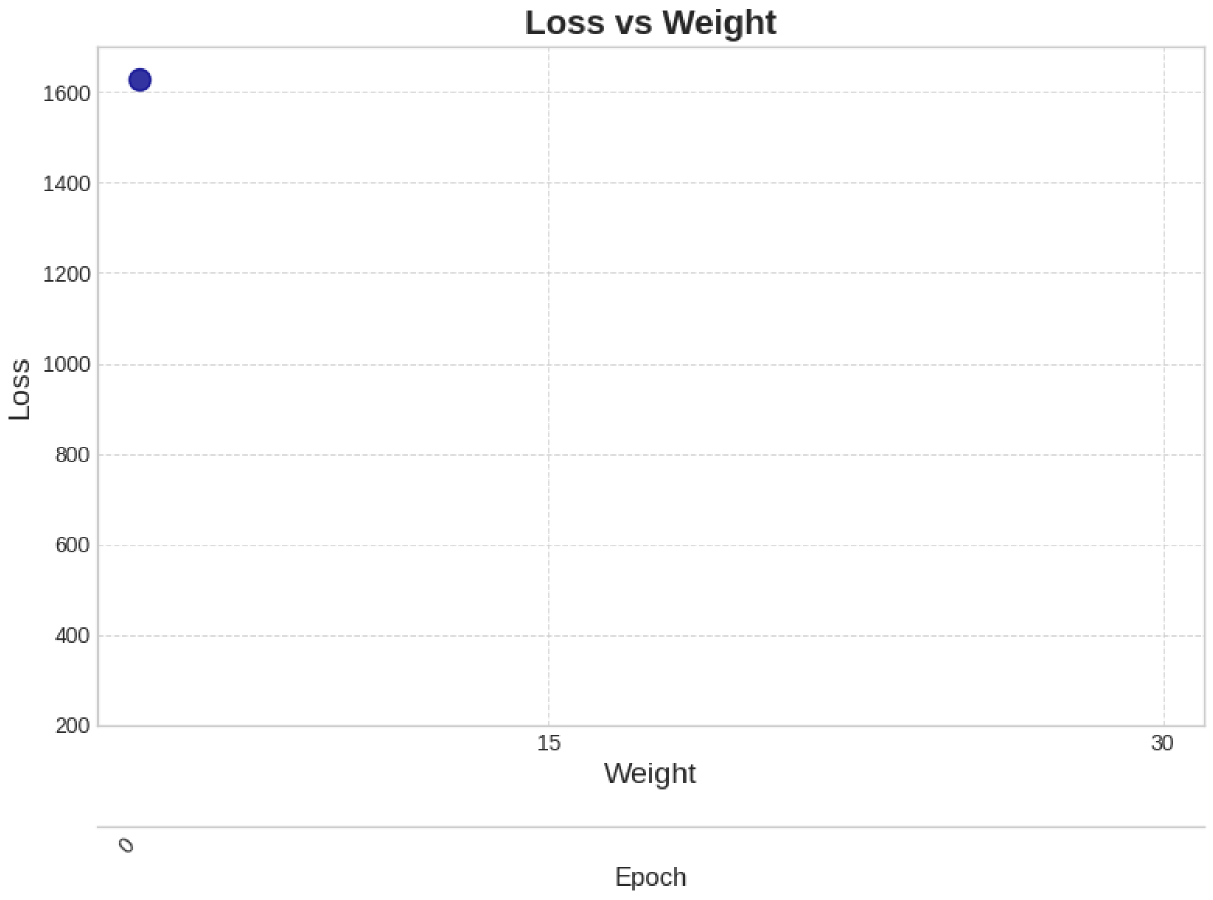
$$w = 10$$
$$b = 5$$

$$Loss = \sum_i (\hat{y}_i - y_i)^2 = \sum_i error_i$$

	Experience	Salary	Predicted	Error	Loss
1	3	60	35	625	625 +100 +121 +784 = 1630
2	4	55	45	100	
3	5	66	55	121	
4	6	93	65	784	

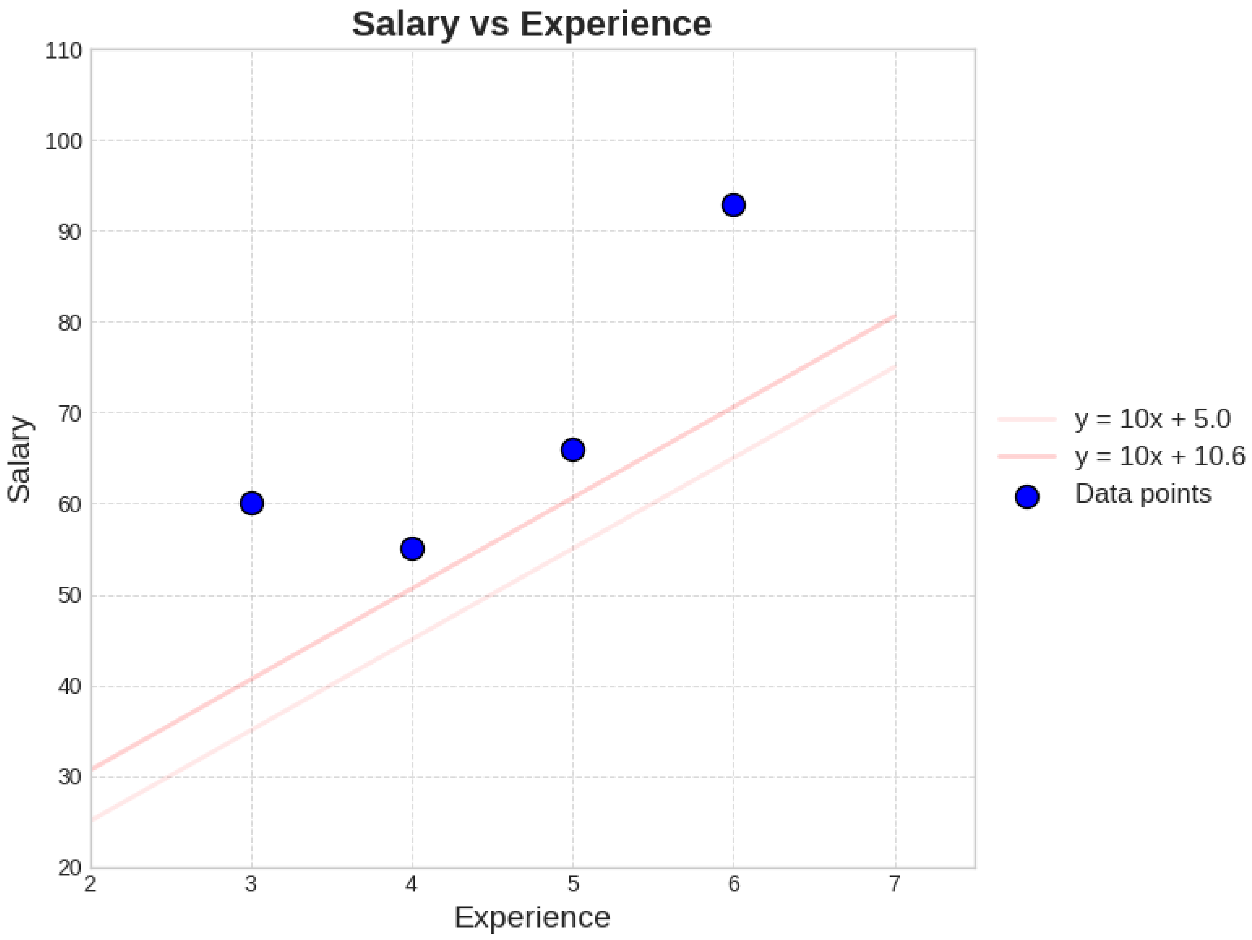
Simple Linear Regression

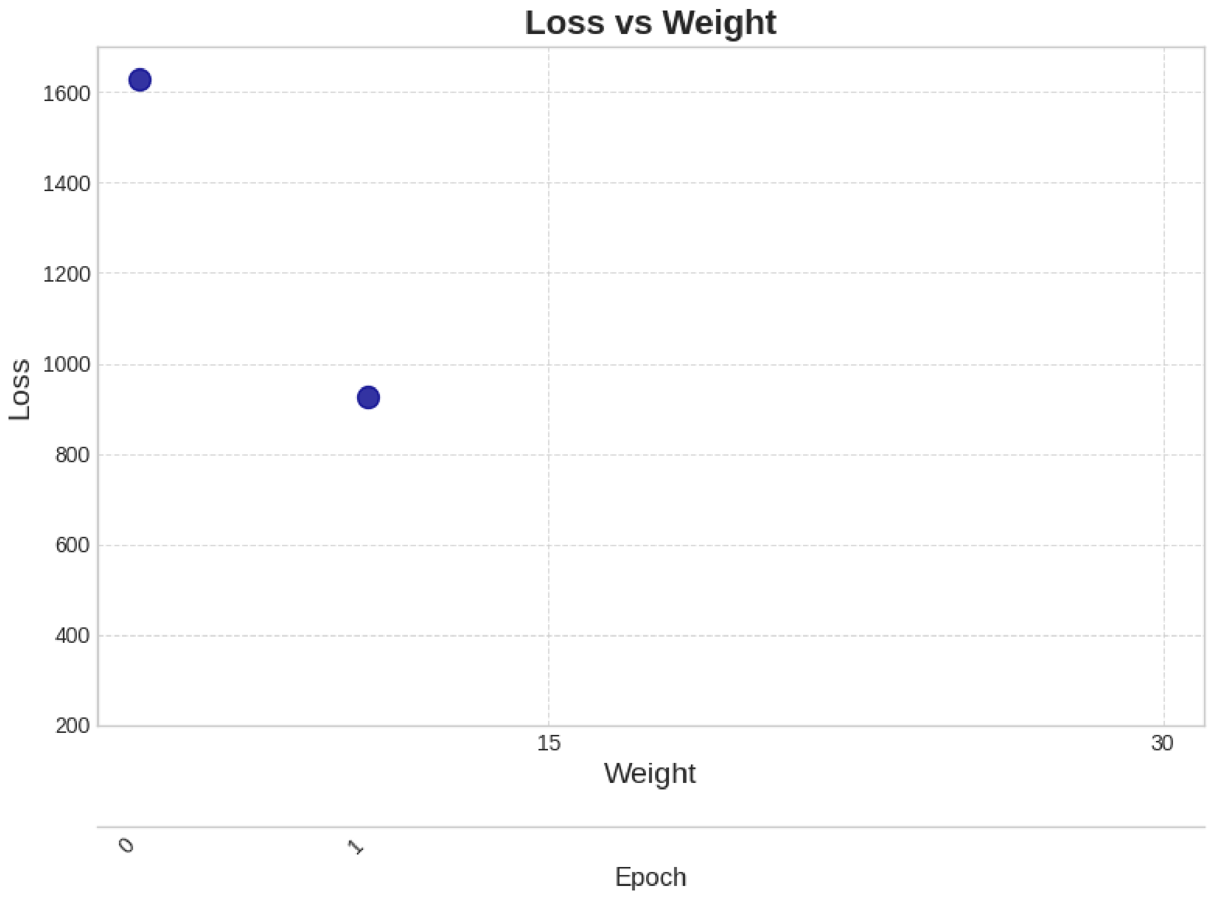
❖ How bias affect the loss function?



Simple Linear Regression

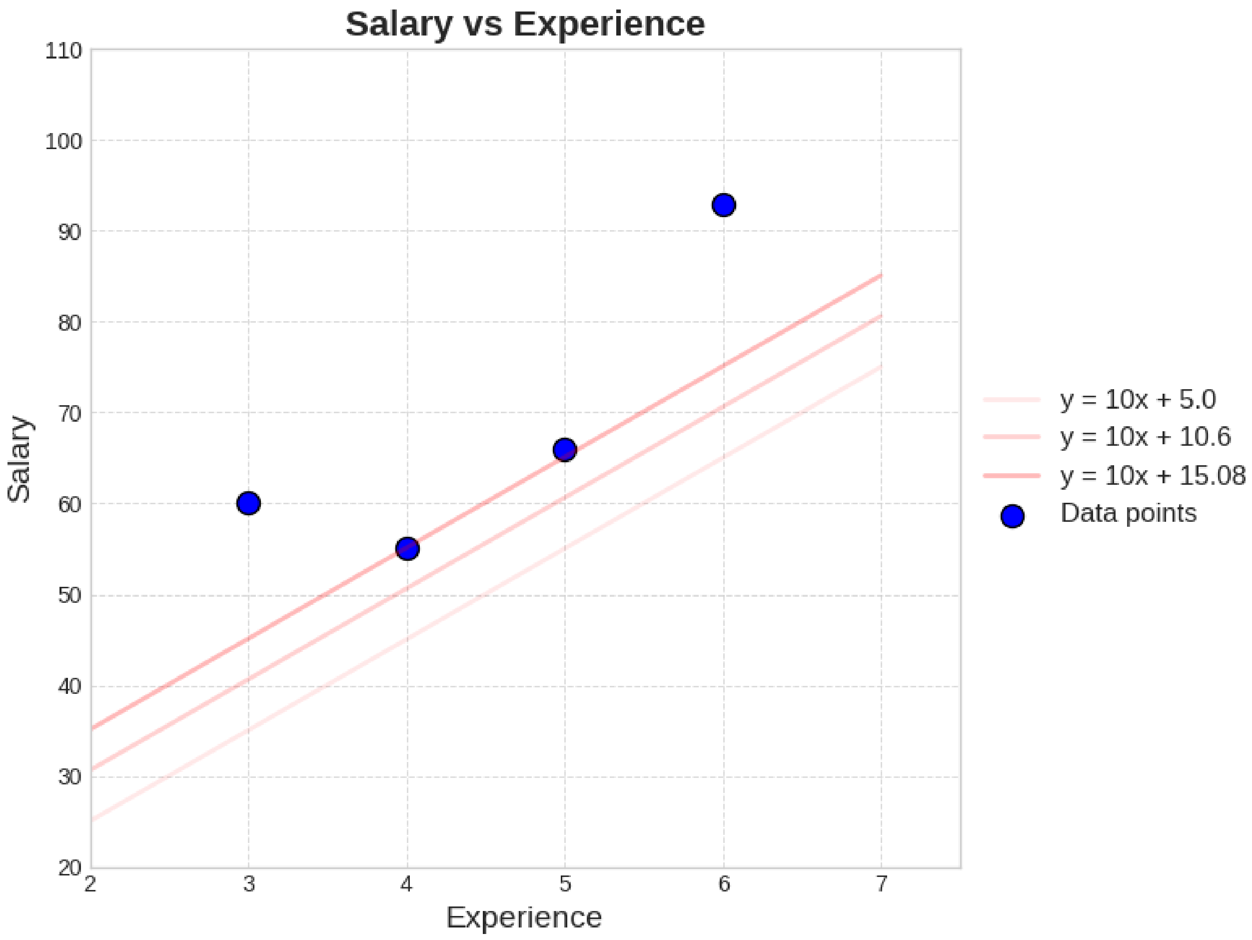
❖ How bias affect the loss function?

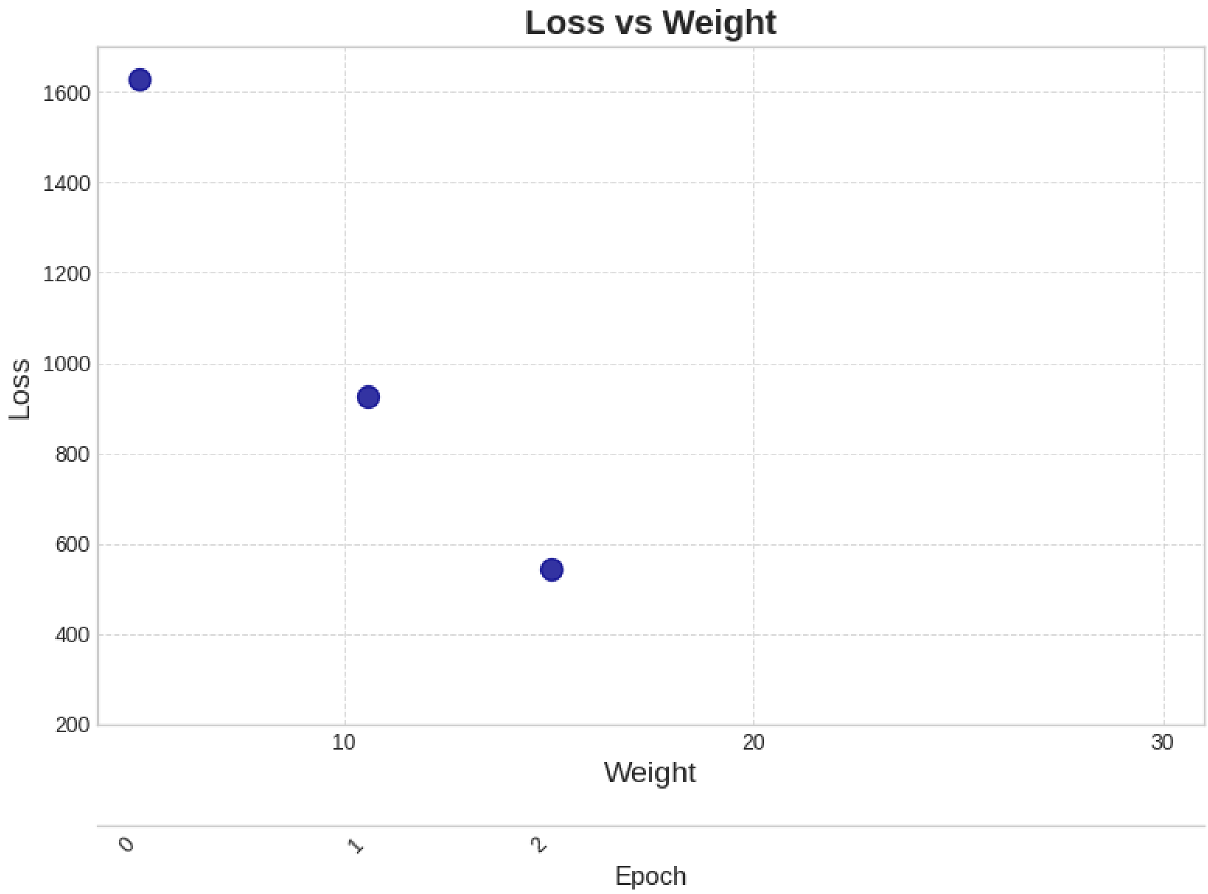




Simple Linear Regression

❖ How bias affect the loss function?

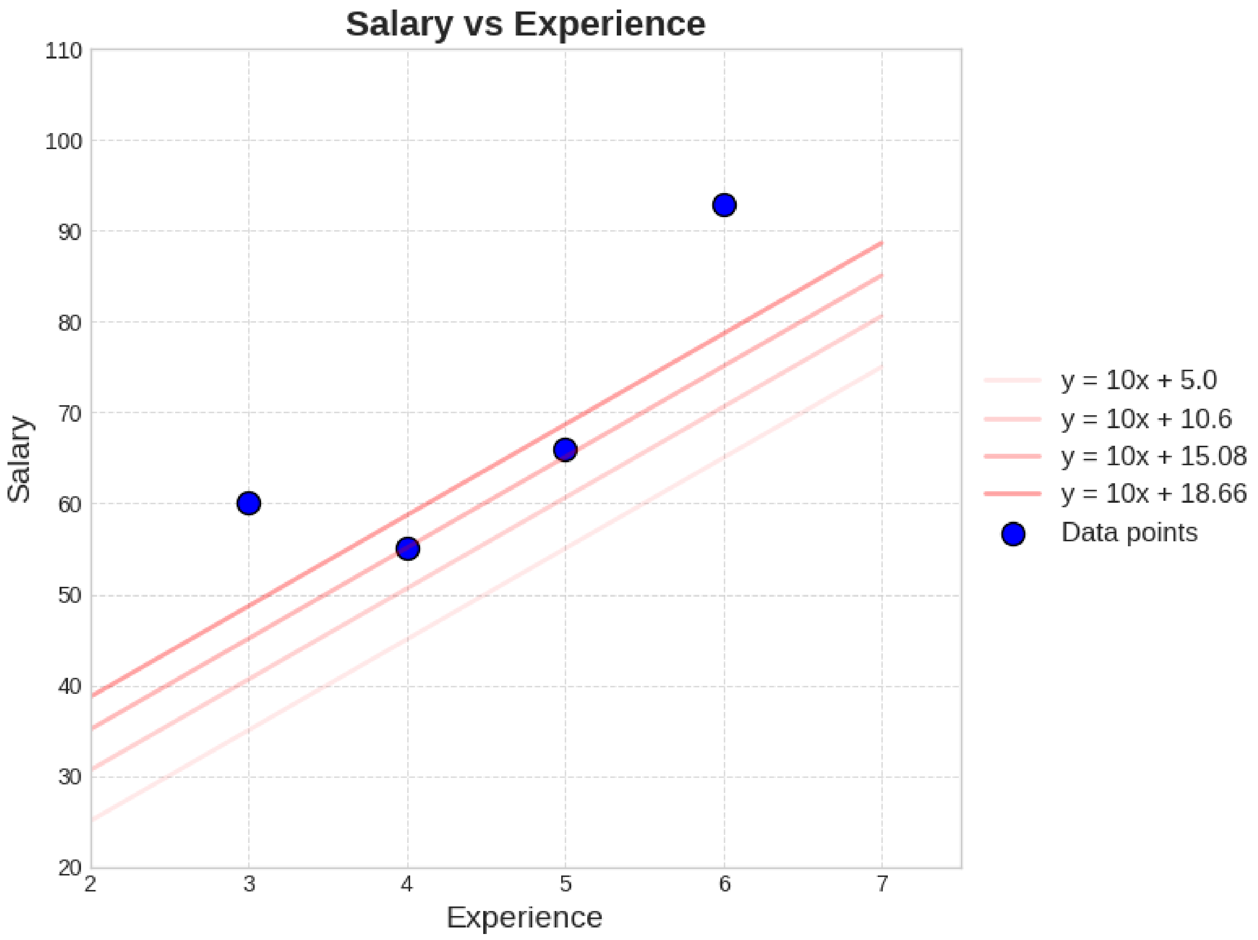


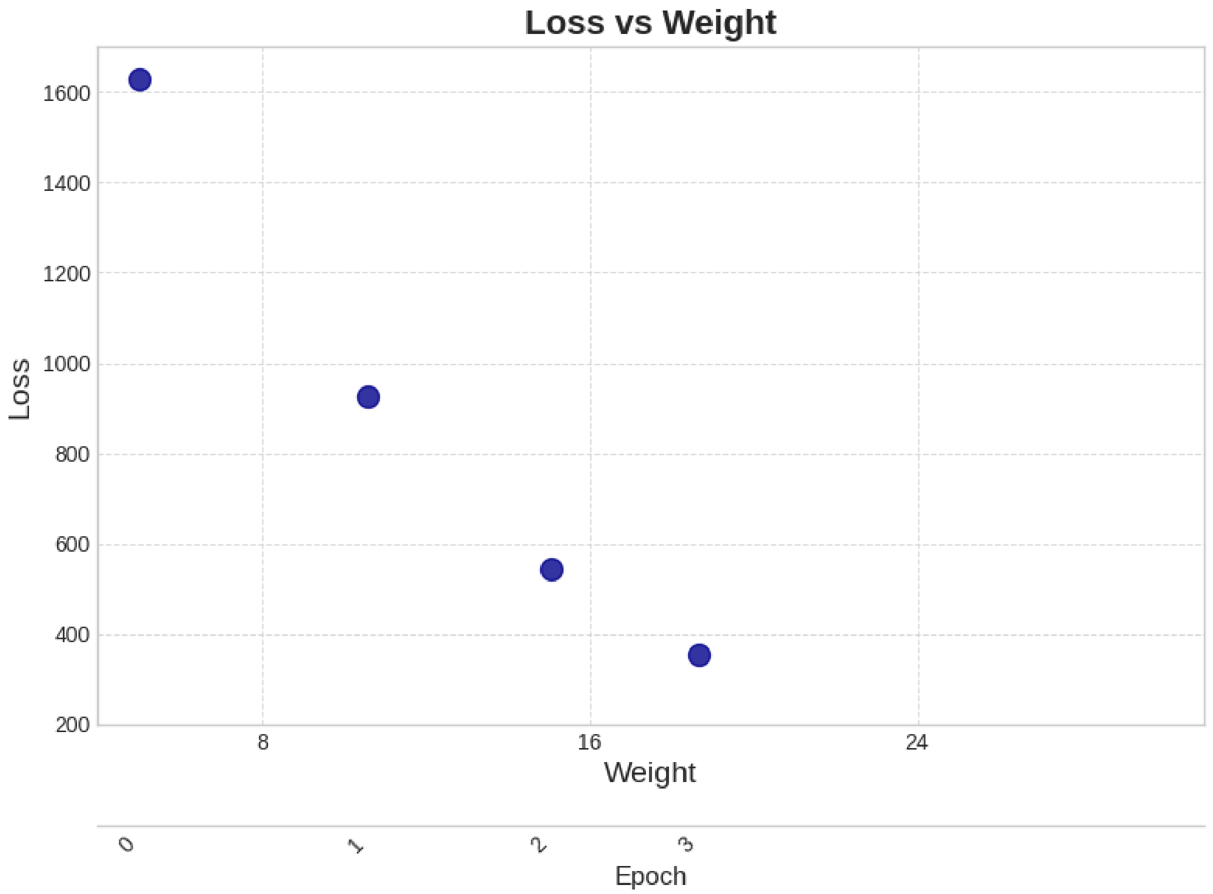


AI VIETNAM
All-in-One Course
(TA Session)

Simple Linear Regression

❖ How bias affect the loss function?

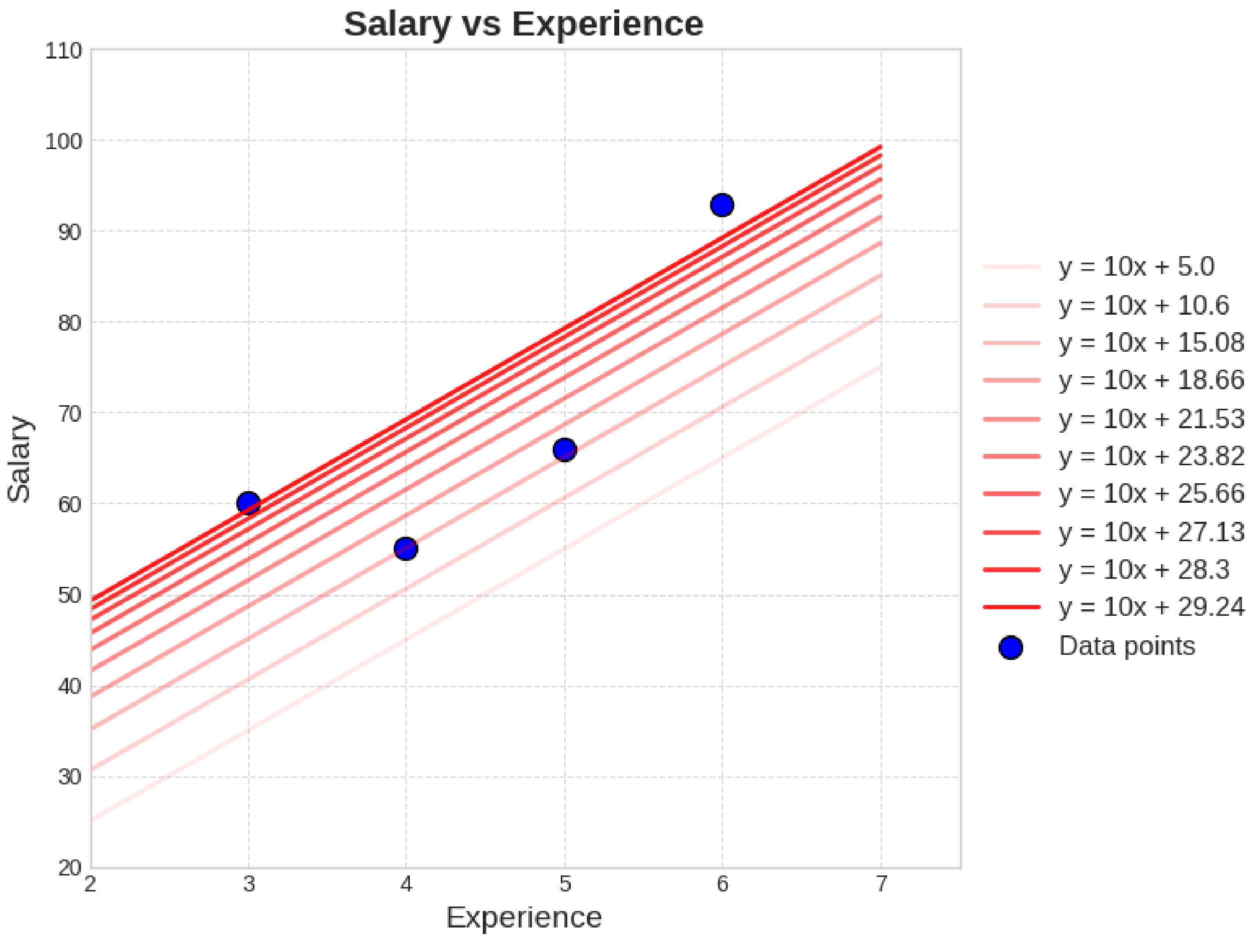


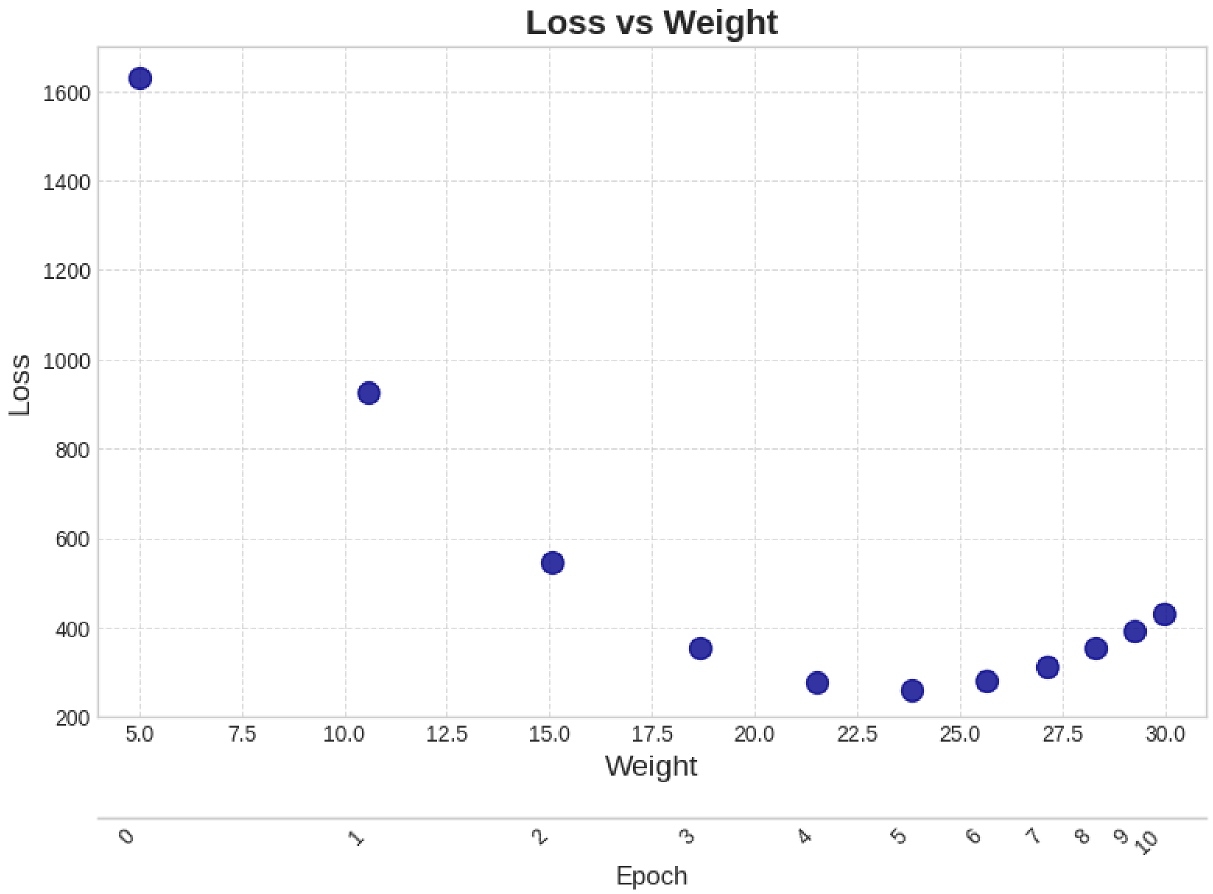


AI VIETNAM
All-in-One Course
(TA Session)

Simple Linear Regression

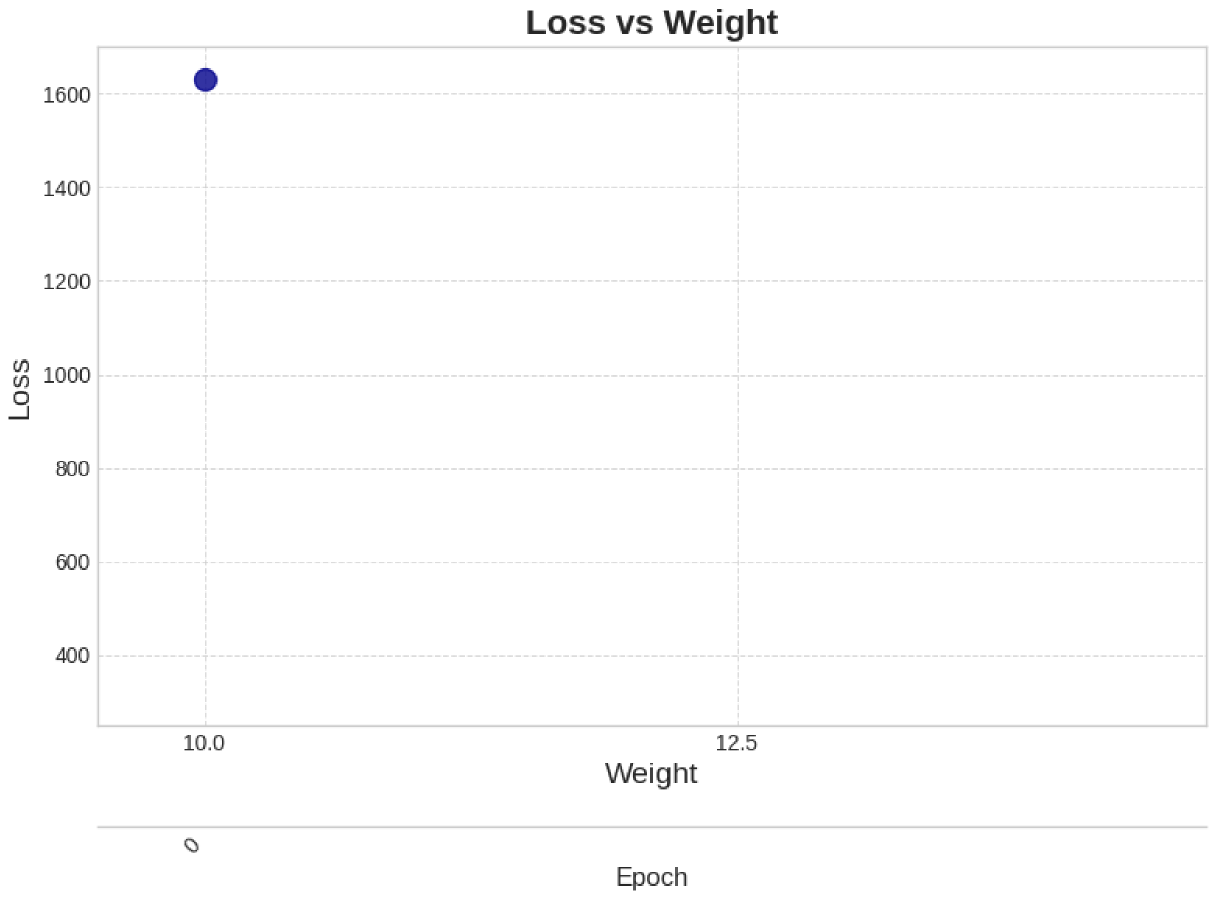
❖ How bias affect the loss function?





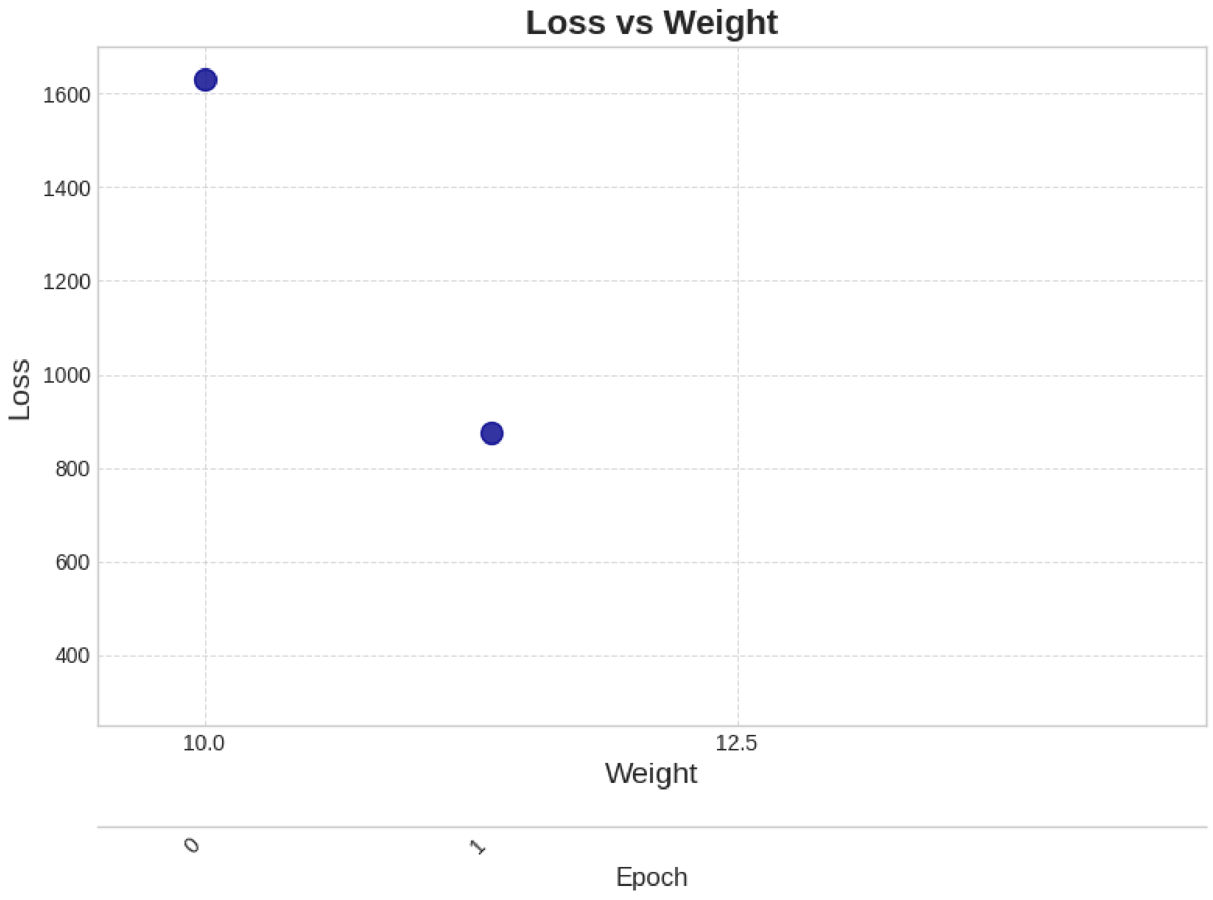
Simple Linear Regression

❖ How bias affect the loss function?



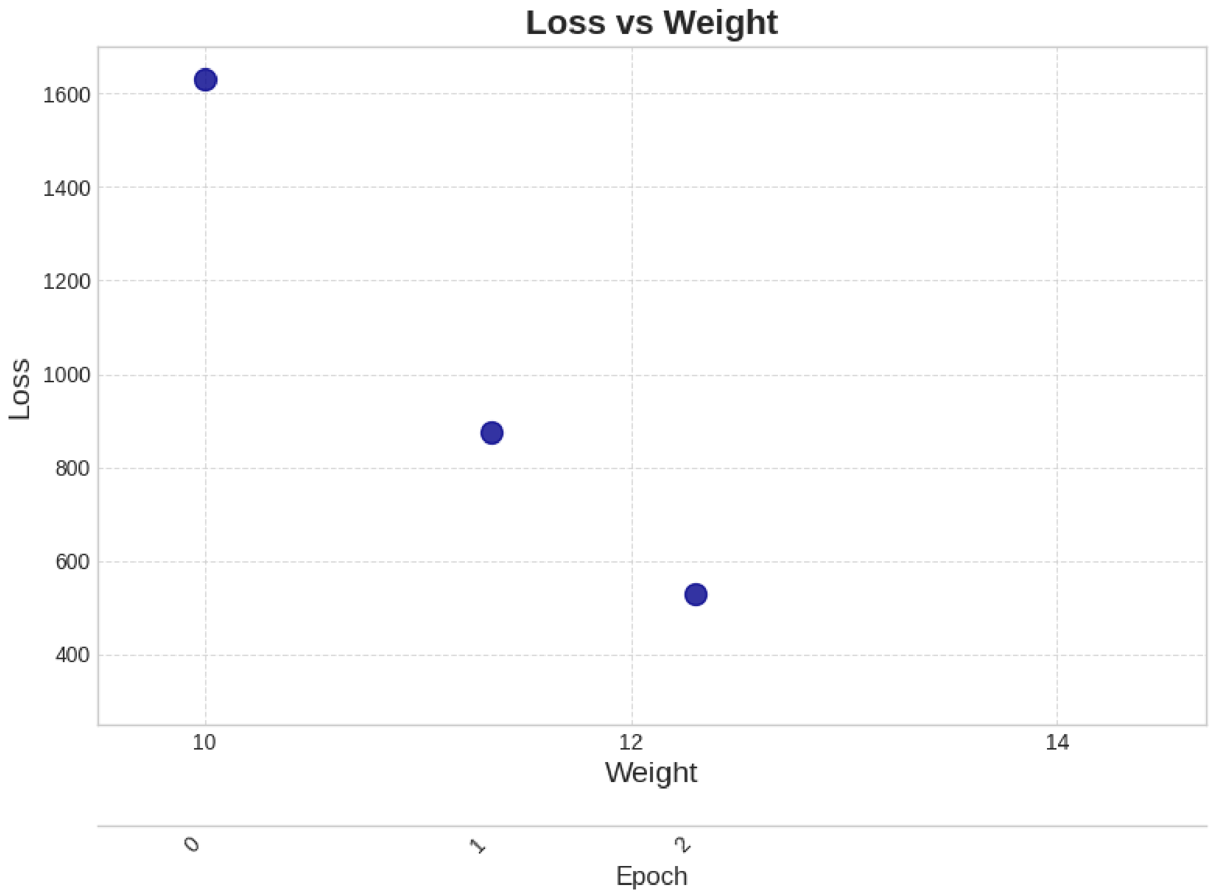
Simple Linear Regression

❖ How bias affect the loss function?



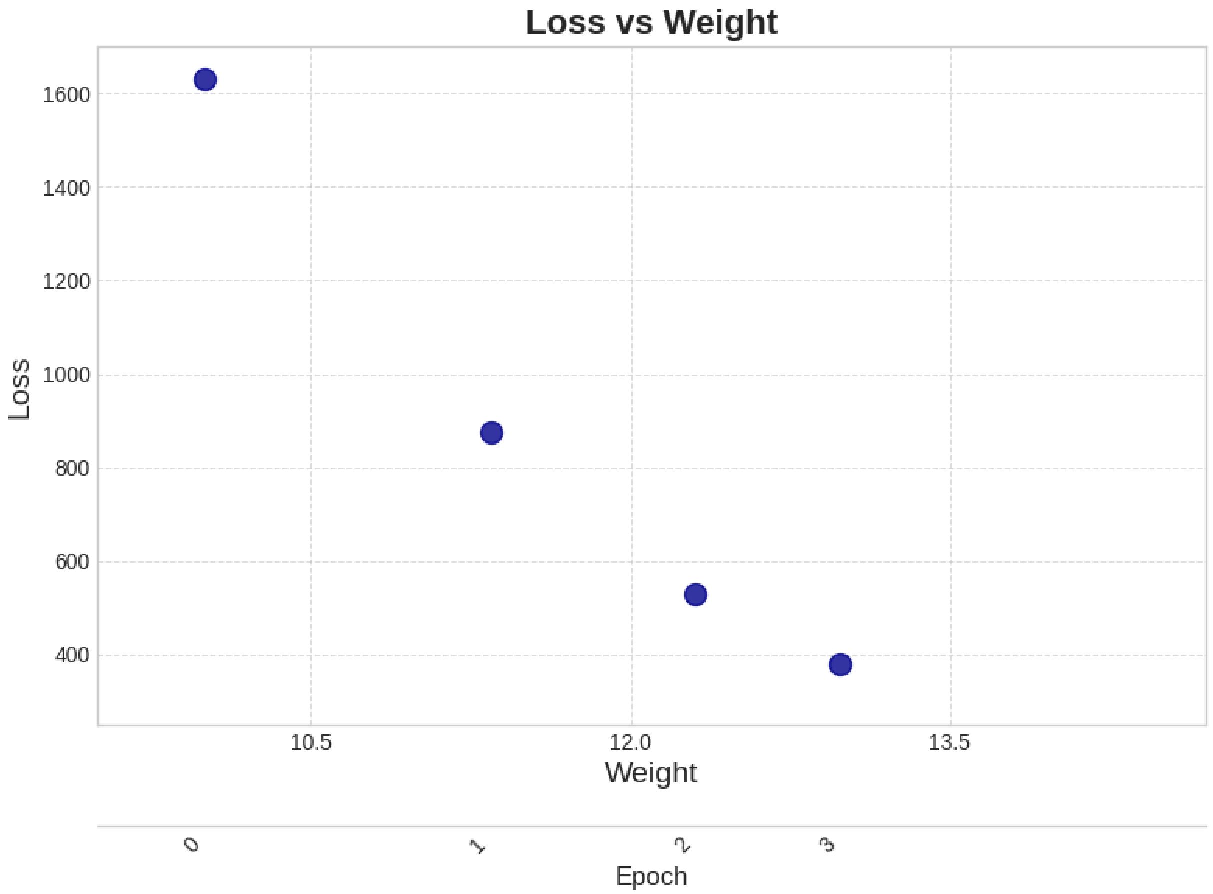
Simple Linear Regression

❖ How bias affect the loss function?



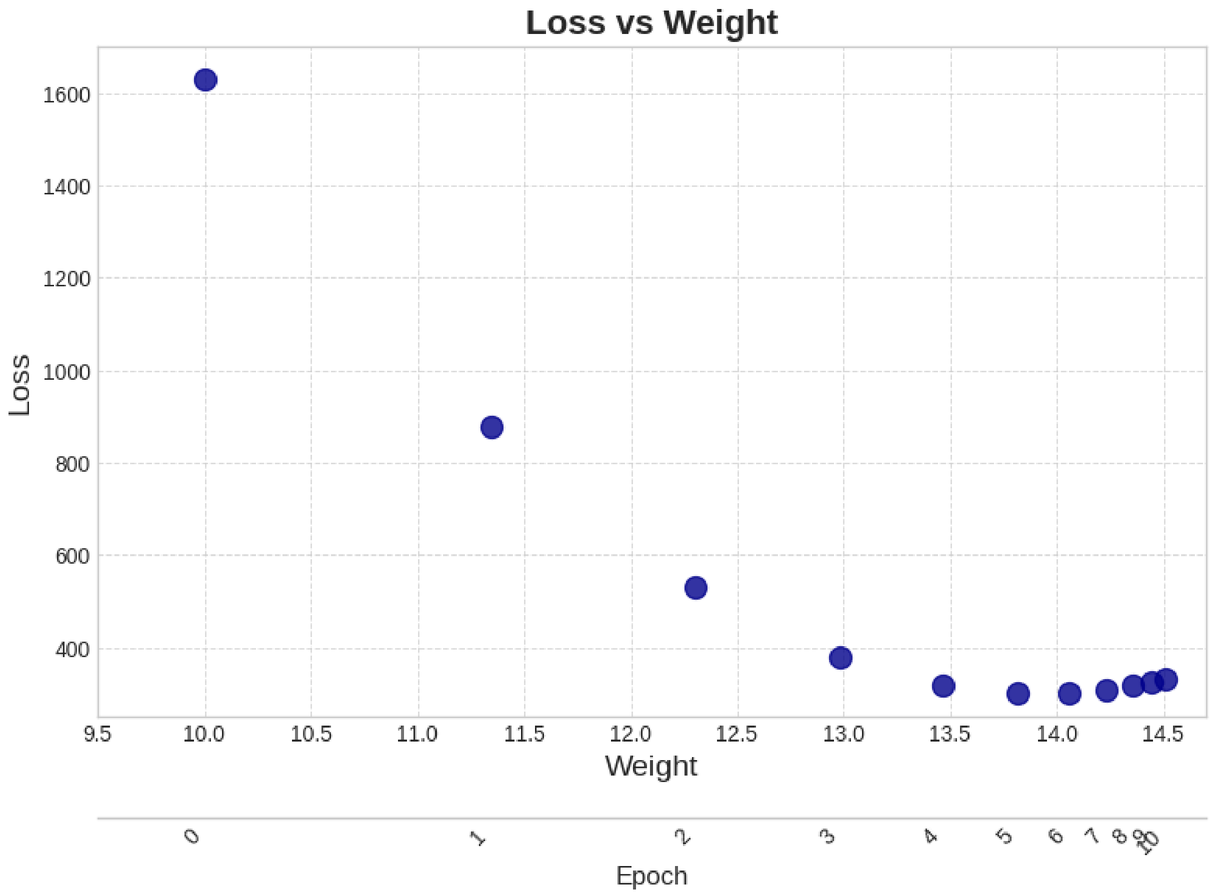
Simple Linear Regression

❖ How bias affect the loss function?



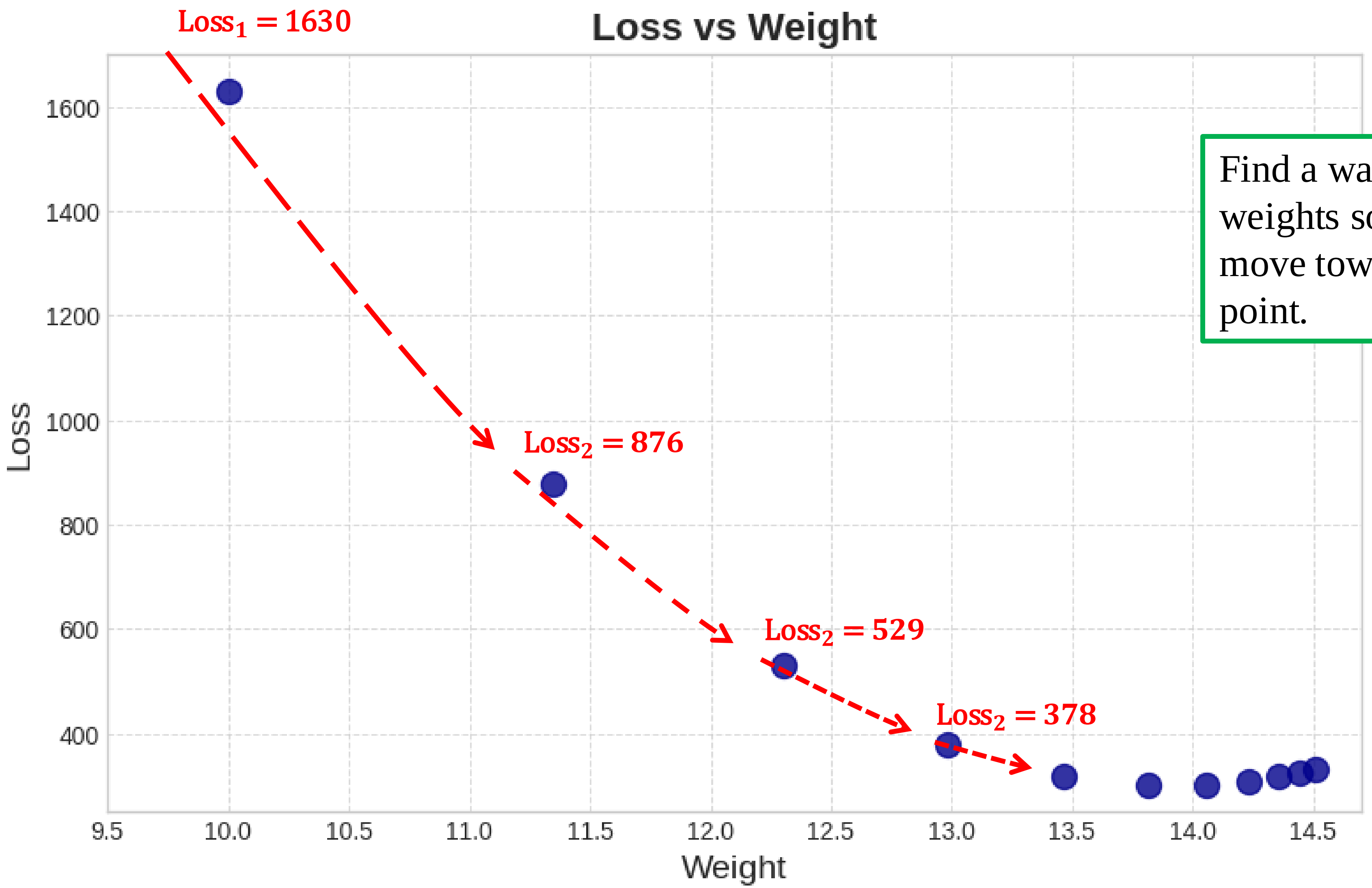
Simple Linear Regression

❖ How bias affect the loss function?



Simple Linear Regression

❖ How bias affect the loss function?



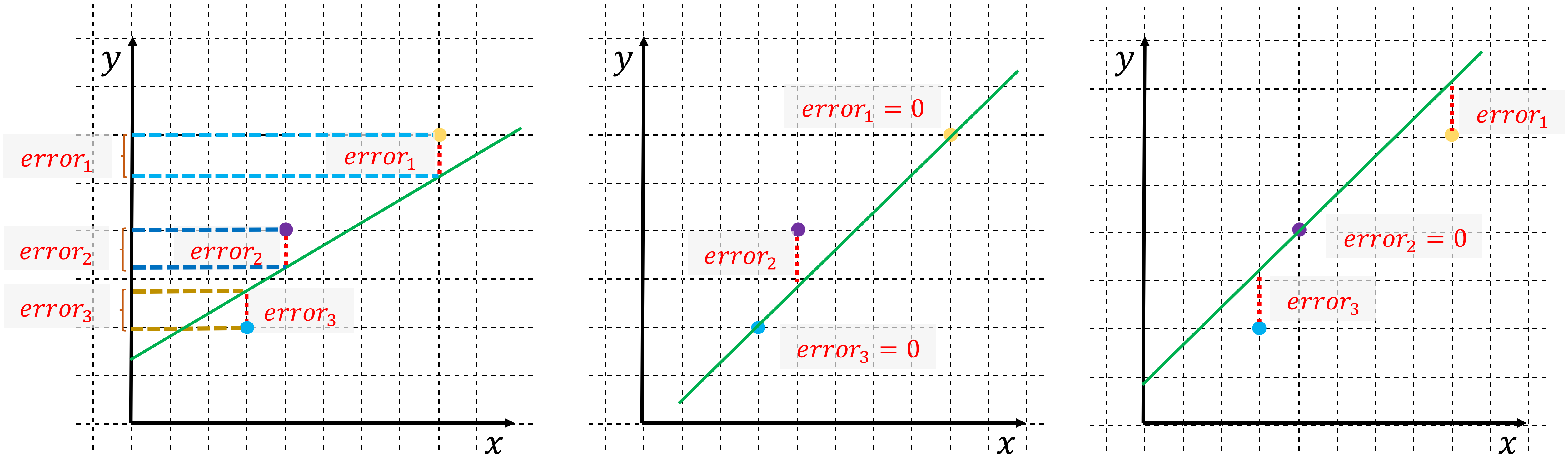
Find a way to change the weights so that loss will move toward minima point.

Simple Linear Regression

❖ Toward Gradient Descent

Training data

$$error_i = distance(\hat{y}_i, y_i)$$



$\hat{y} = w_1 x + b_1$ $\quad\quad$ $\hat{y} = w_2 x + b_2$ $\quad\quad$ $\hat{y} = w_3 x + b_3$

Find a and b whose model has the smallest error, where $error = \sum_i error_i$

How?

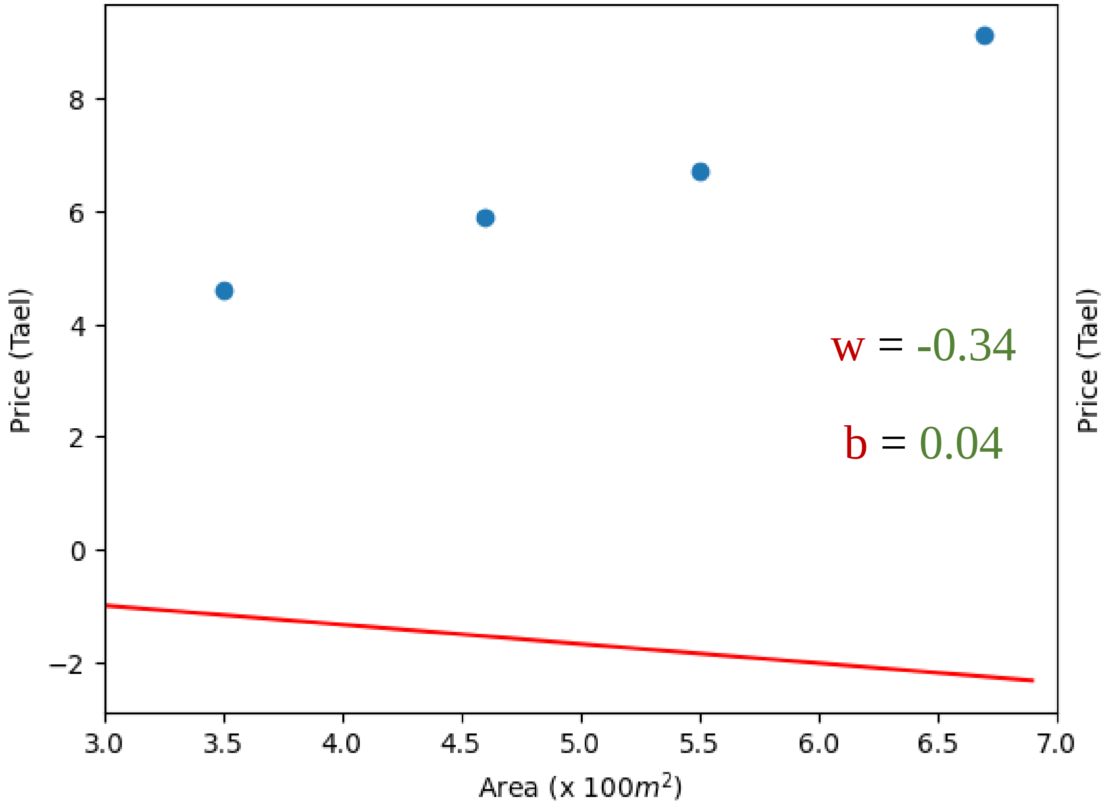
Simple Linear Regression

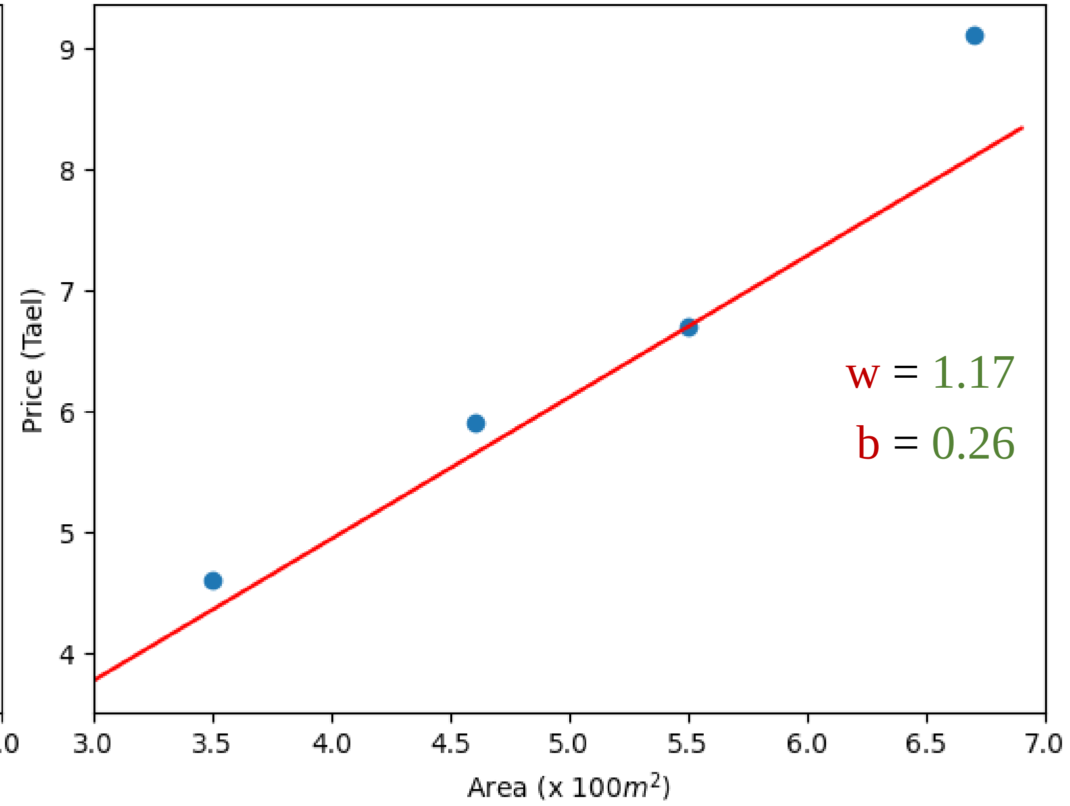
❖ Toward Gradient Descent

$$\hat{y}_i = wx_i + b$$

$$L(\hat{y}_i, y_i) = (\hat{y}_i - y_i)^2$$

How to change w and b so that $L(\hat{y}_i, y_i)$ reduces





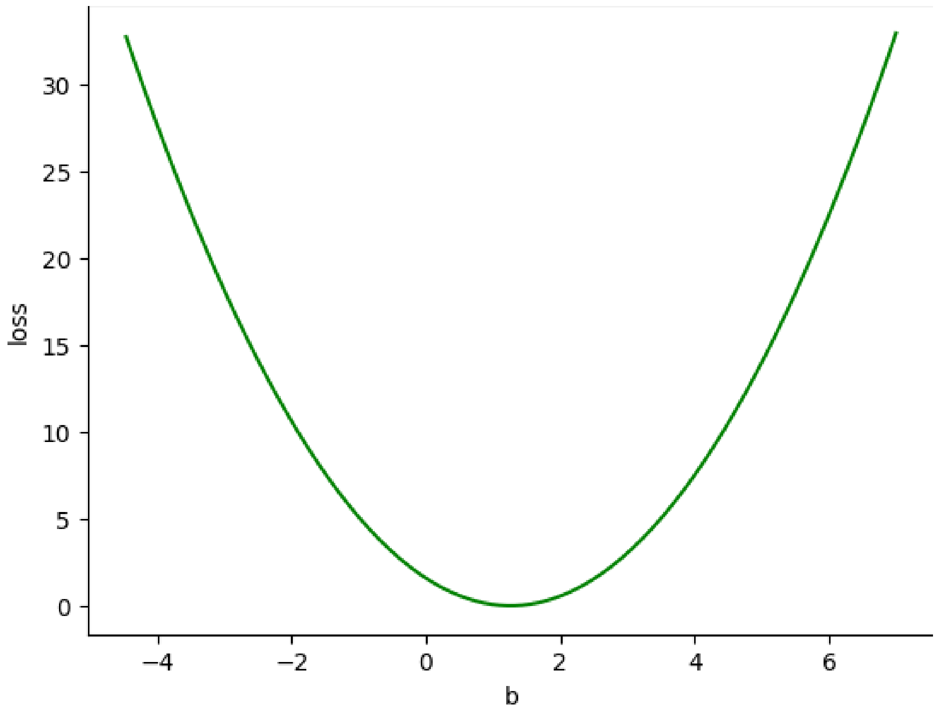
Simple Linear Regression

❖ Toward Gradient Descent

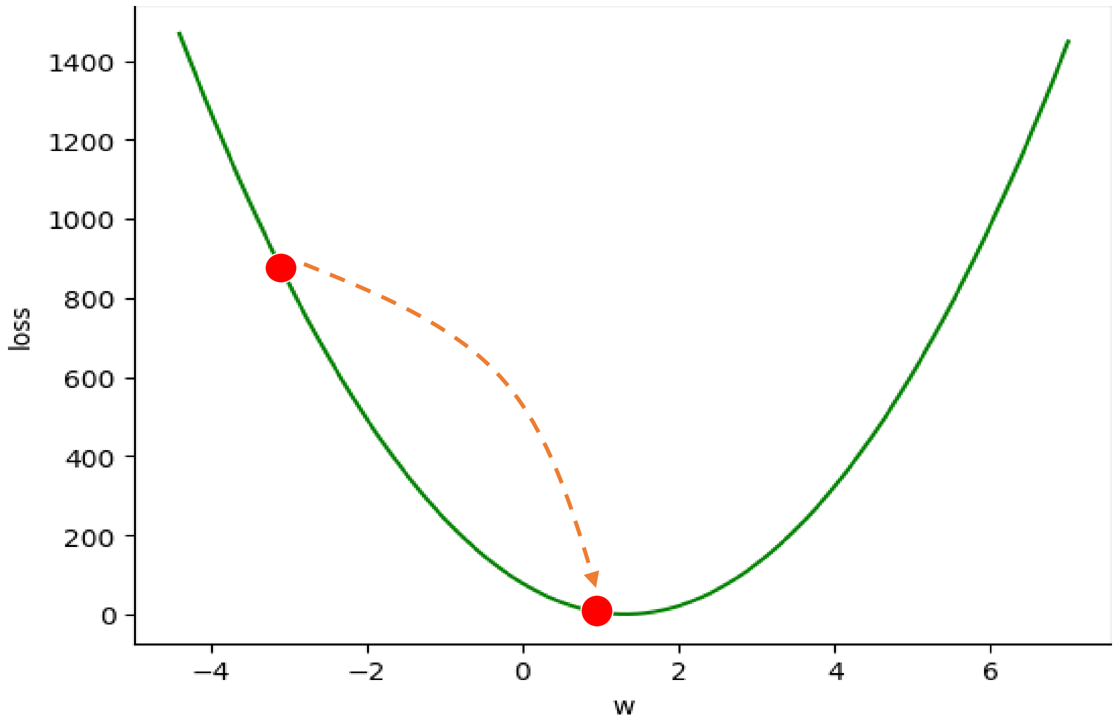
$$\hat{y}_i = wx_i + b$$

$$L(\hat{y}_i, y_i) = (\hat{y}_i - y_i)^2$$

How to change w and b so that $L(\hat{y}_i, y_i)$ reduces



Different b values with a fixed w value



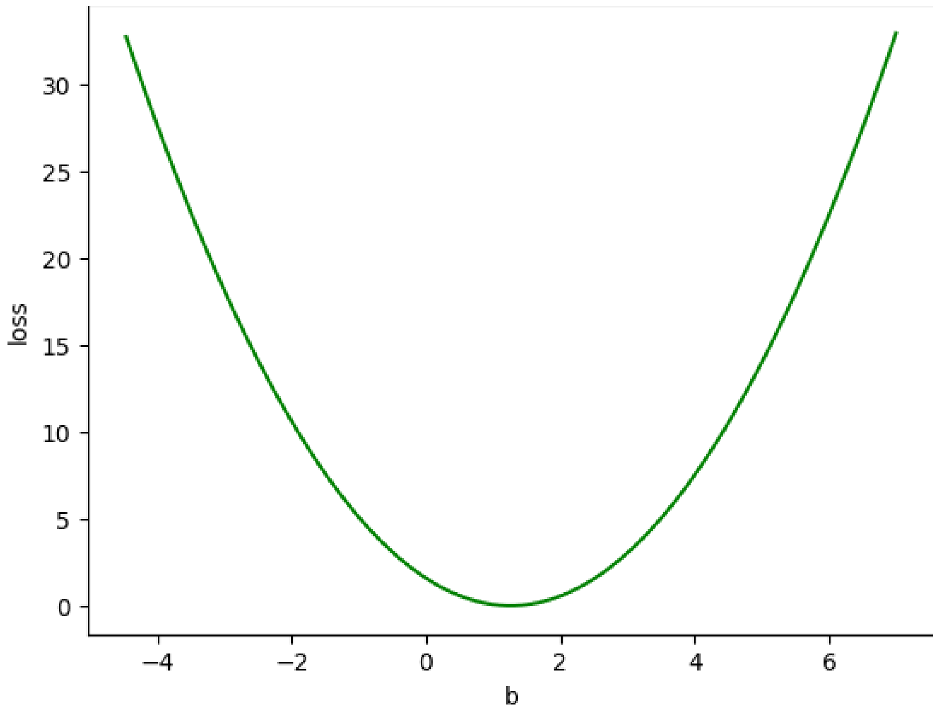
Different w values with a fixed b value

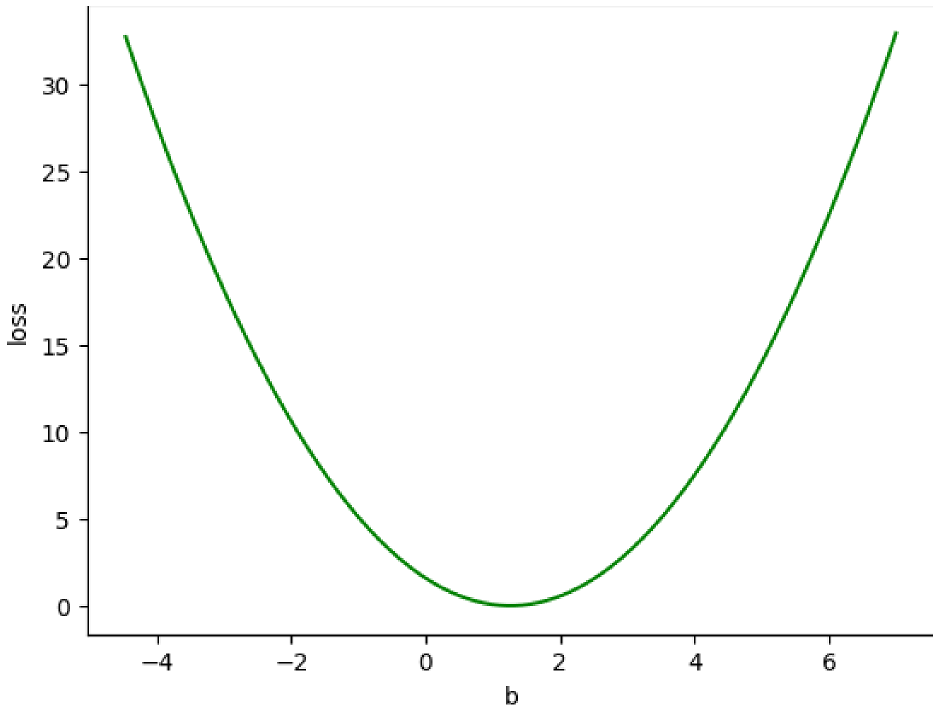
Simple Linear Regression

❖ Toward Gradient Descent

$$\hat{y}_i = wx_i + b$$

$$L(\hat{y}_i, y_i) = (\hat{y}_i - y_i)^2$$

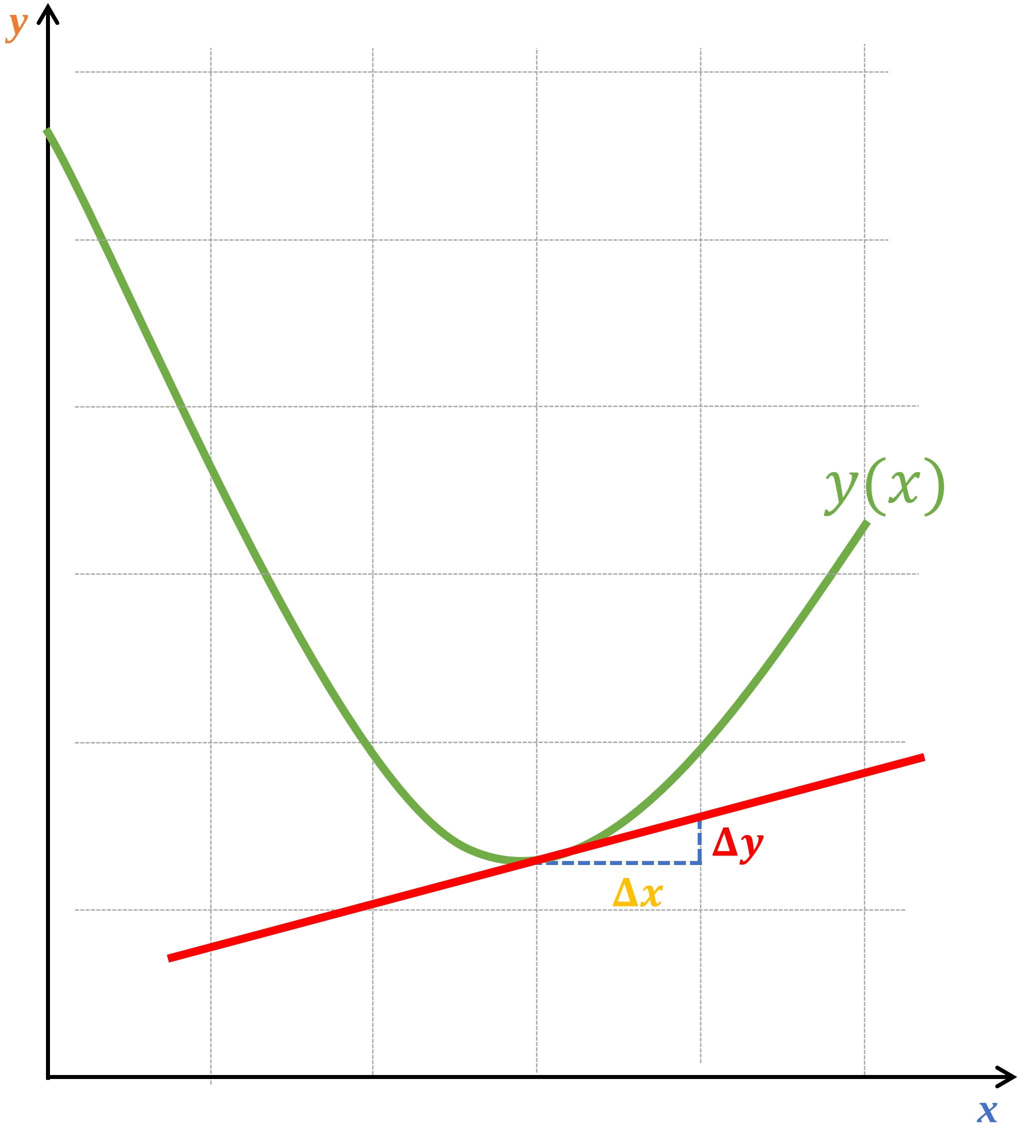




Different b values with a fixed w value

Simple Linear Regression

❖ Derivative of a function

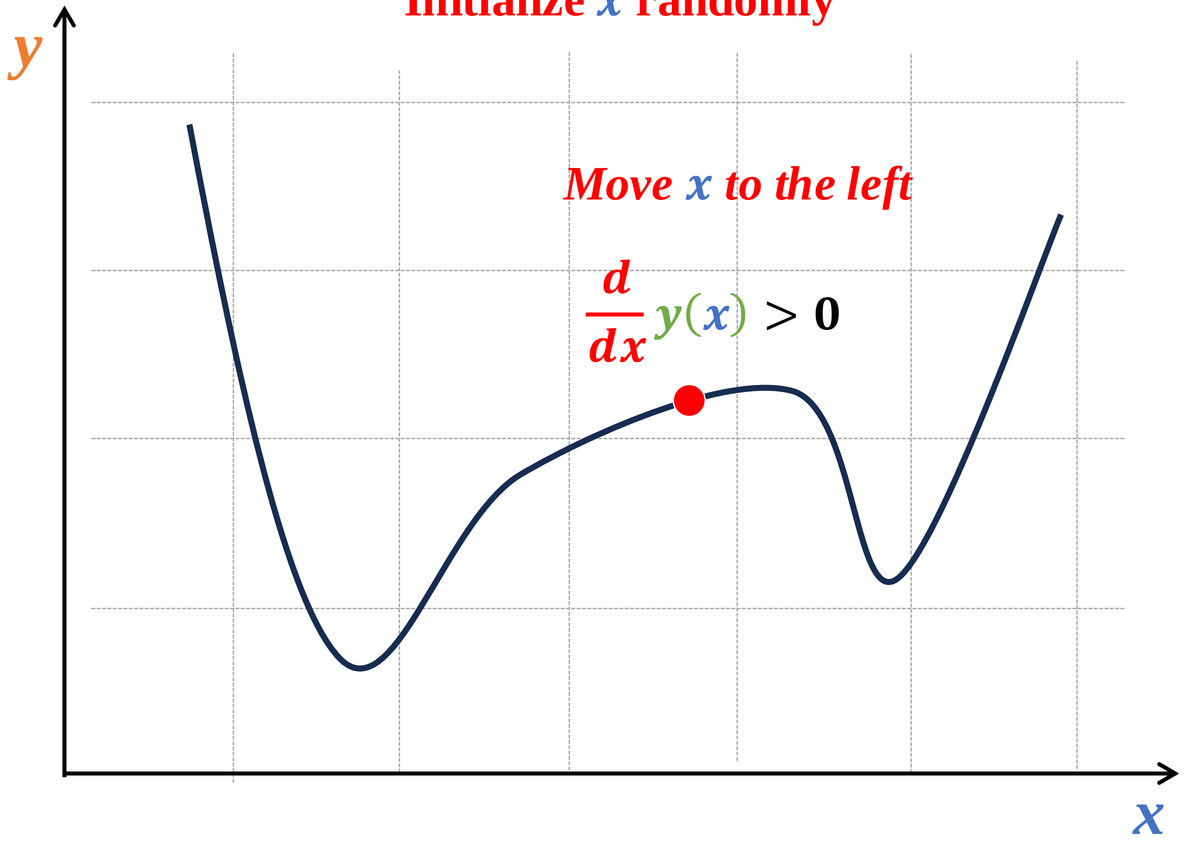


$$\frac{d}{dx}y(x) = \lim_{\Delta x \to 0} \frac{y(x + \Delta x) - y(x)}{\Delta x}$$

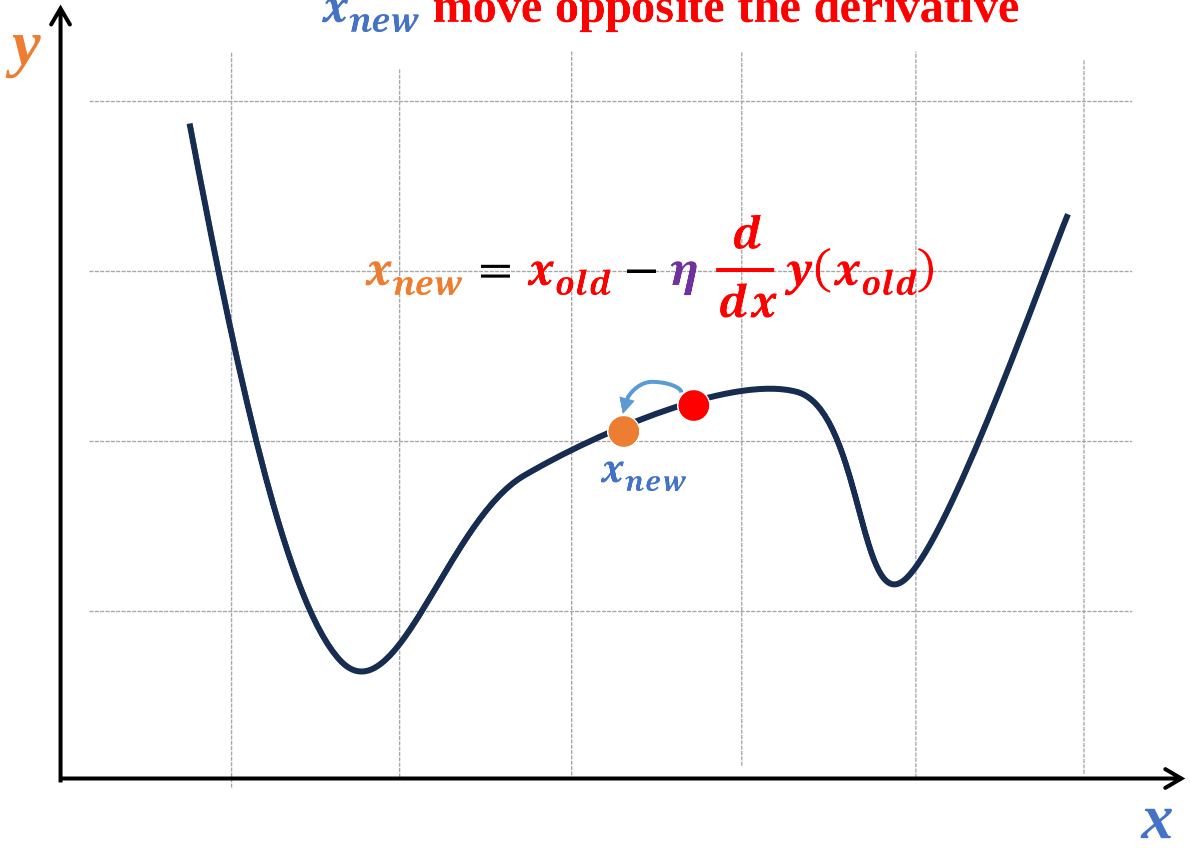
Simple Linear Regression

❖ How gradient descent works?

Initialize x randomly



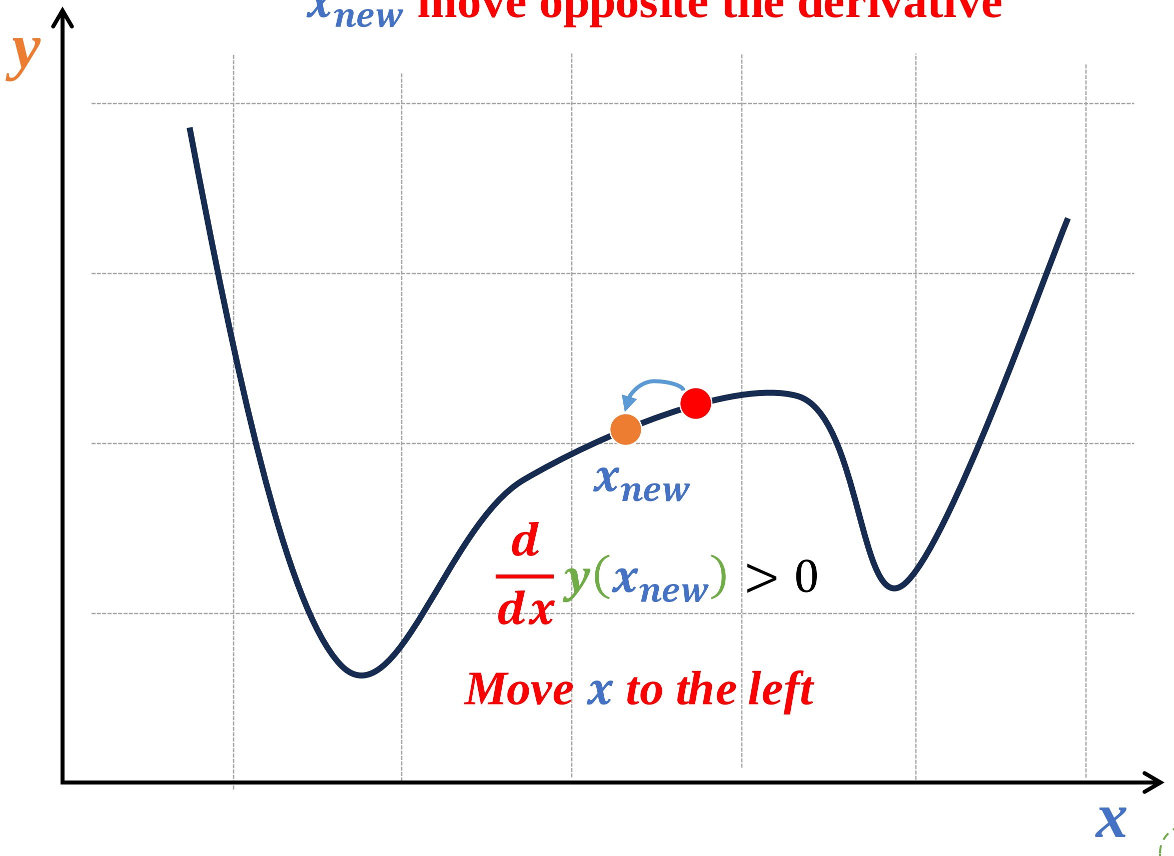
x_{new} move opposite the derivative



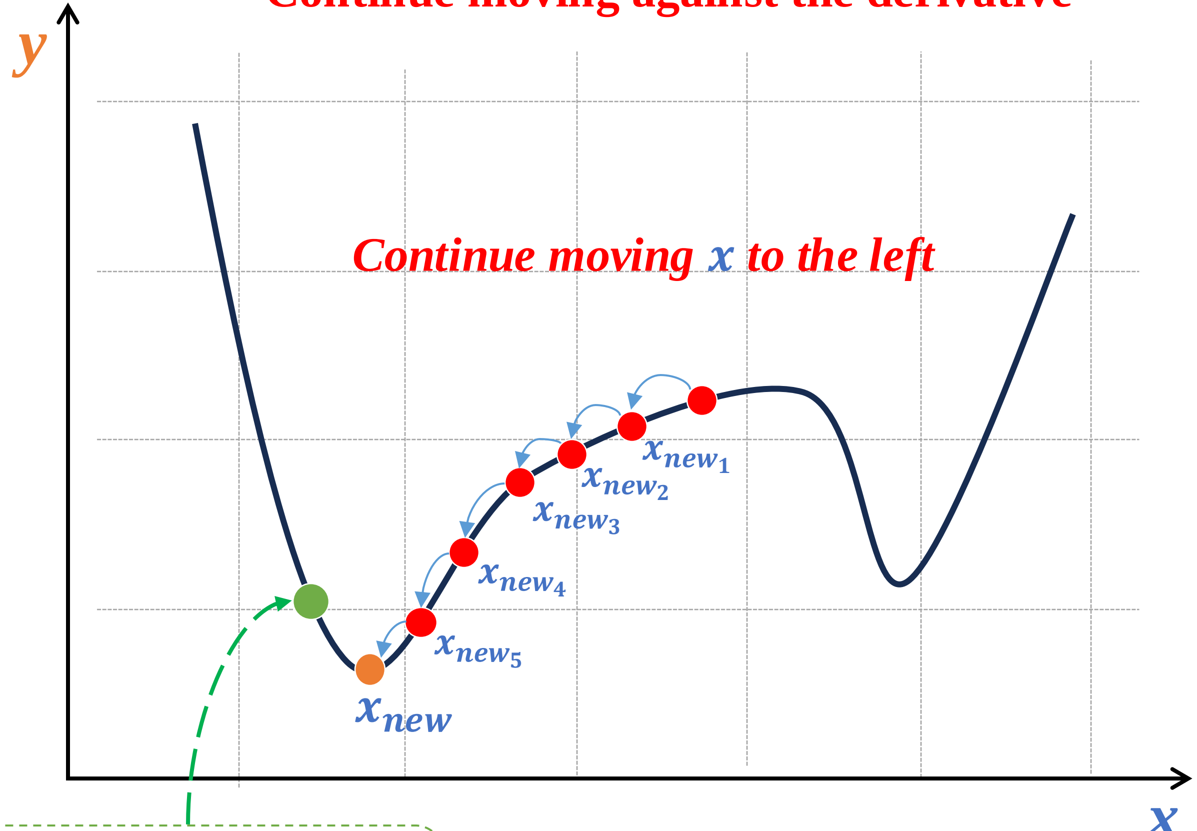
Simple Linear Regression

❖ How gradient descent works?

x_{new} move opposite the derivative



Continue moving against the derivative

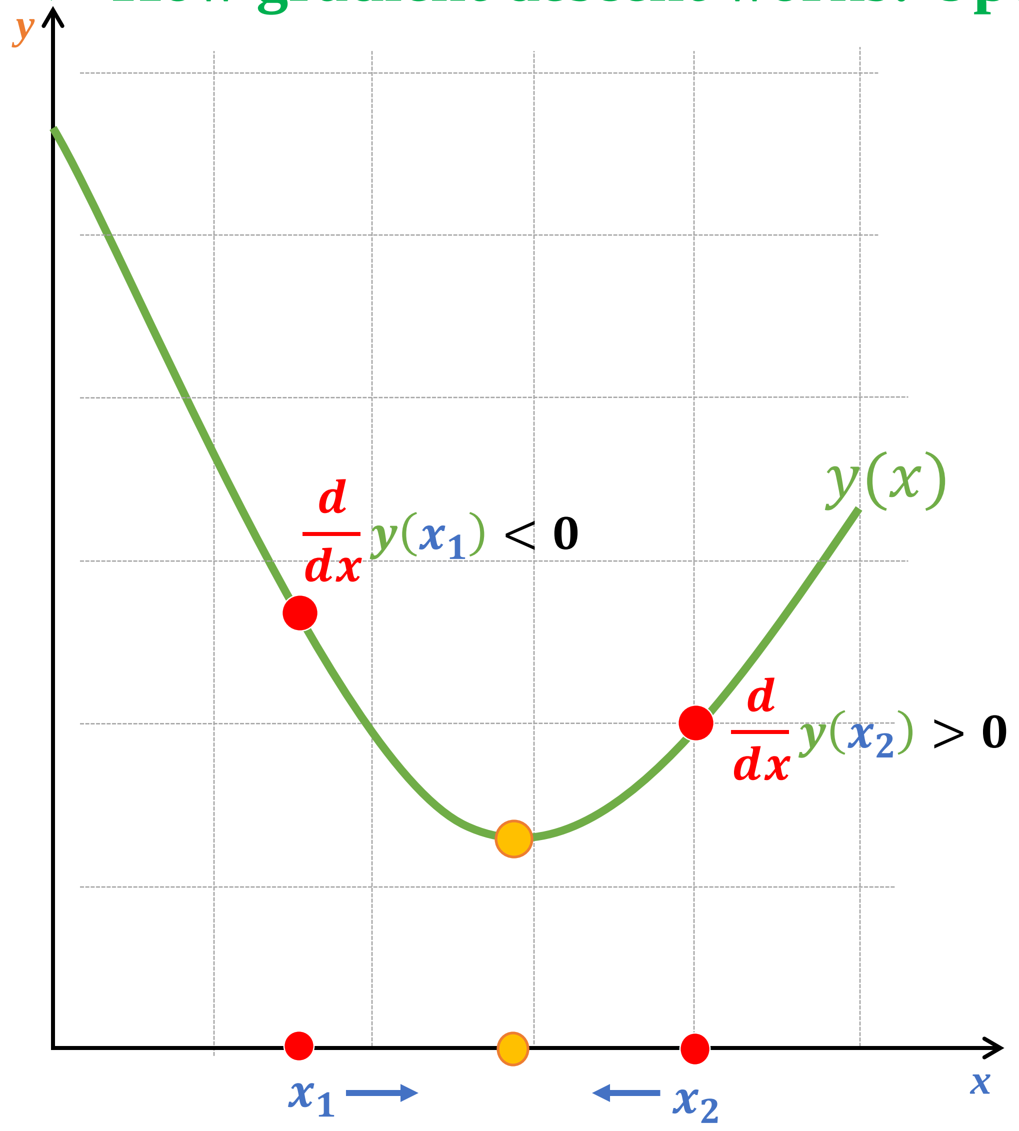


$\frac{d}{dx}y(x_{new}) > 0$

Move x to the right

Simple Linear Regression

❖ How gradient descent works: Update gradient



$$\boxed{\frac{d}{dx}y(x)} = \lim_{\Delta x \to 0} \frac{y(x+\Delta x) - y(x)}{\Delta x}$$

$$x_{new} = x_{old} - \eta \boxed{\frac{d}{dx}y(x_{old})}$$

learning rate

Simple Linear Regression

❖ Gradient Descent

Linear equation

$$\hat{y} = wx + b$$

where $\hat{y}$ is a predicted value,

w and b are parameters

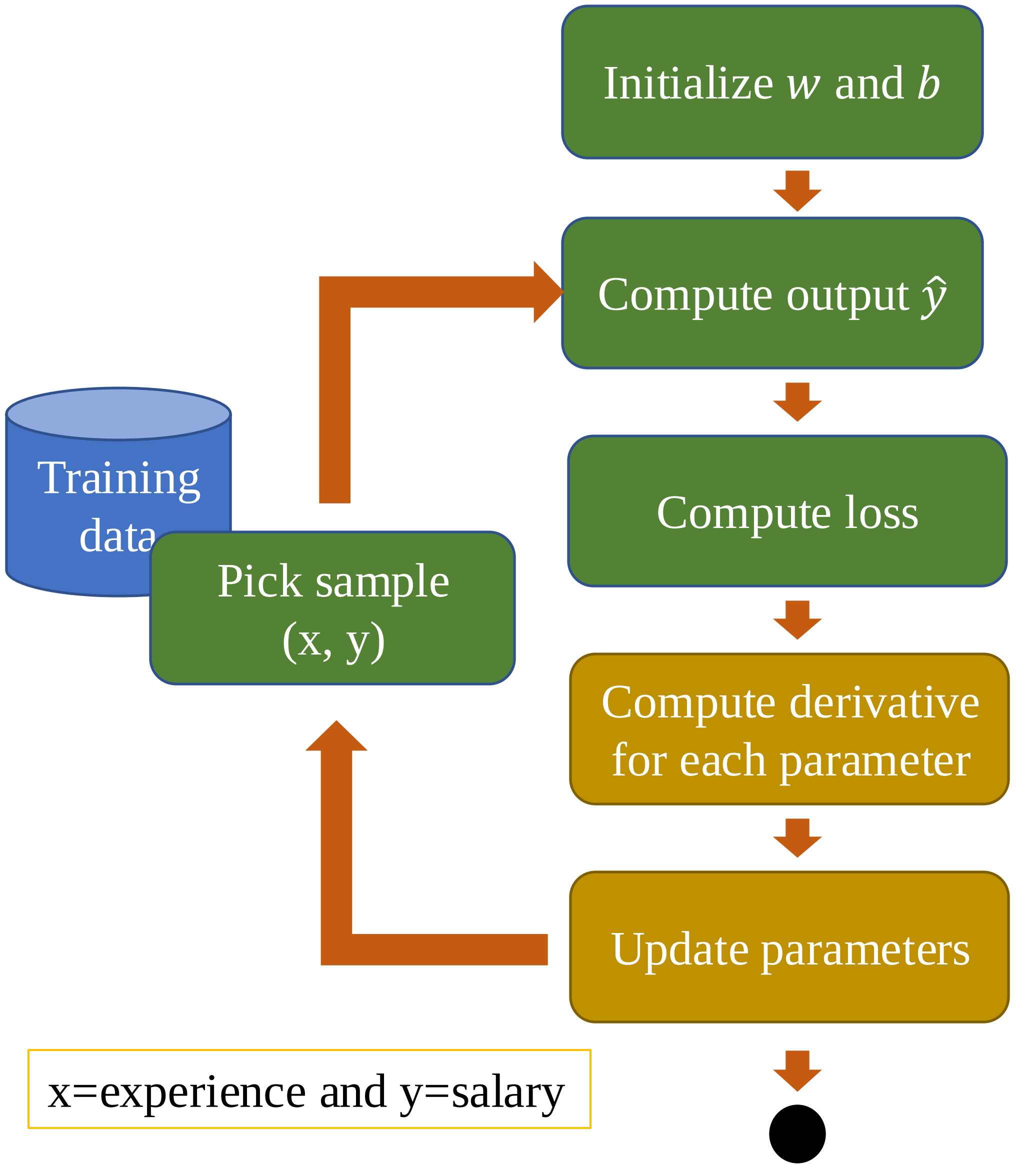
and x is input feature

Error (loss) computation

Idea: compare predicted values $\hat{y}$ and label values y

Squared loss

$$L(\hat{y}, y) = (\hat{y} - y)^2$$



Simple Linear Regression

❖ Gradient Descent

Linear equation

$$\hat{y} = wx + b$$

where $\hat{y}$ is a predicted value,

w and b are parameters

and x is input feature

Error (loss) computation

Idea: compare predicted values $\hat{y}$ and label values y

Squared loss

$$\mathrm{L}(\hat{y}, y) = (\hat{y} - y)^2$$

Find better w and b

Use gradient descent to minimize the loss function

Compute derivate for each parameter

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w} = 2x(\hat{y} - y)$$

$$\frac{\partial L}{\partial b} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial b} = 2(\hat{y} - y)$$

Update parameters

$$w = w - \eta \frac{\partial L}{\partial w} \qquad b = b - \eta \frac{\partial L}{\partial b}$$

η is learning rate

Simple Linear Regression

❖ Proof: Bias (Intercept)

$$\text{Loss} = \sum_i (\hat{y} - y)^2 = \sum_i ((wx + b) - y)^2$$

$$\frac{\partial L}{\partial b} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial b} = \frac{\partial}{\partial \hat{y}} (\hat{y} - y)^2 \cdot \frac{\partial}{\partial b} (wx + b)$$

$$\frac{\partial L}{\partial \hat{y}} = \frac{\partial}{\partial \hat{y}} (\hat{y} - y)^2 = 2 \cdot (\hat{y} - y) \cdot \frac{\partial (\hat{y}_i - y_i)}{\partial \hat{y}_i}$$

$$= 2 \cdot (\hat{y} - y) \cdot \frac{\partial}{\partial \hat{y}_i} (\hat{y}_i - y_i)$$

$$= 2 \cdot (\hat{y} - y) \cdot 1$$

$$= 2(\hat{y} - y)$$

$$\frac{\partial \hat{y}}{\partial b} = \frac{\partial}{\partial b} (wx + b)$$

$$= 1$$

$$\Rightarrow \frac{\partial L}{\partial b} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial b} = 2(\hat{y} - y)$$

Simple Linear Regression

❖ Update Bias (Intercept)

$$\frac{\partial L}{\partial b} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial b} = 2(\hat{y} - y)$$

$$x_{new} = x_{old} - \eta \frac{d}{dx} y(x_{old})$$

$$b_{new} = b_{old} - \eta \frac{\partial L}{\partial b}$$

$$b_{new} = b_{old} - \eta \, 2(\hat{y} - y)$$

Simple Linear Regression

❖ Proof: Slope

$$\text{Loss} = \sum_i (\hat{y} - y)^2 = \sum_i ((wx + b) - y)^2$$

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w} = \frac{\partial}{\partial \hat{y}} (\hat{y} - y)^2 \cdot \frac{\partial}{\partial w} (wx + b)$$

$$\frac{\partial L}{\partial \hat{y}} = \frac{\partial}{\partial \hat{y}} (\hat{y} - y)^2 = 2 \cdot (\hat{y} - y) \cdot \frac{\partial (\hat{y}_i - y_i)}{\partial \hat{y}_i}$$

$$= 2 \cdot (\hat{y} - y) \cdot \frac{\partial}{\partial \hat{y}_i} (\hat{y}_i - y_i)$$

$$= 2 \cdot (\hat{y} - y) \cdot 1$$

$$= 2(\hat{y} - y)$$

$$\frac{\partial \hat{y}}{\partial w} = \frac{\partial}{\partial w} (wx + b)$$

$$= x$$

$$\Rightarrow \frac{\partial L}{\partial w} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w} = 2x(\hat{y} - y)$$

Simple Linear Regression

❖ Update Slope

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w} = 2x(\hat{y} - y)$$

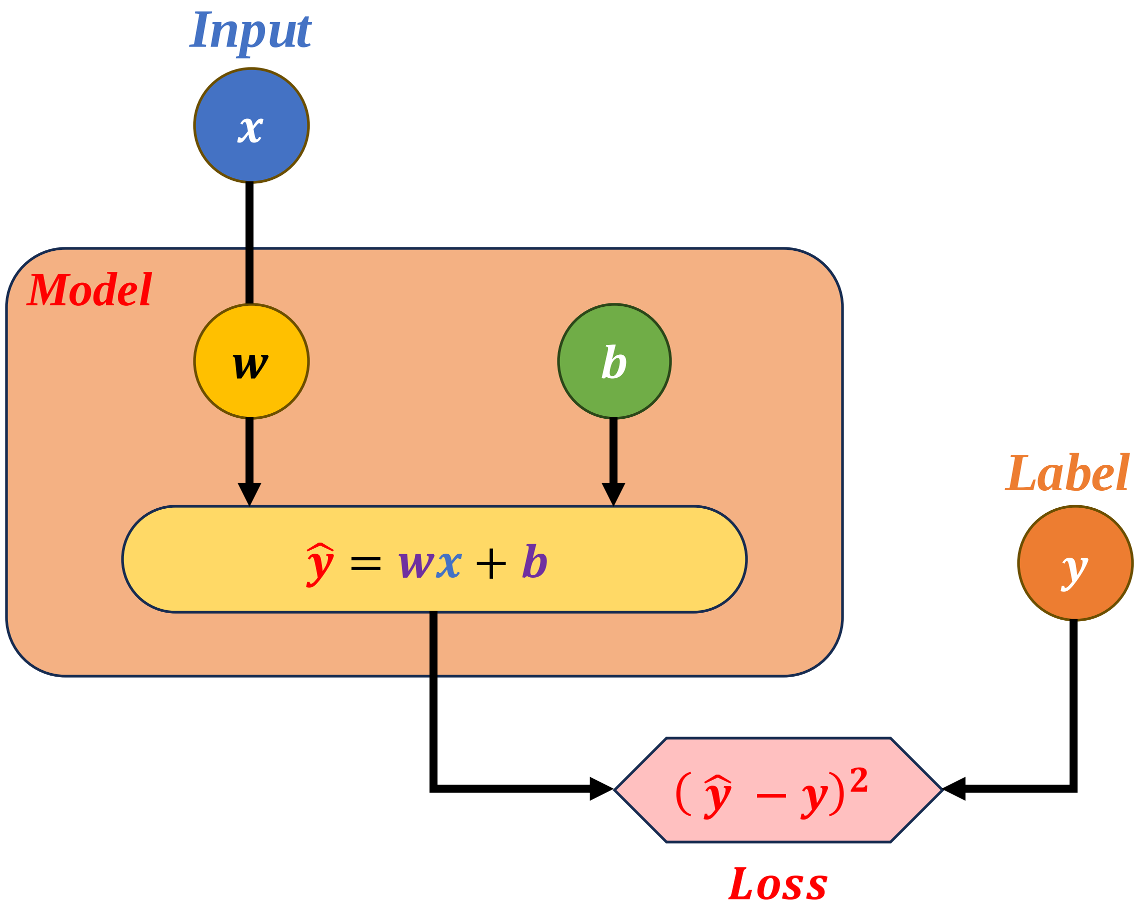
$$x_{new} = x_{old} - \eta \frac{d}{dx} y(x_{old})$$

$$w_{new} = w_{old} - \eta \frac{\partial L}{\partial w}$$

$$w_{new} = w_{old} - \eta \, 2x(\hat{y} - y)$$

Simple Linear Regression

❖ Example



$$\hat{y} = (wx + b)$$

$$\text{Loss} = (\hat{y} - y)^2$$

$$w_{new} = w_{old} - \eta \frac{\partial L}{\partial w}$$

$$b_{new} = b_{old} - \eta \frac{\partial L}{\partial b}$$

$$\frac{\partial L}{\partial w} = 2x(\hat{y} - y)$$

$$\frac{\partial L}{\partial b} = 2(\hat{y} - y)$$

Simple Linear Regression

❖ Example: Forward 1

	Experience	Salary
1	3	60
2	4	55
3	5	66
4	6	93

<u>Initialize</u>

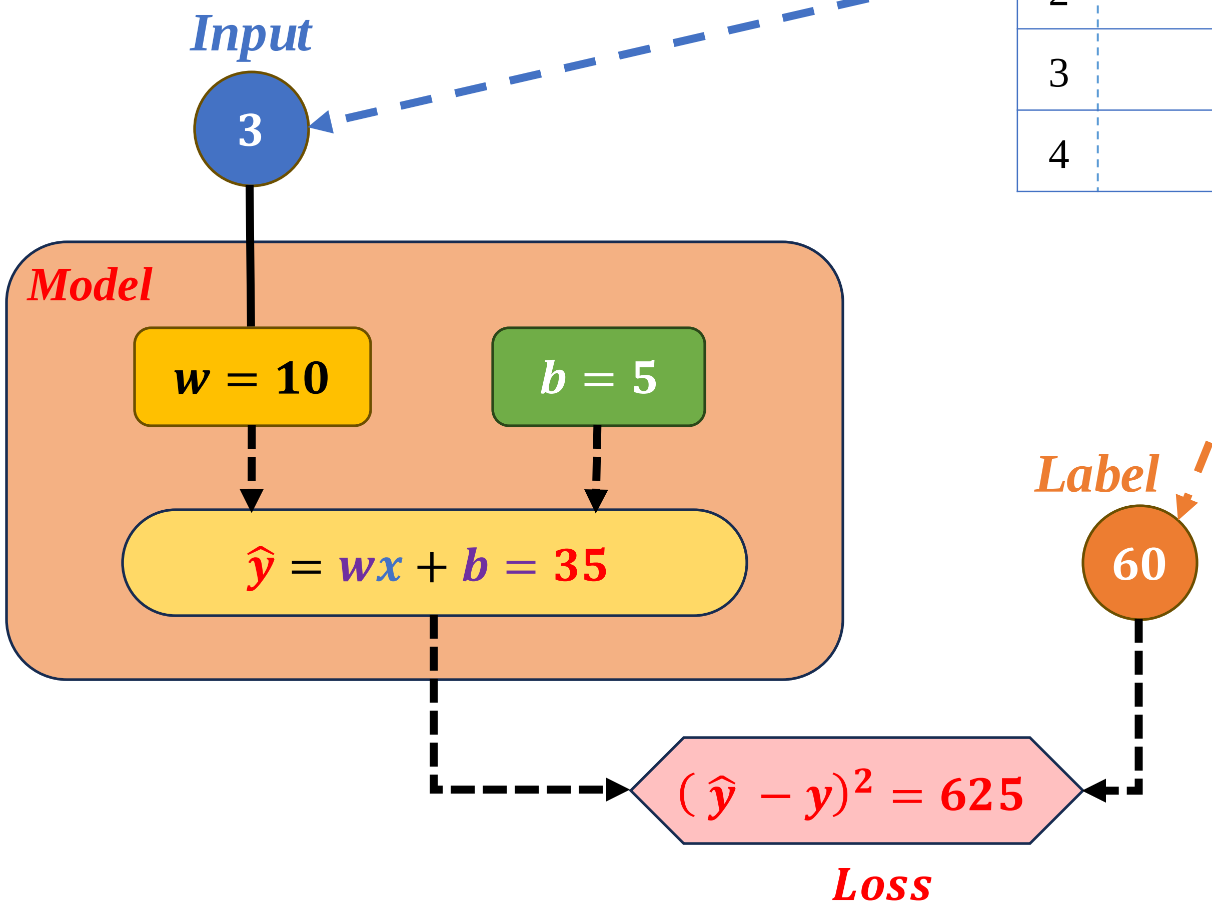
$w = 10$

$b = 5$

$\eta = 0.01$

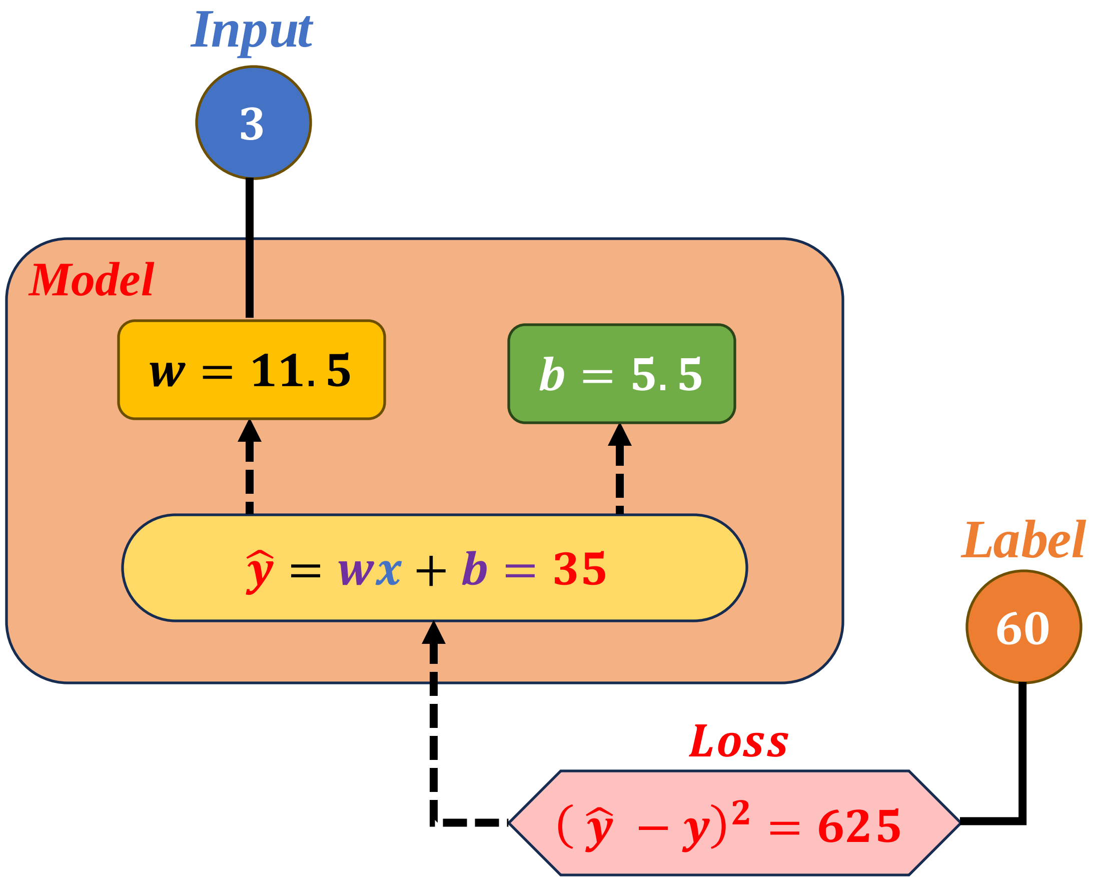
$\hat{y} = (wx + b)$

$\text{Loss} = (\hat{y} - y)^2$



Simple Linear Regression

❖ Example: Backpropagation 1



$$\frac{\partial L}{\partial w} = 2x(\hat{y} - y) \qquad \frac{\partial L}{\partial w} = 2 \times 3(35 - 60) = -150$$

$$\frac{\partial L}{\partial b} = 2(\hat{y} - y) \qquad \frac{\partial L}{\partial b} = 2(35 - 60) = -50$$

$$w_{new} = w_{old} - \eta \frac{\partial L}{\partial w} \qquad 11.5 = 10 - 0.01 \times (-150)$$

$$b_{new} = b_{old} - \eta \frac{\partial L}{\partial b} \qquad 5.5 = 5 - 0.01 \times (-50)$$

Simple Linear Regression

❖ Example: Forward 2

	Experience	Salary
1	3	60
2	4	55
3	5	66
4	6	93

<u>Initialize</u>

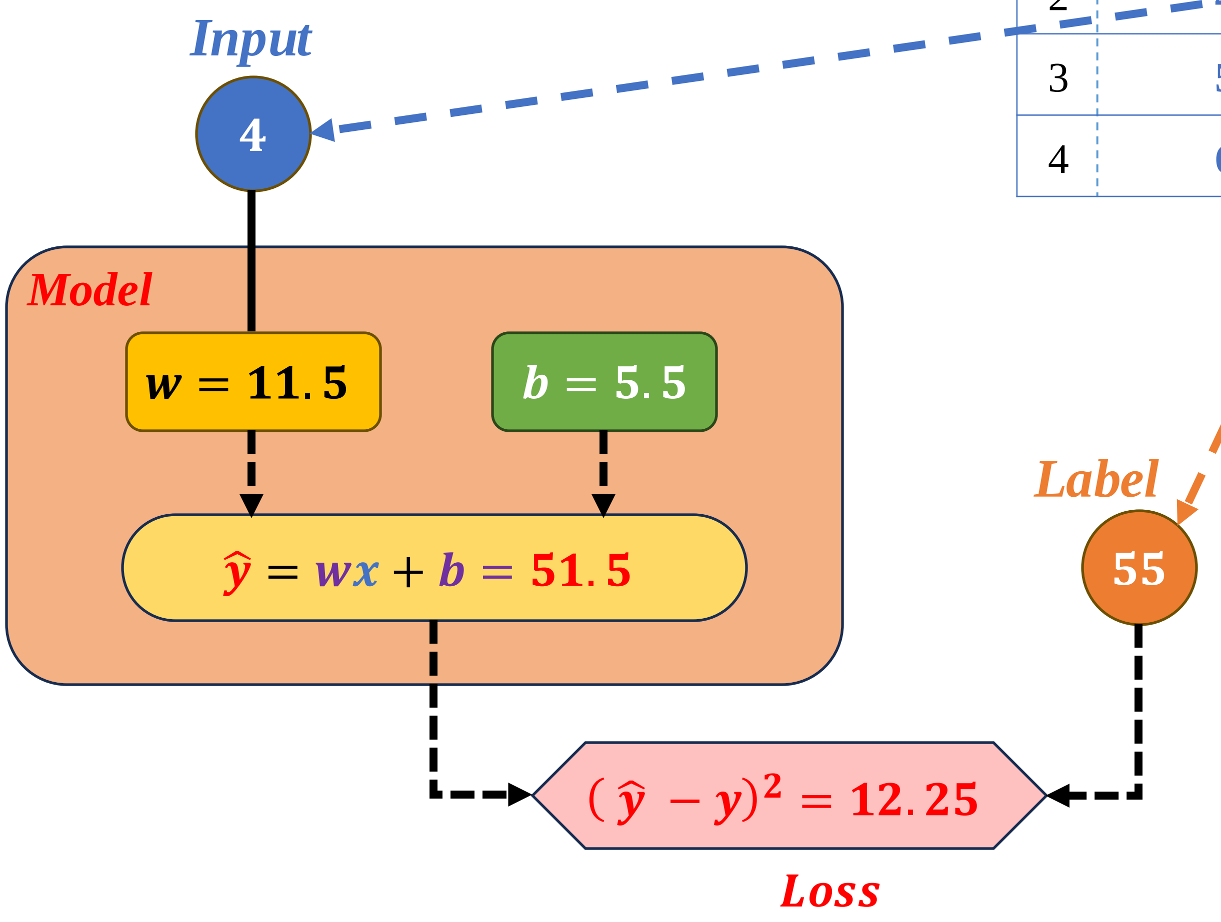
$w = 10$

$b = 5$

$\eta = 0.1$

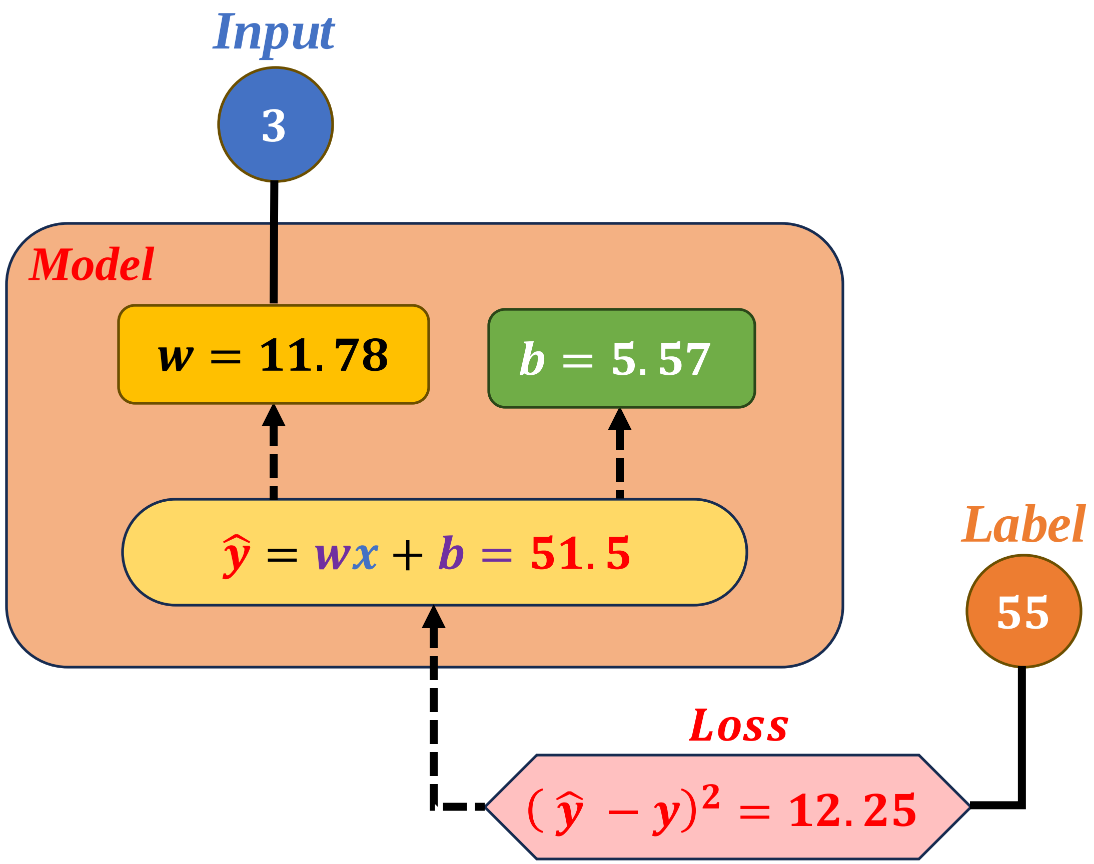
$$\hat{y} = (wx + b)$$

$$\text{Loss} = (\hat{y} - y)^2$$



Simple Linear Regression

❖ Example: Backpropagation 2



$$\frac{\partial L}{\partial w} = 2x(\hat{y} - y)$$

$$\frac{\partial L}{\partial b} = 2(\hat{y} - y)$$

$$\frac{\partial L}{\partial w} = 2 \times 3(51.5 - 55) = -28$$

$$\frac{\partial L}{\partial b} = 2(51.5 - 55) = -7$$

$$w_{new} = w_{old} - \eta \frac{\partial L}{\partial w}$$

$$b_{new} = b_{old} - \eta \frac{\partial L}{\partial b}$$

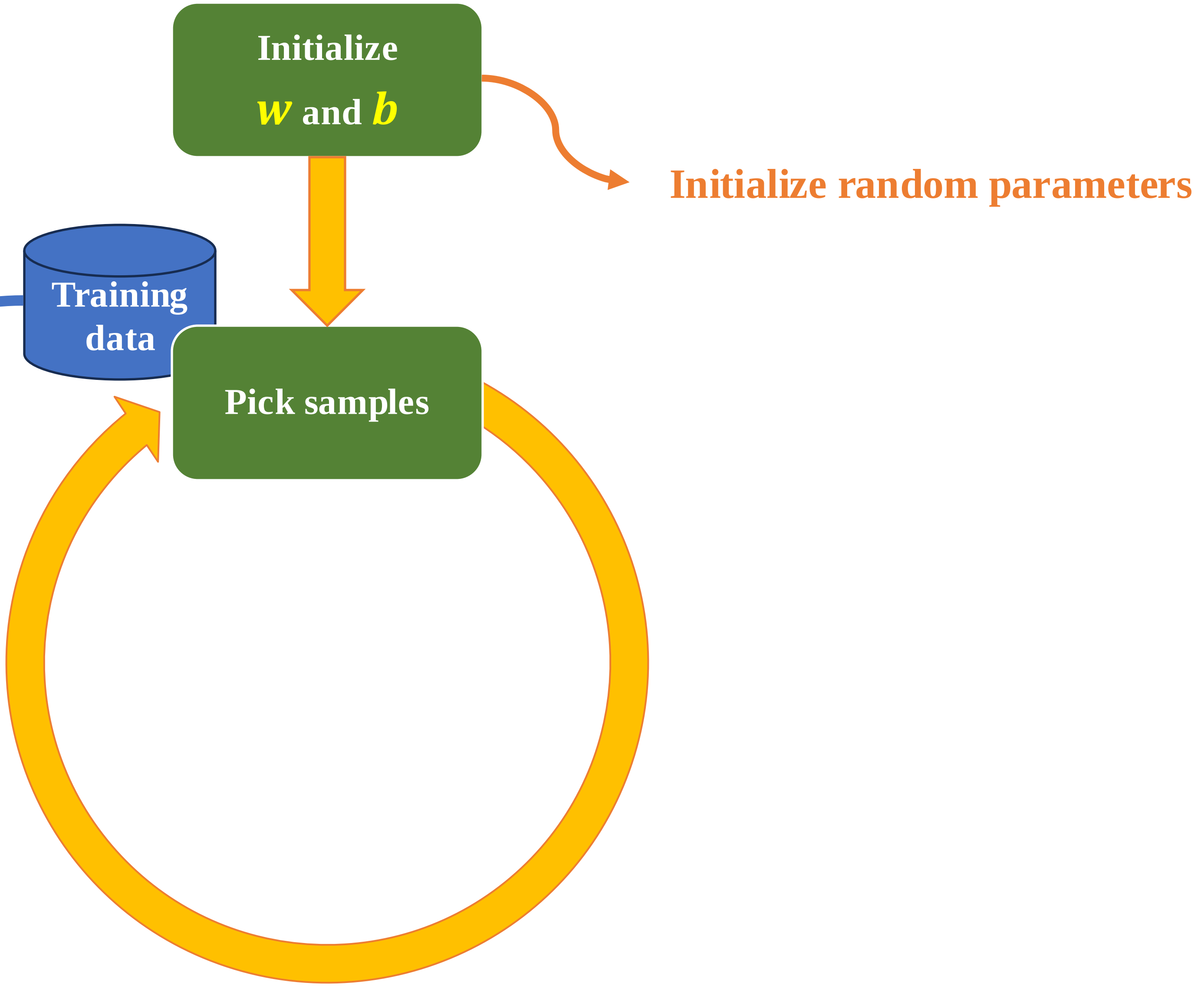
$$11.78 = 11.5 - 0.01 \times (-28)$$

$$5.57 = 5.5 - 0.01 \times (-7)$$

Simple Linear Regression

❖ Update gradient: Proof

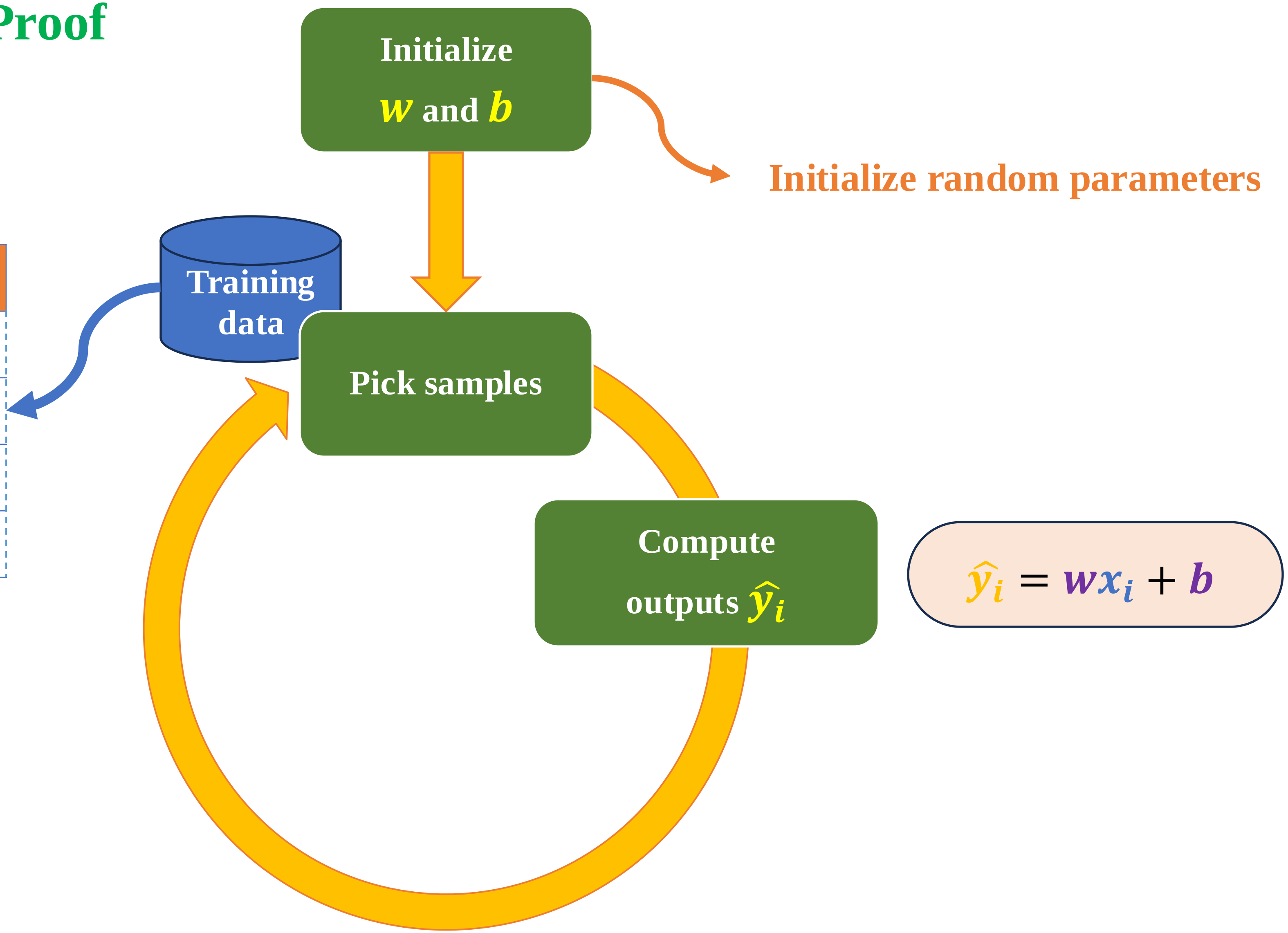
	Experience	Salary
1	3	60
2	4	55
3	5	66
4	6	93



Simple Linear Regression

❖ Update gradient: Proof

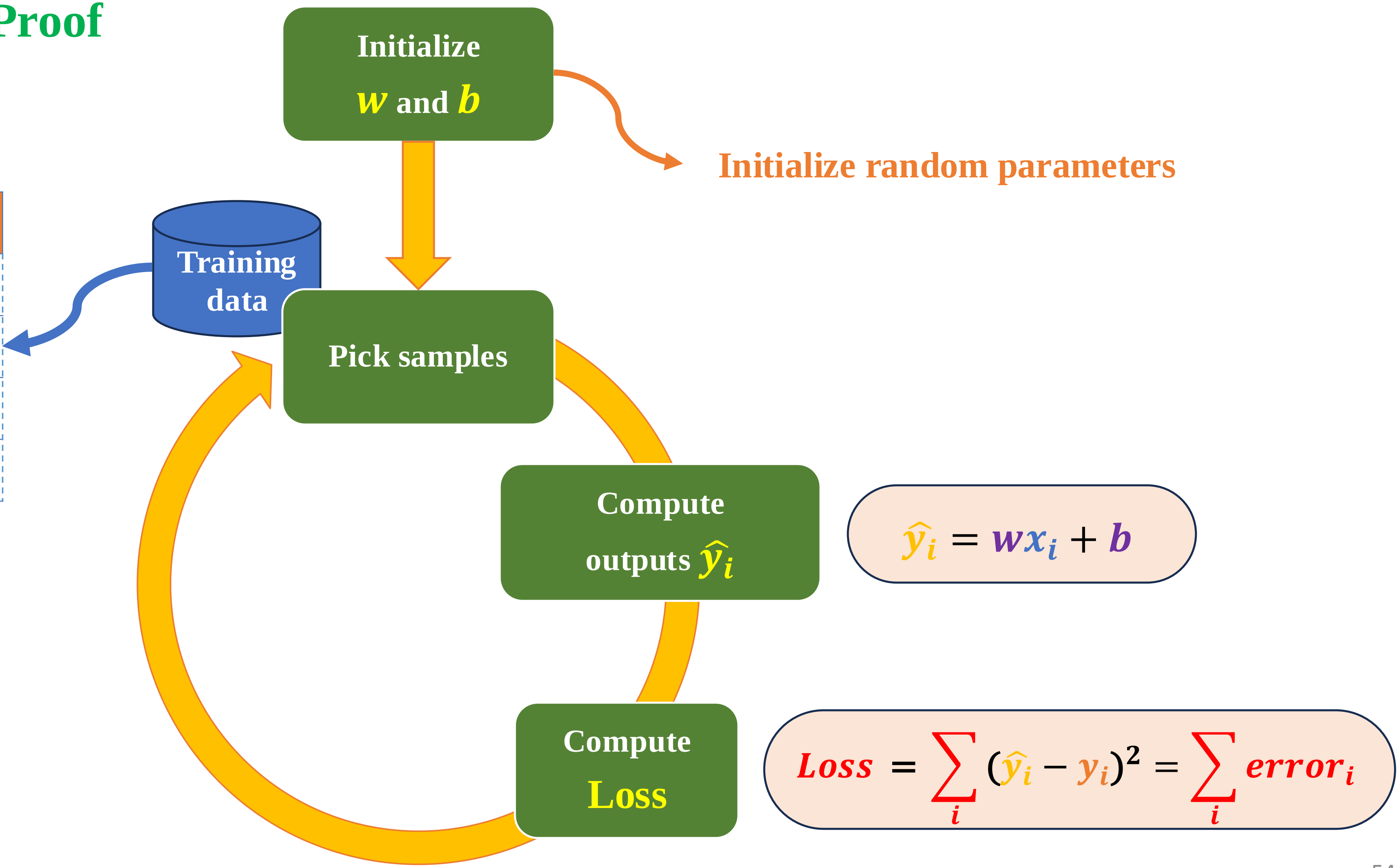
	Experience	Salary
1	3	60
2	4	55
3	5	66
4	6	93



Simple Linear Regression

❖ Update gradient: Proof

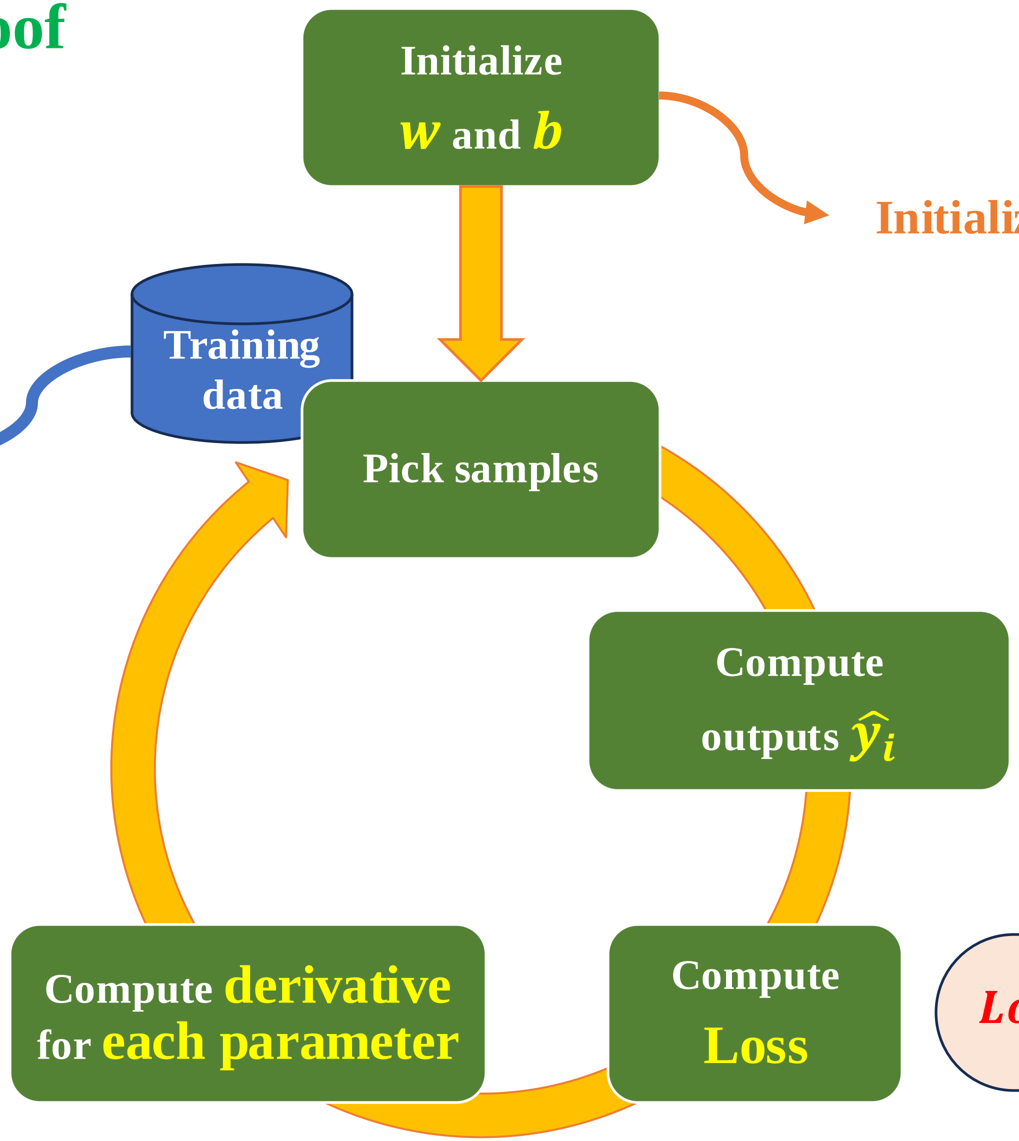
	Experience	Salary
1	3	60
2	4	55
3	5	66
4	6	93



Simple Linear Regression

❖ Update gradient: Proof

	Experience	Salary
1	3	60
2	4	55
3	5	66
4	6	93



Initialize random parameters

$$\hat{y}_i = wx_i + b$$

$$Loss = \sum_i (\hat{y}_i - y_i)^2 = \sum_i error_i$$

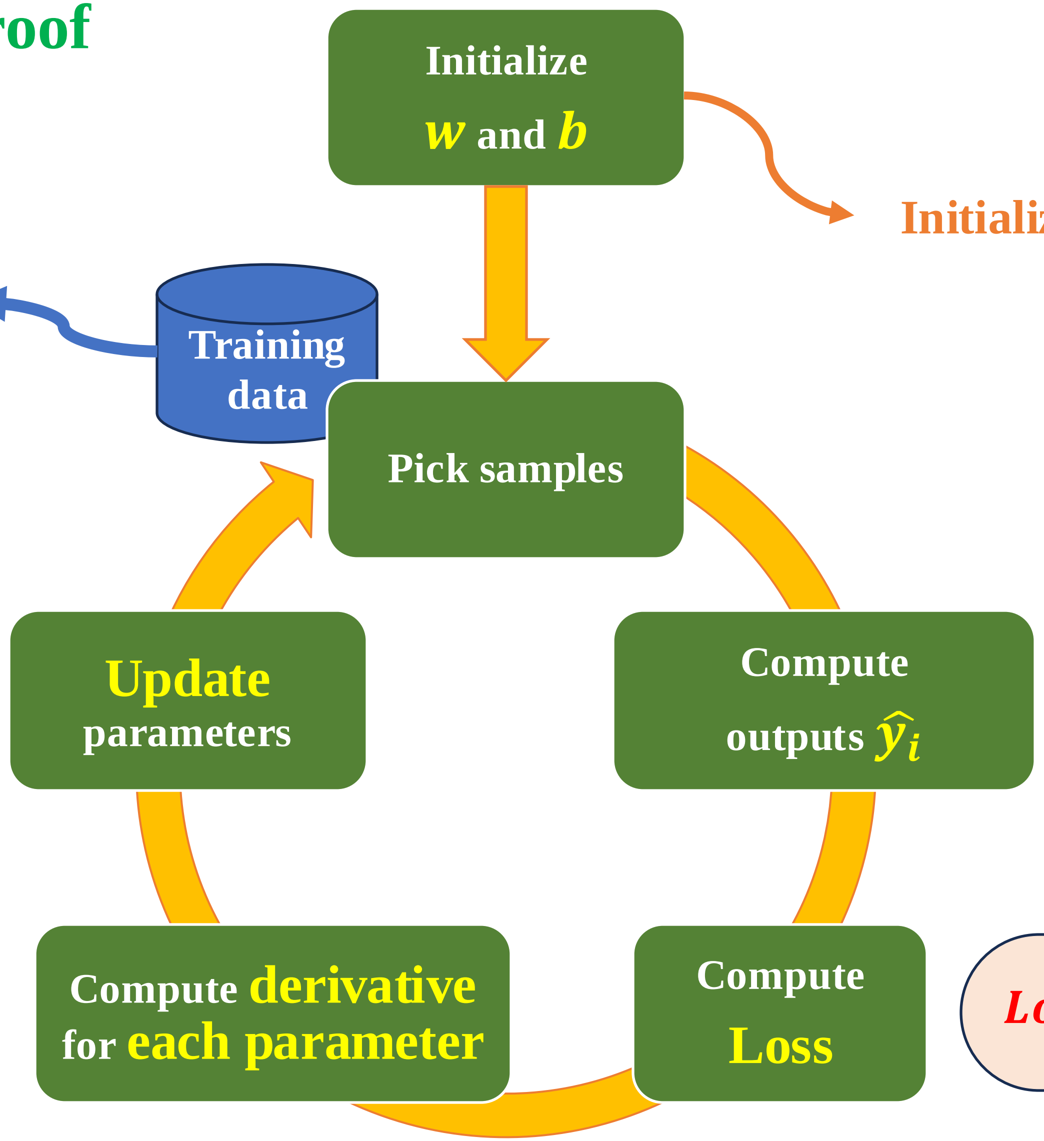
$$\frac{\partial L}{\partial b} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial b} = 2(\hat{y} - y)$$

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w} = 2x(\hat{y} - y)$$

Simple Linear Regression

❖ Update gradient: Proof

	Experience	Salary
1	3	60
2	4	55
3	5	66
4	6	93



Initialize random parameters

$$\hat{y}_i = wx_i + b$$

$$Loss = \sum_i (\hat{y}_i - y_i)^2 = \sum_i error_i$$

$$w_{new} = w_{old} - \eta \frac{\partial L}{\partial w}$$

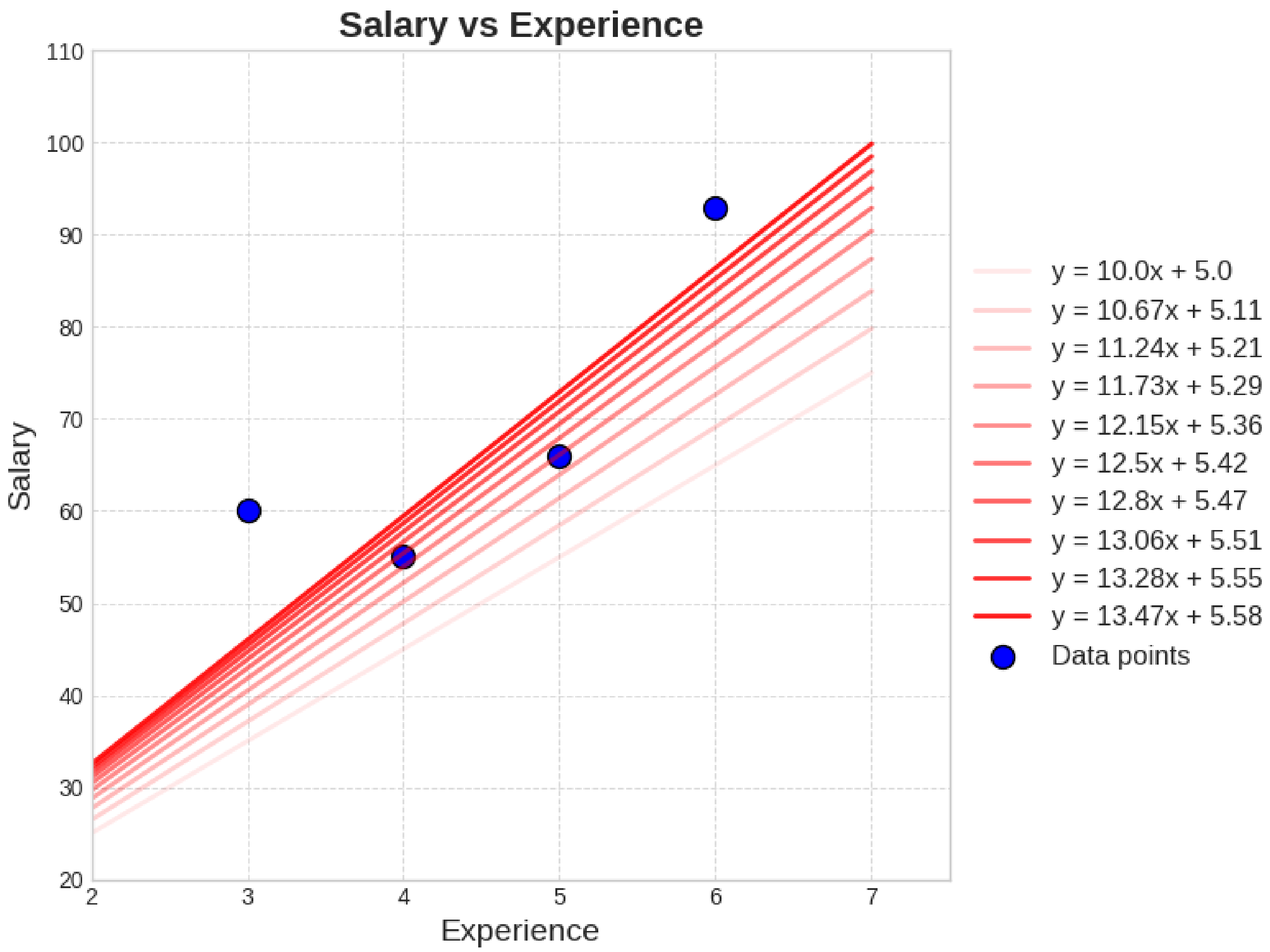
$$b_{new} = b_{old} - \eta \frac{\partial L}{\partial b}$$

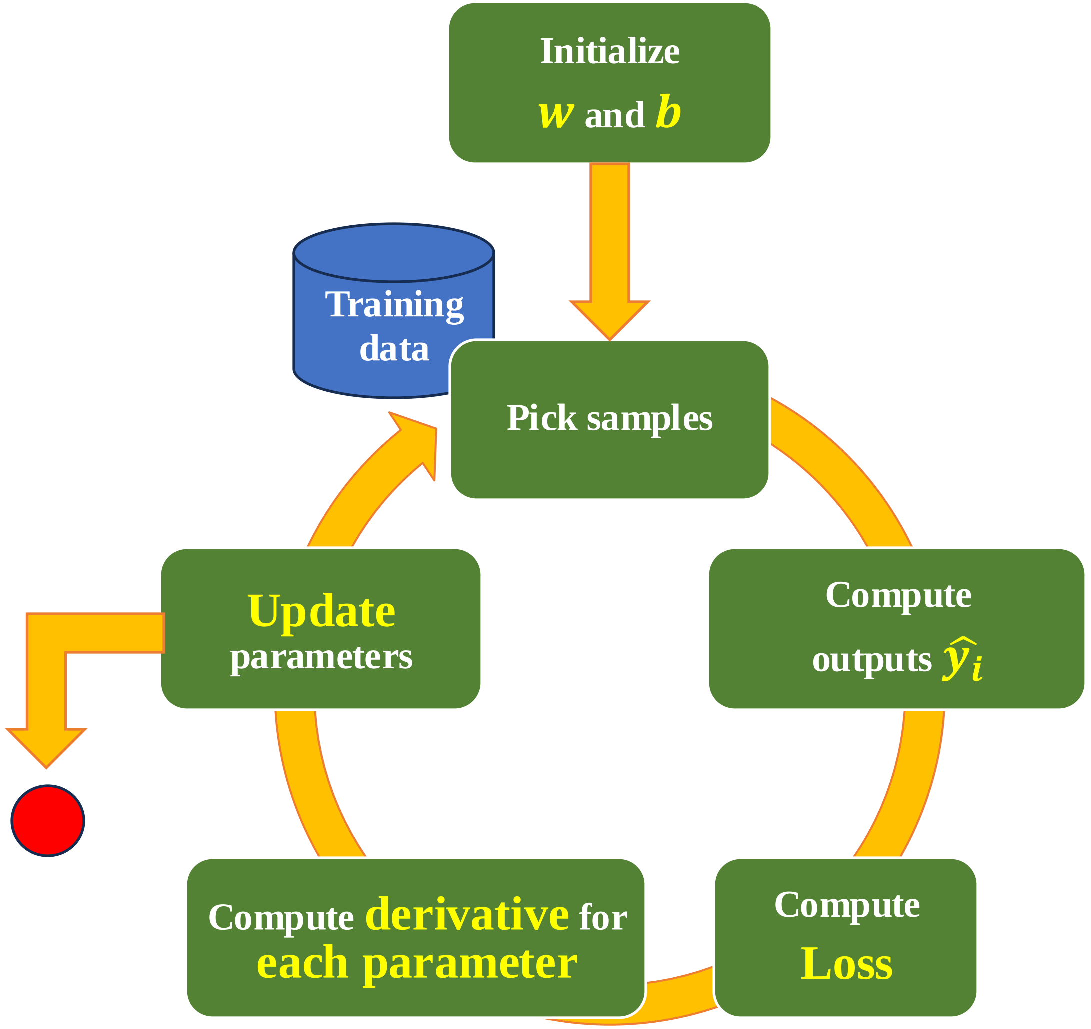
$$\frac{\partial L}{\partial b} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial b} = 2(\hat{y} - y)$$

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w} = 2x(\hat{y} - y)$$

Simple Linear Regression

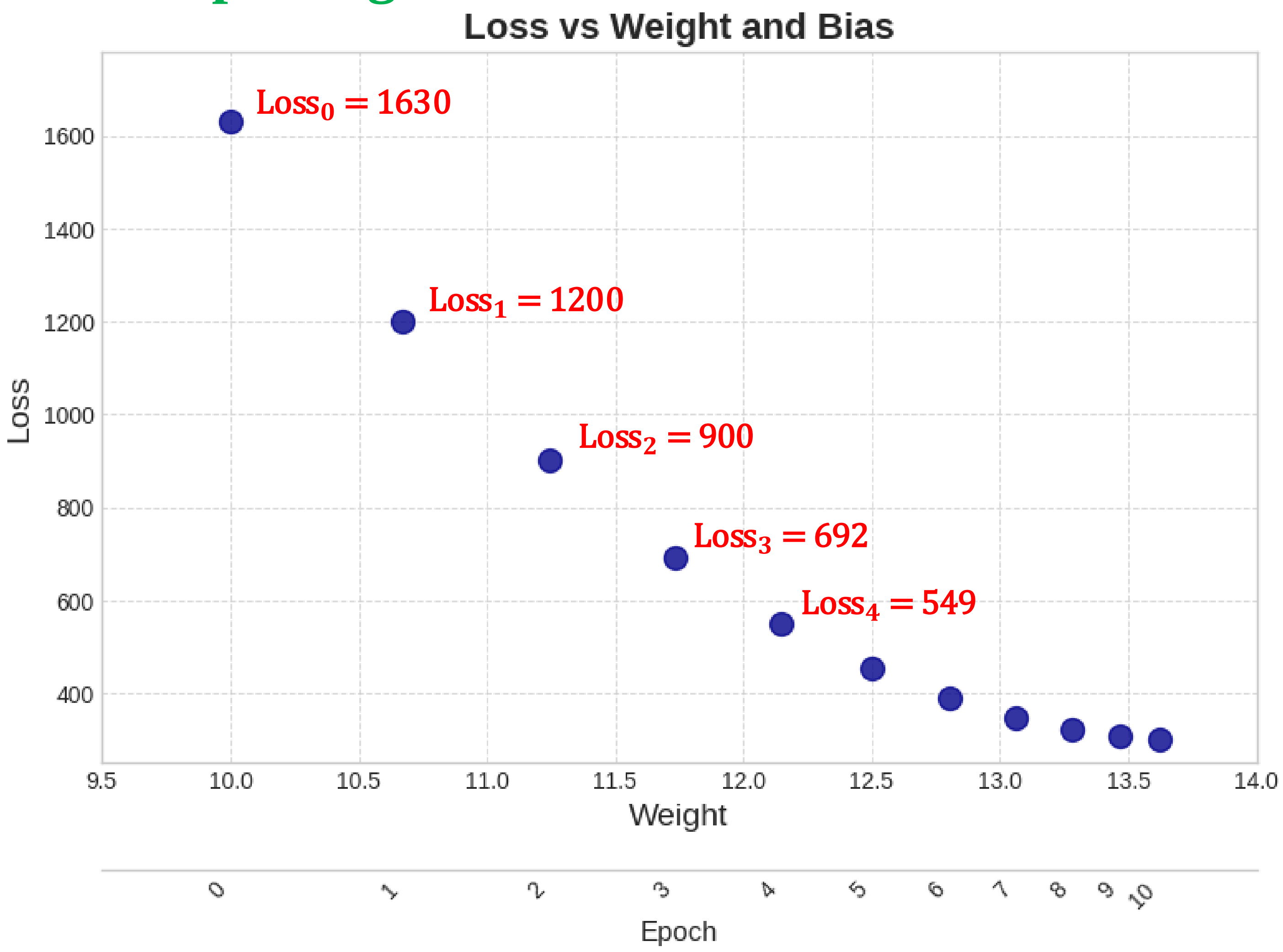
❖ Update gradient: Proof

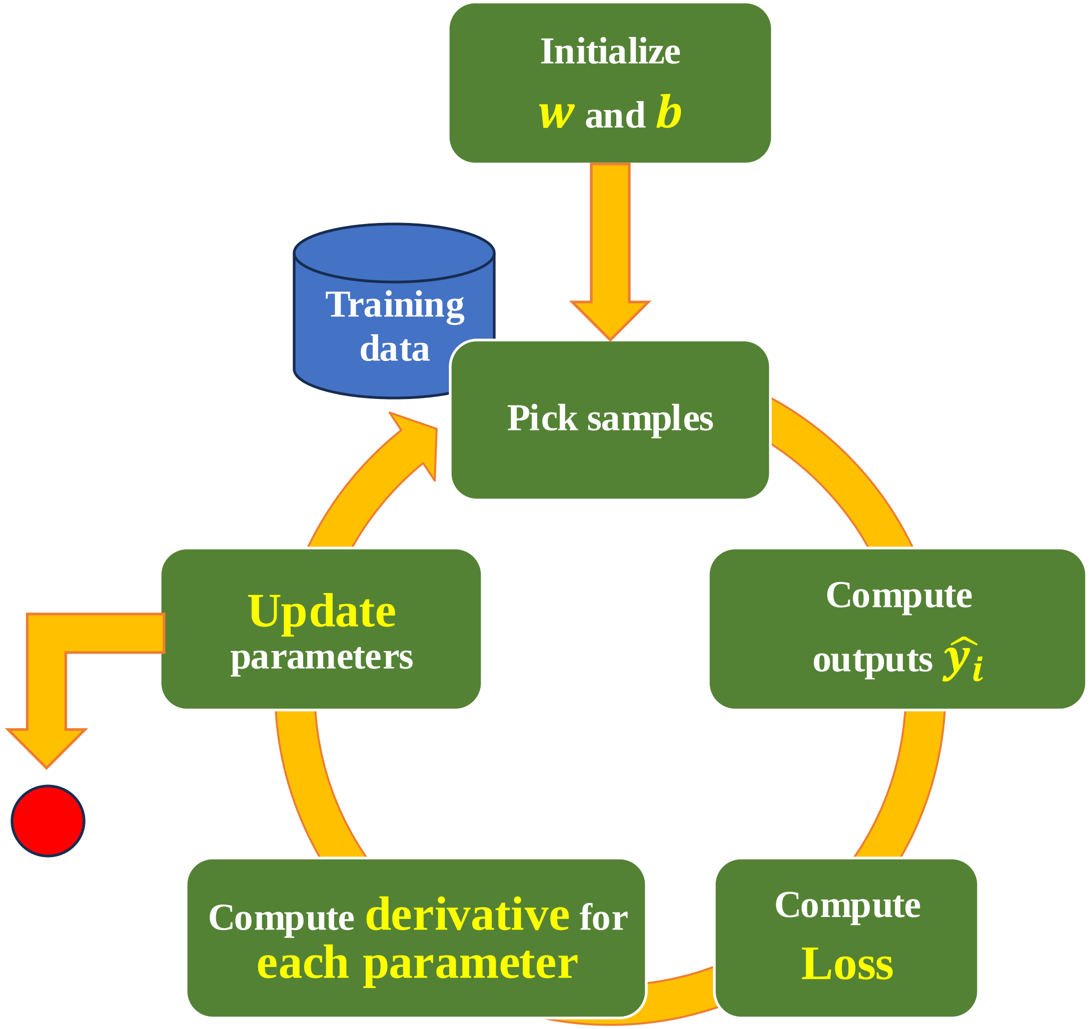




Simple Linear Regression

❖ Update gradient: Proof



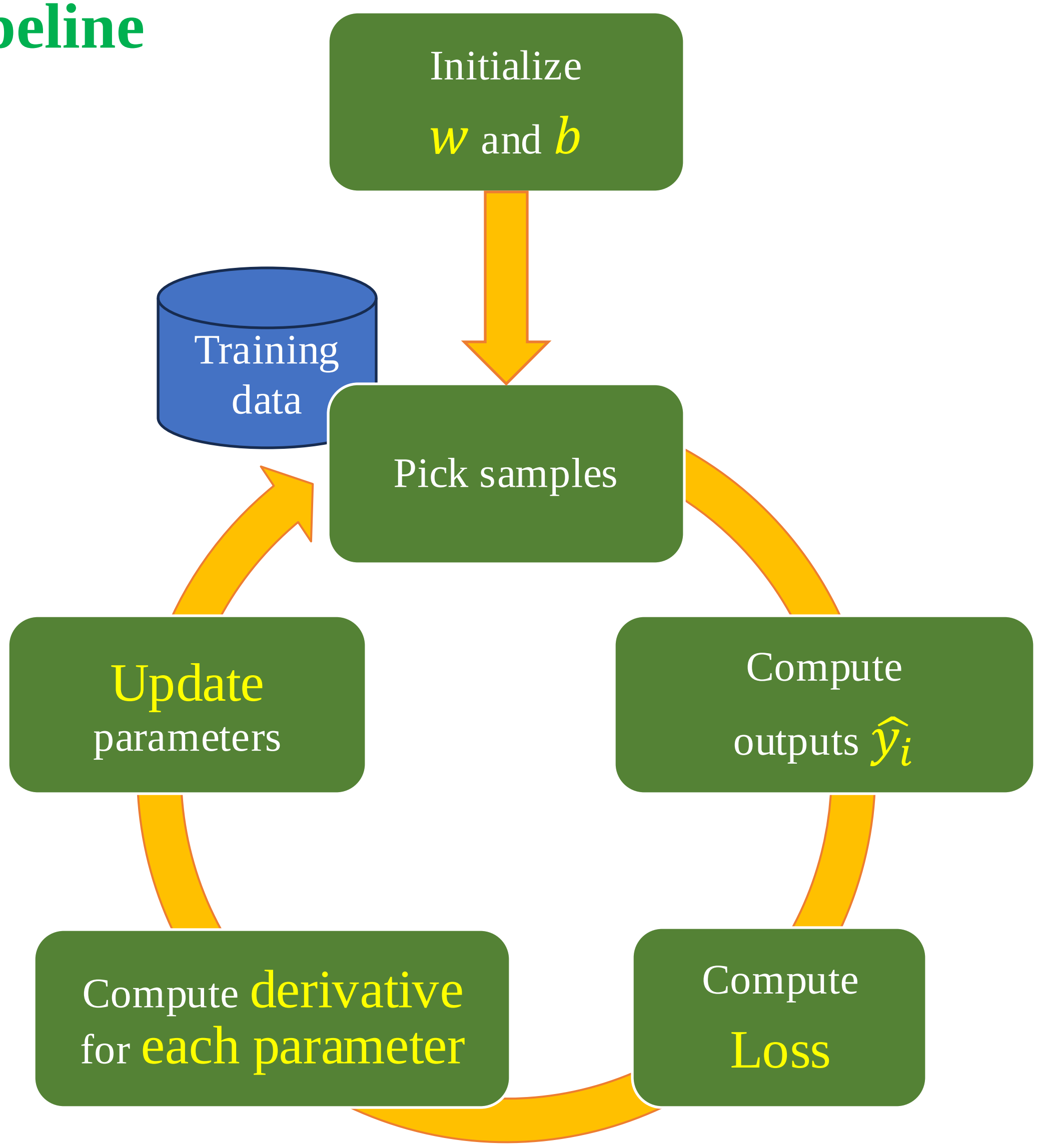


QUIZ

Code Implementation

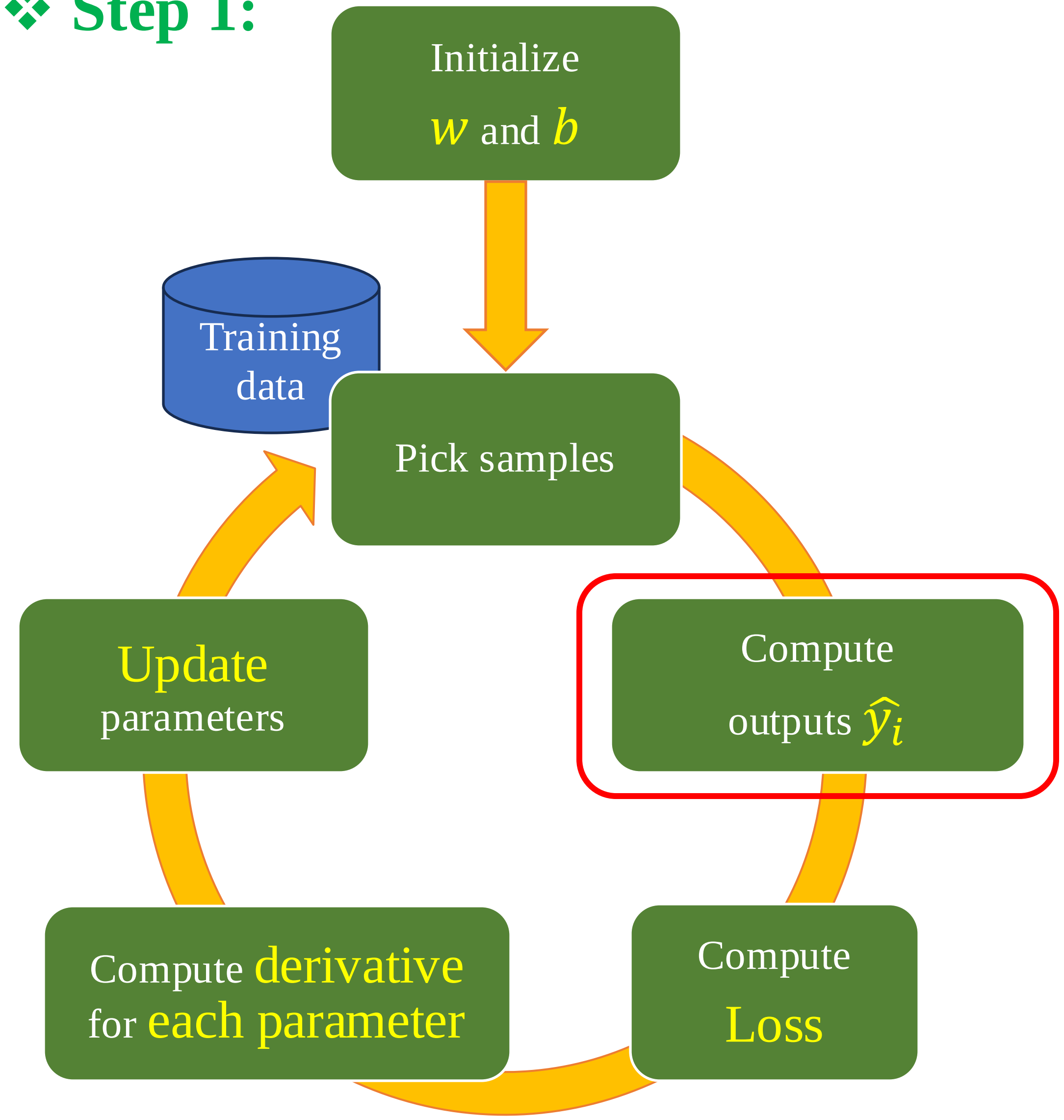
Code Implementation

❖ Gradient Descent Pipeline



Code Implementation

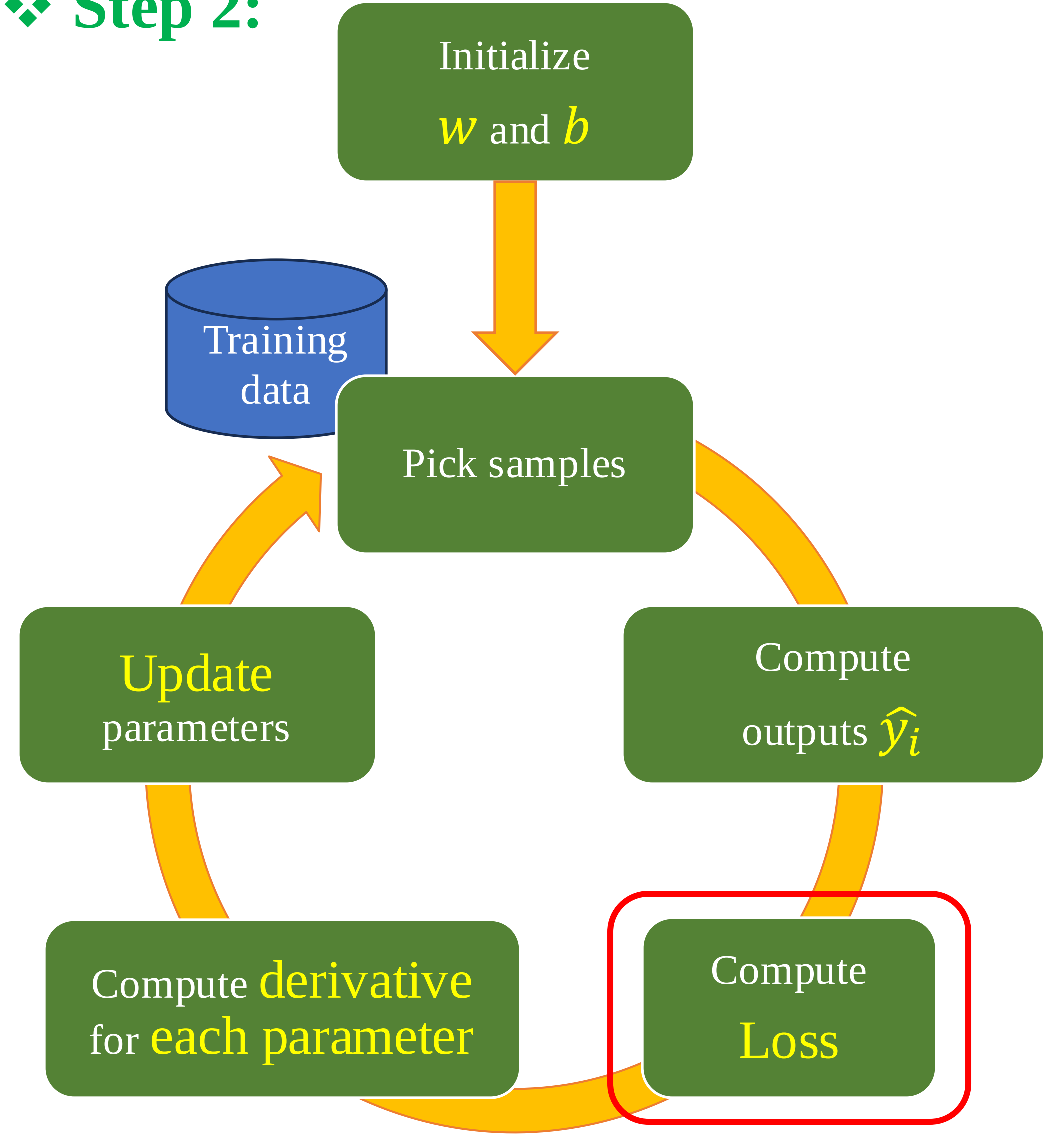
❖ Step 1:




```python
def predict(x, w, b):
    return x * w + b

def compute_loss(y_hat, y):
    return (y_hat - y) ** 2

def compute_gradient(y_hat, y, x):
    dw = 2 * x * (y_hat - y)
    db = 2 * (y_hat - y)

    return dw, db

def update_weight(w, b, dw, db, lr):
    new_w = w - lr * dw
    new_b = b - lr * db

    return new_w, new_b
```

Code Implementation

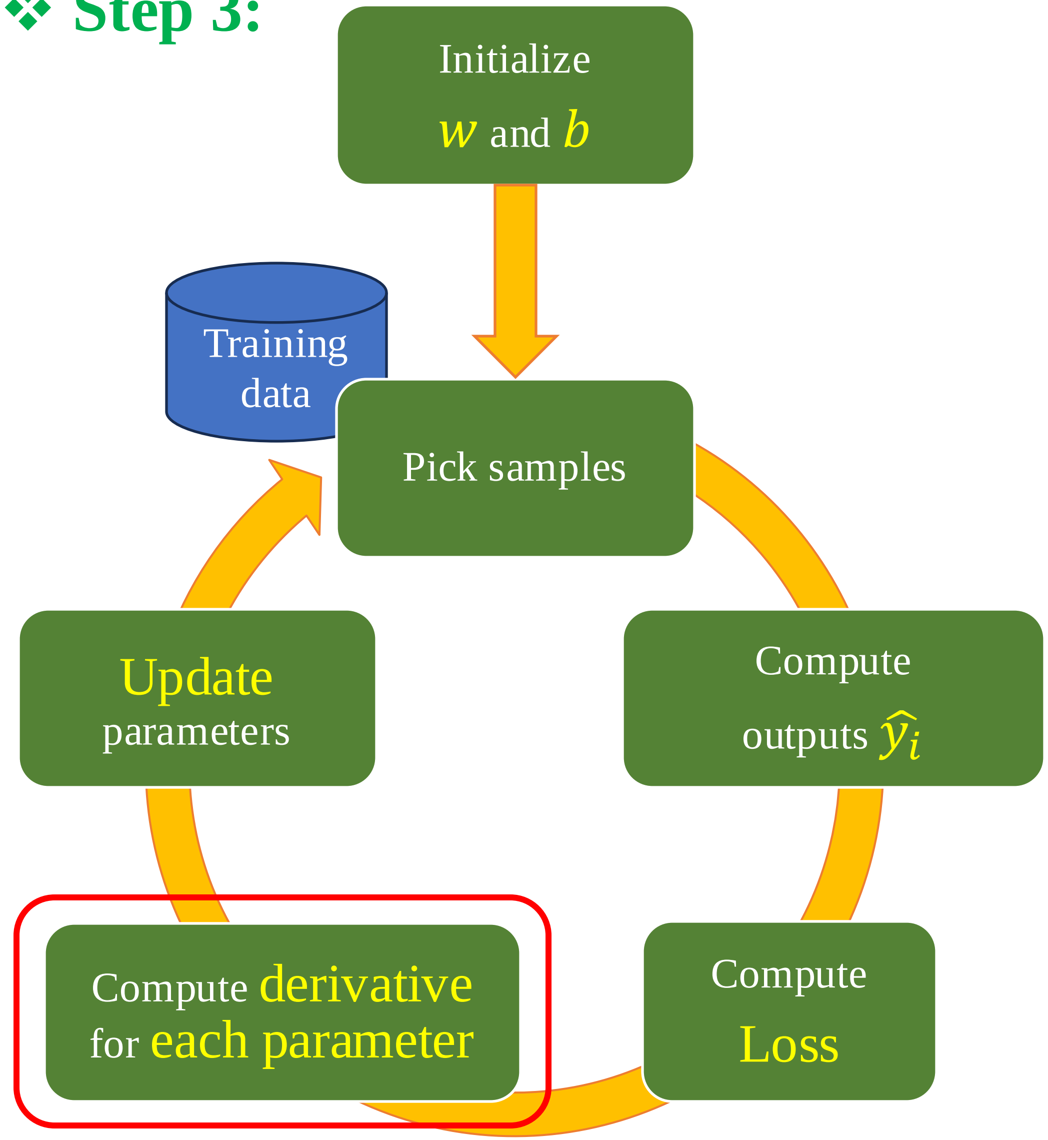
❖ **Step 2:**




```python
def predict(x, w, b):
    return x * w + b

def compute_loss(y_hat, y):
    return (y_hat - y) ** 2

def compute_gradient(y_hat, y, x):
    dw = 2 * x * (y_hat - y)
    db = 2 * (y_hat - y)

    return dw, db

def update_weight(w, b, dw, db, lr):
    new_w = w - lr * dw
    new_b = b - lr * db

    return new_w, new_b
```

Code Implementation

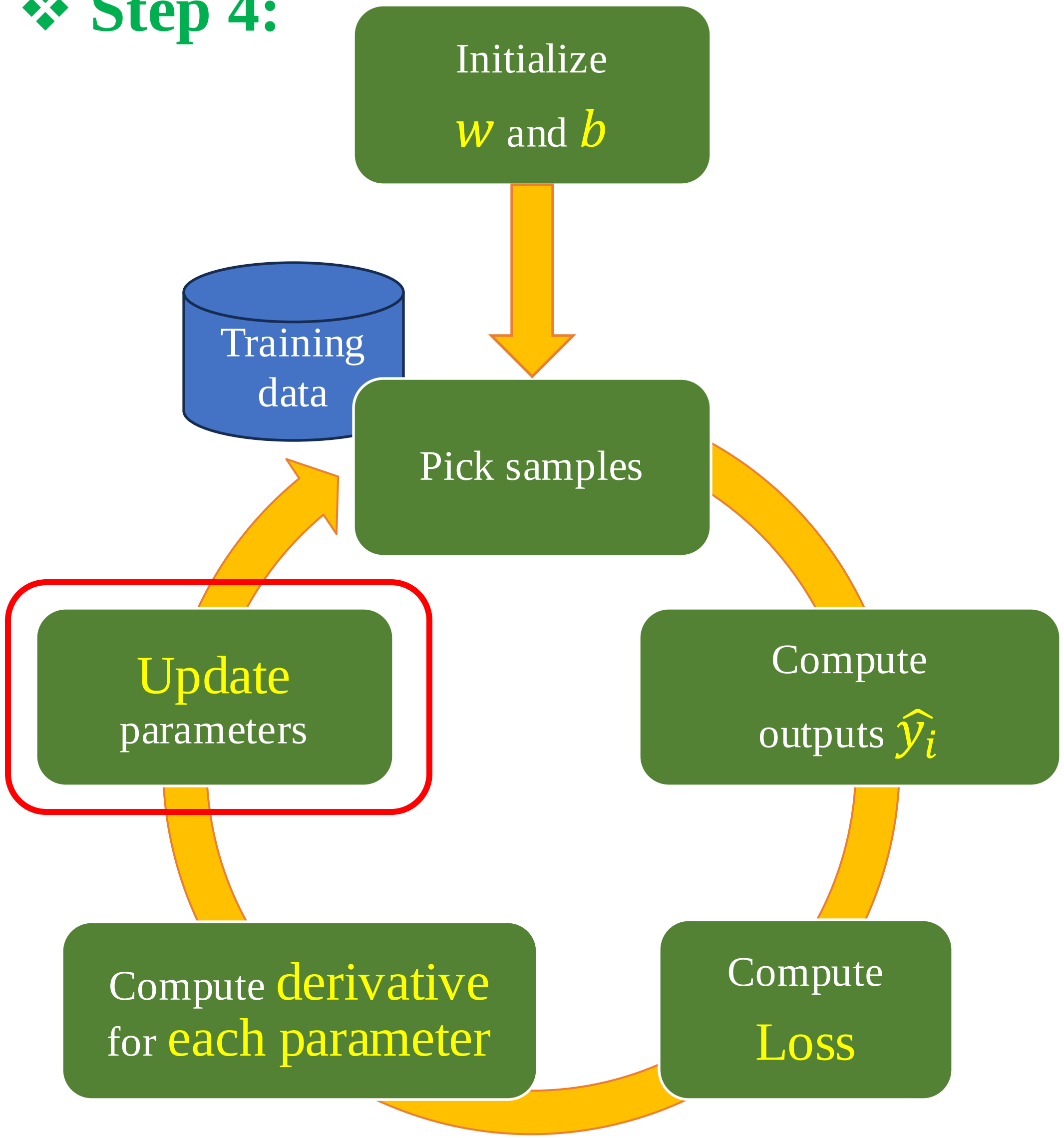
❖ Step 3:




```python
def predict(x, w, b):
    return x * w + b

def compute_loss(y_hat, y):
    return (y_hat - y) ** 2

def compute_gradient(y_hat, y, x):
    dw = 2 * x * (y_hat - y)
    db = 2 * (y_hat - y)

    return dw, db

def update_weight(w, b, dw, db, lr):
    new_w = w - lr * dw
    new_b = b - lr * db

    return new_w, new_b
```

Code Implementation

❖ Step 4:




```python
def predict(x, w, b):
    return x * w + b

def compute_loss(y_hat, y):
    return (y_hat - y) ** 2

def compute_gradient(y_hat, y, x):
    dw = 2 * x * (y_hat - y)
    db = 2 * (y_hat - y)

    return dw, db

def update_weight(w, b, dw, db, lr):
    new_w = w - lr * dw
    new_b = b - lr * db

    return new_w, new_b
```

Code Implementation

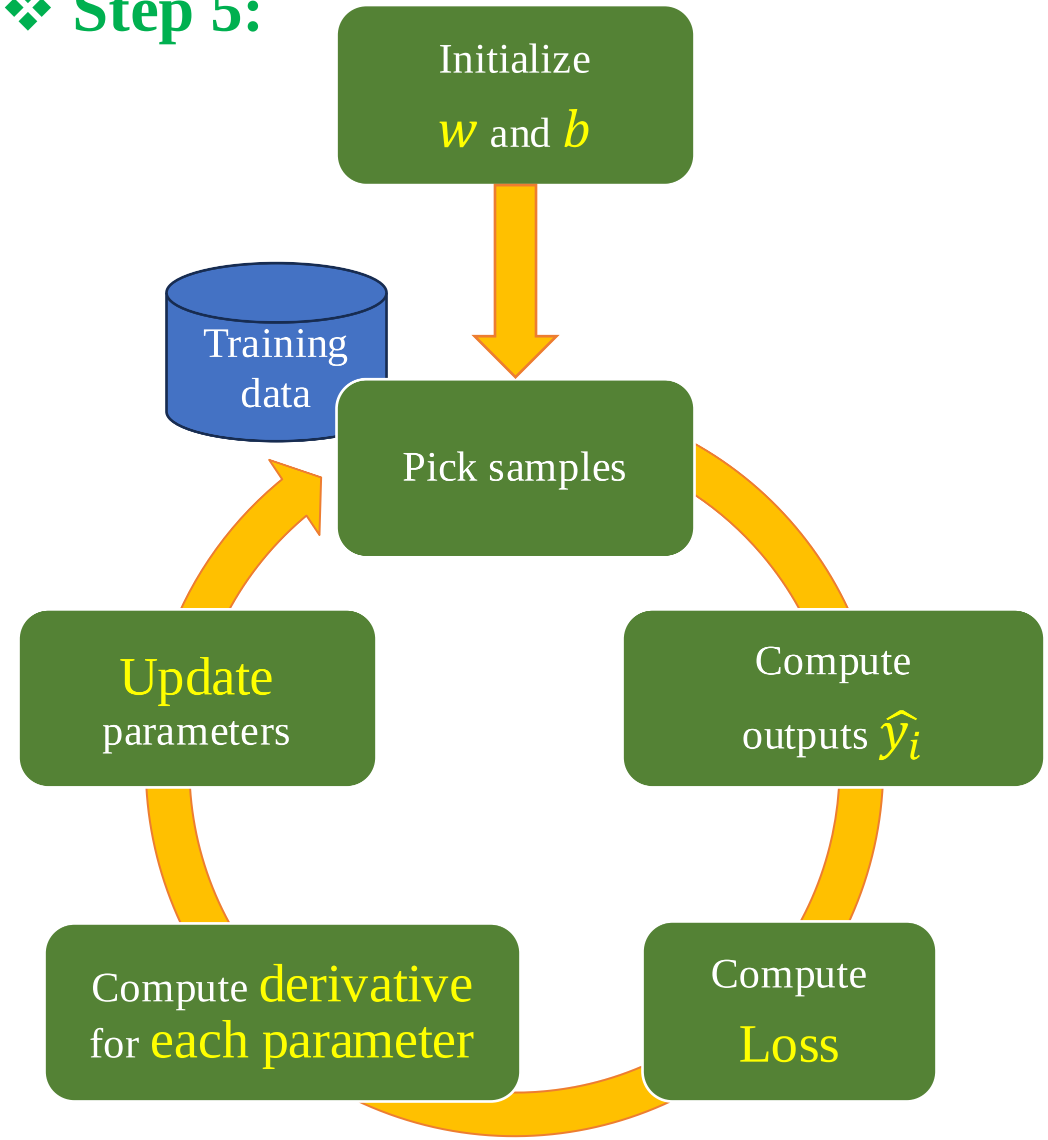
❖ Test functions

```python
x = 5
y = 66

b = 5
w = 10
lr = 0.01

y_hat = predict(x, w, b)
print(f'Prediction: {y_hat}')

loss = compute_loss(y_hat, y)
print(f'Loss: {loss}')

dw, db = compute_gradient(y_hat, y, x)
print(f'Gradient: dw = {dw}, db = {db}')

new_w, new_b = update_weight(w, b, dw, db, lr)
print(f'Updated weight: w = {new_w}, b = {new_b}')
```

```
Prediction: 55
Loss: 121
Gradient: dw = -110, db = -22
Updated weight: w = 11.1, b = 5.22
```

Code Implementation

❖ Step 5:




```python
X = [3, 4, 5, 6]
y = [60, 55, 66, 93]

epochs = 5

b = 5
w = 10
lr = 0.001

for epoch in range(epochs):
    for idx in range(len(X)):
        x = X[idx]
        y_hat = predict(x, w, b)

        loss = compute_loss(y_hat, y[idx])
        dw, db = compute_gradient(y_hat, y[idx], x)
        w, b = update_weight(w, b, dw, db, lr)

        print(f'Epoch: {epoch}, Loss: {loss}, w: {w}, b: {b}')
```

Code Implementation

❖ Test functions

```python
X = [3, 4, 5, 6]
y = [60, 55, 66, 93]

epochs = 5

b = 5
w = 10
lr = 0.001

for epoch in range(epochs):
    for idx in range(len(X)):
        x = X[idx]
        y_hat = predict(x, w, b)

        loss = compute_loss(y_hat, y[idx])
        dw, db = compute_gradient(y_hat, y[idx], x)
        w, b = update_weight(w, b, dw, db, lr)

        print(f'Epoch: {epoch}, Loss: {loss}, w: {w}, b: {b}')
```

```
Epoch: 0, Loss: 625, w: 10.15, b: 5.05
Epoch: 0, Loss: 87.42250000000003, w: 10.2248, b: 5.0687
Epoch: 0, Loss: 96.18313328999996, w: 10.322873, b: 5.0883145999999995
Epoch: 0, Loss: 674.671917735367, w: 10.6345663688, b: 5.140263494799999
Epoch: 1, Loss: 526.9796530551046, w: 10.7723025931928, b: 5.186175569597599
Epoch: 1, Loss: 45.22043422409116, w: 10.826099505653849, b: 5.199624797712861
Epoch: 1, Loss: 44.48726818636239, w: 10.892798282394027, b: 5.212964553060897
Epoch: 1, Loss: 503.1159245209058, w: 11.161961231424927, b: 5.257825044566047
Epoch: 2, Loss: 451.8299181792317, w: 11.289498978991881, b: 5.300337627088366
Epoch: 2, Loss: 20.626734206131236, w: 11.325832310647435, b: 5.309420960002254
Epoch: 2, Loss: 16.495112001764532, w: 11.366446485515041, b: 5.317543794975775
Epoch: 2, Loss: 379.617577561682, w: 11.600251813018248, b: 5.356511349559643
Epoch: 3, Loss: 393.73406129822536, w: 11.719308212286561, b: 5.396196815982414
Epoch: 3, Loss: 7.434185791000412, w: 11.741120774965532, b: 5.401649956521565
Epoch: 3, Loss: 3.5824880584478613, w: 11.760048236650734, b: 5.405435448989197
Epoch: 3, Loss: 290.1665292422299, w: 11.96445953822401, b: 5.43950399925141
Epoch: 4, Loss: 348.46127150556185, w: 12.07646224254047, b: 5.476838234023563
Epoch: 4, Loss: 1.4818504428538535, w: 12.086200744906987, b: 5.479272859615192
Epoch: 4, Loss: 0.008050291351769556, w: 12.087097979065486, b: 5.479452306446892
Epoch: 4, Loss: 224.93879873714388, w: 12.267073496895408, b: 5.509448226085213
```

Code Implementation

❖ Before and after training

Before training:

```python
x = 5
y = 66

y_hat = predict(x, w, b)
print(f'Prediction: {y_hat}')

loss = compute_loss(y_hat, y)
print(f'Loss: {loss}')

print(f'w: {w}, b: {b}')
```

```
Prediction: 55
Loss: 121
w: 10, b: 5
```

After training:

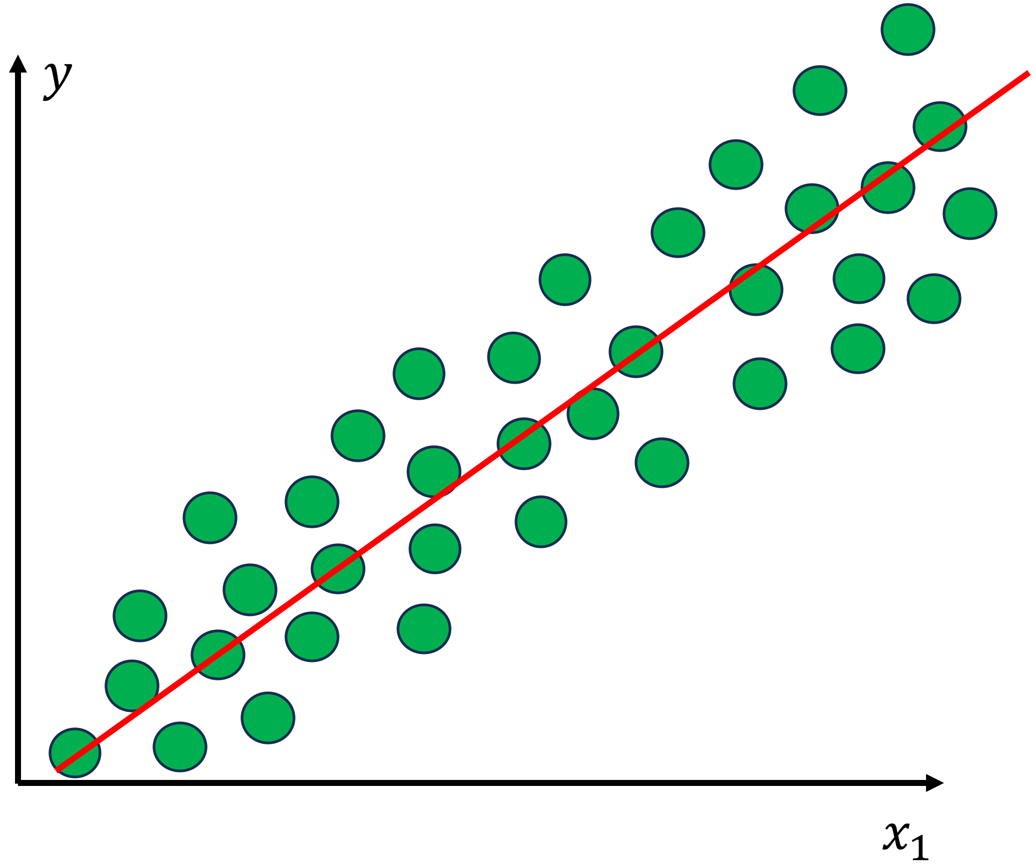
```python
x = 5
y = 66

y_hat = predict(x, w, b)
print(f'Prediction: {y_hat}')

loss = compute_loss(y_hat, y)
print(f'Loss: {loss}')

print(f'w: {w}, b: {b}')
```

```
Prediction: 66.84481571056226
Loss: 0.713713584812813 7
w: 12.267073496895408, b: 5.509448226085213
```

Summarization and QA

Summarization

❖ Summarization



In this lecture, we have discussed:

1. Introduction to Linear Regression.
2. Simple
3. Use SVM to solve a classification and a regression tasks.

Question

?


Thank you!