

# Lecture 2

## Linear Regression

Dong Kook Kim

# Acknowledgement

## 모두를 위한 머신러닝/딥러닝 강의

### 모두를 위한 머신러닝과 딥러닝의 강의

알파고와 이세돌의 경기를 보면서 이제 머신 러닝이 인간이 잘 한다고 여겨진 직관과 의사 결정능력에서도 충분한 데이터가 있으면 어느정도 또는 우리보다 더 잘할수도 있다는 생각을 많이 하게 되었습니다. Andrew Ng 교수님이 말씀하신것 처럼 이런 시대에 머신 러닝을 잘 이해하고 잘 다룰수 있다면 그야말로 "Super Power"를 가지게 되는 것이 아닌가 생각합니다.

더 많은 분들이 머신 러닝과 딥러닝에 대해 더 이해하고 본인들의 문제를 이 멋진 도구들 이용해서 풀수 있게 하기위해 비디오 강의를 준비하였습니다. 더 나아가 이론에만 그치지 않고 최근 구글이 공개한 머신러닝을 위한 오픈소스인 TensorFlow를 이용해서 이론을 구현해 볼수 있도록 하였습니다.

수학이나 컴퓨터 공학적인 지식이 없이도 쉽게 볼수 있도록 만들려고 노력하였습니다.



시즌 RL - Deep Reinforcement Learning

<https://hunkim.github.io/ml/>

# Linear Models : Summary

Supervised Learning	Input $x$	Target $y$	Distribution for Target	Classifier $p(y x)$	Linear models	Loss function
Regression	vector binary integer real	real	Gaussian	Gaussian	Linear regression	MSE
Binary Classification		binary	Bernoulli	sigmoid	Logistic classification	Binary Cross-entropy
Multi-class Classification		integer	multinoulli	softmax	Softmax classification	Multi-class Cross-entropy

# Predicting exam score: regression

x (hours)	y (score)
10	90
9	80
3	50
2	30

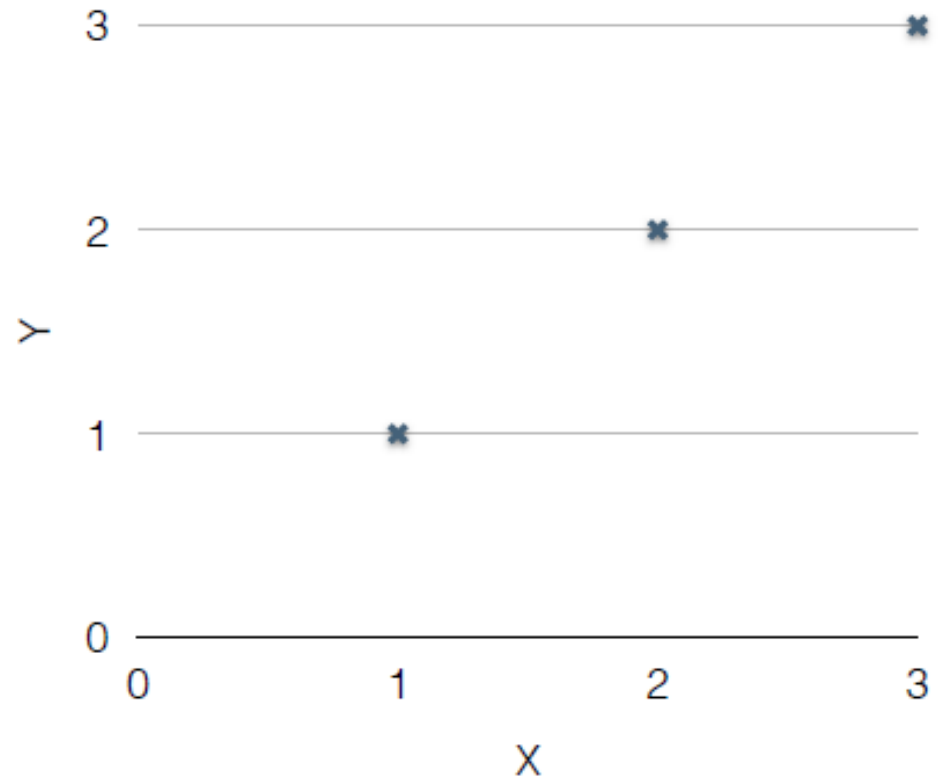
# Regression : Step 1. data

- input (real)   - target (real)

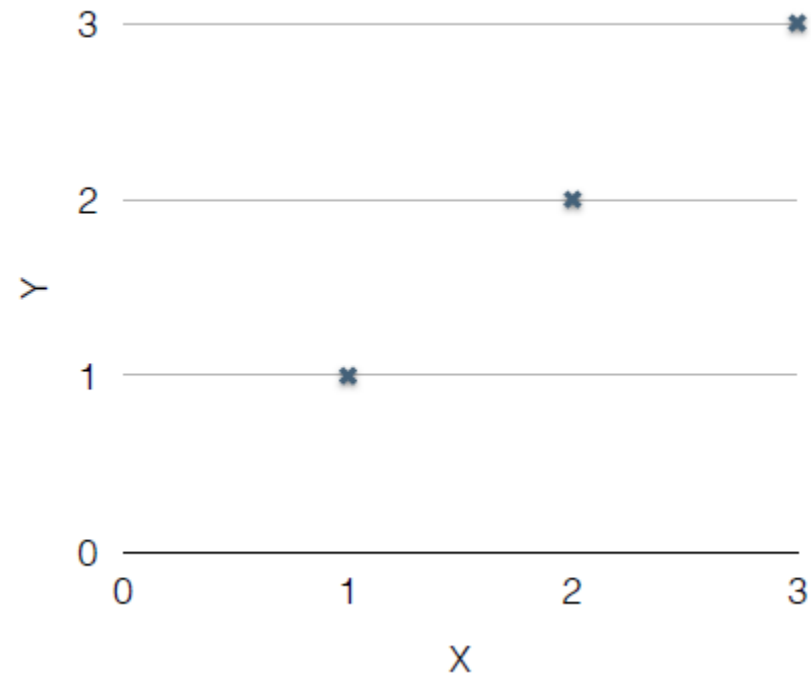
x	y
1	1
2	2
3	3

# Regression (Graph)

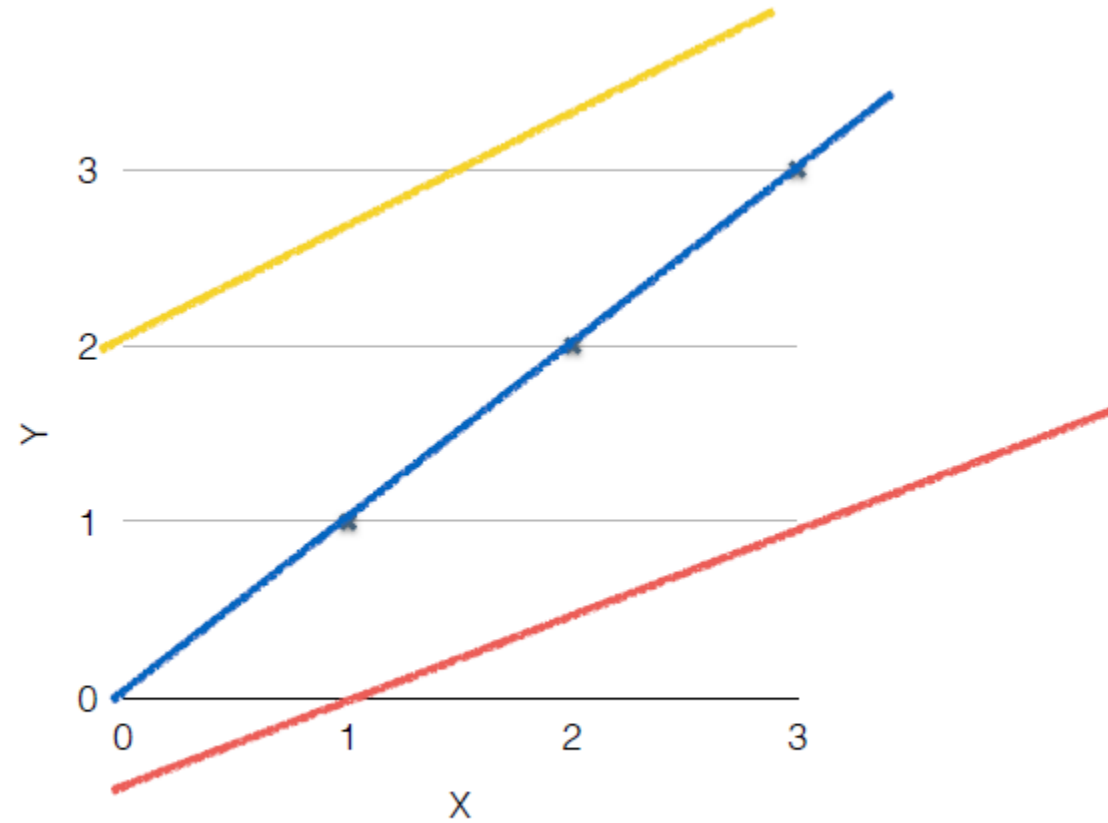
X	Y
1	1
2	2
3	3



# Linear Model (or Hypothesis)



## Step 2. Linear Models



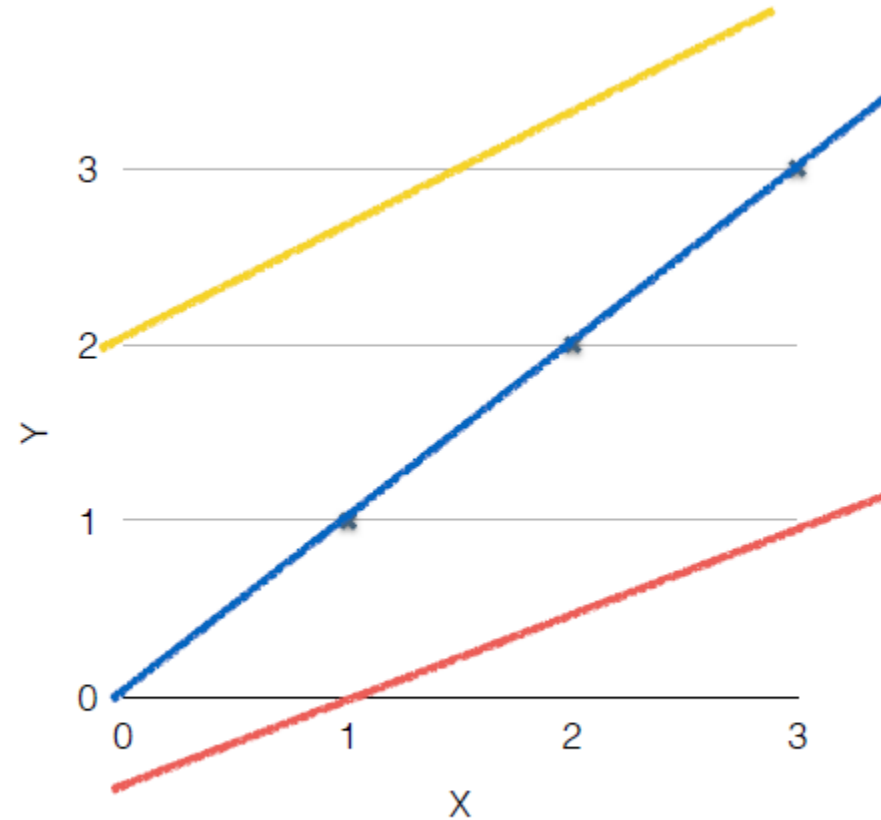


## Step 2. Linear Models

$$H(x) = Wx + b$$

$W, b$  : model parameters

$W$  : weight  
 $b$  : bias

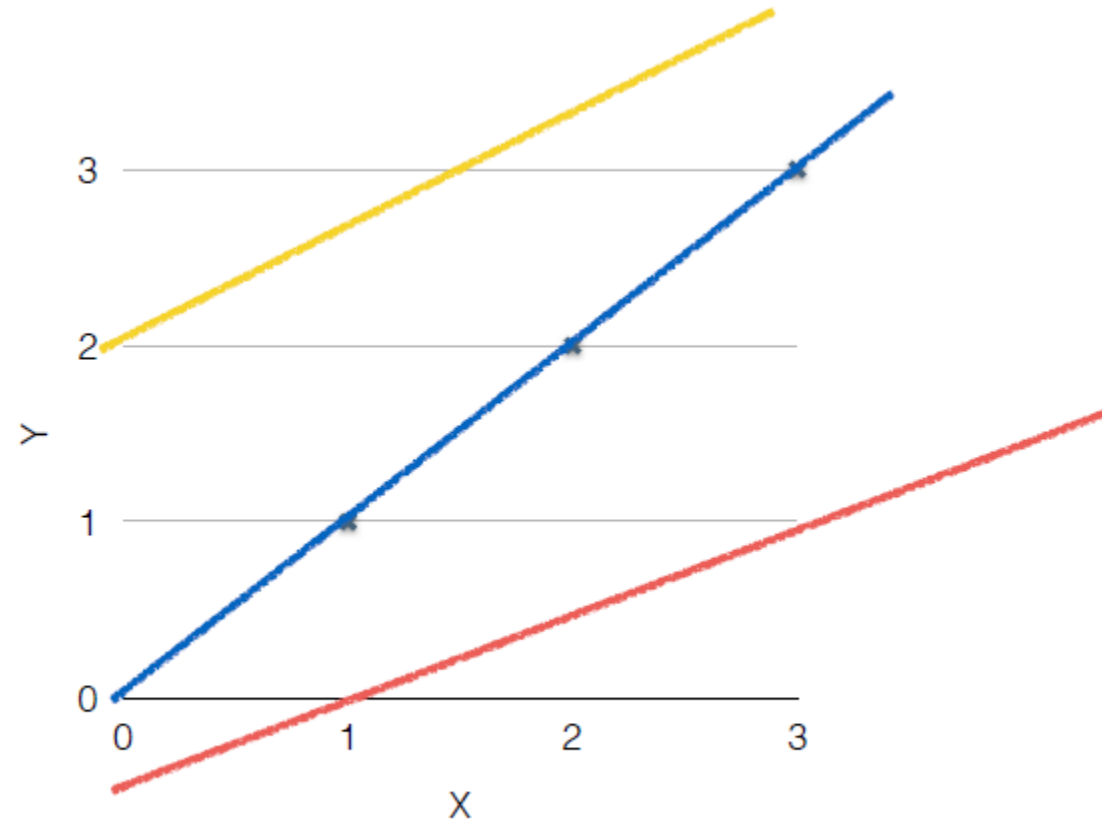


$W1, b1$

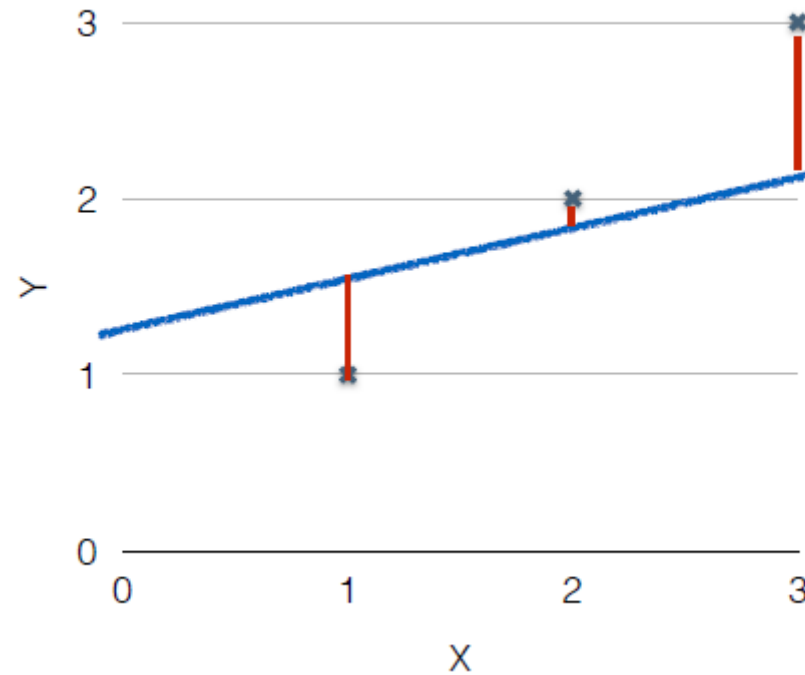
$W2, b2$

$W3, b3$

# Which hypothesis is better



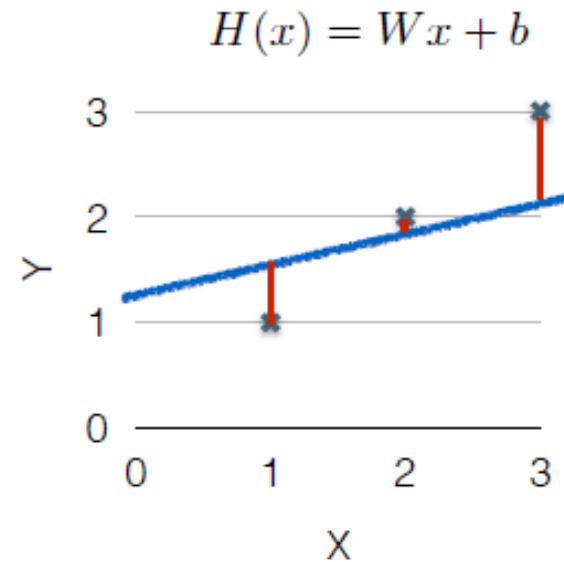
# Which hypothesis is better : Error



## Step 3. Loss function

- How fit the line to our (training) data

error:  $H(x) - y$



# Step 3. Loss function

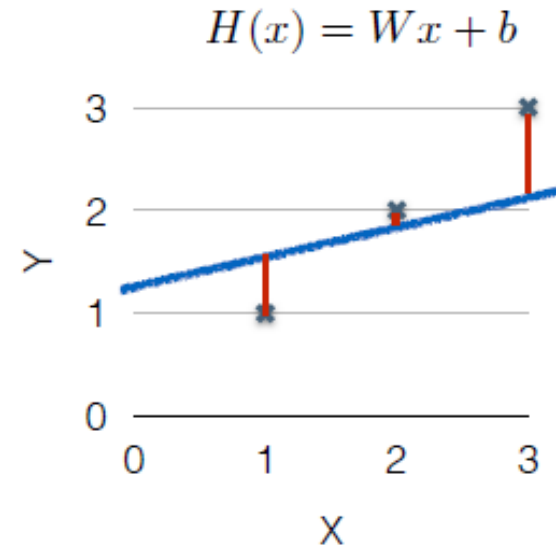
- How fit the line to our (training) data

$$\frac{(H(x^{(1)}) - y^{(1)})^2 + (H(x^{(2)}) - y^{(2)})^2 + (H(x^{(3)}) - y^{(3)})^2}{3}$$

>

$$cost = \frac{1}{m} \sum_{i=1}^m (H(x^{(i)}) - y^{(i)})^2$$

- MSE (mean square error)



## Step 3. Loss function : MSE

$$cost = \frac{1}{m} \sum_{i=1}^m (H(x^{(i)}) - y^{(i)})^2$$

$$H(x) = Wx + b$$

$$cost(W, b) = \frac{1}{m} \sum_{i=1}^m (H(x^{(i)}) - y^{(i)})^2$$

- Cost function is a function of model parameters,  $W, b$

# Hypothesis and Loss

$$H(x) = Wx + b$$

$$cost(W, b) = \frac{1}{m} \sum_{i=1}^m (H(x^{(i)}) - y^{(i)})^2$$

# Simplified hypothesis

$$H(x) = Wx$$

$$\text{cost}(W) = \frac{1}{m} \sum_{i=1}^m (Wx^{(i)} - y^{(i)})^2$$



# What $cost(W)$ looks like?

$$cost(W) = \frac{1}{m} \sum_{i=1}^m (W x^{(i)} - y^{(i)})^2$$

x	Y
1	1
2	2
3	3

- $W=1, cost(W)=?$

# What $\text{cost}(W)$ looks like?

$$\text{cost}(W) = \frac{1}{m} \sum_{i=1}^m (W x^{(i)} - y^{(i)})^2$$

x	Y
1	1
2	2
3	3

- $W=1, \text{cost}(W)=0$

$$\frac{1}{3}((1 * 1 - 1)^2 + (1 * 2 - 2)^2 + (1 * 3 - 3)^2)$$

- $W=0, \text{cost}(W)=4.67$

$$\frac{1}{3}((0 * 1 - 1)^2 + (0 * 2 - 2)^2 + (0 * 3 - 3)^2)$$

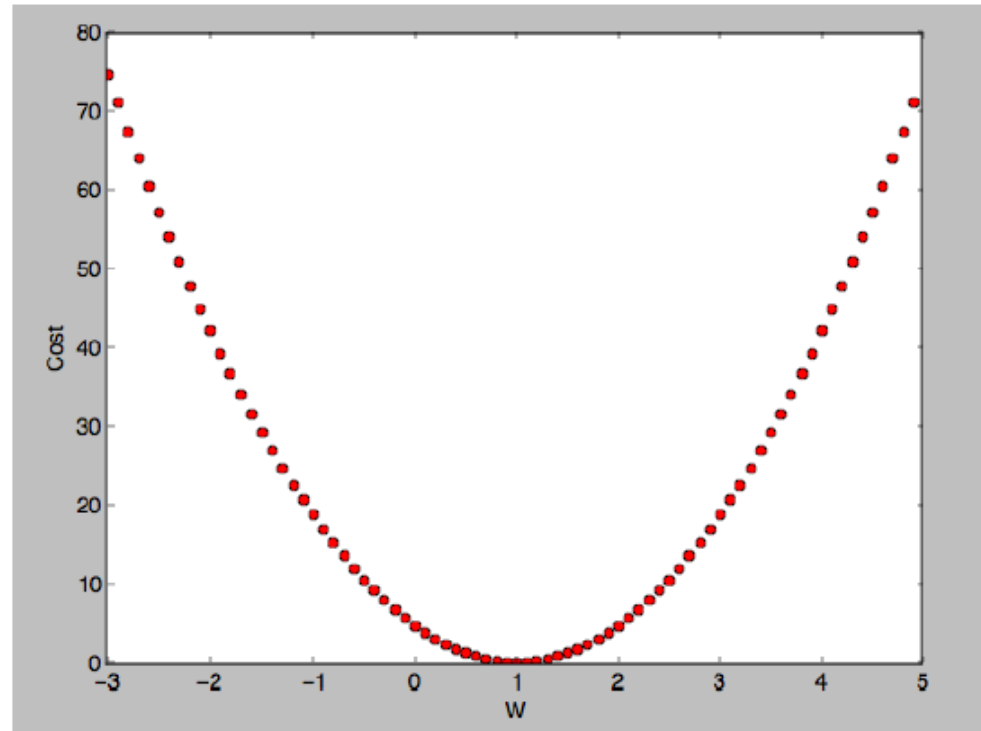
- $W=2, \text{cost}(W)=?$

# What $\text{cost}(W)$ looks like?

- $W=1, \text{cost}(W)=0$
- $W=0, \text{cost}(W)=4.67$
- $W=2, \text{cost}(W)=4.67$

# What $\text{cost}(W)$ looks like?

$$\text{cost}(W) = \frac{1}{m} \sum_{i=1}^m (Wx^{(i)} - y^{(i)})^2$$



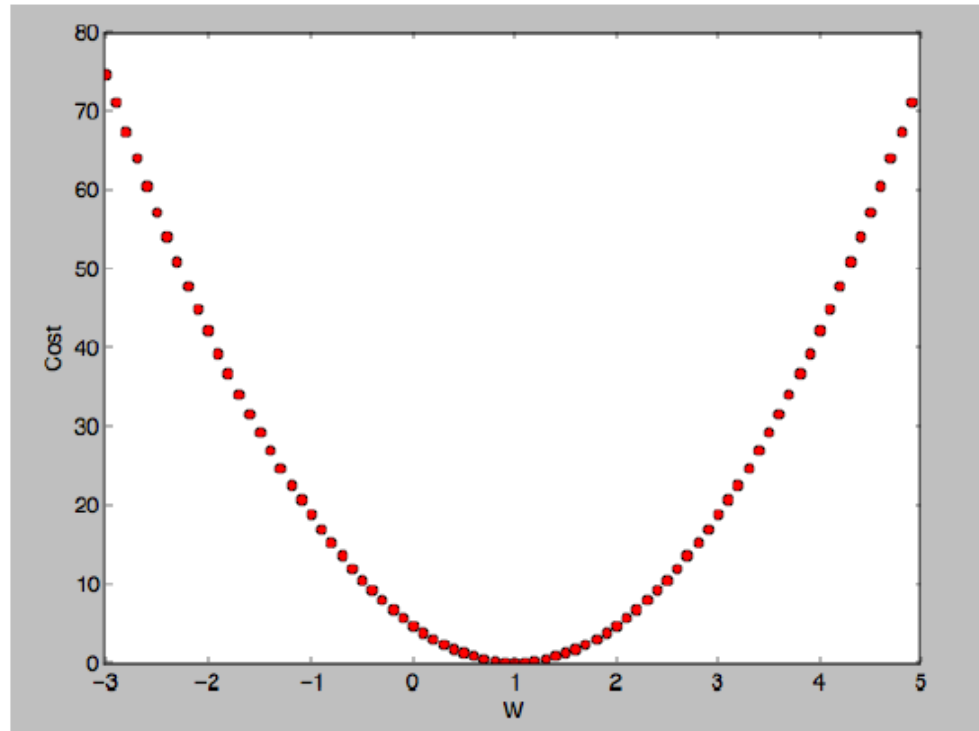
Goal : Minimize Loss

$$\underset{W, b}{\text{minimize}} \text{cost}(W, b)$$

## Step 4. Optimization

- How to minimize loss ?

$$\text{cost}(W) = \frac{1}{m} \sum_{i=1}^m (Wx^{(i)} - y^{(i)})^2$$



# Step 4. Optimization

- **Optimization** : minimizing the loss function  $J(w)$  parameterized by the model's parameters  $w$
- Optimization Goals
  - Find the **global minimum** of the loss function. This is feasible if the objective function is **convex**, i.e. any local minimum is a global minimum.
  - Find the **lowest possible value** of the objective function within its neighborhood. That's usually the case if the objective function is **not convex** as the case in most deep learning problems.

# Step 4. Optimization

- 3 Optimization Algorithms

1. **Not iterative** and simply solves for one point (closed-form solution)
2. **Iterative** and converges to global minimum regardless of the parameters init.
  - gradient descent applied to logistic regression.
3. **Iterative** and applied to a set of problems that have **non-convex** loss functions such as NNs
  - parameters' **initialization** plays a critical role in speeding up convergence and achieving lower error rates.



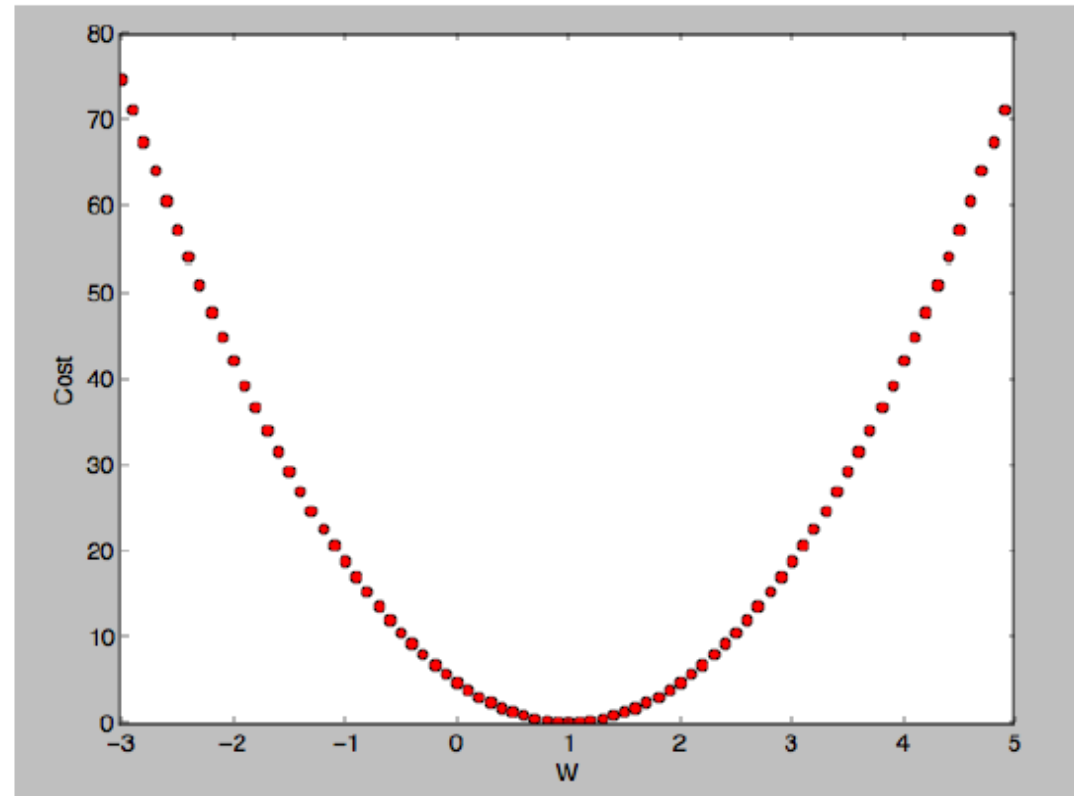
# Step 4. Optimization

## - Gradient descent algorithm

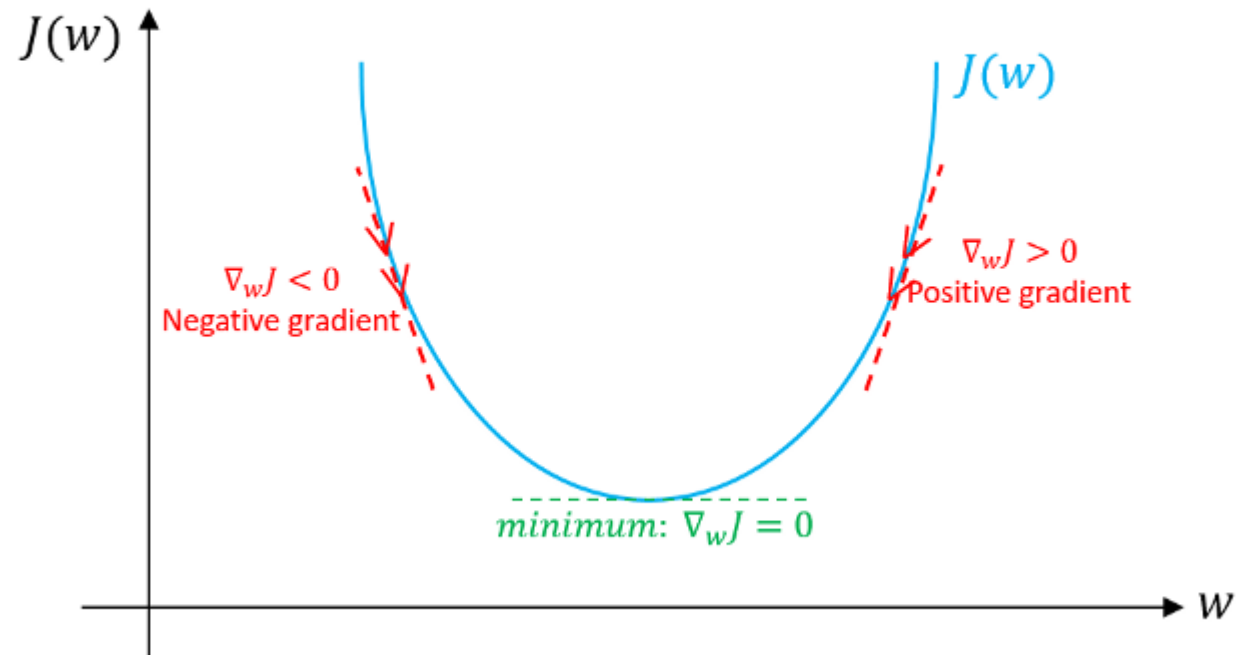
- Most common optimization algorithm in *ML* and *DL*
- First-order optimization algorithm – use the **first derivative** when performing the updates on the parameters.
- Minimize loss function – given  $J(W, b)$ , find  $W, b$  to minimize loss
- Update **the parameters in the opposite direction of the gradient** of the objective function  $J(W, b)$  w.r.t the parameters
- The size of the step we take on each iteration to reach the local minimum is determined by the **learning rate  $\alpha$**

# How it works?

How would you find the lowest point?

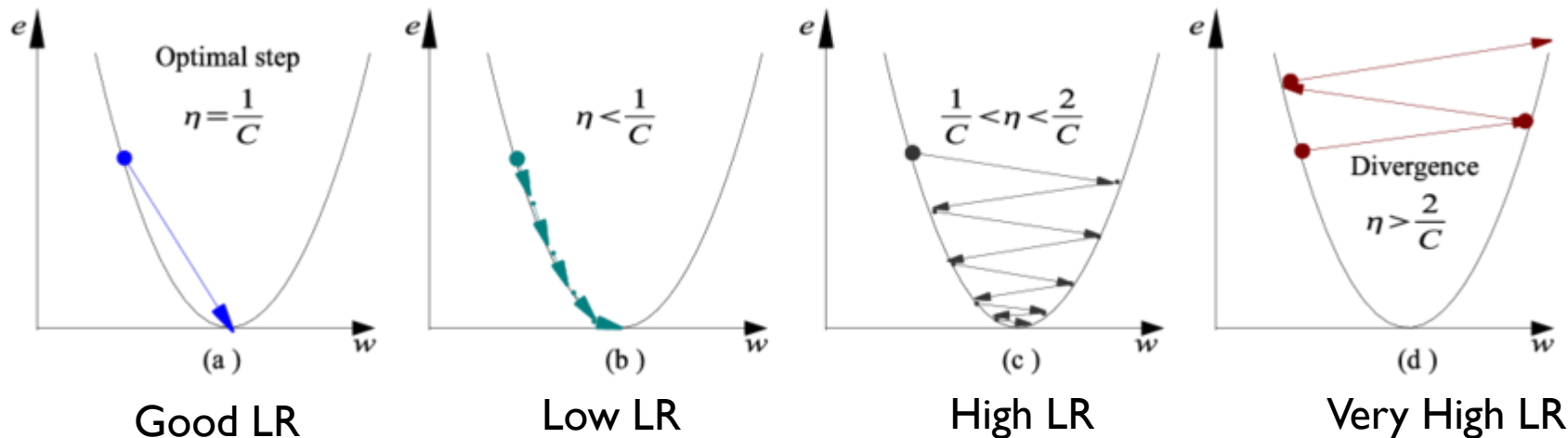


# How it works? : Gradient Descent



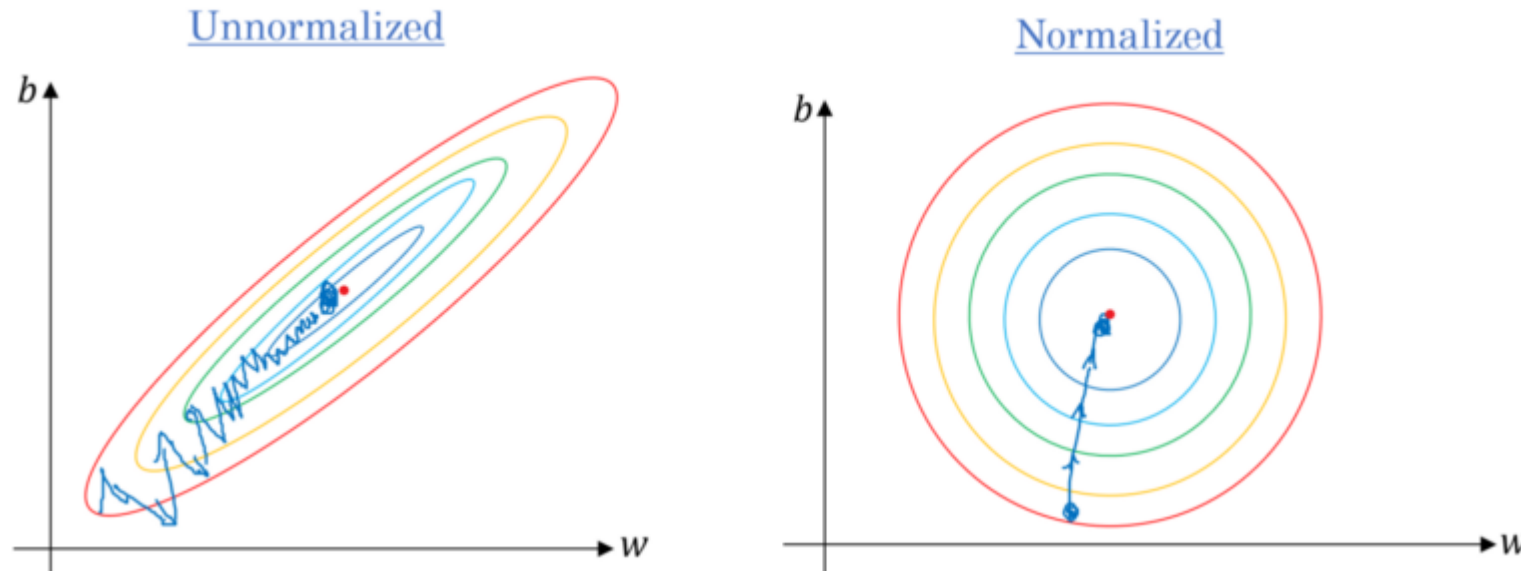
# How it works?

1. **Initialize** weight  $w$  and bias  $b$  to any random numbers.
2. Pick a value for **the learning rate  $\alpha$** . The learning rate determines how big the step would be on each iteration.  
(The most commonly used rates are :  $0.001, 0.003, 0.01, 0.03, 0.1$ )



# How it works?

3. Make sure to **scale the data** if it's on a very different scales. If we don't scale the data, the level curves (contours) would be narrower and taller which means it would take longer time to converge



# How it works?

4. On each iteration, take the partial derivative of the cost function  $J(w)$  w.r.t each parameter (gradient)

$$\frac{\partial}{\partial w} J(w) = \nabla_w J$$

$$\frac{\partial}{\partial b} J(w) = \nabla_b J$$

5. Update equations

$$w = w - \alpha \nabla_w J$$

$$b = b - \alpha \nabla_b J$$

# Gradient Descent

1. Batch GD : use all data
2. Mini-Batch GD : use batch size data
  - faster than batch version
3. Stochastic GD (SGD) : use each sample

# Formal definition

$$\text{cost}(W) = \frac{1}{m} \sum_{i=1}^m (Wx^{(i)} - y^{(i)})^2$$



$$\text{cost}(W) = \frac{1}{2m} \sum_{i=1}^m (Wx^{(i)} - y^{(i)})^2$$



# Formal definition

$$cost(W) = \frac{1}{2m} \sum_{i=1}^m (W x^{(i)} - y^{(i)})^2$$

$$W := W - \alpha \frac{\partial}{\partial W} cost(W)$$

# Formal definition

$$W := W - \alpha \frac{\partial}{\partial W} \frac{1}{2m} \sum_{i=1}^m (W x^{(i)} - y^{(i)})^2$$

$$W := W - \alpha \frac{1}{2m} \sum_{i=1}^m 2(W x^{(i)} - y^{(i)}) x^{(i)}$$

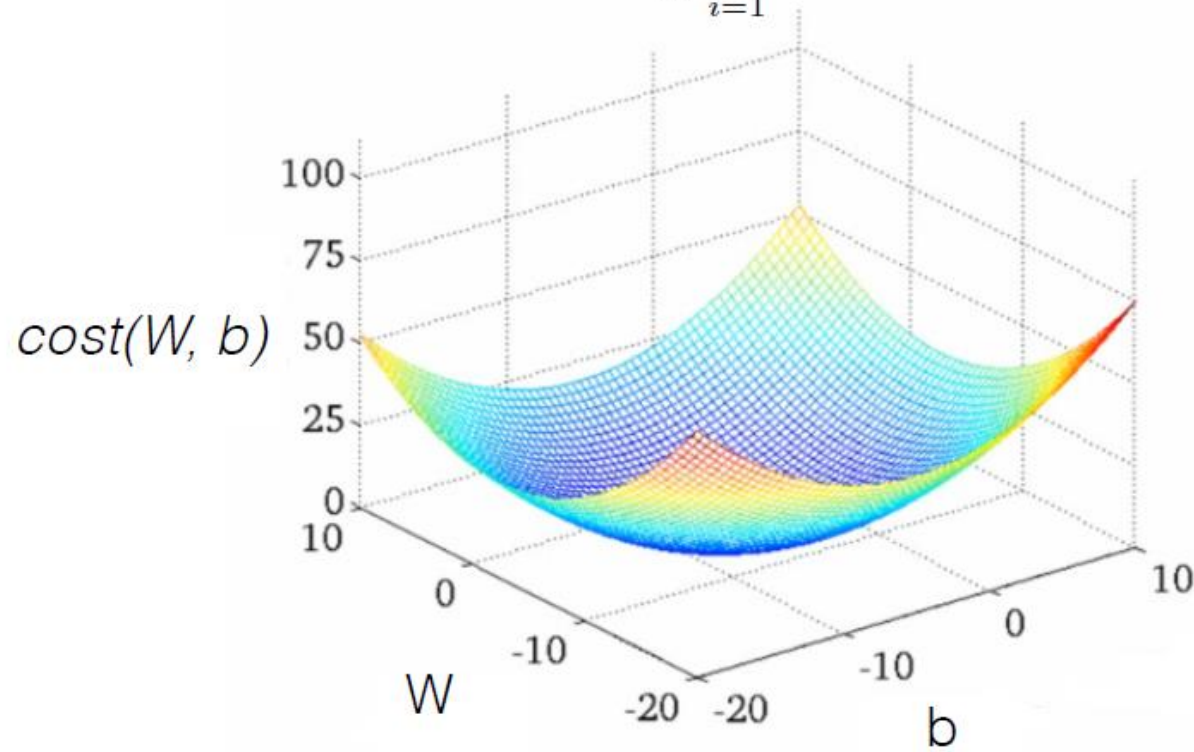
$$W := W - \alpha \frac{1}{m} \sum_{i=1}^m (W x^{(i)} - y^{(i)}) x^{(i)}$$

# Gradient descent algorithm

$$W := W - \alpha \frac{1}{m} \sum_{i=1}^m (W x^{(i)} - y^{(i)}) x^{(i)}$$

# Convex function

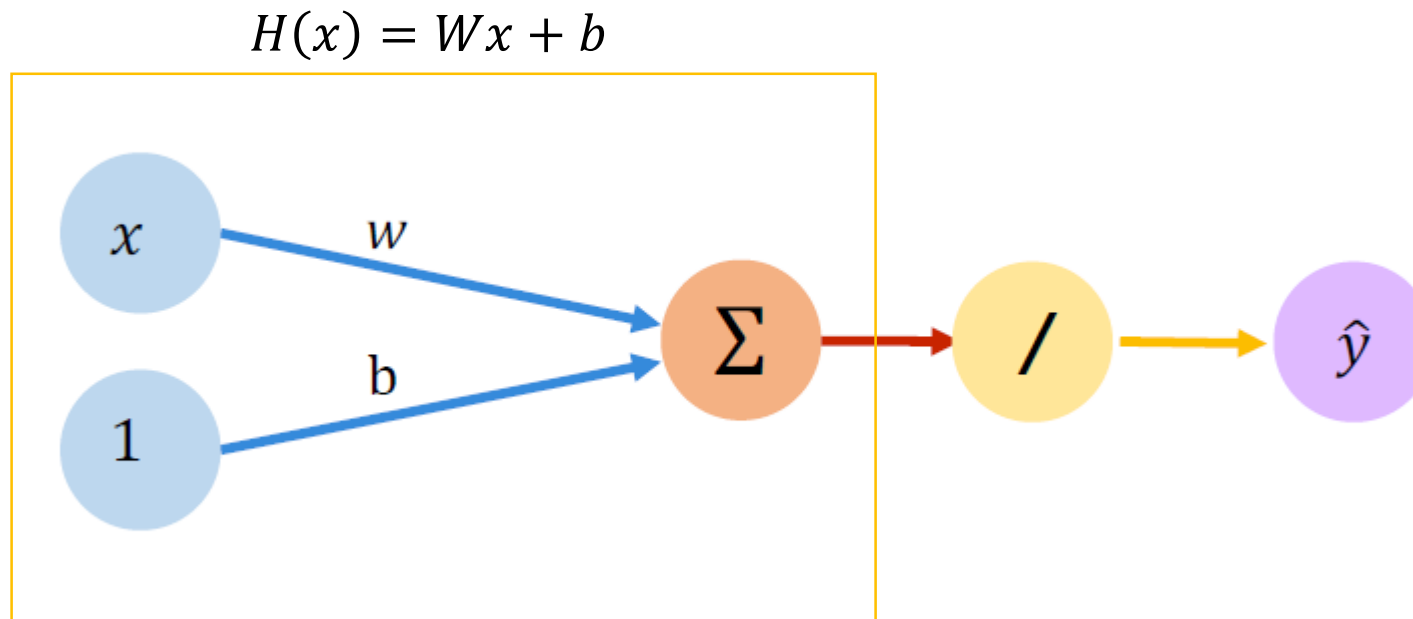
$$cost(W, b) = \frac{1}{m} \sum_{i=1}^m (H(x^{(i)}) - y^{(i)})^2$$



[www.holehouse.org/mlclass/](http://www.holehouse.org/mlclass/)

# Graph for Linear Regression

- Single input/single output : parameters  $W, b$



Keras : `Dense(1, input_dim=1)`

# (scalar) Linear Regression : Summary

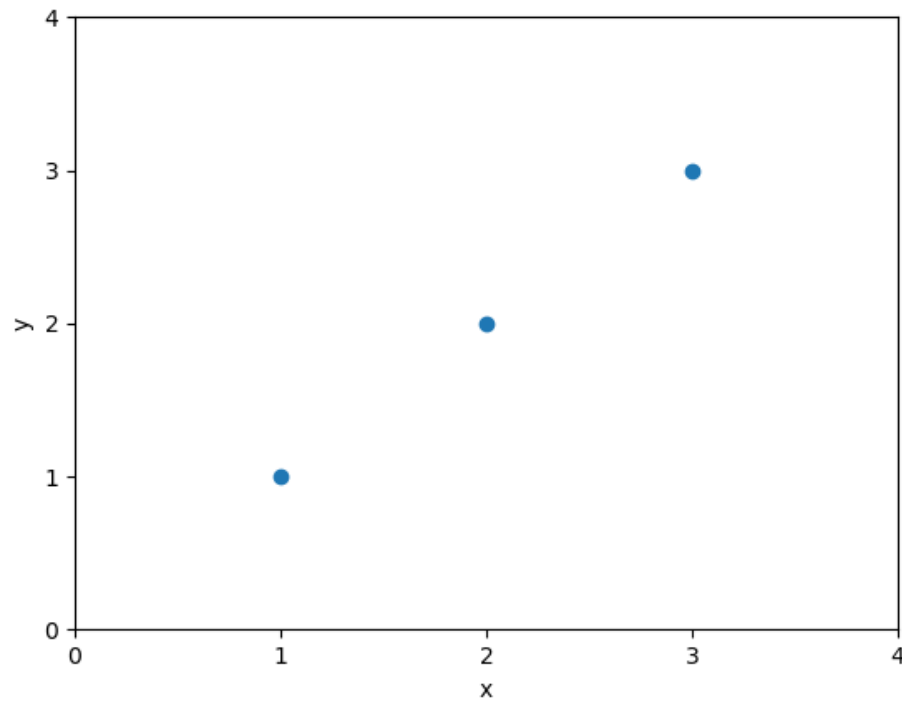
Data Set :	$x^{(i)}, y^{(i)}, i = 1, \dots, m$	Input & target
Model :	$H(x) = Wx + b$	Linear model W : weight, b: bias
Cost Function	$cost(W, b) = \frac{1}{m} \sum_{i=1}^m (H(x^{(i)}) - y^{(i)})^2$	MSE
Optimization	$W := W - \alpha \frac{1}{m} \sum_{i=1}^m (Wx^{(i)} - y^{(i)}) x^{(i)}$	GD
Testing	$\hat{y} = Wx + b$	Metric : MSE

# Exercise 02-1.

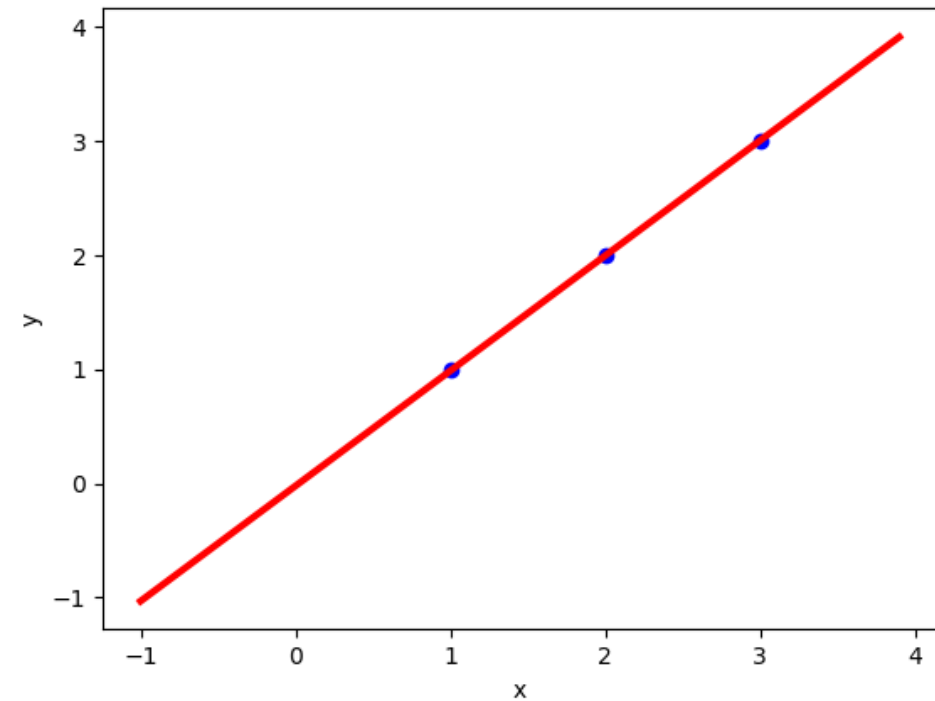
tf2-02-1-linear\_regression1.py

# Exercise 02-1.

- Data: input and target



- Regression results



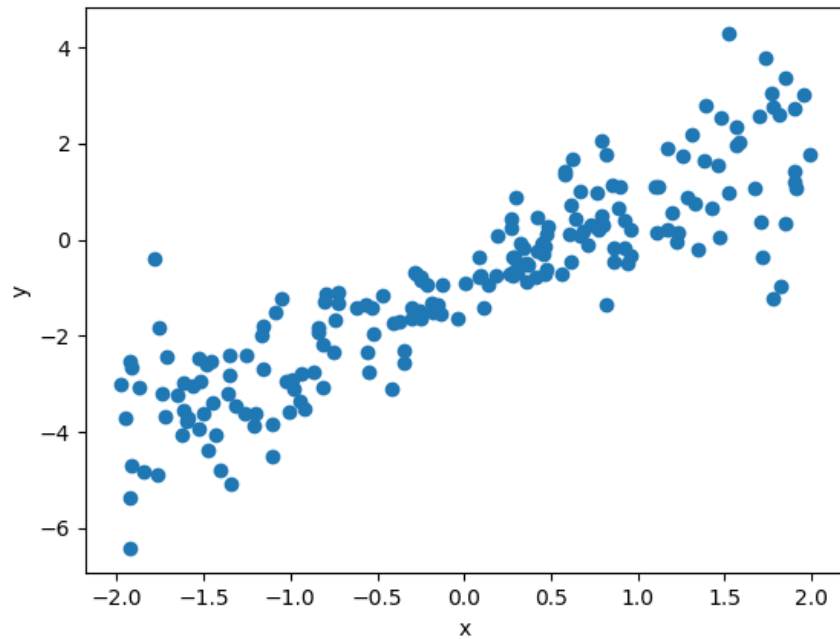


## Exercise 02-2.

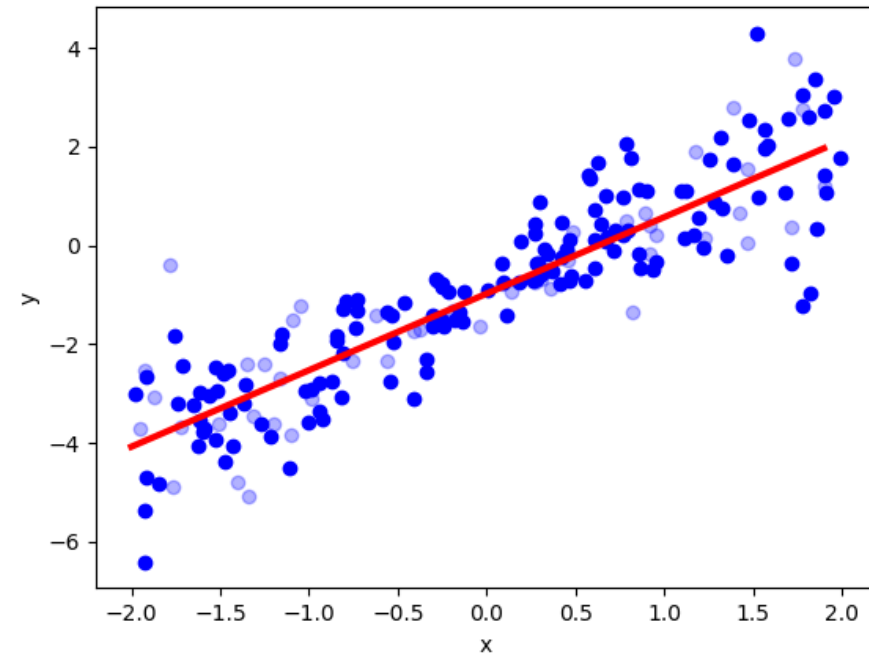
tf2-02-2-linear\_regression2.py

# Exercise 02-2.

- Data: input and target



- Regression results



# Lecture 2.1

## Multivariable Linear Regression

Dong Kook Kim

# Recap

- Hypothesis : Linear model

$$H(x) = Wx + b$$

- Cost function : MSE

$$cost(W, b) = \frac{1}{m} \sum_{i=1}^m (H(x^{(i)}) - y^{(i)})^2$$

- Optimization : GD algorithm

$$W := W - \alpha \frac{1}{m} \sum_{i=1}^m (Wx^{(i)} - y^{(i)}) x^{(i)}$$

# Predicting exam score: regression using one input (x)

one-variable  
one-feature

x (hours)	y (score)
10	90
9	80
3	50
2	60
11	40

# Predicting exam score: regression using three inputs ( $x_1$ , $x_2$ , $x_3$ )

multivariable input (feature) & one target

$x_1$ (quiz 1)	$x_2$ (quiz 2)	$x_3$ (midterm 1)	Y (final)
73	80	75	152
93	88	93	185
89	91	90	180
96	98	100	196
73	66	70	142

Test Scores for General Psychology

# Model

$$H(x) = Wx + b$$

$x$  : multivariate (vector),  $W$  : vector,  $b$  : scalar,  $H(x)$  : scalar

# Model

: three-variable inputs, one-variable output

$$H(x) = Wx + b$$

$$H(x_1, x_2, x_3) = w_1x_1 + w_2x_2 + w_3x_3 + b$$

$$W = [w_1 \quad w_2 \quad w_3] \quad x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$



# Loss function : MSE

$$H(x_1, x_2, x_3) = w_1x_1 + w_2x_2 + w_3x_3 + b$$

$$cost(W, b) = \frac{1}{m} \sum_{I=1}^m (H(x_1^{(i)}, x_2^{(i)}, x_3^{(i)}) - y^{(i)})^2$$

# Multi-variable LR

: n-variable inputs, scalar output

$$H(x_1, x_2, x_3) = w_1x_1 + w_2x_2 + w_3x_3 + b$$

$$H(x_1, x_2, x_3, \dots, x_n) = w_1x_1 + w_2x_2 + w_3x_3 + \dots + w_nx_n + b$$

$$H(x) = Wx + b$$

$$W = [w_1 \quad \cdots \quad w_n] \quad x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}$$

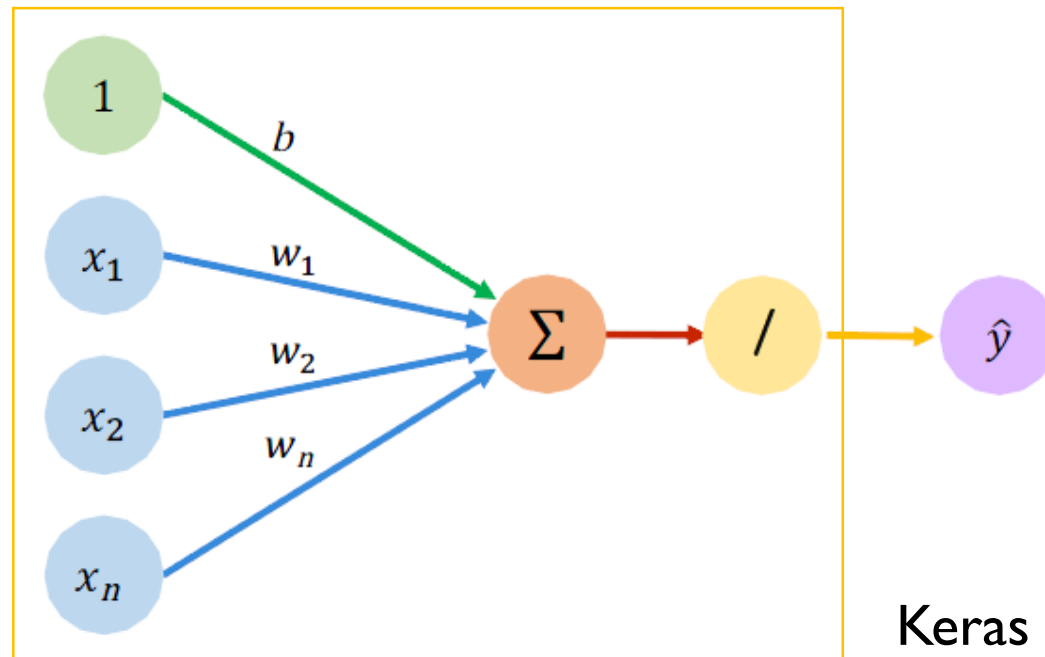
$W$  : n-dim weight vector

$b$  : bias

$x$  : n-dim vectors

# Graph for LR

- **Muti-variate Input/Scalar Target:**  $W = [w_1 \quad \cdots \quad w_n], b \quad x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, y$



Keras : **Dense(1, input\_dim=n)**

Inputs

Weights

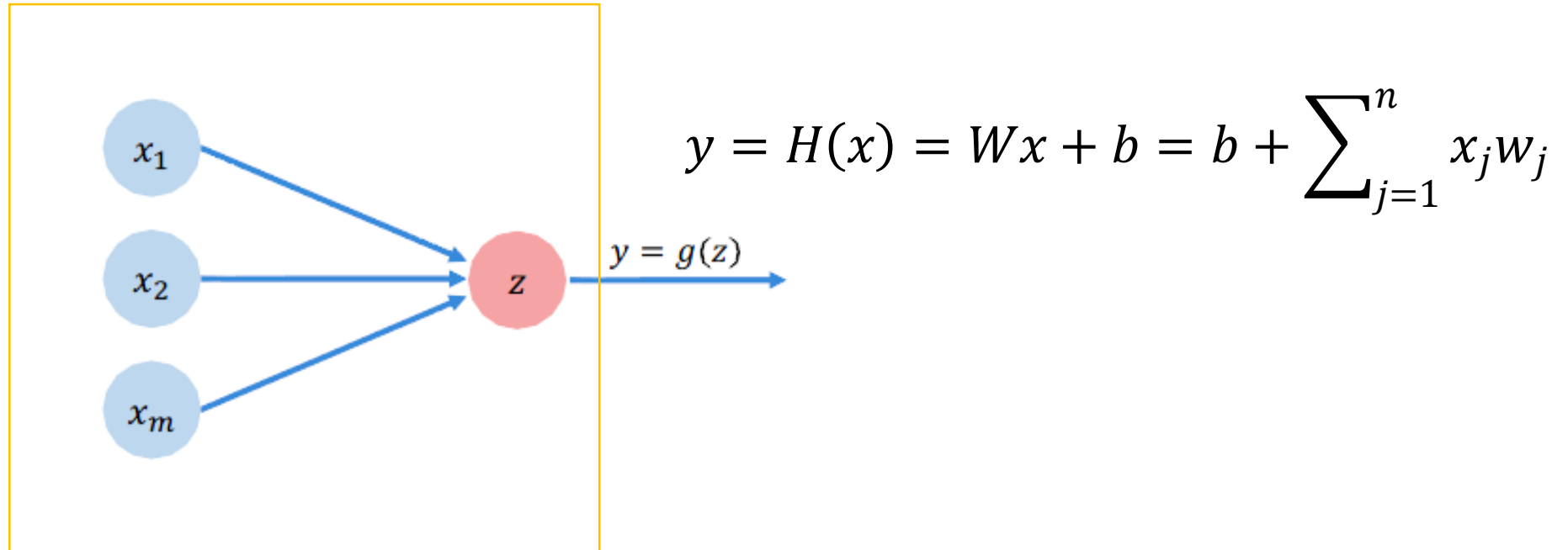
Sum

Linearity

Output

# Graph for LR : Simplified

- Multi-variate Input/Scalar Target:



Keras : `Dense(l, input_dim=n)`

# Multi-variable LR

: n-variable inputs, k-variable output

$$H(x) = Wx + b$$

$$W = \begin{bmatrix} w_{11} & \dots & w_{1n} \\ \dots & \ddots & \dots \\ w_{k1} & \dots & w_{kn} \end{bmatrix}$$

$W$  : n X n weight  
matrix

$$b = \begin{bmatrix} b_1 \\ \vdots \\ b_k \end{bmatrix}$$

$b$  : n-dim bias v  
ector

$$x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix},$$

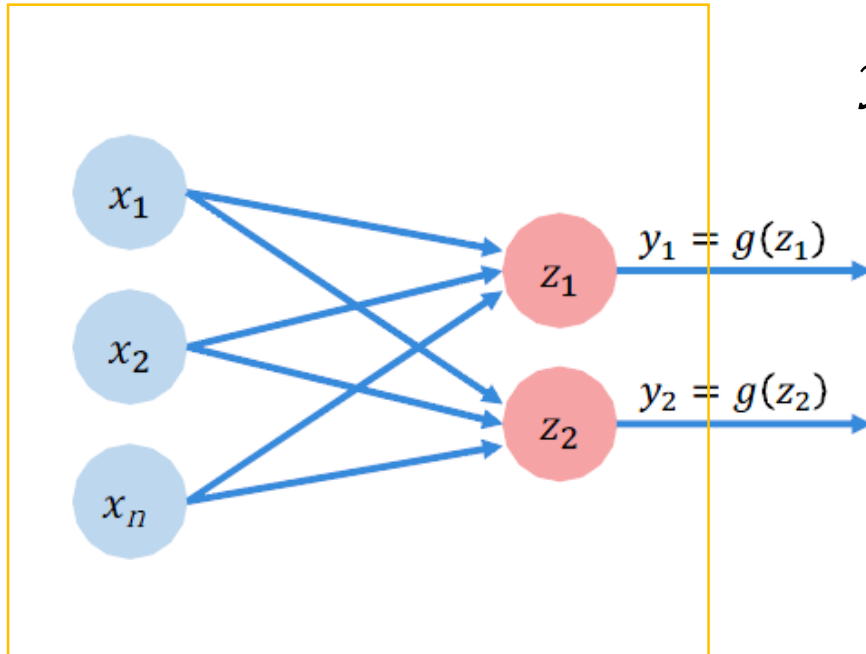
$x$  : n-dim vector

$$y = \begin{bmatrix} y_1 \\ \vdots \\ y_k \end{bmatrix}$$

$y$  : n-dim vector

# Graph for LR : Simplified

- Muti-variate Input/Muti-variate Target:



$$y_k = H_k(x) = W_k x + b_k = b_k + \sum_{j=1}^n x_j w_{k,j}$$

Keras : `Dense(2, input_dim=n)`

# Loss function

: n-variable inputs, k-variable output

$$cost(W, b) = \frac{1}{m} \sum_{i=1}^m \|H(x^{(i)}) - y^{(i)}\|^2$$

$$H(x) = Wx + b$$

$$W = \begin{bmatrix} w_{11} & \dots & w_{1n} \\ \dots & \ddots & \dots \\ w_{k1} & \dots & w_{kn} \end{bmatrix}$$

$W$  : n X n weight  
matrix

$$b = \begin{bmatrix} b_1 \\ \vdots \\ b_k \end{bmatrix}$$

$b$  : n-dim bias v  
ector

$$x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix},$$

$x$  : n-dim vector

$$y = \begin{bmatrix} y_1 \\ \vdots \\ y_k \end{bmatrix}$$

$y$  : n-dim vector

# Optimization

$$H(x) = Wx + b$$

n-variable inputs, scalar output

$$cost(W, b) = \frac{1}{m} \sum_{i=1}^m (H(x^{(i)}) - y^{(i)})^2$$

n-variable inputs, k-variable output

$$cost(W, b) = \frac{1}{m} \sum_{i=1}^m \|H(x^{(i)}) - y^{(i)}\|^2$$

- Gradient Descent

$$W := W - \alpha \frac{\partial cost(W, b)}{\partial W}$$



# Matrix

$$w_1x_1 + w_2x_2 + w_3x_3 + \dots + w_nx_n$$

# Matrix multiplication

"Dot Product"

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \times \begin{bmatrix} 7 & 8 \\ 9 & 10 \\ 11 & 12 \end{bmatrix} = \begin{bmatrix} 58 & \\ & \end{bmatrix}$$

# Hypothesis using matrix

$$w_1x_1 + w_2x_2 + w_3x_3 + \dots + w_nx_n$$

$$(x_1 \quad x_2 \quad x_3) \cdot \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix} = (x_1w_1 + x_2w_2 + x_3w_3)$$

$$H(X) = XW$$

$$X = [x_1 \quad \dots \quad x_n] \quad W = \begin{bmatrix} w_1 \\ \vdots \\ w_n \end{bmatrix}$$

# Hypothesis using matrix

$$H(x_1, x_2, x_3) = x_1w_1 + x_2w_2 + x_3w_3$$

$x_1$	$x_2$	$x_3$	Y
73	80	75	152
93	88	93	185
89	91	90	180
96	98	100	196
73	66	70	142

Test Scores for General Psychology

# Hypothesis using matrix

$$H(x_1, x_2, x_3) = x_1w_1 + x_2w_2 + x_3w_3$$

$x_1$	$x_2$	$x_3$	Y
73	80	75	152
93	88	93	185
89	91	90	180
96	98	100	196
73	66	70	142

Test Scores for General Psychology

$$(x_1 \quad x_2 \quad x_3) \cdot \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix} = (x_1w_1 + x_2w_2 + x_3w_3)$$

$$H(X) = XW$$

# Many x instances

$x_1$	$x_2$	$x_3$	Y
73	80	75	152
93	88	93	185
89	91	90	180
96	98	100	196
73	66	70	142

Test Scores for General Psychology

$$(x_1 \quad x_2 \quad x_3) \cdot \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix} = (x_1 w_1 + x_2 w_2 + x_3 w_3)$$

$x_1$	$x_2$	$x_3$	Y
73	80	75	152
93	88	93	185
89	91	90	180
96	98	100	196
73	66	70	142

# Hypothesis using matrix

$$w_1x_1 + w_2x_2 + w_3x_3 + \dots + w_nx_n$$

$$\begin{pmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \\ x_{41} & x_{42} & x_{43} \\ x_{51} & x_{52} & x_{53} \end{pmatrix} \cdot \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix} = \begin{pmatrix} x_{11}w_1 + x_{12}w_2 + x_{13}w_3 \\ x_{21}w_1 + x_{22}w_2 + x_{23}w_3 \\ x_{31}w_1 + x_{32}w_2 + x_{33}w_3 \\ x_{41}w_1 + x_{42}w_2 + x_{43}w_3 \\ x_{51}w_1 + x_{52}w_2 + x_{53}w_3 \end{pmatrix}$$

$$H(X) = XW$$

# Hypothesis using matrix

$$\begin{pmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \\ x_{41} & x_{42} & x_{43} \\ x_{51} & x_{52} & x_{53} \end{pmatrix} \cdot \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix} = \begin{pmatrix} x_{11}w_1 + x_{12}w_2 + x_{13}w_3 \\ x_{21}w_1 + x_{22}w_2 + x_{23}w_3 \\ x_{31}w_1 + x_{32}w_2 + x_{33}w_3 \\ x_{41}w_1 + x_{42}w_2 + x_{43}w_3 \\ x_{51}w_1 + x_{52}w_2 + x_{53}w_3 \end{pmatrix}$$

[5, 3]

[3, 1]

[5, 1]

$$H(X) = XW$$



# Hypothesis using matrix

$$\begin{array}{ccc} \left( \begin{array}{|c|} \hline \mathbf{X} \\ \hline \end{array} \right) \times \left( \begin{array}{|c|} \hline \mathbf{W} \\ \hline \end{array} \right) = \left( \begin{array}{|c|} \hline H(\mathbf{X}) \\ \hline \end{array} \right) \\ [5, 3] \qquad \qquad [?, ?] \qquad \qquad [5, 1] \end{array}$$

$$H(X) = XW$$

# Hypothesis using Matrix (one output)

$$\begin{pmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \\ x_{41} & x_{42} & x_{43} \\ x_{51} & x_{52} & x_{53} \end{pmatrix} \cdot \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix} = \begin{pmatrix} x_{11}w_1 + x_{12}w_2 + x_{13}w_3 \\ x_{21}w_1 + x_{22}w_2 + x_{23}w_3 \\ x_{31}w_1 + x_{32}w_2 + x_{33}w_3 \\ x_{41}w_1 + x_{42}w_2 + x_{43}w_3 \\ x_{51}w_1 + x_{52}w_2 + x_{53}w_3 \end{pmatrix}$$

[n, 3]

[3, 1]

[n, 1]

$$H(X) = XW$$

# Hypothesis using Matrix (n output)

$$\begin{pmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \\ x_{41} & x_{42} & x_{43} \\ x_{51} & x_{52} & x_{53} \end{pmatrix} \cdot \begin{matrix} ? \\ ? \\ ? \\ ? \\ ? \end{matrix} = \begin{pmatrix} x_{11}w_{11} + x_{12}w_{21} + x_{13}w_{31} & x_{11}w_{12} + x_{12}w_{22} + x_{13}w_{32} \\ x_{21}w_{11} + x_{22}w_{21} + x_{23}w_{31} & x_{21}w_{12} + x_{22}w_{22} + x_{23}w_{32} \\ x_{31}w_{11} + x_{32}w_{21} + x_{33}w_{31} & x_{31}w_{12} + x_{32}w_{22} + x_{33}w_{32} \\ x_{41}w_{11} + x_{42}w_{21} + x_{43}w_{31} & x_{41}w_{12} + x_{42}w_{22} + x_{43}w_{32} \\ x_{51}w_{11} + x_{52}w_{21} + x_{53}w_{31} & x_{51}w_{12} + x_{52}w_{22} + x_{53}w_{32} \end{pmatrix}$$

$[n, 3] \quad \quad [?, ?] \quad \quad [n, 2]$

$$H(X) = XW$$

# Hypothesis using Matrix (n output)

$$\begin{pmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \\ x_{41} & x_{42} & x_{43} \\ x_{51} & x_{52} & x_{53} \end{pmatrix} \cdot \begin{pmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \\ w_{31} & w_{32} \end{pmatrix} = \begin{pmatrix} x_{11}w_{11} + x_{12}w_{21} + x_{13}w_{31} & x_{11}w_{12} + x_{12}w_{22} + x_{13}w_{32} \\ x_{21}w_{11} + x_{22}w_{21} + x_{23}w_{31} & x_{21}w_{12} + x_{22}w_{22} + x_{23}w_{32} \\ x_{31}w_{11} + x_{32}w_{21} + x_{33}w_{31} & x_{31}w_{12} + x_{32}w_{22} + x_{33}w_{32} \\ x_{41}w_{11} + x_{42}w_{21} + x_{43}w_{31} & x_{41}w_{12} + x_{42}w_{22} + x_{43}w_{32} \\ x_{51}w_{11} + x_{52}w_{21} + x_{53}w_{31} & x_{51}w_{12} + x_{52}w_{22} + x_{53}w_{32} \end{pmatrix}$$

**[n, 3]                  [3, 2]                                  [n, 2]**

$$H(X) = XW$$

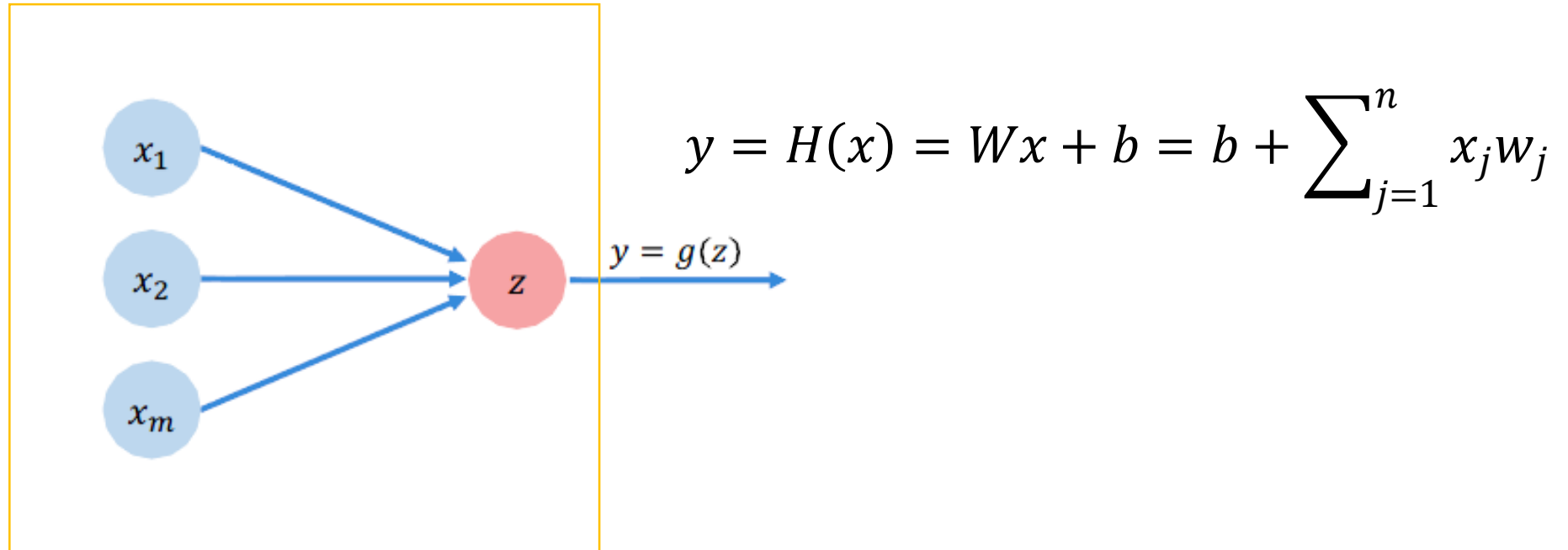
# $WX$ vs $XW$

- Lecture (theory):  $H(x) = Wx + b$
- Implementation (TensorFlow)

$$H(X) = XW$$

# Graph for LR : Simplified

- Muti-variate Input/Scalar Target:



Keras : `Dense(l, input_dim=n)`

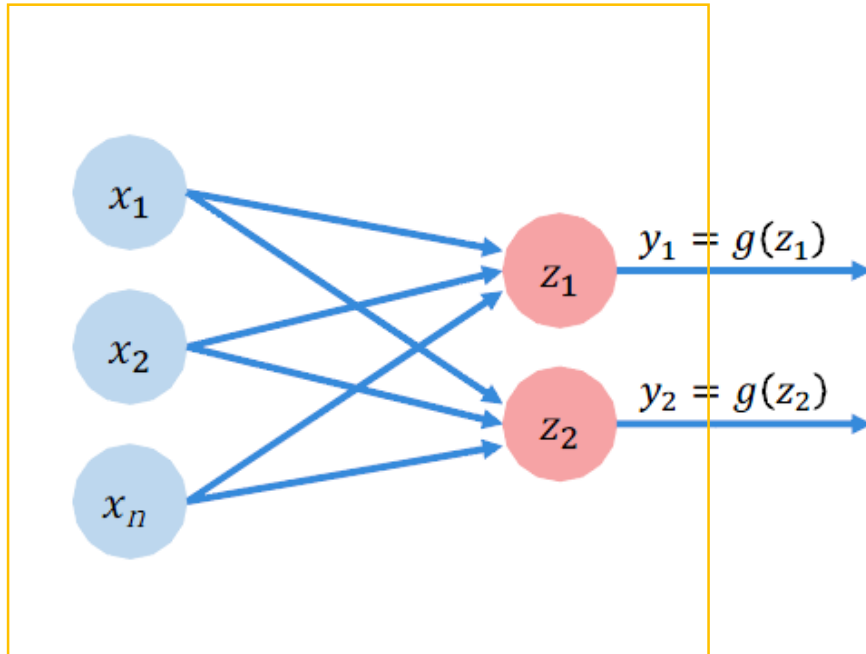
# Linear Regression : Summary

- Multi-variate Input/Scalar Target:

Data Set :	$x^{(i)}, y^{(i)}, i = 1, \dots, m$	Input vector & scalar target
Model :	$H(x) = Wx + b$	Linear model W : weight vector b : bias
Cost Function	$cost(W, b) = \frac{1}{m} \sum_{i=1}^m (H(x^{(i)}) - y^{(i)})^2$	MSE
Optimization	$W := W - \alpha \frac{\partial cost(W, b)}{\partial W}$	GD
Testing	$\hat{y} = Wx + b$	Metric : MSE

# Graph for LR : Simplified

- **Muti-variate Input/Muti-variate Target:**  $H(x) = Wx + b$



$$y_k = H_k(x) = W_k x + b_k = b_k + \sum_{j=1}^n x_j w_{k,j}$$

$$W = \begin{bmatrix} w_{11} & \dots & w_{1n} \\ \dots & \ddots & \dots \\ w_{k1} & \dots & w_{kn} \end{bmatrix} \quad b = \begin{bmatrix} b_1 \\ \vdots \\ b_k \end{bmatrix}$$

$W$  :  $n \times n$  weight  
matrix

$b$  :  $n$ -dim bias v  
ector

Keras : **Dense(2, input\_dim=n)**



# Linear Regression : Summary

- Multi-variate Input/Mult-variate Target:

Data Set :	$x^{(i)}, y^{(i)}, i = 1, \dots, m$	Input vector & target vector
Model :	$H(x) = Wx + b$	Linear model W : weight matrix b : bias vector
Cost Function	$cost(W, b) = \frac{1}{m} \sum_{i=1}^m   H(x^{(i)}) - y^{(i)}  ^2$	MSE
Optimization	$W := W - \alpha \frac{\partial cost(W, b)}{\partial W}$	GD
Testing	$\hat{y} = Wx + b$	Metric : MSE

## Exercise 02-3.

tf2-02-3-multivariable\_linear\_regression.py

# Exercise 02-3.

- Data: input and target

$x_1$ (quiz 1)	$x_2$ (quiz 2)	$x_3$ (midterm 1)	Y (final)
73	80	75	152
93	88	93	185
89	91	90	180
96	98	100	196
73	66	70	142

# Exercise 02-4.

tf2-02-4-file\_input\_linear\_regression.py

- data read from file