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 데이타가 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 도구를 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.

NEW 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

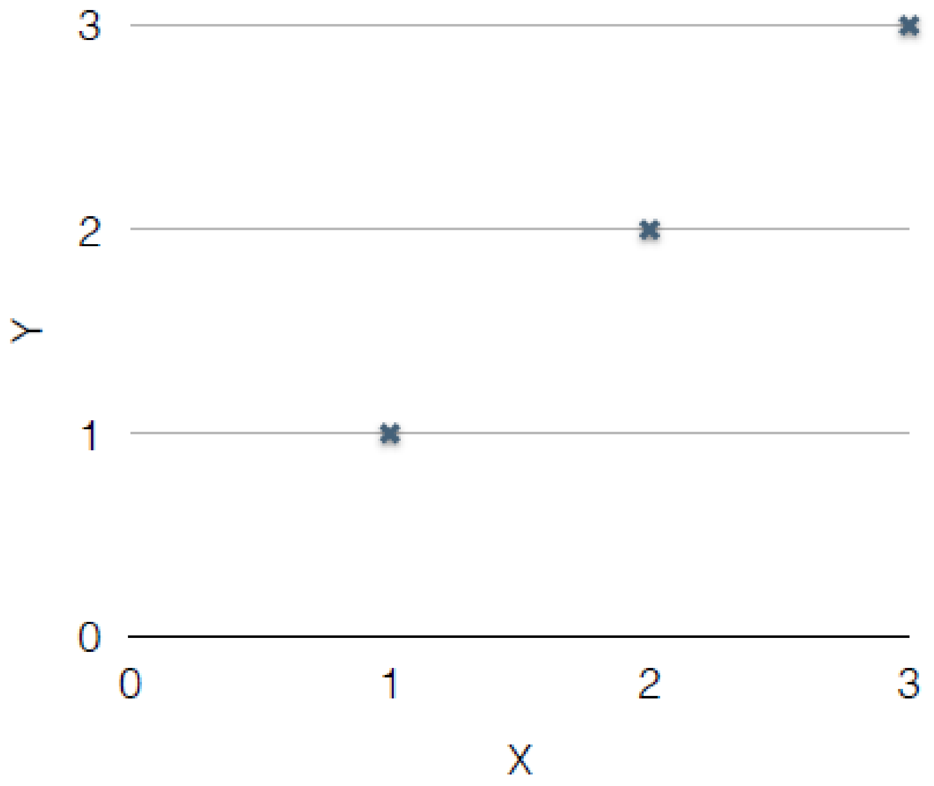
Regression : Step 1. data

- input (real) - target (real)

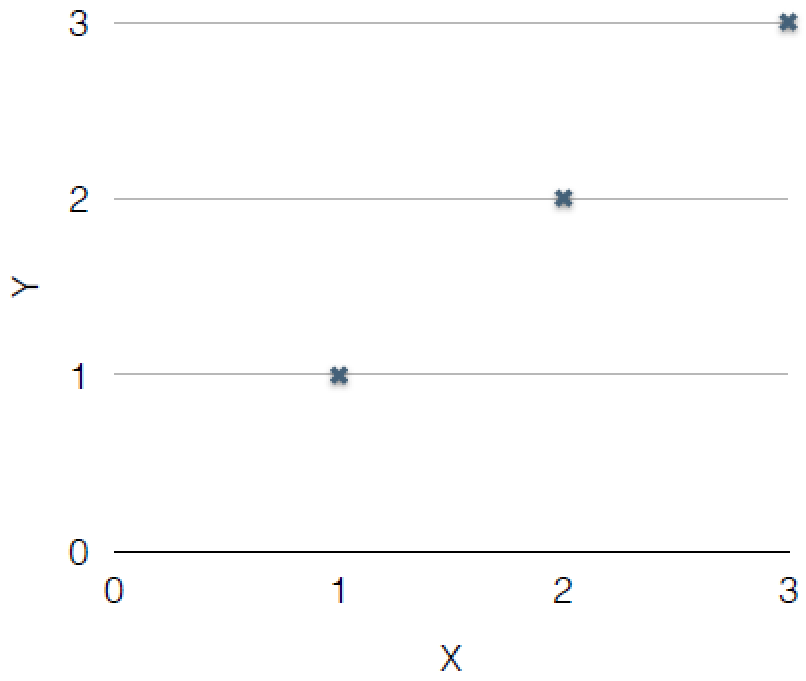
x	y
1	1
2	2
3	3

Regression (Graph)

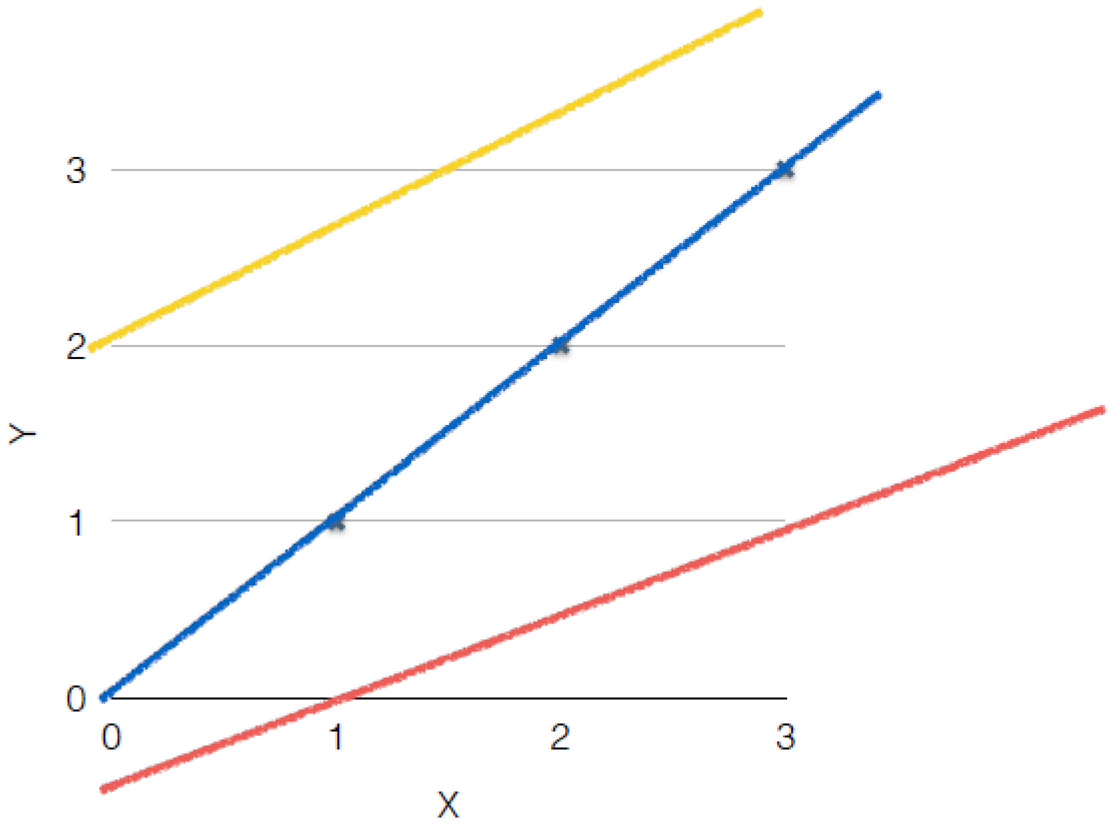
X	Y
1	1
2	2
3	3



Linear Model (or Hypothesis)



Step 2. Linear Models



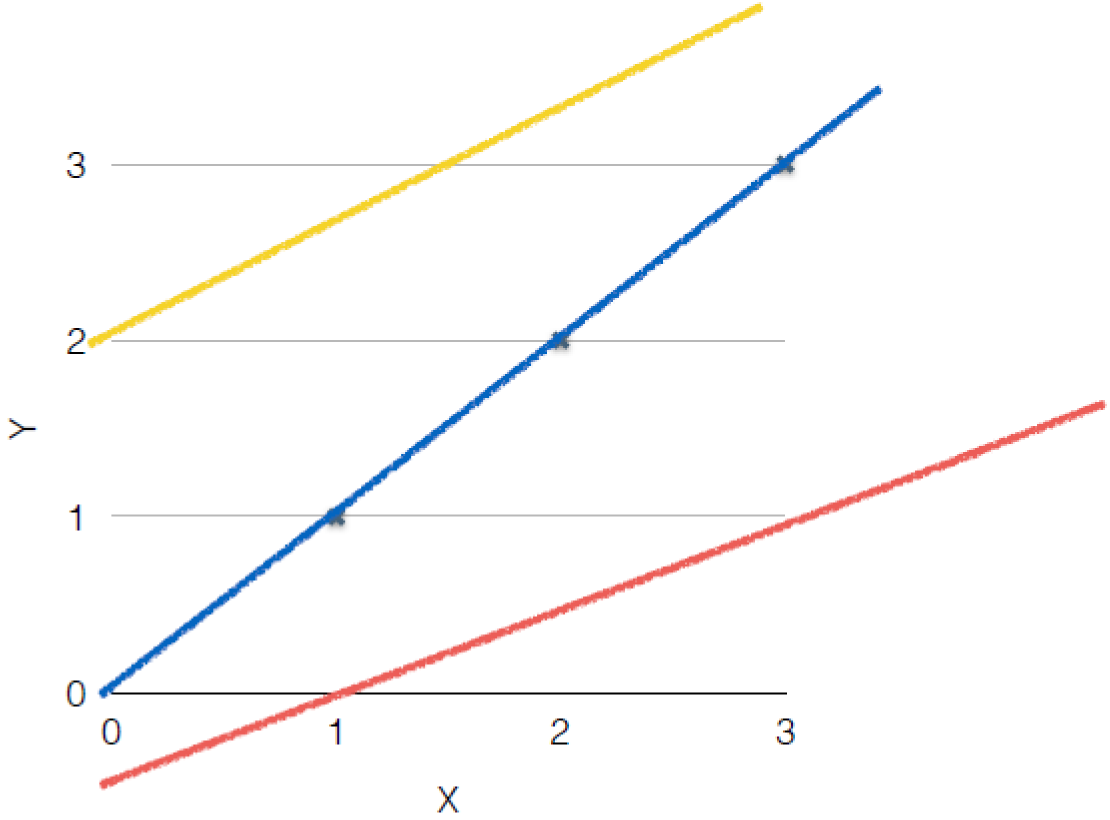
Step 2. Linear Models

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

W, *b* : model parameters

W : weight

b : bias

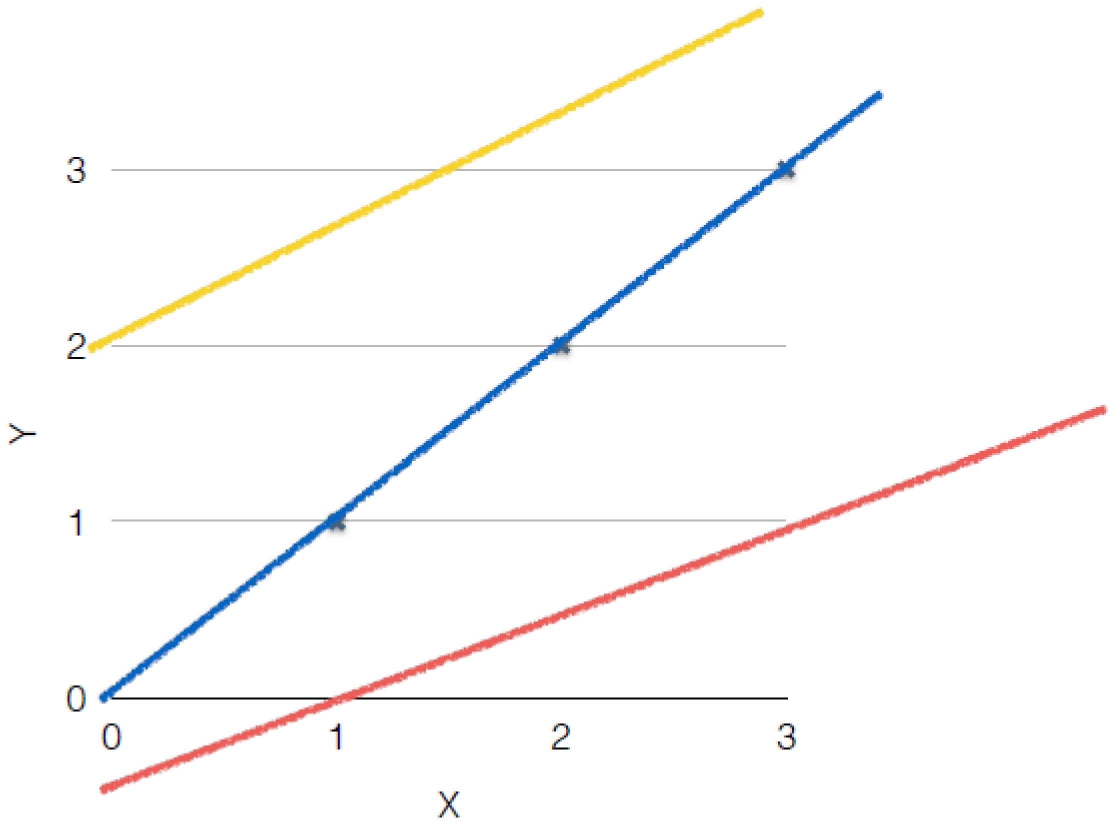


$W1$, b 1

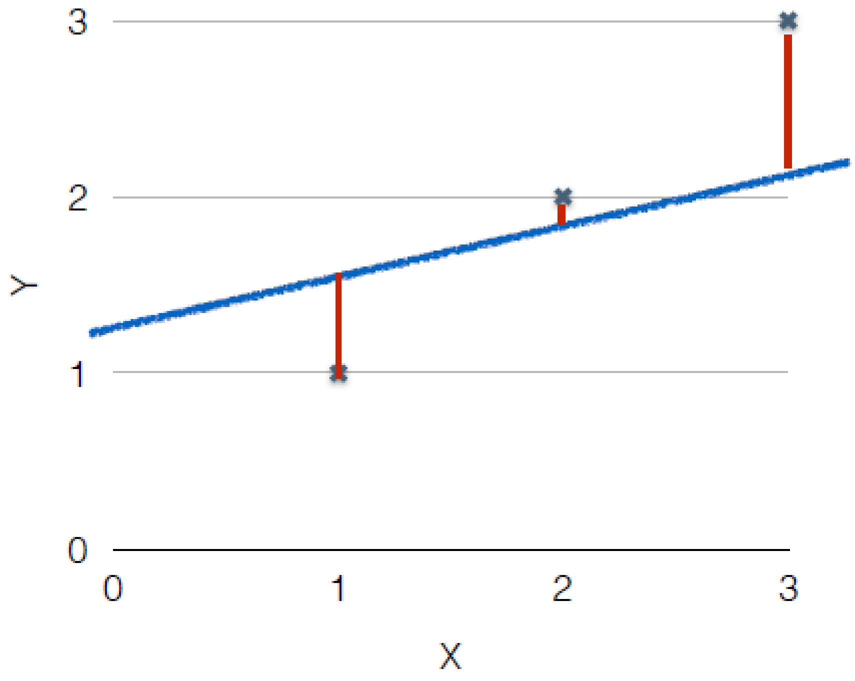
$W2$, b 2

$W3$, b 3

Which hypothesis is better



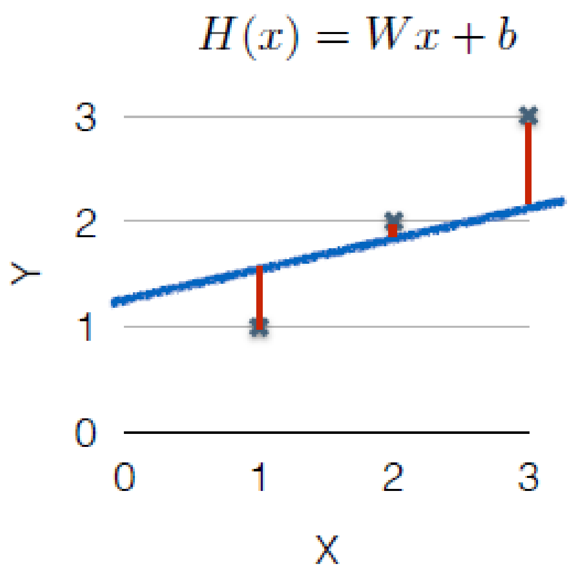
Which hypothesis is better : Error



Step 3. Loss function

- How fit the line to our (training) data

$$\text{error} : \ H(x) - \ y$$



Step 3. Loss function

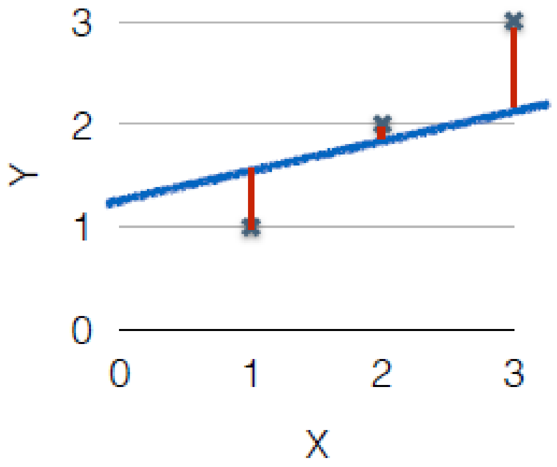
- How fit the line to our (training) data

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

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

- MSE (mean square error)

$H(x) = Wx + b$



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

$$H(x) = Wx$$

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

What *cost(W)* looks like?

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

X	Y
1	1
2	2
3	3

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

What *cost(W)* looks like?

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

x	Y
1	1
2	2
3	3

- W=1, cost(W)=0

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

- W=0, cost(W)=4.67

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

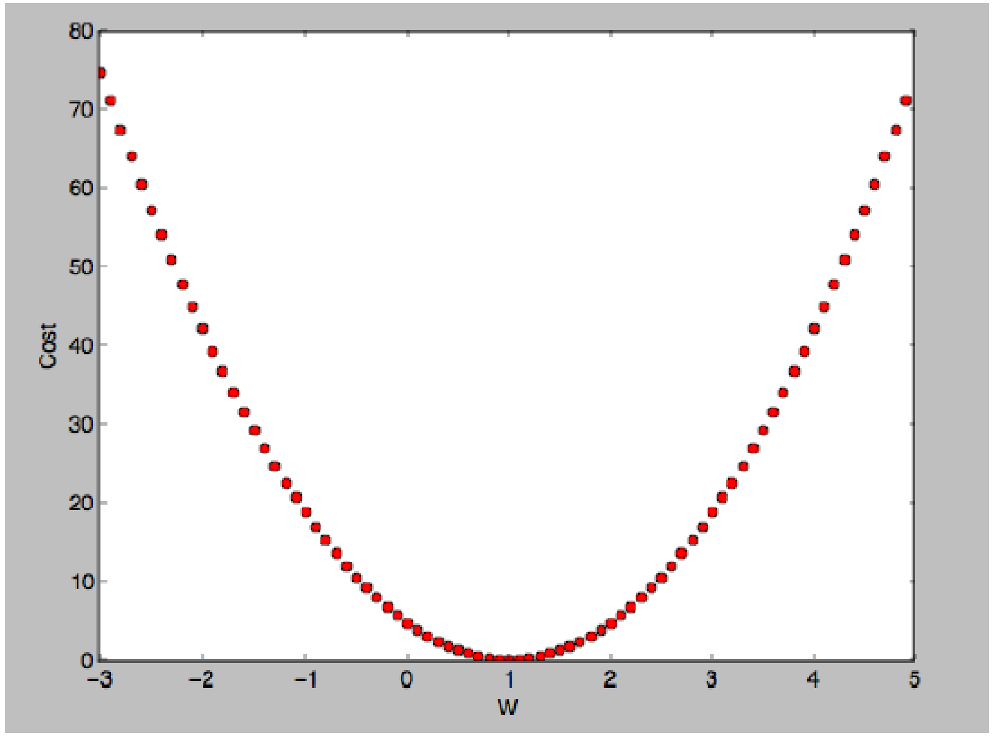
- W=2, cost(W)=?

What *cost(W)* looks like?

- W=1, cost(W)=0
- W=0, cost(W)=4.67
- W=2, cost(W)=4.67

What *cost(W)* looks like?

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

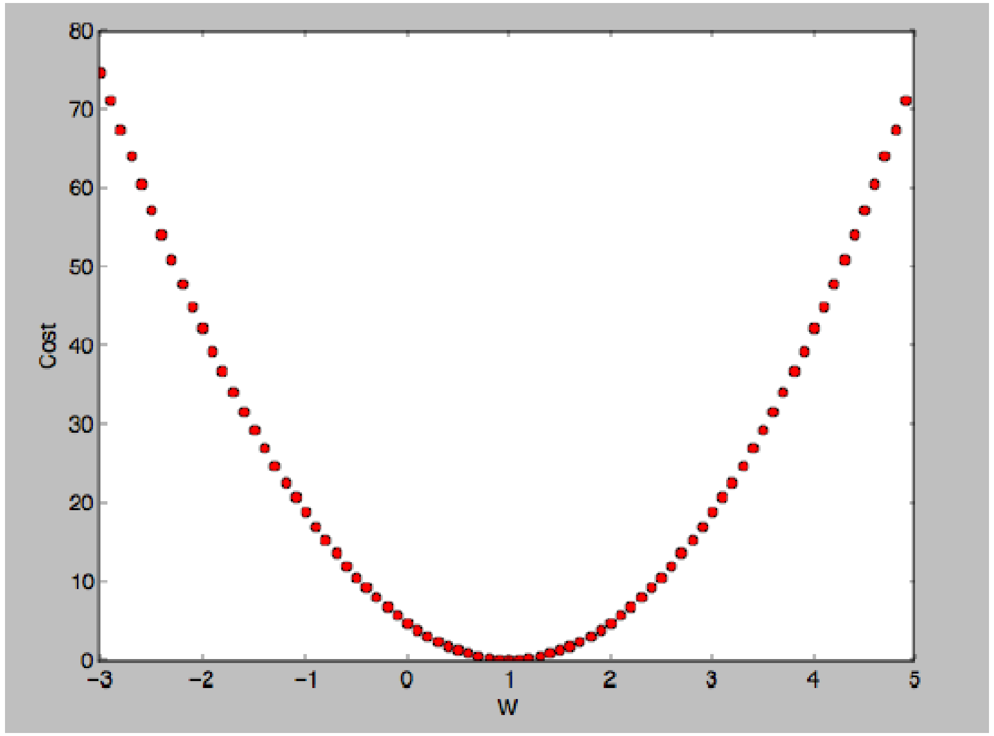


Goal : Minimize Loss

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

Step 4. Optimization
- How to minimize loss ?

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



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.

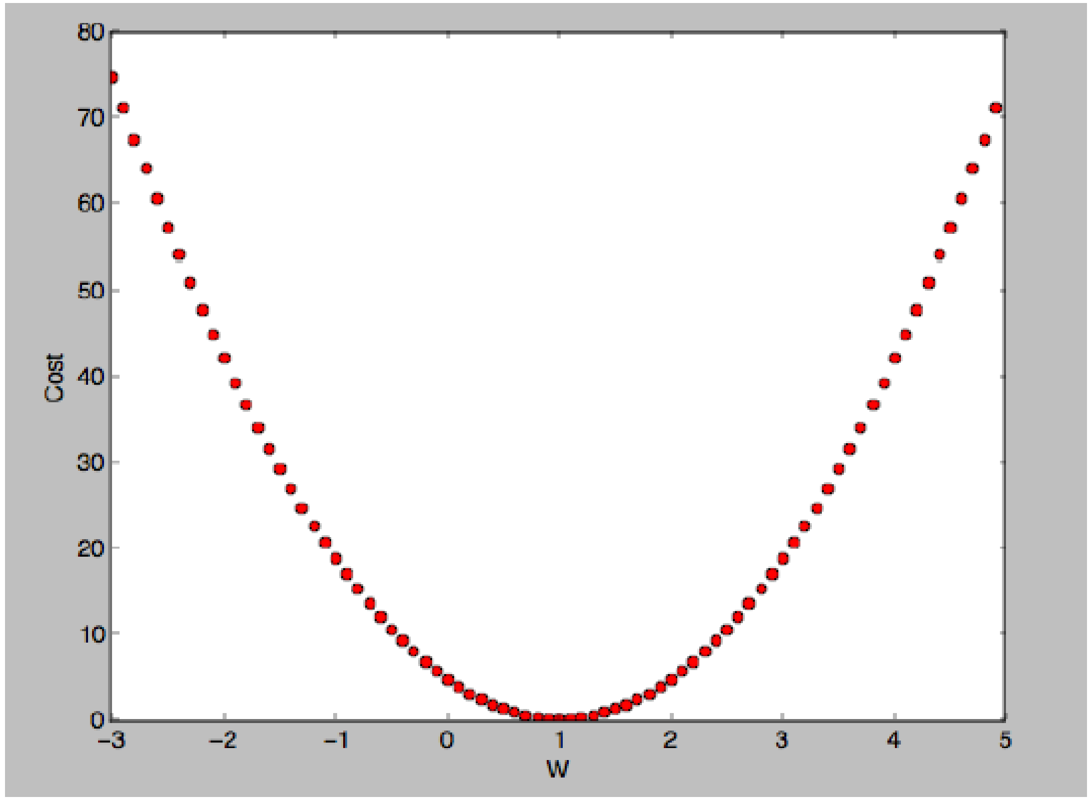
3. Iterative and applied to a set of problems that have non-convex loss functions such as NNs
   - parameters' initialization plays a critical role in speeding up convergence and achieving lower error rates.

Step 4. Optimization
- Gradient descent algorithm

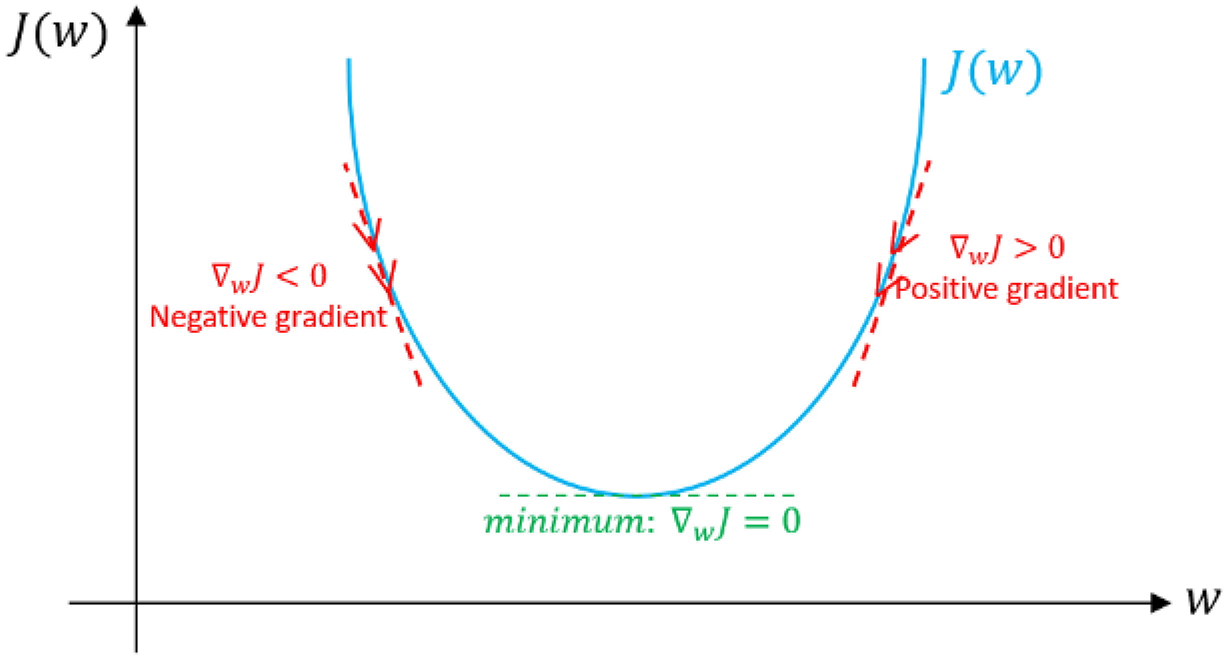
- Most common optimization algorithm in *ML* and *DL*
- First-order optimization algorithm – use the first derivative when performing the updates on the parameters.
- Minimize loss function – given $J(W, b)$, find W, b to minimize loss
- Update the parameters in the opposite direction of the gradient 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 α

How it works?

How would you find the lowest point?



How it works? : Gradient Descent



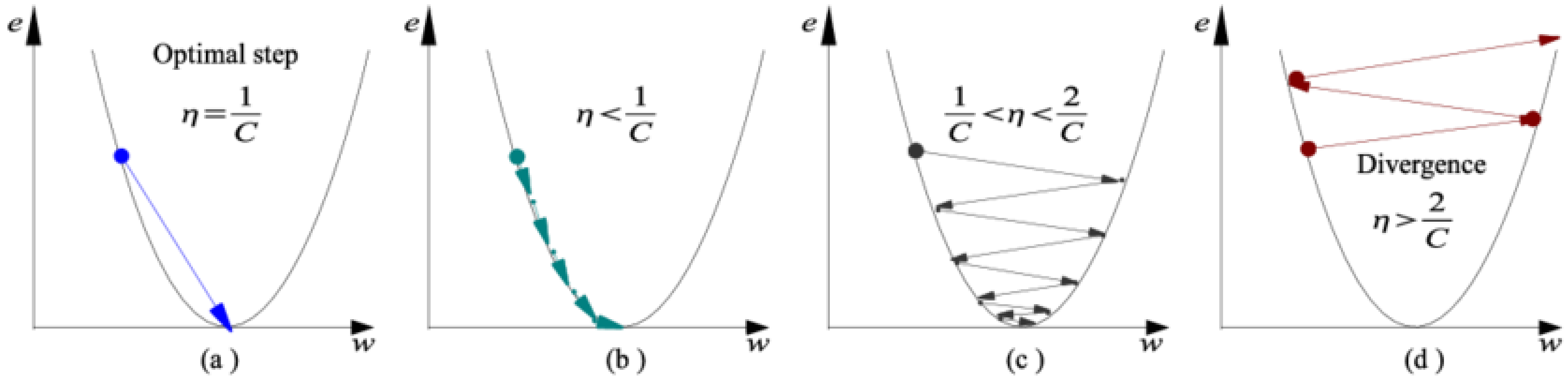
https://towardsdatascience.com/gradient-descent-algorithm-and-its-variants-10f652806a3

How it works?

1. Initialize weight *w* and bias *b* to any random numbers.

2. Pick a value for the learning rate α. The learning rate determines how big the step would be on each iteration.

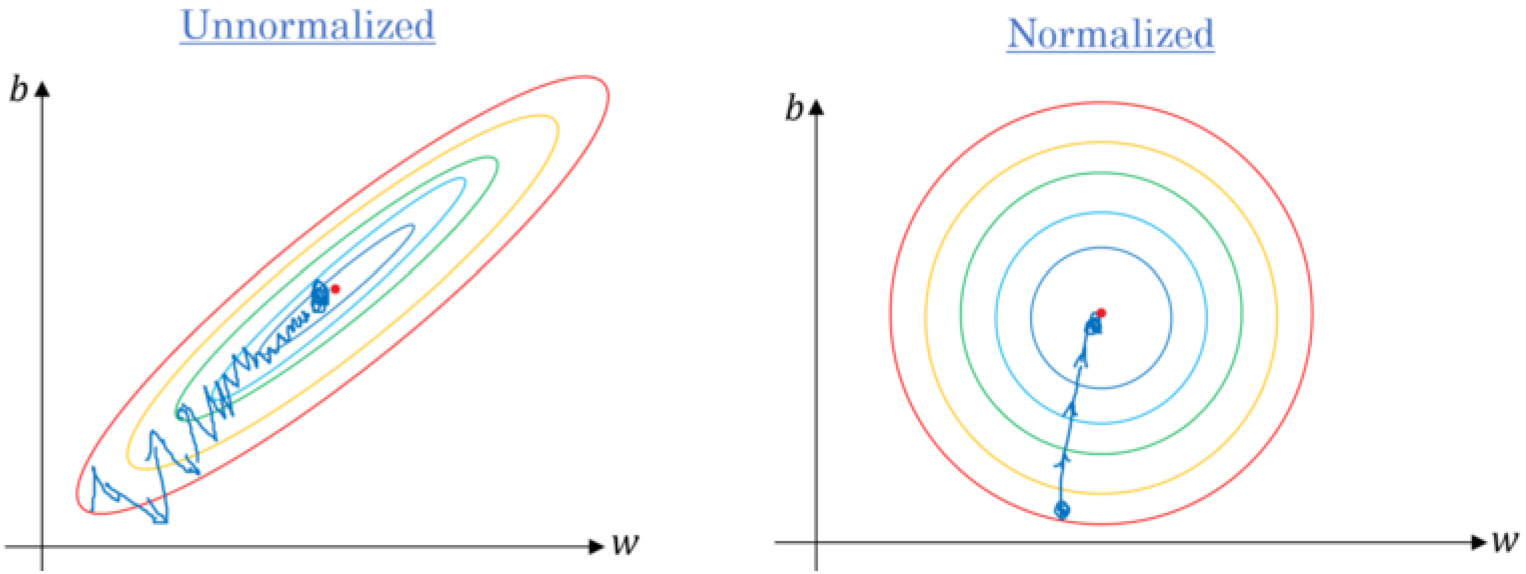
   (The most commonly used rates are : *0.001, 0.003, 0.01, 0.03, 0.1*)



Good LR | Low LR | High LR | Very High LR

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

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

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

Formal definition

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

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

Formal definition

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

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

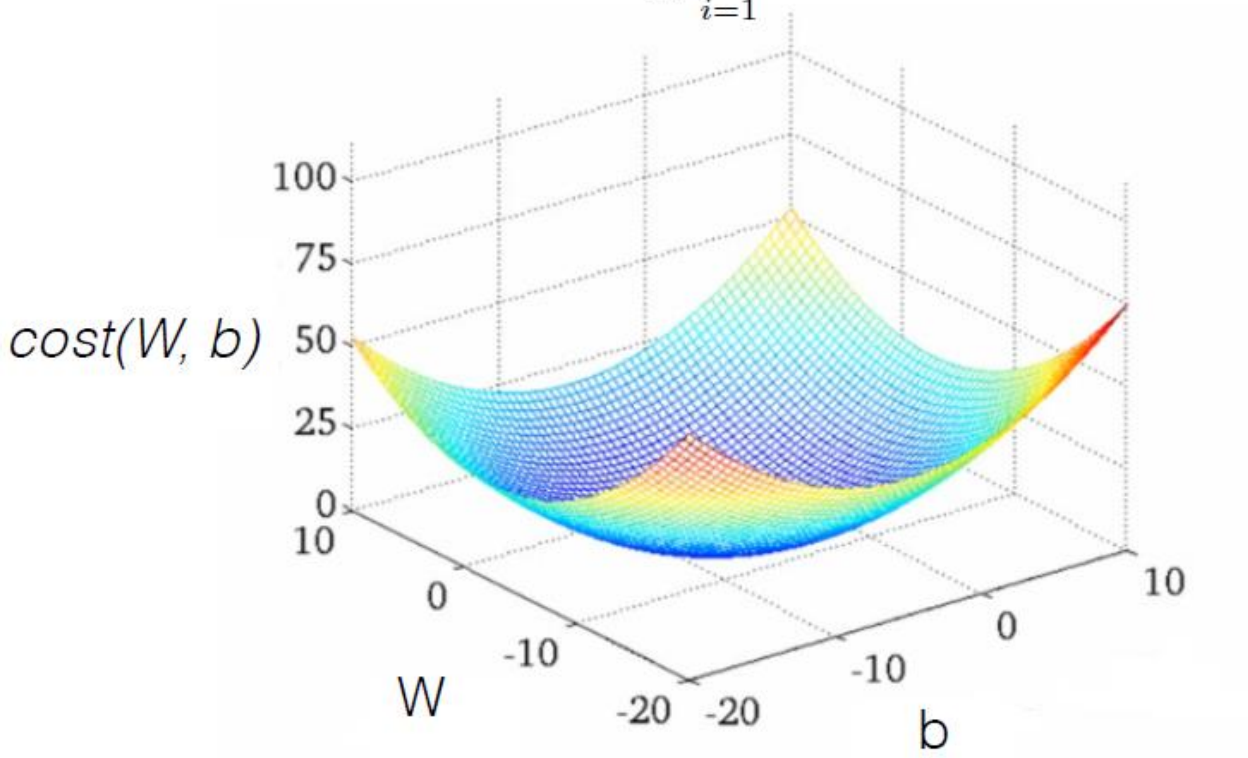
$$W := W - \alpha \frac{1}{m} \sum_{i=1}^{m} (Wx^{(i)} - y^{(i)})x^{(i)}$$

Gradient descent algorithm

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

Convex function

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

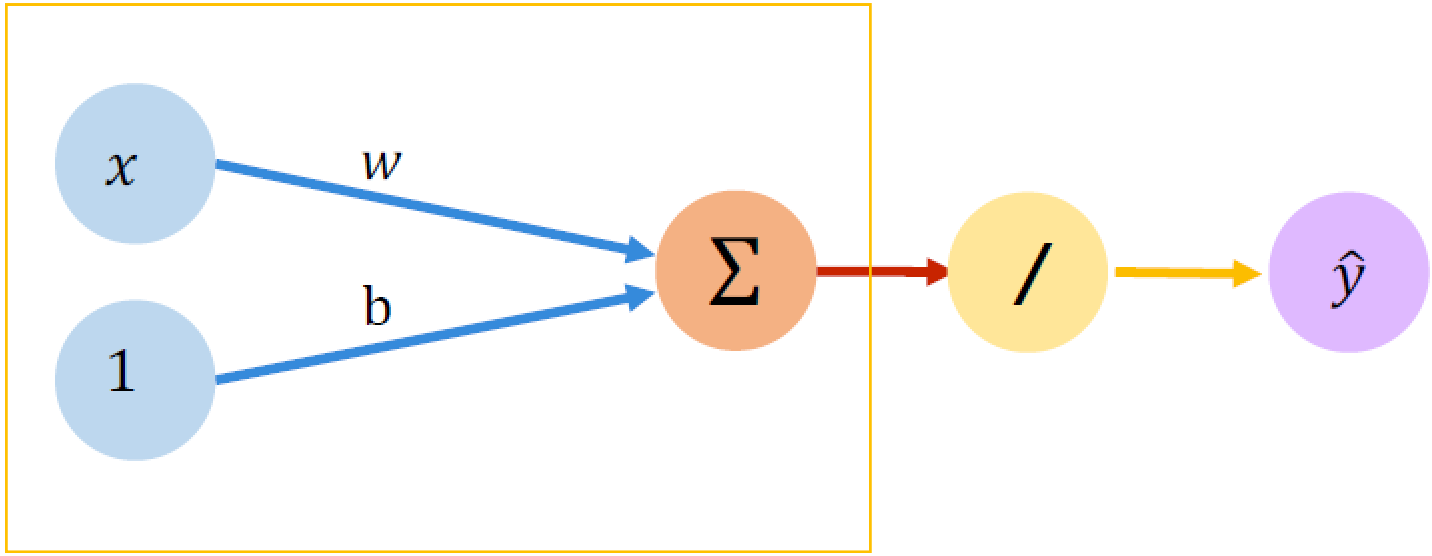


www.holehouse.org/mlclass/

Graph for Linear Regression

- Single input/single output : parameters W, b

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



Keras : Dense(1, input_dim=1)

(scalar) Linear Regression : Summary

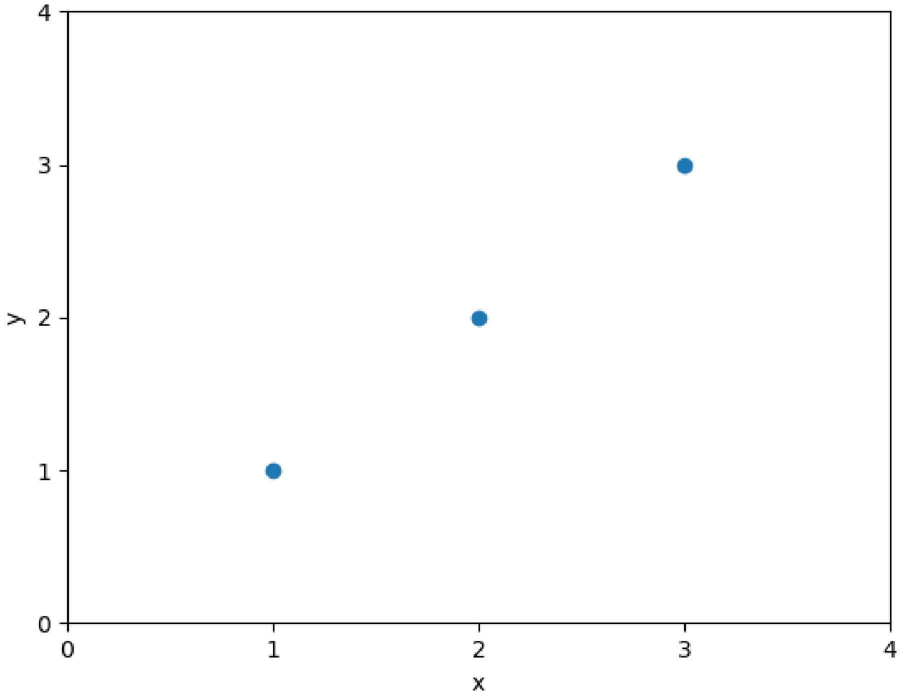
Data Set :	$x^{(i)}, y^{(i)}, i = 1, \dots, m$	Input & target
Model :	$H(x) = Wx + b$	Linear model W : weight, b: bias
Cost Function	$cost(W, b) = \frac{1}{m}\sum_{i=1}^{m}(H(x^{(i)}) - y^{(i)})^2$	MSE
Optimization	$W := W - \alpha\frac{1}{m}\sum_{i=1}^{m}(Wx^{(i)} - y^{(i)})\, x^{(i)}$	GD
Testing	$\hat{y}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

Exercise 02-1.

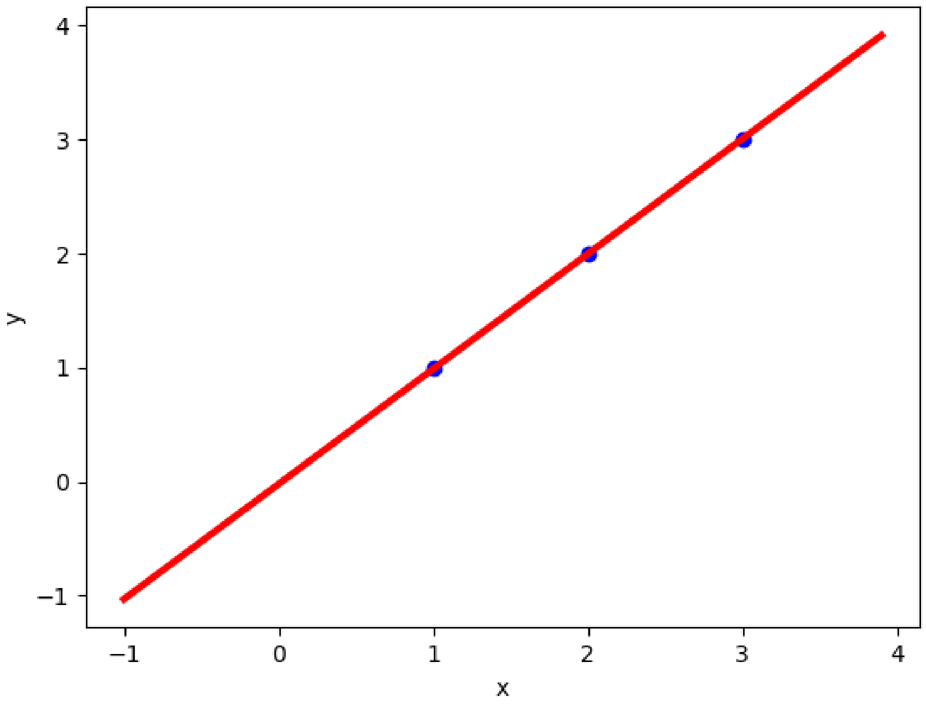
tf2-02-1-linear_regression1.py

Exercise 02-1.

- Data: input and target



- Regression results

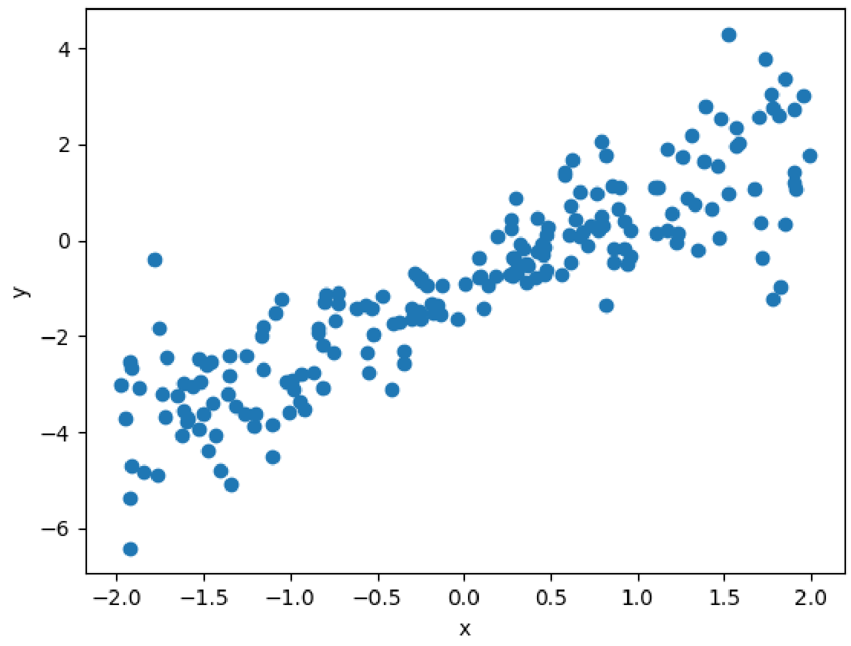


Exercise 02-2.

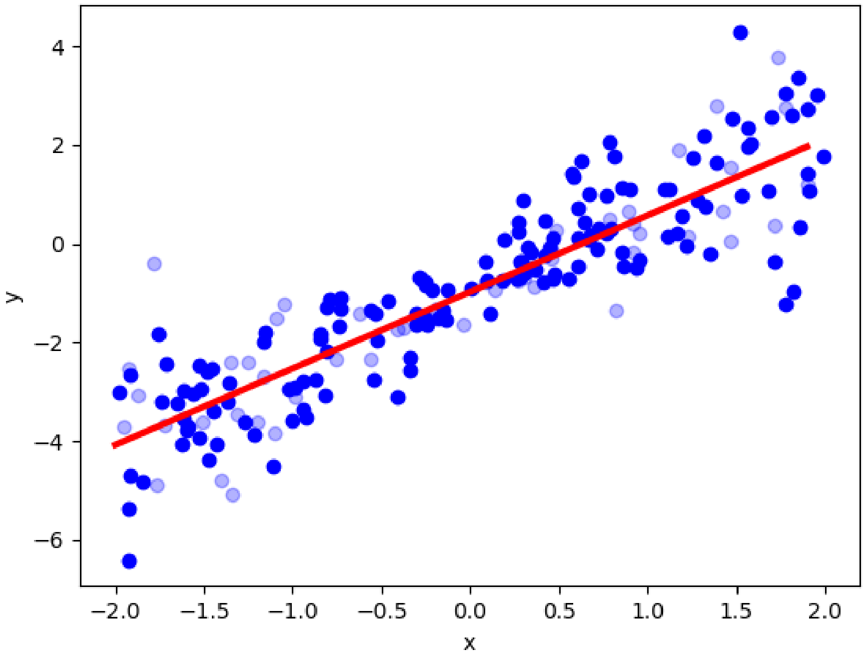
tf2-02-2-linear_regression2.py

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 (x1, x2, x3)

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

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

$$H(x_1, x_2, x_3) = w_1 x_1 + w_2 x_2 + w_3 x_3 + b$$

$$W = \begin{bmatrix} w_1 & w_2 & w_3 \end{bmatrix} \qquad x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

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

$$H(x_1, x_2, x_3) = w_1 x_1 + w_2 x_2 + w_3 x_3 + b$$

$$H(x_1, x_2, x_3, ..., x_n) = w_1 x_1 + w_2 x_2 + w_3 x_3 + ... + w_n x_n + b$$

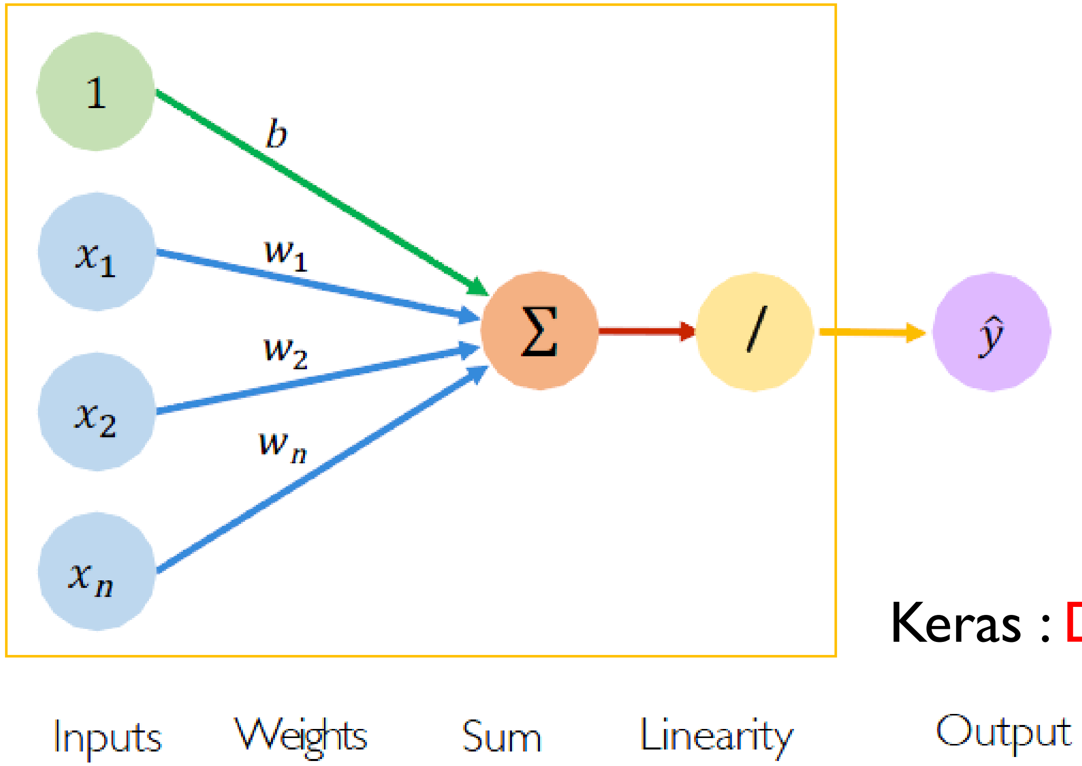
$$H(x) = Wx + b$$

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

W : n-dim weight vector
b : bias
x : n-dim vectors

Graph for LR

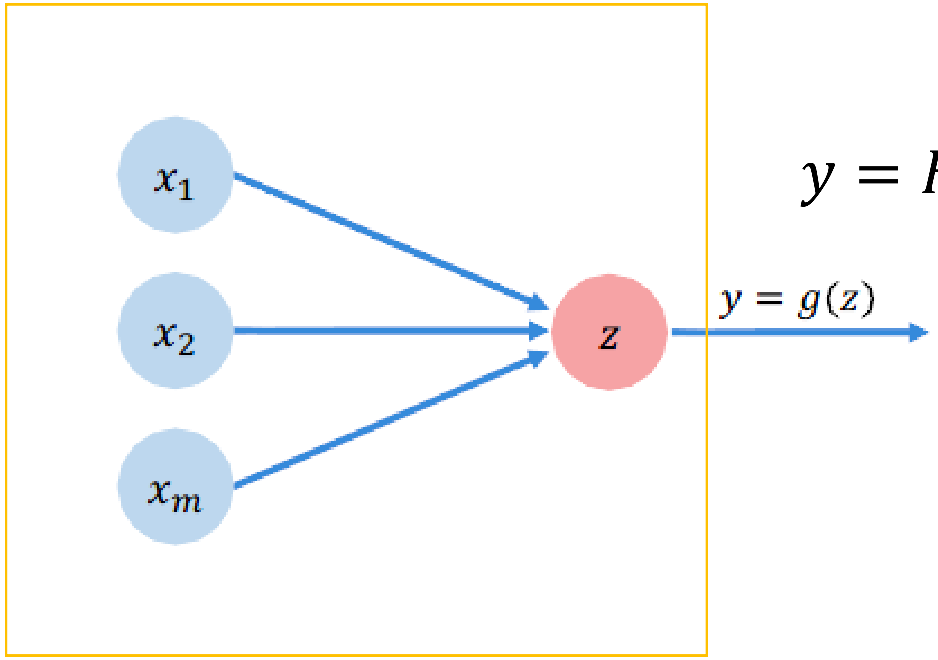
- Muti-variate Input/Scalar Target: $W = [w_1 \quad \cdots \quad w_n], b \quad x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, y$



Keras : Dense(1, input_dim=n)

Graph for LR : Simplified

- Muti-variate Input/Scalar Target:



$$y = H(x) = Wx + b = b + \sum_{j=1}^{n} x_j w_j$$

Keras : Dense(1, input_dim=n)

Multi-variable LR
: n-variable inputs, k-variable output

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

$$W = \begin{bmatrix} w_{11} & \cdots & w_{1n} \\ \cdots & \ddots & \cdots \\ w_{k1} & \cdots & w_{kn} \end{bmatrix} \qquad b = \begin{bmatrix} b_1 \\ \vdots \\ b_k \end{bmatrix} \qquad x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, \qquad y = \begin{bmatrix} y_1 \\ \vdots \\ y_k \end{bmatrix}$$

W : k X n weight matrix

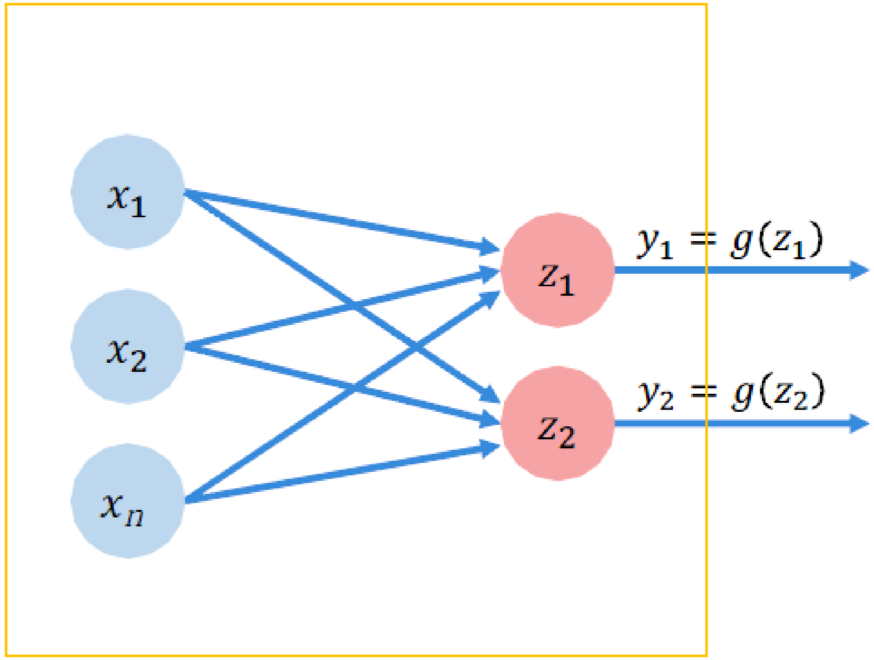
b : k-dim bias v ector

x : n-dim vector

y : k-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

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

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

$$W = \begin{bmatrix} w_{11} & \cdots & w_{1n} \\ \cdots & \ddots & \cdots \\ w_{k1} & \cdots & w_{kn} \end{bmatrix} \quad b = \begin{bmatrix} b_1 \\ \vdots \\ b_k \end{bmatrix} \quad x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, \quad y = \begin{bmatrix} y_1 \\ \vdots \\ y_k \end{bmatrix}$$

W : k X n weight matrix

b : k-dim bias v ector

x : n-dim vector

y : k-dim vector

Optimization

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

n-variable inputs, scalar output

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

n-variable inputs, k-variable output

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

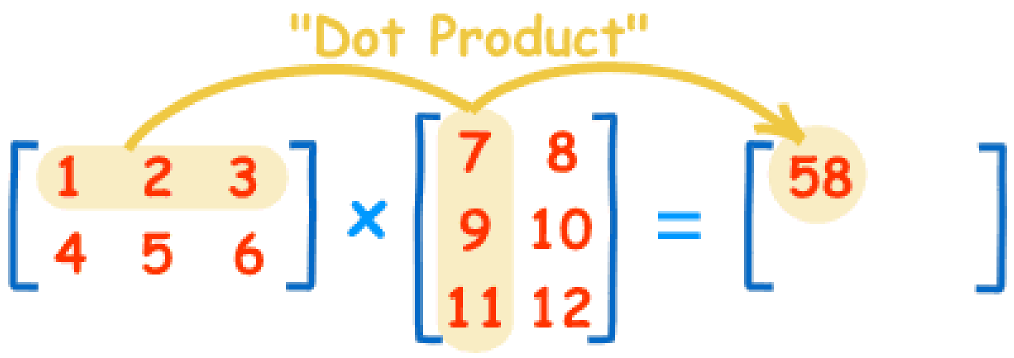
- Gradient Descent

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

Matrix

$$w_1 x_1 + w_2 x_2 + w_3 x_3 + ... + w_n x_n$$

Matrix multiplication



https://www.mathsisfun.com/algebra/matrix-multiplying.html

Hypothesis using matrix

$$w_1x_1 + w_2x_2 + w_3x_3 + \ldots + w_nx_n$$

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

$$H(X) = XW$$

$$X = \begin{bmatrix} x_1 & \cdots & x_n \end{bmatrix} \qquad W = \begin{bmatrix} w_1 \\ \vdots \\ w_n \end{bmatrix}$$

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

https://www.symbolab.com/solver/matrix-calculator

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

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

$$H(X) = XW$$

https://www.symbolab.com/solver/matrix-calculator

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

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

http://college.cengage.com/mathematics/brase/understandable_statistics/7e/students/datasets/mlr/frames/frame.html

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

$$w_1x_1 + w_2x_2 + w_3x_3 + \ldots + w_nx_n$$

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

$$H(X) = XW$$

https://www.symbolab.com/solver/matrix-calculator

Hypothesis using matrix

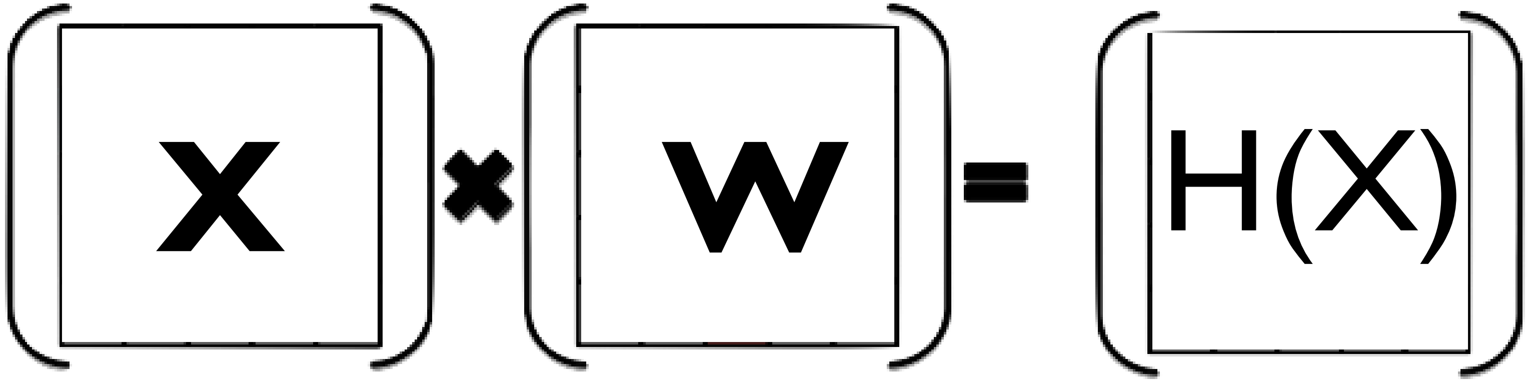
$$\begin{pmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \\ x_{41} & x_{42} & x_{43} \\ x_{51} & x_{52} & x_{53} \end{pmatrix} \cdot \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix} = \begin{pmatrix} x_{11}w_1 + x_{12}w_2 + x_{13}w_3 \\ x_{21}w_1 + x_{22}w_2 +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

[5, 3] **[3, 1]** [5, 1]

$$H(X) = XW$$

https://www.symbolab.com/solver/matrix-calculator

Hypothesis using matrix



[5, 3]　　[?, ?]　　[5, 1]

$$H(X) = XW$$

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]

$$H(X) = XW$$

https://www.symbolab.com/solver/matrix-calculator

Hypothesis using Matrix (n output)

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

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

$$H(X) = XW$$

https://www.symbolab.com/solver/matrix-calculator

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]

$$H(X) = XW$$

https://www.symbolab.com/solver/matrix-calculator

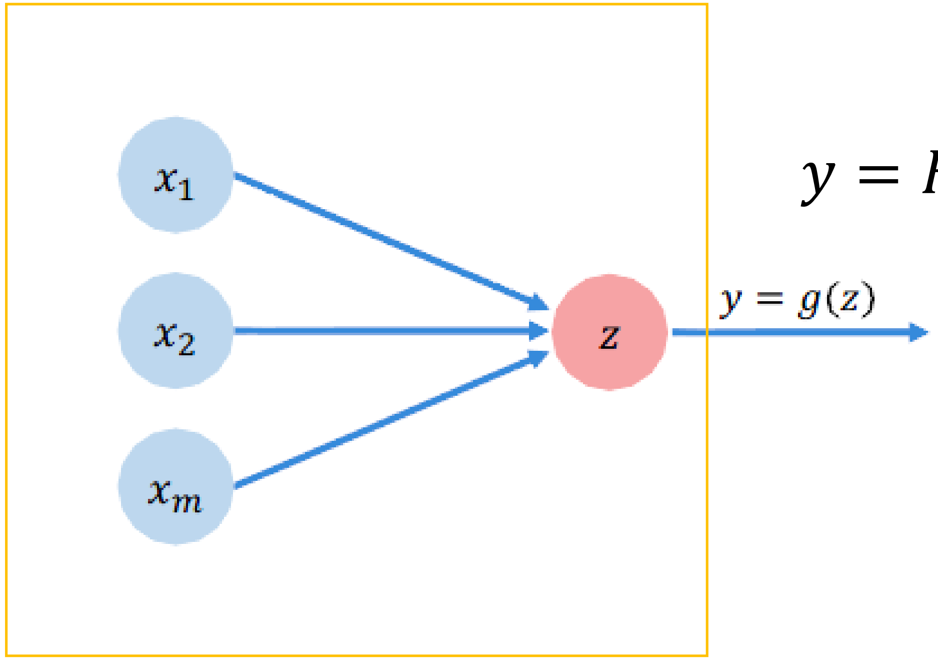
WX vs **XW**

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

$$H(X) = XW$$

Graph for LR : Simplified

- Muti-variate Input/Scalar Target:



$$y = H(x) = Wx + b = b + \sum_{j=1}^{n} x_j w_j$$

Keras : Dense(1, input_dim=n)

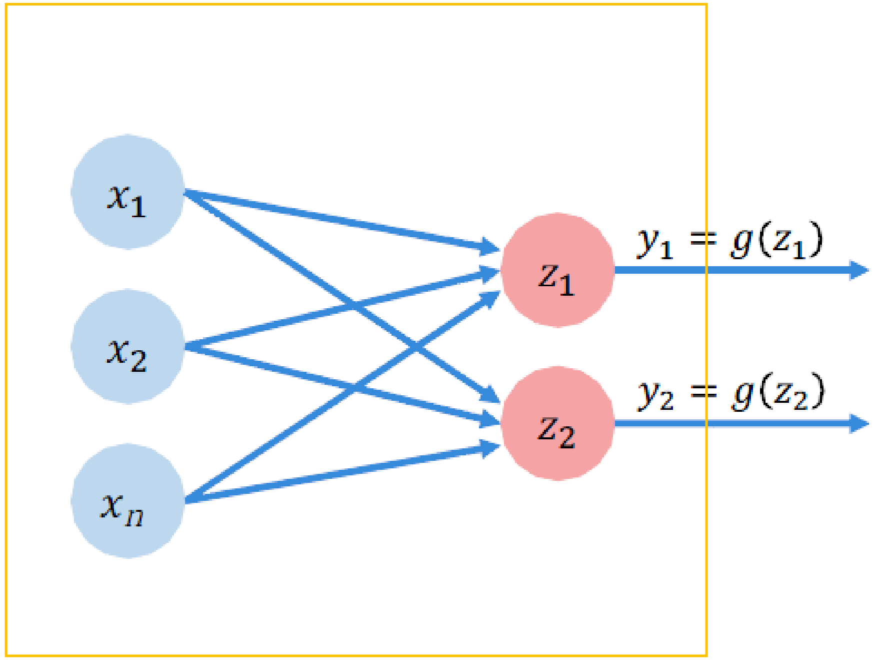
Linear Regression : Summary

- Muti-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: $H(x) = Wx + b$



Keras : Dense(2, input_dim=n)

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

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

W : k x n weight matrix

b : k-dim bias vector

Linear Regression : Summary

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