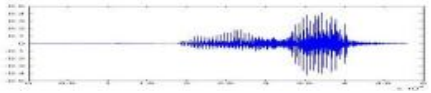
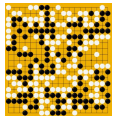
A Brief Introduction to ML

credit:http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML2020/introduction.pdf
https://speech.ee.ntu.edu.tw/~hylee/ml/ml2021-course-data/regression%20(v16).pdf
https://www.reddit.com/r/ProgrammerHumor/comments/dw1yoj/my_requirement_is_fairly_easy_and_i_already_have/

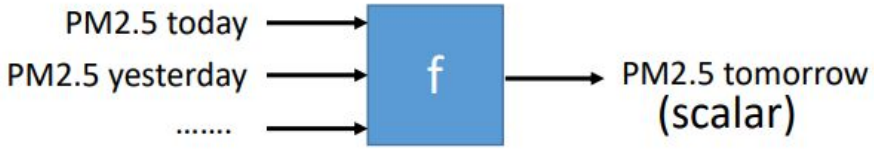
Machine learning ->looking for a function

- Speech Recognition : f () = “How are you”
- Image Recognition :f () = “Cat”
- Playing Go :f () = “5-5”
- Dialogue System :f (“How are you?”) = “I am fine.”

Types of Functions

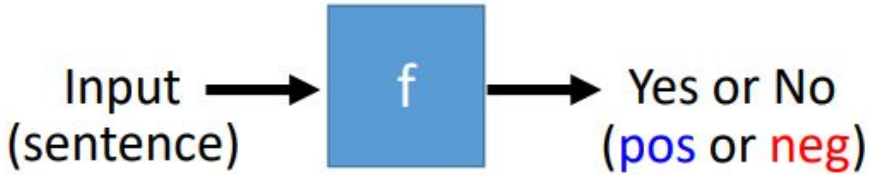
1.Regression

Output is a scalar



2.Classification

Output is a class



3. Harder Problems….
Generation …..

Case Study

We have a gold price data

	date	nominal_effective_exchange_rate	effective_federal_funds_rate	cpi_index	djia_adj_close	gold_price
0	1985-02-01	171.491304	8.35	105.300	1286.770020	299.1
1	1985-03-01	176.187000	8.50	105.800	1284.010010	303.9
2	1985-05-01	170.949545	8.27	106.500	1258.060059	316.4
3	1985-06-01	171.393913	7.97	107.300	1315.410034	316.5
4	1985-08-01	164.653478	7.88	108.300	1347.449951	329.8
...	...	...	...	...	...	...
291	2019-02-01	117.455217	2.40	262.284	24999.669922	1320.1
292	2019-03-01	117.676500	2.40	263.057	25916.000000	1300.9
293	2019-05-01	118.825000	2.42	264.452	26592.910156	1284.0
294	2019-06-01	119.594783	2.39	265.137	24815.039063	1359.0
295	2019-08-01	118.413913	2.40	267.101	26864.269531	1498.8

296 rows × 6 columns

1. Function with Unknown Parameters

y= f(

	date	nominal_effective_exchange_rate	effective_federal_funds_rate	cpi_index	djia_adj_close	gold_price
0	1985-02-01	171.491304	8.35	105.300	1286.770020	299.1
1	1985-03-01	176.187000	8.50	105.800	1284.010010	303.9
2	1985-05-01	170.949545	8.27	106.500	1258.060059	316.4
3	1985-06-01	171.393913	7.97	107.300	1315.410034	316.5
4	1985-08-01	164.653478	7.88	108.300	1347.449951	329.8
...	...	...	...	...	...	...
291	2019-02-01	117.455217	2.40	262.284	24999.669922	1320.1
292	2019-03-01	117.676500	2.40	263.057	25916.000000	1300.9
293	2019-05-01	118.825000	2.42	264.452	26592.910156	1284.0
294	2019-06-01	119.594783	2.39	265.137	24815.039063	1359.0
295	2019-08-01	118.413913	2.40	267.101	26864.269531	1498.8

296 rows × 6 columns

)

$y=wx_1+b$ —model

y is output, the gold price of 3/1/1986

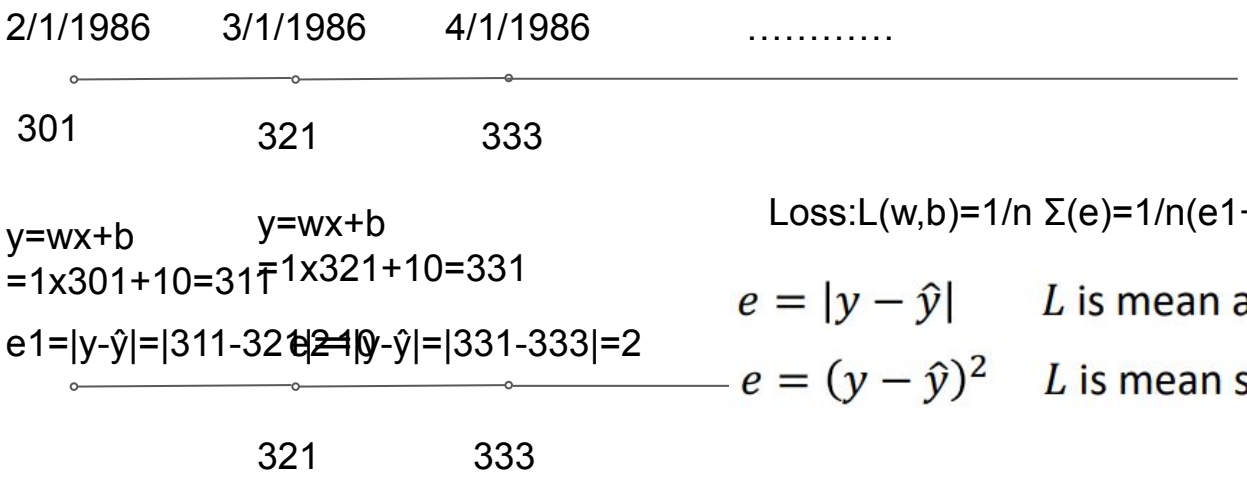
X_1 is the input, the gold price of 2/1/1986

w (weight) and b(bias) are unknown parameters learned from data

2.Define the Loss Function

Loss Function: L(w,b)

L(1,10) -> w=1,b=10



Loss:L(w,b)=1/n Σ(e)=1/n(e1+e2+...+en)

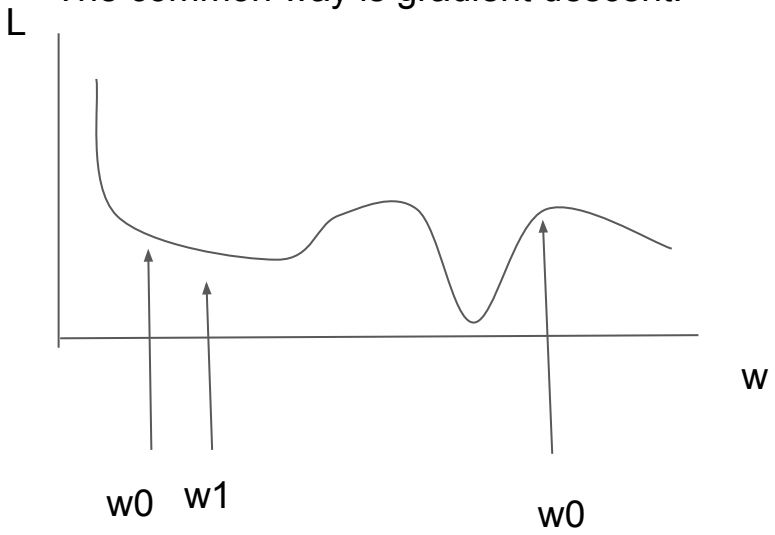
$e = |y - \hat{y}|$ L is mean absolute error (MAE)

$e = (y - \hat{y})^2$ L is mean square error (MSE)

3.Optimization

We need to find the w*,b*=argminL(w,b)

The common way is gradient descent.



1. Randomly pick a initial value
2. Calculate the partial derivative dL/dw
   a. If dL/dw is negative go right
   b. If dL/dw is positive go left

Potential Problem？

w1=w0+ηdL/dw η: learning rate (hyperparemeter)

Machine Learning is as simple as putting elephant into fridge

y=wx+b

L(w,b)=1/n Σ(e)

w=1.1,b=20

Step 1: function with unknown

Step 2: define loss from training data

Step 3: optimization

1. Open the door

2. Put it in

3. Close the door

How to improve?

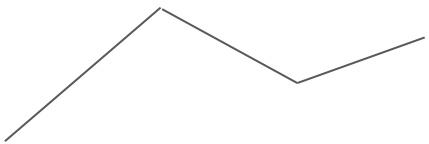
We can use more inputs.

New function: We use the data from previous year instead of the data from previous month.
$y=\Sigma(w_i x_i)+b$
1 to 12

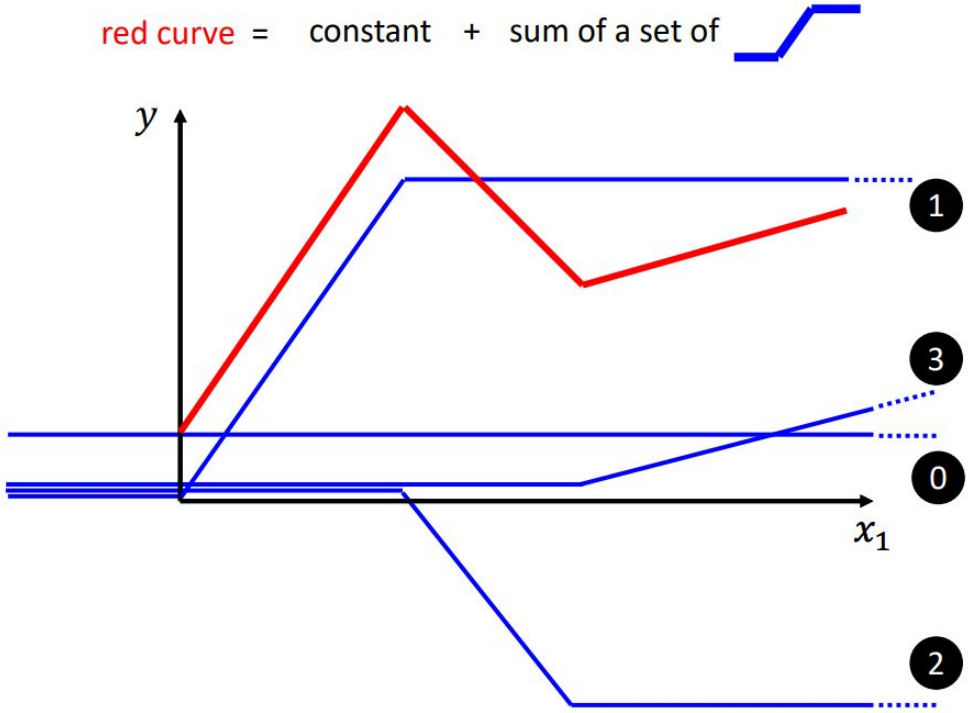
Or we can use the feature in other columns

But it is still a linear model…

If we have a function like below it is hard for us fit it by $y=\Sigma(w_i x_i)+b$



New Solution



Sigmoid Function

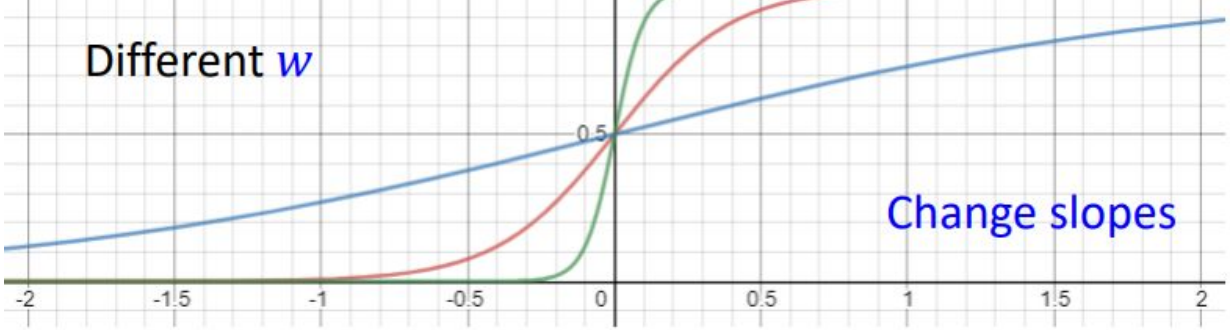
sum of a set of 

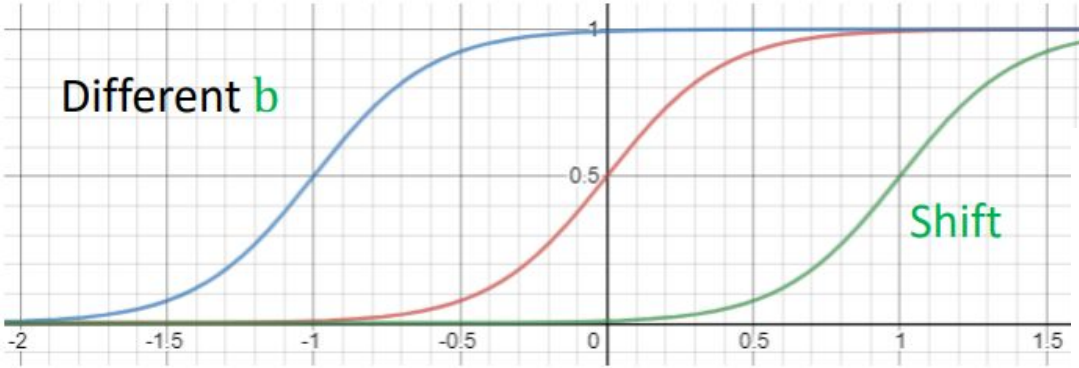
(activation function):sigmoid function $S(x) = \frac{1}{1+e^{-x}}$

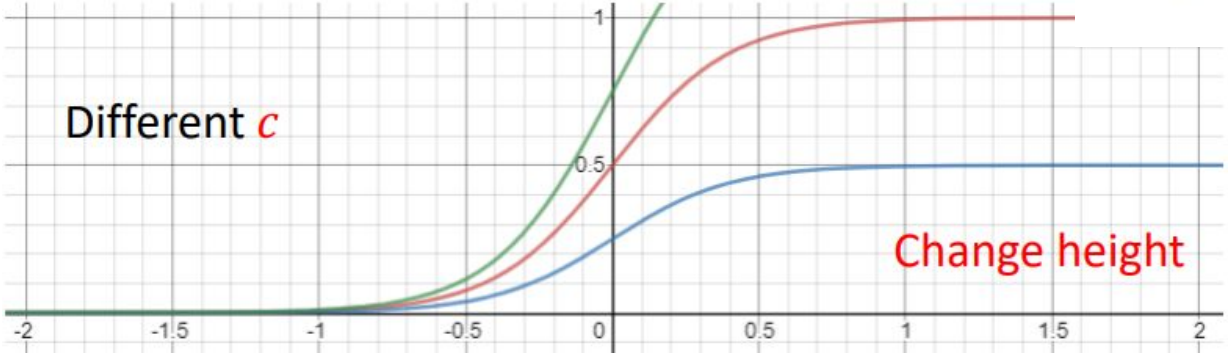
Our model becomes:

$$y = c\frac{1}{1+e^{-(b+wx_1)}}$$

$$= c\ sigmoid(b + wx_1)$$

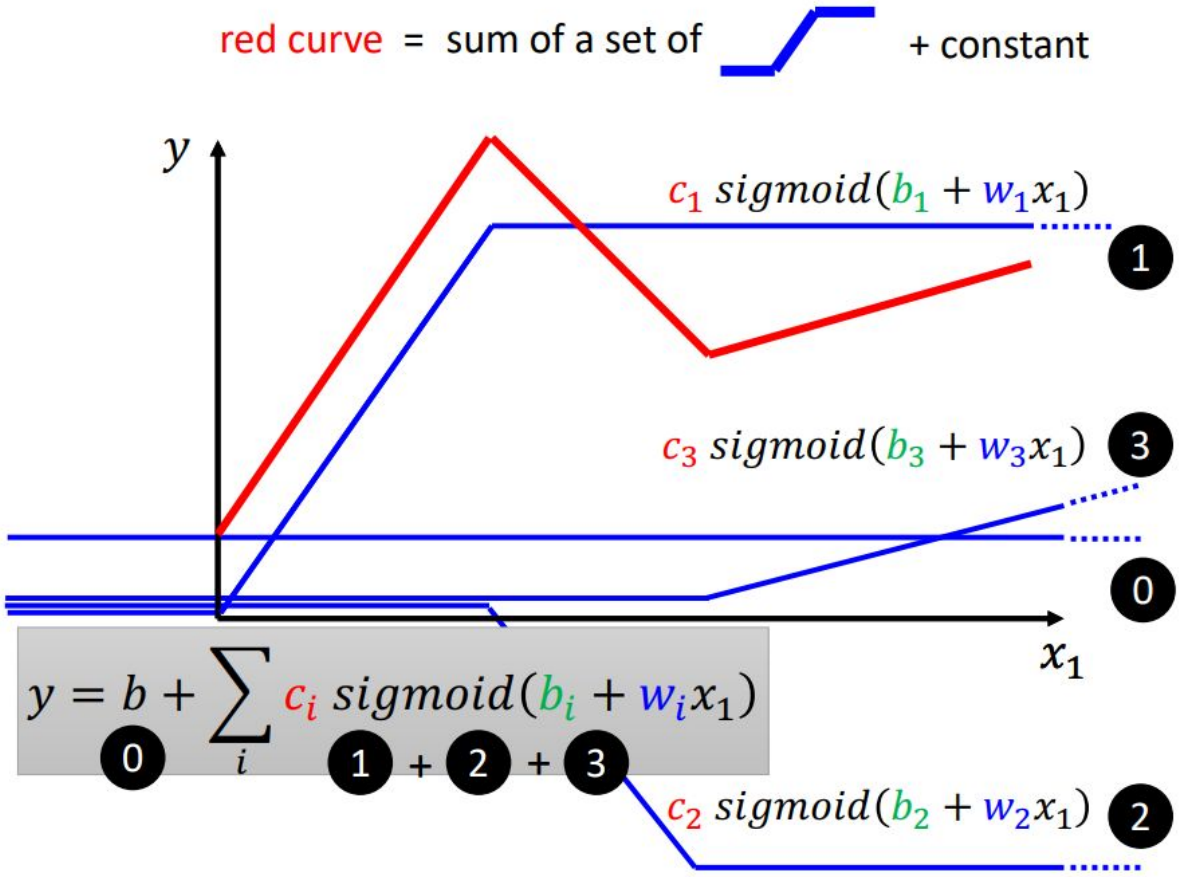






$$y = c\frac{1}{1+e^{-(b+wx_1)}}$$

$$= c\ sigmoid(b + wx_1)$$


red curve = sum of a set of
+ constant
y
$c_1\ sigmoid(b_1 + w_1 x_1)$
1
$c_3\ sigmoid(b_3 + w_3 x_1)$
3
0
x_1
$y = b + \sum_i c_i\ sigmoid(b_i + w_i x_1)$
0
1 + 2 + 3
$c_2\ sigmoid(b_2 + w_2 x_1)$
2

We need to use a set of sigmoid function and each function have different parameters. Suppose we have i sigmoid functions and j features.

For a single input we have: $y=b+\sum_i c_i \text{ sigmoid}(w_i x+b_i)$

For multiple inputs: $y=b+\sum_i c_i \text{ sigmoid}(\sum_j w_{ij} x_j+b_i)$

$$y = b + \sum_i c_i \; sigmoid\left(b_i + \sum_j w_{ij} x_j\right)$$

j: 1,2,3
no. of features

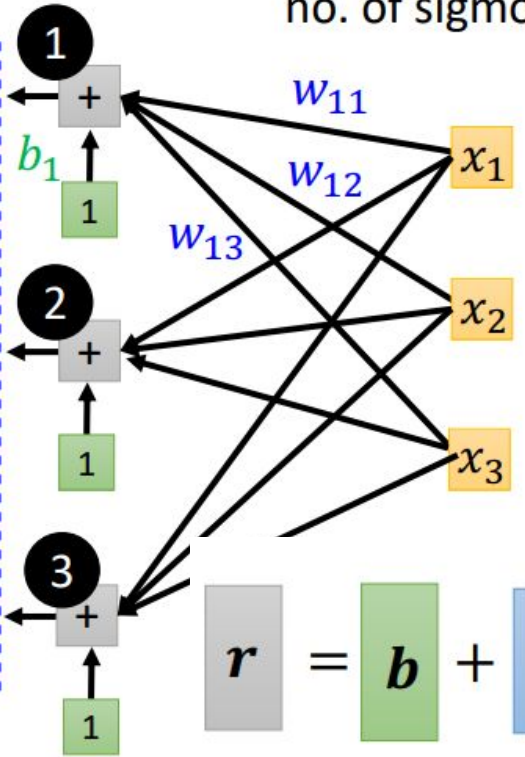
i: 1,2,3
no. of sigmoid

$r_1 = b_1 + w_{11}x_1 + w_{12}x_2 + w_{13}x_3$

w_{ij}: weight for x_j for i-th sigmoid

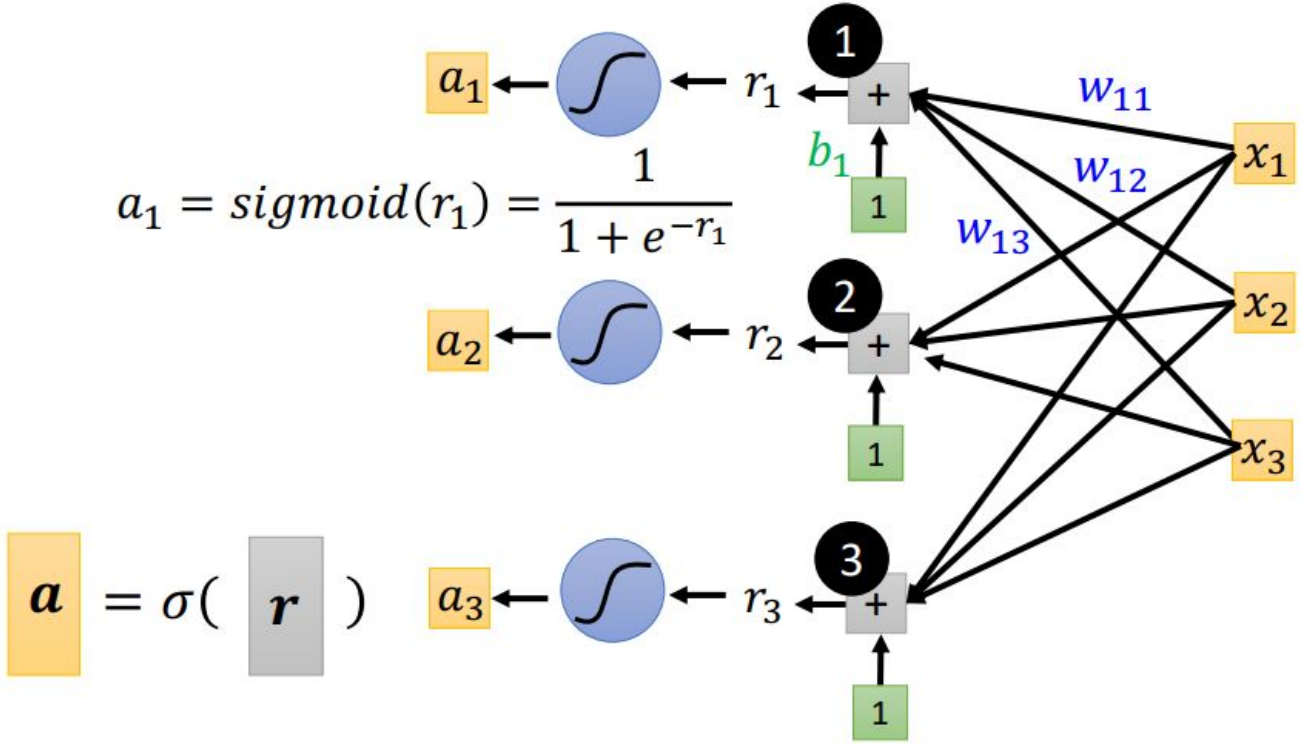
$r_2 = b_2 + w_{21}x_1 + w_{22}x_2 + w_{23}x_3$

$r_3 = b_3 + w_{31}x_1 + w_{32}x_2 + w_{33}x_3$



$$\boldsymbol{r} = \boldsymbol{b} + W\,\boldsymbol{x}$$

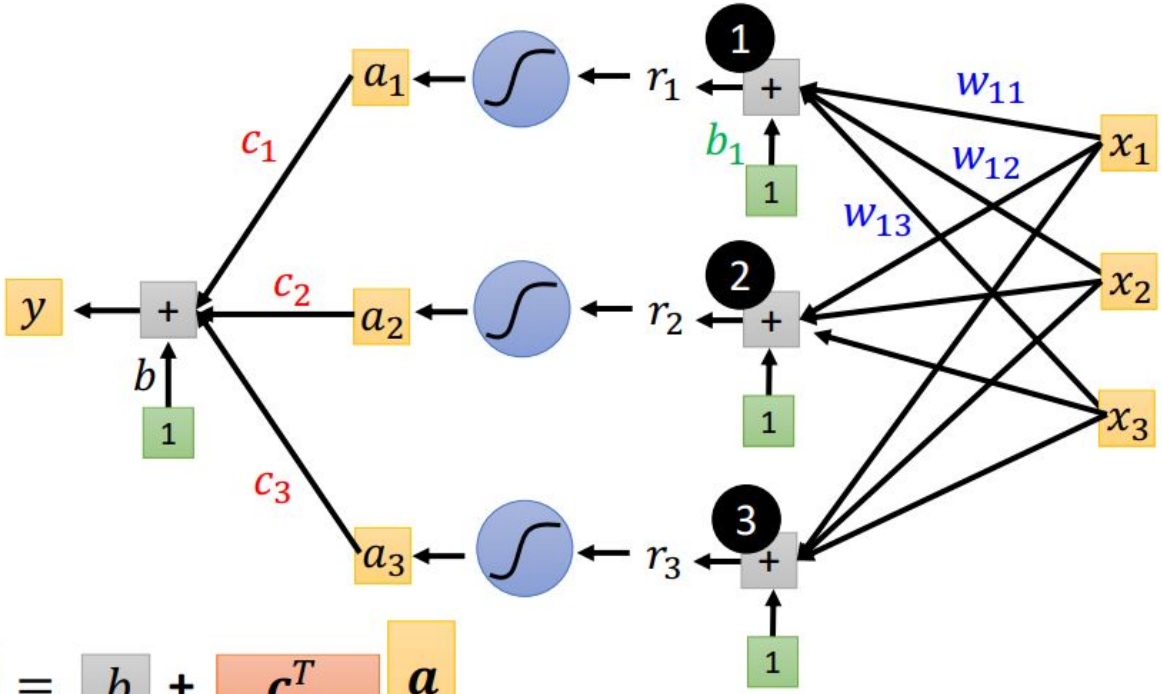
$$y = b + \sum_i c_i \; sigmoid\left(b_i + \sum_j w_{ij} x_j\right) \qquad i: 1,2,3 \quad j: 1,2,3$$



$$y = b + \sum_i c_i \, sigmoid\left(b_i + \sum_j w_{ij} x_j\right)$$

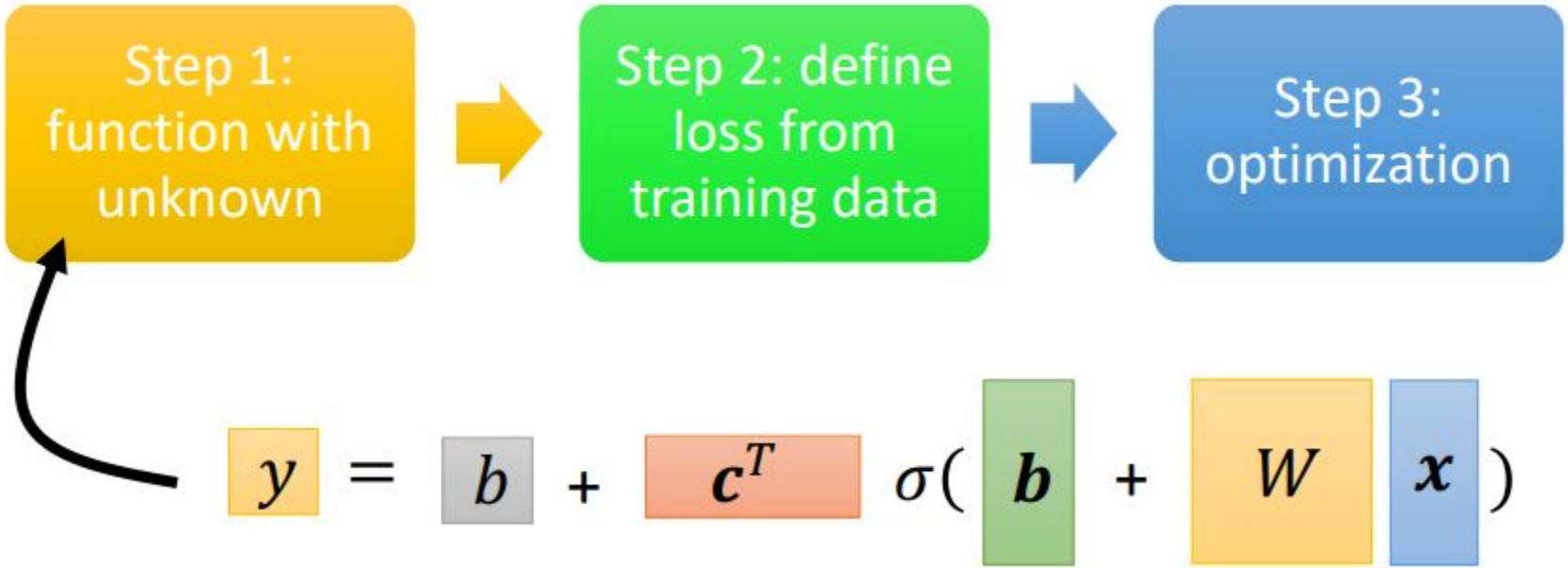
i: 1,2,3

j: 1,2,3

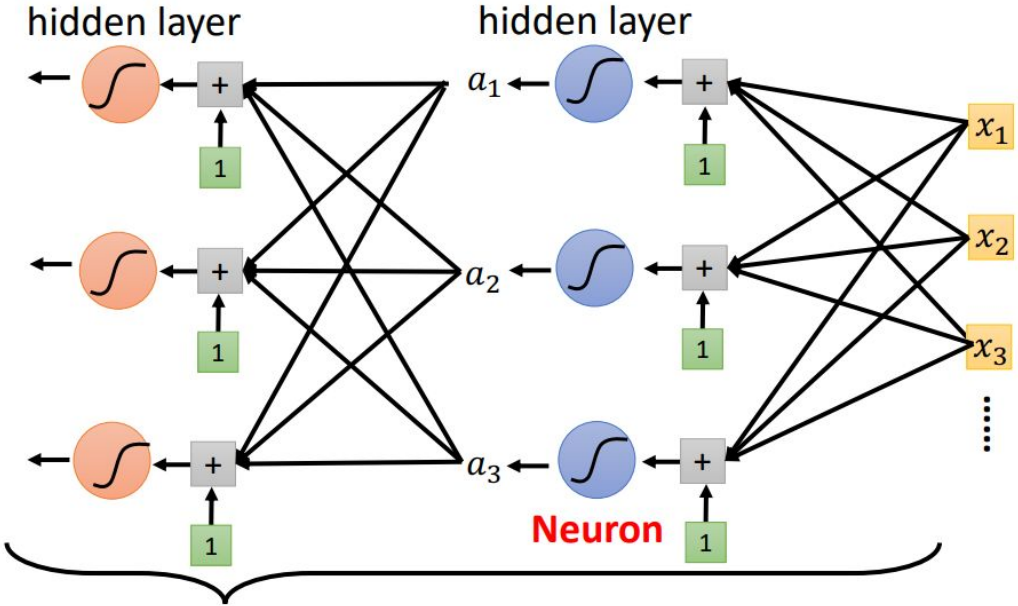


$$y = b + \boldsymbol{c}^T \boldsymbol{a}$$

Back to Framework



Deep Learning



Neural Network This mimics human brains … (???)

Many layers means **Deep** ➡ Deep Learning