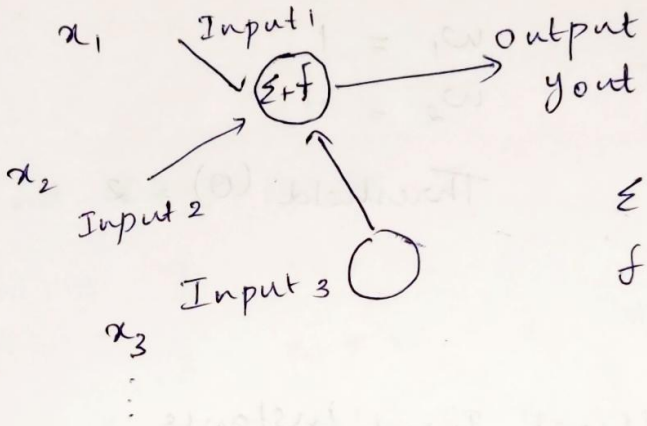
7/9

Deep Learning

Basics of Neural network

- MP Neuron model - First foundation
- Single Layer perceptron neural network

- MP Neuron - synapsis, dendrites, soma
  n number of inputs



Σ - Summation
f - Activation
- Relu
- sigmoid
- Soft max

M/P neuron modul
AND gate
OR gate
- accepts only binary input and binary o/put

AND gate

x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

$y = x_1 \cdot w_1 + x_2 w_2$

$f(x) = x_1 \times w_1 + x_2 w_2$

$$b(x) = \sum_{i=1}^{n} x_i w_i$$

Summation

Activation function ($\int$)
- Buiary
- Sigmoid
- Soft max
- Relu
- unit step
- tanh

$$= \begin{cases} 1, g(f(x)) \geq \theta \\ 0, g(f(x)) \leq \theta \end{cases}$$

θ - Threshold values

Construct the m/p neuron model for AND gate

↳

x_1	x_2	y
Input		output
0	0	0
0	1	0
1	0	0
1	1	1

$w_1 = 1$

$w_2 = 1$

Threshold (θ) = 2

First observation / First Input instance

$x_1 = 0 \quad x_2 = 0 \quad w_1 = 1 \quad w_2 = 1 \Rightarrow$ Parameters

Summation

$$f(x) = \sum_{i=1}^{n} x_i w_i$$

$$= x_1 w_1 + x_2 w_2$$

$$= 0(1) + 0(1) = 0$$

Activation function

$$g(x) = \begin{cases} 1, & f(x) \geq \theta \\ 0, & f(x) < \theta \end{cases}$$

$$= \begin{cases} 1, \geq 2 \\ 0 < 2 \end{cases}$$

$$g(f(x)) = y_{out} = 0$$

x_1	x_2	y actual	y_{out} predicted
0	0	0	0
0	1	0	0
1	0	0	0

Second Observation

$$y = x_i * w_i$$

$$y = x_1 * w_1 + x_2 \times w_2$$

$$= 0(1) + 1(1)$$

$$f(x) = 1$$

Activation $y_{out} = 0$

Third observation

$x_1 = 1 \quad x_2 = 0 \quad w_1 = w_2 = 1$ ~~0=0~~

$$\sum_{i=1}^{n} x_i w_i$$

$$= x_1 w_1 + x_1 w_2$$

$$= (1)(1) + (0)(0)$$

$$\therefore f(x) = 1$$

Activation $y_{out} = 0$

Fourth observation

$x_1 = 1 ; \quad x_2 = 1 , \quad w_1 = w_2 = 1$

$$f(x) = x_1 w_1 + x_2 w_2$$
$$= (1 \times 1) + (1 \times 1)$$
$$f(x) = 2$$
$$y_{out} = 1$$

Construct a mp model for OR gate $w_1, w_2 = 2$
threshold = 1

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

to

No learning,
No error

I $f(x) = x_1 w_1 + x_2 w_2$
$= 0(2) + 0(2) = 0$

Activation fn: $\begin{cases} 1, & \geq 1 \\ 0, & < 1 \end{cases}$

$y_{out} = 0$

II $f(x) = x_1 w_1 + x_2 w_2$
$= 0(2) + 1(2) = 2$

Activation fn: $\begin{cases} 1, & \geq 1 \\ 0, & < 1 \end{cases}$

$y_{out} = 1$

iii $f(x) = 1(2) + 0(2) = 2$

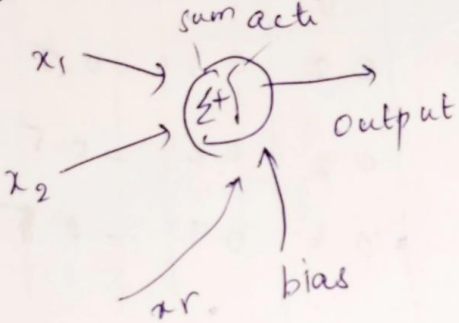
$y_{out} = 1$

iv $f(x) = 1(2) + 1(2) = 4$

$y_{out} = 1$

x_1	x_2	y_{act}	y_{pred}
0	0	0	0
0	1	1	1
1	0	1	1
1	1	1	

Single layer perceptron (Delta rule)



- Only one computational layer

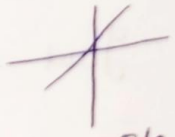
error = Actual - predicted

Learning rate (l) → 0 to 1 range

0.4, 0.5, 0.6 - Stable learning rates.

$f(x) = y_{out} = x_i \cdot w_i$

becomes : $x_i \times$ learning rate $\times$ error rate $\times w_i$



$y = mx + c$

O/P ✓ Slope Intercept bias.

AND gate

x_1	x_2	x_0	y	y_{out}	error actual - predicted	w_1	w_2	w_0

Construct single layer perceptron for AND gate

$\alpha = 0.4$ (learning rate) $\quad w_1 = 0.2 \quad w_2 = 0.3 \quad w_0 = -0.5 \quad x_0 = 1$ bias input

Activation function (unit step)

$$= \begin{cases} 1, & f(x) > 0 \\ 0, & f(x) \leq 0 \end{cases}$$

AND gate

$$w_{new} = w_{old} + \text{learning rate} \times \text{error} \times x_i$$

Weight updation

x_1	x_2	x_0	Actual y	predicted y_{out}	error $a-p$	w_1	w_2	w_0
0	0	1	0	0	0	0.2	0.3	-0.5
0	1	1	0	0	0	0.2	0.3	-0.5
1	0	1	0	0	0	0.2	0.3	-0.5
1	1	1	1	0	1	0.6	0.7	-0.4

$$f(x) = x_i w_i + (\text{bias input})(\text{bias weight})$$

I. $y_{out} = 0(0.2) + 0(0.3) + 1(-0.5)$

$= -0.5 \Rightarrow 0$

II. $y_{out} = 0(0.2) + 1(0.3) + 1(-0.5)$

$= 0.3 - 0.5 = -0.2 \Rightarrow 0$

III. $y_{out} = 1(0.2) + 0(0.3) + 1(-0.5)$

$= -0.3 \Rightarrow 0$

IV. $y_{out} = 1(0.2) + 0(0.3) + 1(-0.5)$

$= 0 \Rightarrow 0$

$$w_1 = 0.2 + 0.4(1)(1) = 0.6$$

$$w_2 = 0.3 + 0.4(1)(1) = 0.7$$

$$w_0 = -0.5 + 0.4(1)(1)$$

$$= -0.1$$

OR gate

x_1	x_2	x_0	Actual y	predicted y_{out}	error $a-p$	w_1	w_2	w_0
						0.2	0.3	-0.5
0	0	1	0	0	0			
0	1	1	1	0	1	0.2	0.7	-0.1
1	0	1	1	1	0	0.2	0.7	-0.1
1	1	1	1	1	0	0.2	0.7	-0.1

I $y_{out} = 0(0.2) + 0(0.3) + 1(-0.5)$

$= -0.5 \Rightarrow 0$

II $y_{out} = 0(0.2) + 1(0.3) + 1(-0.7)$

$= -0.2 \Rightarrow 0$

$$w_1 = 0.2 + 0.4(1)(0)$$

$$= 0.2$$

$$w_2 = 0.3 + 0.4(1)(1)$$

$$= 0.7$$

$$w_0 = -0.5 + 0.4(1)(1)$$

$$= -0.1$$

III $y_{out} = 1(0.2) + 0(0.7) + 1(-0.1)$

$= 0.3 \Rightarrow 1$

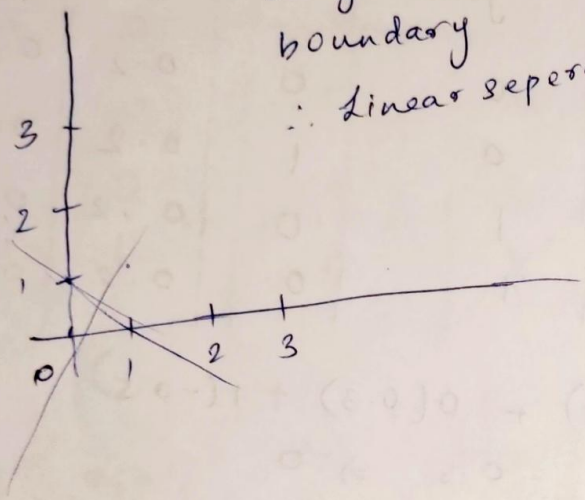
IV $y_{out} = 1(0.2) + 1(0.7) + 1(-0.1) = 0.8 \Rightarrow 1$

Linear seperable

		AND	OR	XOR
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

→ Not Linear seperable

Single boundary
∴ Linear seperable



10.09.22

Neural Network
- MC neural network - Hebbian learning
- Single layer Perceptron

Hebbian learning

AND gate:

x_1	x_2	O/P
0	0	0
0	1	0
1	0	0
1	1	1

SLP

$w_{new} = w_{old} + \Delta w \times input \times learning\ rate$

HL

$w_{new} = w_{old} + \Delta w_i$

$\Delta w_i = x_i \times y_i$

y_i → Actual O/P

We use bipolar input and bipolar o/p instead of binary i/p and o/p

[Else $x_i x y_i = 0$ if $x_i = 0$]

x_1	x_2	o/p
-1	-1	-1
-1	1	-1
1	-1	-1
1	1	1

Activation function $\begin{cases} 1, & \geq 2 \\ 0, & < 2 \end{cases}$

$w_1 = w_2 = 0$

x_1	x_2	Bias (x_b)	Actual o/p	Δw_1	Δw_2	Δw_b	w_1 w_2 w_b
							1 1 -1
-1	-1	1	-1	1	1	-1	2 0 -2
-1	1	1	-1	1	-1	-1	+1 1 -3
1	-1	1	-1	-1	1	-1	2 2 -2
1	1	1	1	1	1	1	

2, 2, -2

Q. Construct a Hebbian learning Rule for OR gate.

OR gate:

x_1	x_2	o/p
-1	-1	-1
-1	1	1
1	-1	1
1	1	1

Activation function $\begin{cases} 1, & \geq 2 \\ 0, & < 2 \end{cases}$

$w_1 = w_2 = 0$

x_1	x_2	x_b	Actual o/p	Δw_1	Δw_2	Δw_b	w_1	w_2	w_b
							1	1	-1
-1	-1	1	-1	1	1	-1			
							0	2	0
-1	1	1	1	-1	1	1			
							1	1	1
1	-1	1	1	1	-1	1			
							2	2	2
1	1	1	1	1	1	1			

o op

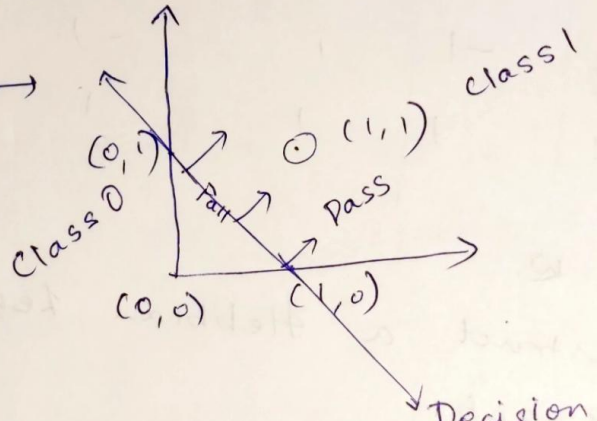
$y_{pred}(-1,-1) = -2-2+2 = -2 \Rightarrow -1$

$y_{pred}(-1,1) = -2+2+2 = 2 \Rightarrow 1$

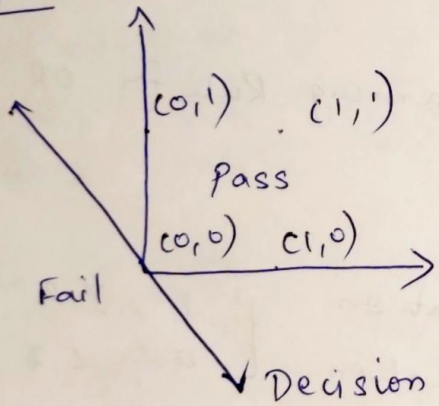
$y_{pred}(1,-1) = 2-2+2 = 2 \Rightarrow 1$

$y_{pred}(1,1) = 2+2+2 = 6 \Rightarrow 1$

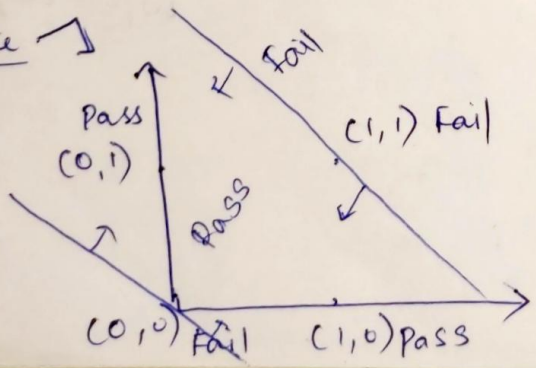
AND Gate →



OR Gate ↴



XOR Gate ↴

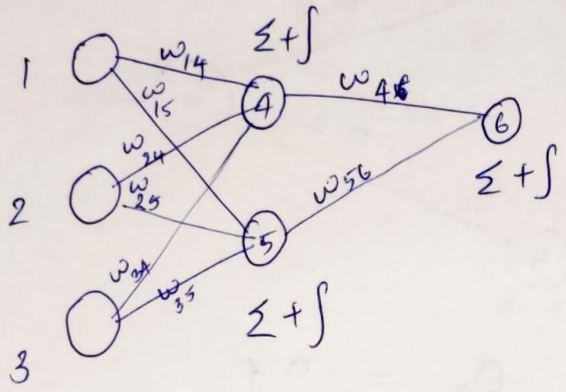


- 2 Decision boundary lines
- Not able to construct a single decision boundary line to seperate 2 classes
- Non linear seperable

Multi layer perceptron

Input Node Output Node

Hidden node



- computations on hidden & o/p
- 2 layer feed forward neural network

- Summation (4, 5, 6) I_j
- Activation (4, 5, 6)
- weight value ($w_{14} \dots w_{56}$)
- Error value (4, 5, 6)
- Bias value (4, 5, 6) ↓ nox

$$I_j = \sum x_i w_{ij}$$

Summation

$$I_4 = x_1 \cdot w_{14} + x_2 w_{24} + x_3 \cdot w_{34} + \theta_4$$

Back propogation

Activation

$$O_4 = \frac{1}{1 + e^{-I_4}} \qquad O_j = \frac{1}{1 + e^{-I_j}}$$

Error value Target = 1

O/p Node $Err_6 = O_6 (1 - O_6)(T_j - O_6)$

T_j - Target / Actual output.

Hidden Node

$$Err_5 = O_5 (1 - O_5)\, Err_6 \cdot w_{56}$$

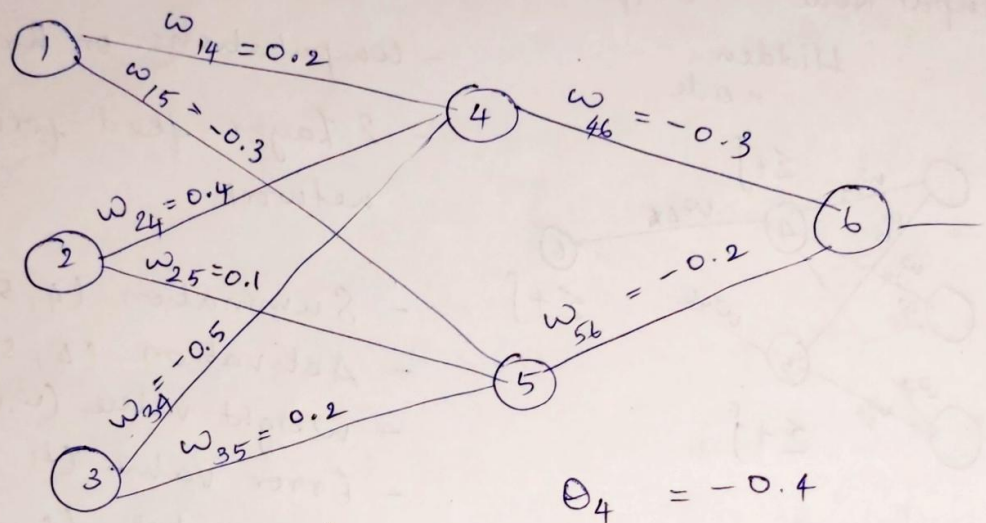
$$Err_4 = O_4 (1 - O_4)\, Err_6 \cdot w_{46}$$

Weight value

New weight = Old weight + Δw_{ij}

Δw_{ij} = learningrate × $error_j$ × x_{i_4}

Construct a multi layer Perceptron



$\eta = 0.9$

$x_1 = 1$
$x_2 = 0$
$x_3 = 1$

$\theta_4 = -0.4$
$\theta_5 = 0.2$
$\theta_6 = 0.1$

$I_4 = 1(0.2) + 0(0.4) + 1(-0.5) + -0.4$
$= 0.2 - 0.5 - 0.4 = -0.7$

$I_5 = 1(-0.3) + 0(0.1) + 1(0.2) + 0.2$
$= 0.1$

$O_4 = \frac{1}{1+e^{-I_4}} = \frac{1}{1+e^{+0.7}} = 0.332$

$O_5 = \frac{1}{1+e^{-I_5}} = \frac{1}{1+e^{-0.1}} = 0.525$

$I_6 = 0.332(-0.3) + 0.525(-0.2) + 0.1$
$= -0.1046$

$$Err_6 = O_6(1-O_6)(T-O_6)$$

$$O_6 = \frac{1}{1+e^{-I_6}} = \frac{1}{1+e^{0.1046}} = 0.473$$

$$Err_6 = 0.473(1-0.473)(1-0.473$$
$$= 0.131$$

$$Err_5 = O_5(1-O_5)\, err_6 \times w_{56}$$
$$= 0.525(1-0.525)(0.131)(-0.2)$$
$$= -0.00653$$

$$Err_4 = 0.332(1-0.332)(0.131)(-0.3)$$
$$= -0.0087$$

$$\text{New } w_{ij} = \text{Old } w_{ij} + \eta \times err_j \times x_i$$

$$w_{46} = -0.3 + 0.9 \times 0.131 \times 0.332 = -0.26$$

$$w_{56} = -0.2 + 0.9 \times 0.131 \times 0.525 = -0.138$$

$$w_{35} = 0.2 + 0.9 \times -0.0065 \times 1 = 0.194$$

$$w_{34} = -0.5 + 0.9 \times -0.0087 \times 1 = -0.507$$

$$w_{25} = 0.1 + 0.9 \times -0.0065 \times 0 = 0.1$$

$$w_{24} = 0.4 + 0.9 \times -0.0087 \times 0 = 0.4$$

$$w_{15} = -0.3 + 0.9 \times -0.0065 \times 1 = -0.305$$

$$w_{14} = 0.2 + 0.9 \times -0.0087 \times 1 = 0.19$$

$$\theta_5 = 0.2 + 0.9 \times -0.0065 = 0.194$$

$$\theta_6 = 0.1 + 0.9 \times 0.131 = 0.2179$$

$$\theta_4 = -0.4 + 0.9 \times -0.0087 = -0.407$$

$$I_4 = 1(0.193) + 0(0.4) + 1(-0.507) - 0.407$$
$$= -0.721$$

$$O_4 = \frac{1}{1 + e^{+0.721}} = 0.327$$

$$I_5 = 1(-0.305) + 0(0.1) + 1(0.195) + 0.195$$
$$= 0.085$$

$$O_5 = \frac{1}{1 + e^{-0.085}} = 0.521$$

$$I_6 = 0.327(-0.261) + 0.521(-0.138) + 0.218$$
$$= 0.061$$

$$O_6 = \frac{1}{1 + e^{0.061}} = 0.515$$

$$err_6 = 0.515(1 - 0.515)(1 - 0.515) = 0.121$$

$$err_5 = 0.521(1 - 0.521)(0.121)(-0.138) = -0.004$$

$$err_4 = 0.327(1 - 0.327)(0.121)(-0.261) = -0.006$$

$$w_{46} = -0.261 + (0.9)(0.121)(0.327) = -0.225$$

$$w_{56} = -0.138 + (0.9)(0.121)(0.521) = -0.081$$

$$w_{14} = 0.193 + (0.9)(-0.006) \times 1 = 0.188$$

$$w_{15} = -0.305 + (0.9)(-0.004)(1) = -0.309$$

$w_{24} = 0.4 + (0.9)(-0.006)(0) = 0.4$

$w_{25} = 0.1 + (0.9)(-0.004)(0) = 0.1$

$w_{34} = -0.507 + (0.9)(-0.006)(1) = -0.512$

$w_{35} = 0.195 + (0.9)(-0.004)(1) = 0.191$

$\theta_4 = -0.407 + (0.9)(-0.006) = -0.412$

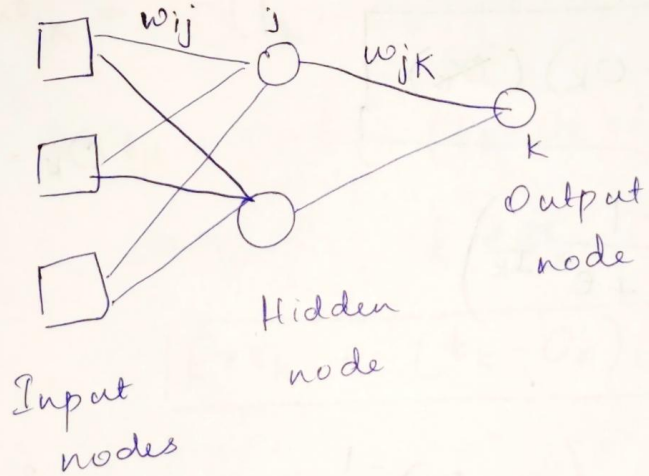
$\theta_5 = 0.193 + (0.9)(-0.004) = 0.191$

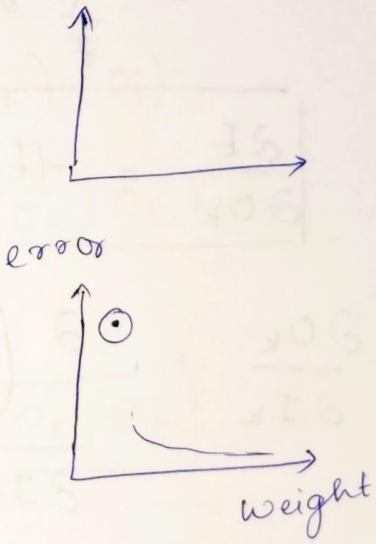
$\theta_6 = 0.218 + (0.9)(0.121) = 0.327$

12/09/22

Backpropogation with gradient Descent

Weight updation Technique





$\therefore \Delta w \propto -\frac{\partial E}{\partial w}$

$\Delta w = -L\left(\frac{\partial E}{\partial w}\right)$

L - learning rate

$w_1 = 0.2$

$w_2 = 0.8$

$w_3 = 0.5$

Error = t - actual

objective: reduce the error

Change the weight w.r.t E and weight

For $error_k \Rightarrow$ check w_{jk}

$$\Delta w_{jk} = \frac{\partial E}{\partial O_k} \cdot \frac{\partial O_k}{\partial I_k} \cdot \frac{\partial I_k}{\partial w_{jk}}$$

Chain rule (Summation, and activation)

$$\frac{\partial E}{\partial O_k} = \frac{\partial \left(\frac{1}{2} (t_k - O_k) \right)^2}{\partial O_k}$$

E = sum-squared error

$(t_k - O_k)^2$

= half sum squared error

$= \frac{1}{2} (t_k - O_k)^2$

$$= \frac{1}{2} \cdot 2 (t_k - O_k)(-1)(\cancel{O_k})$$

$$\boxed{\frac{\partial E}{\partial O_k} = -(t_k - O_k)(\cancel{O_k})}$$

$$O_k = \frac{1}{1 + e^{-I_k}}$$

$$\frac{\partial O_k}{\partial I_k} = \frac{\partial \left(\frac{1}{1 + e^{-I_k}} \right)}{\partial I_k}$$

$$= \frac{\partial (1 + e^{-I_k})^{-1}}{\partial I_k} = (-1)(1 + e^{-I_k})^{-2} (e^{-I_k})(-1)$$

$$= \frac{e^{-I_k}}{(1 + e^{-I_k})^2}$$

$$= \frac{e^{-I_k}}{1 + e^{-I_k}} \cdot \frac{1}{1 + e^{-I_k}} = \frac{e^{-I_k}}{1 + e^{-I_k}} \cdot O_k$$

$$= \left(1 - \frac{1}{1+e^{-I_k}}\right) . O_k$$

$$\boxed{\frac{\partial O_k}{\partial I_k} = (1-O_k)\,O_k}$$

$$\frac{\partial I_k}{\partial w_{jk}} = \frac{\partial (O_j \cdot w_{jk})}{\partial w_{jk}}$$

$$\boxed{\frac{\partial I_k}{\partial w_{jk}} = O_j}$$

$$\Delta w_{jk} = -(t_k - O_k)(1-O_k)(O_k)(O_j)$$

$$\Delta w_{jk} = -l\left(-(t_k - O_k)(1-O_k)O_k\right) \cdot O_j$$

$$\Delta w_{jk} = l\, err_k \cdot O_j$$

$$\boxed{Err_k = (t_k - O_k)(1-O_k)O_k}$$

$$\Delta w \propto -\left(\frac{\partial E}{\partial w}\right)$$

$$\Delta w_{ij} = -l\left(\frac{\partial E}{\partial w_{ij}}\right)$$

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial O_j} \times \frac{\partial O_j}{\partial I_j} \times \frac{\partial I_j}{\partial w_{ij}}$$

$$\frac{\partial E}{\partial O_j} = \frac{\partial E}{\partial O_k} \cdot \frac{\partial O_k}{\partial I_k} \cdot \frac{\partial I_k}{\partial O_j}$$

$$\frac{\partial E}{\partial O_k} = \frac{\partial \left(\frac{1}{2} (t_k - O_k) \right)^2}{\partial O_k} = -(t_k - O_k)$$

$$\frac{\partial O_k}{\partial I_k} = (1 - O_k)(O_k)$$

$$\frac{\partial I_k}{\partial O_j} = \frac{\partial (w_{jk} \times O_j)}{\partial O_j}$$

$$\boxed{\frac{\partial I_k}{\partial O_j} = w_{jk}}$$

$$\frac{\partial O_j}{\partial I_j} = \frac{\partial \left(\cancel{w_{ij}} \frac{1}{1 + e^{-I_j}} \right)}{\partial I_j}$$

$$= \frac{\partial \left(1 + e^{-I_j} \right)^{-1}}{\partial I_j}$$

$$= (-1)\left(1 + e^{-I_j}\right)^{-2} \left(e^{-I_j}\right)(-1)$$

$$= \frac{e^{-I_j}}{\left(1 + e^{-I_j}\right)^2}$$

$$= \frac{e^{-I_j}}{1+e^{-I_j}} \times \frac{1}{1+e^{-I_j}}$$

$$= \frac{e^{-I_j}}{1+e^{-I_j}} \times O_j$$

$$= 1 - \frac{1}{1+e^{-I_j}} \times O_j$$

$$= (1 - O_j)(O_j)$$

$$\frac{\partial I_j}{\partial w_{ij}} = \frac{\partial (O_i \times w_{ij})}{\partial w_{ij}}$$

$$\frac{\partial I_j}{\partial w_{ij}} = O_i$$

$$= -l\left(-((t_k - O_k)(1-O_k)(O_k) \times (w_{jk}) \times (1-O_j)(O_j)\right.$$
$$\left.(O_i)\right.$$

$$= l\underline{(t_k - O_k)(1-O_k)(O_k)} \times w_{jk} \times (1-O_j)(O_j)(O_i)$$

$$= l\,err_k\, w_{jk} \times (1-O_j)(O_j)(O_i)$$

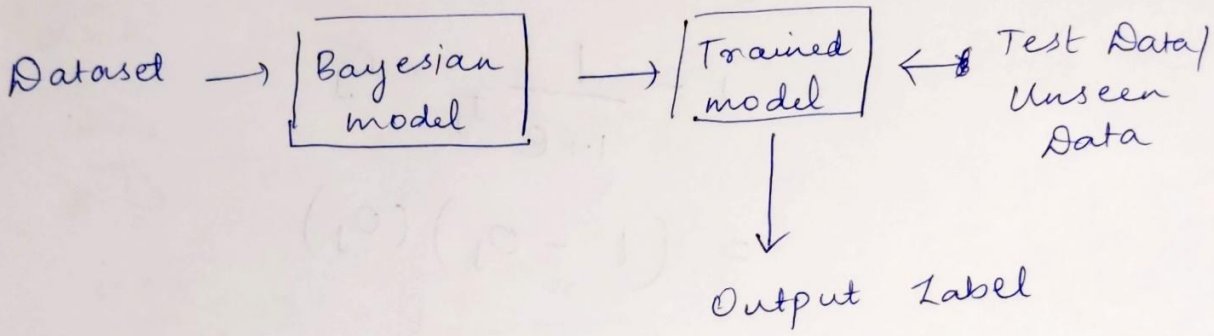
$$Err_j = (w_{jk})(1-O_j)(O_j)$$

14/09/22

Bayesian statistics

Maximum likelihood Estimation

- Statistics model to predict the event
- Supervised learning technique / classification algorithm



Bayesian statistics
↳ Bayes Theorm
- Hypothesis - Bayesian
- Event - ~~ML~~ Statistics
- Class label - ML

Attributes - Evidence

- Prior knowledge
- Likelihood

CGPA / Joboffer

Naive Bayes theorm
- All attributes are independent to each other

$P(H/Evidence)$ Probability of hypothesis given Evidence

Class membership probability value

Posterior probability · Conditional probability

$$P(H/x) = \frac{P(x/H) \cdot P(H)}{P(x)}$$

$P(H)$ → Prior knowledge

$P(x)$ → Evidence

$P(x/H)$ → Likelihood

Hypothesis

x = Evidence

$$P(H/x_1, x_2, x_3 \ldots x_n) = P(x_1/H) \times P(x_2/H) \times \ldots P(x_n/H) \times P(H)$$

Posterior probability = Likelihood × Prior probability

$P(H) \rightarrow P(C)$ (Class label)

	CGPA	Interactiveness	Practical knowledge	Communication skill	Job offer
1.	≥9	Yes	Excellent	Good	Yes
2.	≥8	TVo	Good	Moderate	~~No~~ Yes
3.	≥9	No	Average	Poor	No
4.	<8	NO	Average	Good	NO
5.	≥$8	yes	Good	Moderate	Yes
6.	≥9	Yes	Good	Moderate	Yes
7.	<8	Yes	Good	Poor	NO
8.	~~<8~~ ≥9	~~Yes~~ No	~~Good~~ Excellent	~~Poor~~ Good	Yes ~~No~~ ×
9.	≥8	Yes	Good	Good	Yes ×
10.	≥8	Yes	~~Excellent~~ Good	Good	Yes
	≥9	Yes	Avg	Good	?

$$P(\text{Job offer} = \text{Yes}) = \frac{7}{10}$$

$$P(\text{Job offer} = \text{No}) = \frac{3}{10}$$

CGPA	Job offer = Yes	Job offer = No
≥ 9	3	1
≥ 8	4	—
< 8	0	2

CGPA	Job offer = Yes	Job offer = No
≥ 9	3/7	1/3
≥ 8	4/7	—
< 8	0	2/3

Interactiveness	Job offr = Yes	Job offer = No
Yes	5	1
NO	2	2
Likelihood Yes	5/7	1/3
NO	2/7	2/3

Practical	Yes	NO
Excellent	2/7	0/3
Good	4/7	1/3
Avg	1/7	2/3

Comm.	Yes	NO
Good	4/7	1/3
Moderate	3/7	0/3
Poor	0/7	2/3

$$P(Yes/testdata) = \frac{3}{7} \times \frac{5}{7} \times \frac{2}{7} \times \frac{4}{7} \times \frac{7}{10}$$

$$= 0.0174$$

$$P(No/testdata) = 0.0074$$

Transaction ID	Age	Income	Student	Credit rating	Buys comp
1	Youth	High	No	Fair	No
2	Youth	H	N	E	No
3	Middle	H	No	F	Y
4	Snr	M	No	F	Y
5	Snr	L	yes	F	Y
6	Snr	L	Yes	E	No
7	Midle	L	Yes	E	Y.
8	Youth	M	NO	F	No
9	Youth	L	yes	F	Yes
10	Snr	M	Yes	F	Yes
11	Youth	M	Yes	E	Yes
12	Middle	M	NO	E	Yes
13	Midde	H	Yes	F	No
14	Snr	M	No	Exc	

Test data (x) x = age = youth, income = medium,
student = yes, credit rating = fair

P(Buys computer = Yes) = 9/14

P(Buys computer = No) = 5/14

Age	Yes	No
Youth	2/9	3/5
Middle	4/9	0/5
Sn	3/9	2/5

Income	Yes	No
High	2/9	2/5
Medium	4/9	2/5
Low	3/9	1/5

Student	Yes	No
Yes	6/9	1/5
No	3/9	4/5

Credit rating	Yes	No
Excellent	3/9	3/5
Fair	6/9	2/5

Given a data tuple having the ~~systems~~ values

System 26-30 46K-50K dept, age, salary resp.

	Dept	Status	age	Salary	Count
1.	Sales	Senior	31..35	46-50k	30
2.	Sales	Junior	26.30	26-30k	40
3.	Sales	Jnr	31.35	31-35K	40
4.	Systems	Jnr	21-25	46-50K	20
5.	Systems	Snr	31-35	66-70K	5
6.	Systems	Jnr	26-30	46-50K	3
7.	Systems	Snr	41-45	66-70K	3
8.	Marketing	Snr	36-40	46-50K	10
9.	~~Marketing~~ Secretary	Jnr	31-35	41-45K	4
10.	Secretary	Snr	46-50	36-40K	4
11.	Secretary	Jnr	26-30	26-30K	6

→ Capacity, underfitting, overfitting

↙

Training error

Test error / Generalisation error

Generalisation

Bias — Training error

Variance — Testing error

Underfitting : — High bias

— High variance

Overfitting:

— Low bias

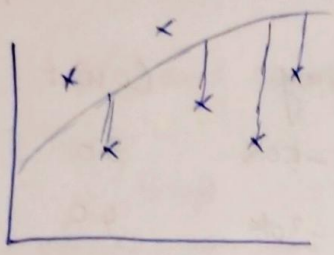
— High variance

Generalised model

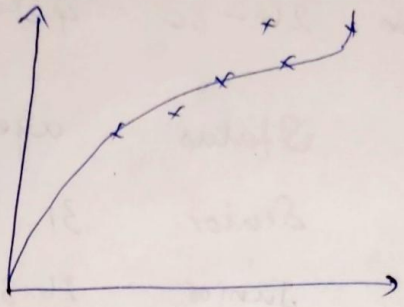
— low bias

— low variance

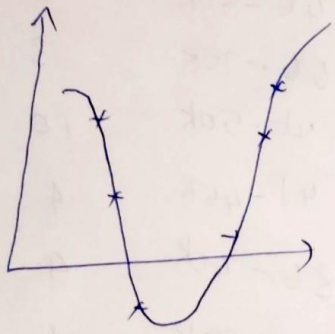
Underfitting

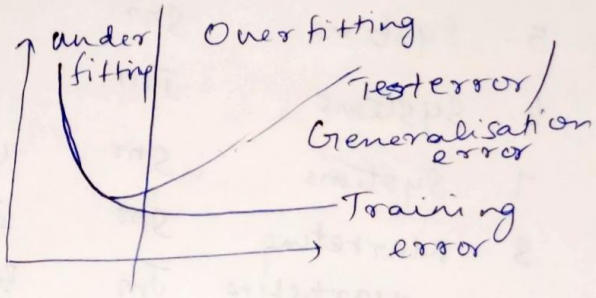


Generalised Model / capacity



Overfitting
(Polynomial degree)





Solution : $P(Snr) = \frac{52}{165}$

$P(Jnr) = \frac{113}{165}$

Department :

	Snr	Jnr
Sales	30/52	80/113
System	8/52	23/113
Marketing	10/52	4/113
Secretary	4/52	6/113

Age

	Snr	Jnr
31 - 35	35/52	44/113
26 - 30	0	49/113
21 - 25	0	20/113
41 - 45	3/52	0
36 - 40	10/52	0
46 - 50	4/52	0

Salary:

	Snr	Jnr
46-50	40/52	23/113
26-30	0	46/113
31-35	0	40/113
66-70	8/52	0
41-45	0	4/113
36-40	4/52	0

$P(\frac{Snr}{Systems, 26-30, 46k-50k})$

$= \frac{8}{52} \times 0 \times \frac{40}{52} \times \frac{52}{165} = 0$

$P(\frac{Jnr}{Systems, 26-30, 46k-50k})$

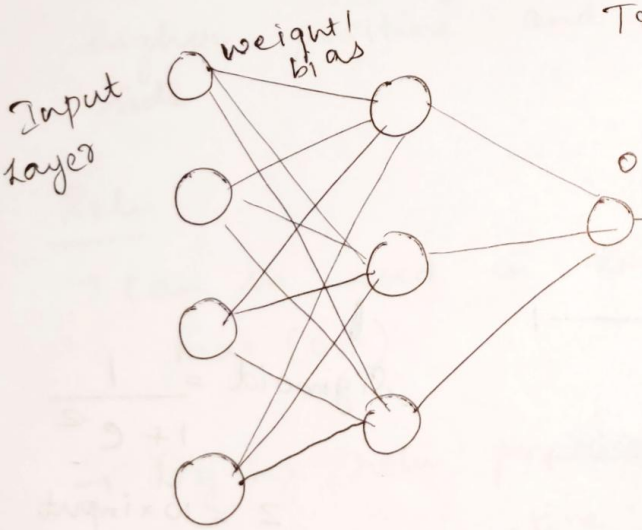
$= \frac{23}{113} \times \frac{49}{113} \times \frac{23}{113} \times \frac{113}{165}$

$= 0.012$

∴ Prediction: Jnr.

24/09

Activation function / Transfer function

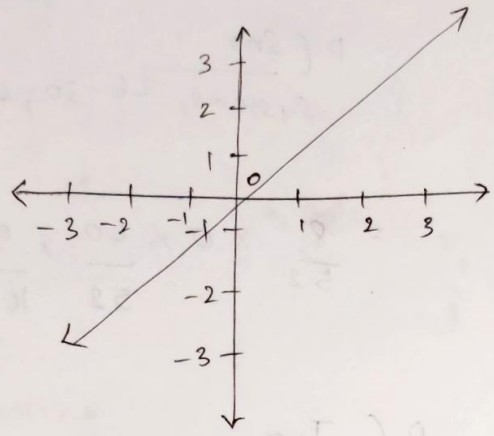


① Sigmoid
② Tanh
③ Relu
④ Threshold
⑤ Binary step

/ Squashing Function

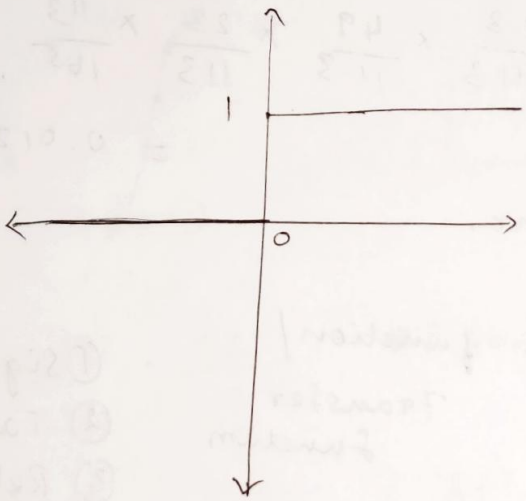
↳ Linear
  ↳ Linear / Identity
  ↳ Binary step
↳ Non Linear

① Linear function



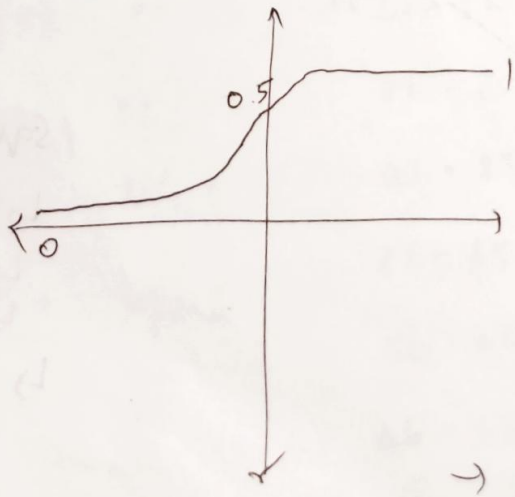
$x = f(x)$

② Binary step activation fn.



$$\begin{cases} 1, & f(x) \geq 0 \\ 0, & f(x) < 0 \end{cases}$$

③ Sigmoid (Non - linear)



$$\text{Sigmoid} = \frac{1}{1+e^{-z}}$$

$z = w \times \text{input} + \text{bias}$

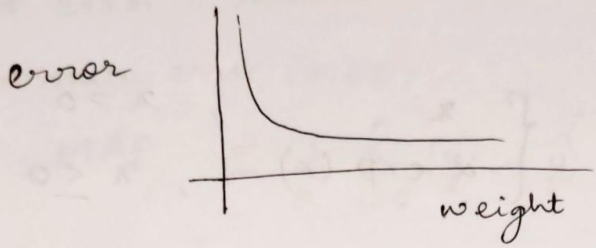
→ between (0 to 1)

→ can be used in o/p layer

→ In binary classification prblm

Drawback:

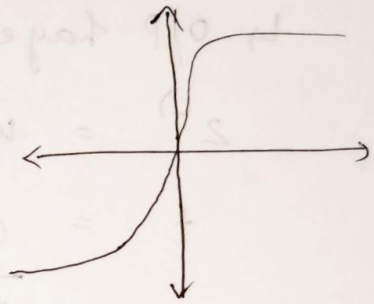
→ Vanishing gradient / Saturation problem
→ No weight updation for higher values, rate of change becomes 0.



→ Mean is around 0.5 ⇒ not zero-centric

Tanh → Tan hypebolic function

→ Non linear adiration function
→ Range can be −1 to 1
→ Mean is zero-centric



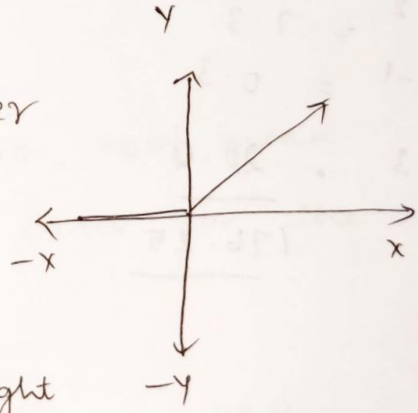
Drawback:

→ Vanishing gradient towards higher positive and higher negative side

$$\tanh(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$$

Relu

→ can be used in hidden layer
max (0, x)

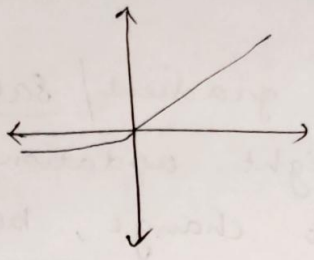


→ Dying relu problem
(Towards negative side, weight is not updated

→ Overcome – (1) Leaky Relu (3) Exponential Relu
(2) Parametric Relu
Leaky

Leaky Relu

$\max(0.01x, x)$



Parametric Relu

$\max(\alpha x, x)$

α - Hyper Parameter

Exponential Relu :

$$\begin{cases} x & , x > 0 \\ \alpha \cdot \exp(x) - 1 & , x \leq 0 \end{cases}$$

Multi class classification problem (4 classes)

↳ Softmax Activation function

↳ O/P layer

$$z^{[L]} = wx + b$$

$$\begin{bmatrix} 5 \\ 2 \\ -1 \\ 3 \end{bmatrix}$$

$$t = \frac{e^{z[L]}}{e^5 + e^2 + e^{-1} + e^3} \Rightarrow \text{normalize}$$

$e^5 = 148.4$

$e^2 = 7.3$

$e^{-1} = 0.3$

$e^3 = \frac{20.0}{176.25}$

- First class : $\frac{148.4}{176.25}$
- II : $\frac{7.3}{176.25}$
- III $\frac{0.3}{176.25}$
- IV : $\frac{20}{176.25}$

$$\begin{bmatrix} 0.84 \\ 0.04 \\ 0.002 \\ 0.113 \end{bmatrix}$$

∴ I is class label

Softmax Activation function

j - number of classes

$$\therefore \quad t = \frac{e^i}{\sum_{j=1}^{4} e^j}$$

(1) Linear / Identity fn
HW application

26/09/22 Error / Residuals

Standard error = Actual - Predicted (Observed)

Mean Absolute error (MAE)

$$MAE = \frac{1}{n} \sum_{i=0}^{n} |y_i - \hat{y}_i|$$

y_i = Actual
$\hat{y}_i$ = Predicted

Mean Square Error (MSE)

$$MSE = \frac{1}{n} \sum_{i=0}^{n} (y_i - \hat{y}_i)^2$$

Root Mean squared error (RMSE)

$$RMSE = \frac{1}{n} \sum_{i=0}^{n} \sqrt{\frac{(y_i - \hat{y}_i)^2}{n}}$$

Actual	Predicted
8	4
2	16
1	1

Find out the mean squared error, root mean squared error, mean absolute error for the given example

Items	Actual Sales
1	80
2	90
3	100
4	110
5	120

$$\therefore \quad t = \sum_{j=1}^{4} \frac{e^j}{\&}$$

(1) Linear / Identity fn
HW application

26/09/22 Error / Residuals

Standard error = Actual − Predicted (Observed)

Mean Absolute error (MAE)

$$MAE = \frac{1}{n} \sum_{i=0}^{n} |y_i - \hat{y}_i|$$

y_i = Actual

$\hat{y}_i$ = Predicted

Mean Square Error (MSE)

$$MSE = \frac{1}{n} \sum_{i=0}^{n} (y_i - \hat{y}_i)^2$$

Root Mean squared error (RMSE)

$$RMSE = \frac{1}{n} \sum_{i=0}^{n} \sqrt{\frac{(y_i - \hat{y}_i)^2}{n}}$$

Actual	Predicted
8	4
2	16
1	1

Find out the mean squared error, root mean squared error, mean absolute error for the given example

Items	Actual Sales
1	80
2	90
3	100
4	110
5	120

The value predicted by the model is 75

Find the MAE, RMSE, MSE

$$MAE = \frac{1}{5}\left[(80-75)+(90-75)+(100-75)+(110-75)+(120-75)\right]$$

$$= \frac{1}{5}(5+15+25+35+45)$$

$$= \frac{125}{5} = 25$$

$$MSE = \frac{1}{5}\left[5^2+15^2+25^2+35^2+45^2\right] = 825$$

$$RMSE = 28.7$$

	Actual	Predicted
D_1	80	75
D_2	75	85

$$MAE = \frac{1}{2}\left[(80-75)+(85-75)\right]$$

$$= \frac{1}{2}[5+10] = 7.5$$

$$MSE = \frac{1}{2}[25+100] = 62.5$$

$$RMSE = 7.90$$

→ Find the sales for week 6 using linear Regression

Weeks (x)	Sales
1	1000
2	3000
3	4000
4	8000
6	?

$$m = \frac{n\Sigma xy - \Sigma x \Sigma y}{n\Sigma x^2 - (\Sigma x)^2}$$

$$c = \frac{\Sigma y - m\Sigma x}{n}$$

$$a_1 = \frac{(\overline{xy}) - \bar{x}\bar{y}}{(\overline{x_i^2}) - (\bar{x})^2}$$

$$a_0 = \bar{y} - a_1\bar{x}$$

x	y	xy	x_i^2
1	1000	1000	1
2	3000	6000	4
3	4000	12k	9
4	8000	32k	16
10	16000	51000	

$$\frac{12750 - (2.5)(4000)}{7.5 - 6.25}$$

$$m = \frac{12750}{1.25} = \frac{10,192}{2200}$$

$$c = 4000 - \frac{2200}{10,192}(2.5)$$

$$= -25,476$$

$$y = 10192x - 25476$$

$$6 = 11700 \quad 8 \Rightarrow 16100 = 35676$$

(2)

x_1	x_2	y
1	4	1
2	5	6
3	8	8
4	2	12

$$\beta = (X^T X)^{-1} X^T y$$

$$X^T X = \begin{bmatrix} 4 & 10 & 19 \\ 10 & 30 & 46 \\ 14 & 46 & 109 \end{bmatrix}$$

$$[X^T X]^{-1} = 366$$

$$X^T y = \begin{bmatrix} 27 \\ 85 \\ 122 \end{bmatrix}$$

When we use Eigen vector, value

Differentiation

Partial derivative

Why we use activation fn

Linear → Non linear → Activation

SVD decomposition - P.

Vectors / Tensors

Basic ~~comp~~ operations

29/09

Loss function

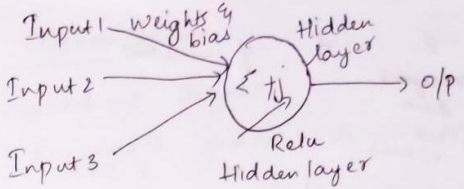
→ Classification

→ Regression

- Mean square error
- Mean absolut error
- Huber loss

O/P : Binary class / Multi class

- Binary cross Entropy loss / Log loss (Two class)
- Categorical Cross Entropy Loss



Relu variants of relu

↳ Leaky

↳ Prelu

↳ Exponential relu

Error value / cost function / loss function

Binary cross entropy loss / log loss

$$\text{Binary cross entropy} = -\frac{1}{n}\sum_{i=1}^{n}\left[y \log p + (1-y)\log(1-p)\right]$$

y represents actual / target value

y : Actual / target value

p : predicted probability / estimated value

Sigmoid (f(x))

↳ 0 to 1

0.5 → 1

below → 0

S.No	x_1	x_2	x_3	Σ Weighted sum (Prob. value)	y	$\hat{y}$	Loss
1	0.1	0.5	0.2	0.26	1	0	0.585
2	0.2	0.3	0.1	0.20	0	0	0.09
3	0.7	0.4	0.2	0.48	1	0	0.31
4	0.1	0.4	0.3	0.30	0	0	0.15
						$\frac{\Sigma}{4}$	0.285

Instance1:

$$-y \log p + (1-y) \log (1-p)$$

$$-1 \log (0.26) + (1-1) \log (1-0.26)$$

$$= 0.585$$

Instance 2:

$$-0 \log p + (1-0) \log (1-0.20)$$

$$= -1 \log (0.80) = 0.096$$

Instance 3

$$-1 \log (0.48)$$

$$= 0.318$$

Instance 4 : 0.154

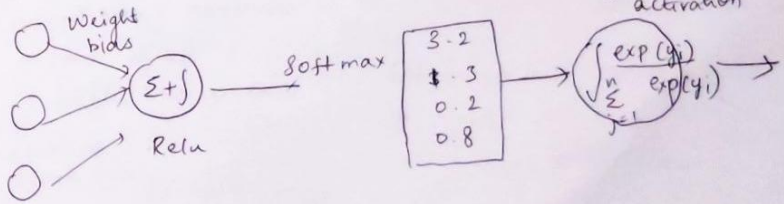
∴ Binary cross entropy loss = 0.285

Optimization

- Gradient descent
- Stochastic gradient descent
- Adadelta
- Ada grad
- RMS prop

Reg No	Actual value	Estimated probability	Loss
1	1	0.94	0.026
2	1	0.90	0.045
3	1	0.78	0.107
4	0	0.586	0.356
5	0	0.51	0.309
6	1	0.47	0.327
7	1	0.32	0.494
8	0	0.10	0.045
9			$\frac{\Sigma}{n}$ 0.213

Categorical cross Entropy



Estimated o/p	Actual o/p
0.775	1
0.116	0
0.039	0
0.070	0

Categorical cross entropy

$$\text{Loss} = -\sum_{i=1}^{n} [T_i \log(P_i)]$$

T_i — Actual target; P_i — Estimated Probability value

$$= -[1 \times \log(0.775) + 0 \times \log(0.116) + 0 \times (\log(0.39)) + 0 \times \log(0.70)] = 0.115$$

Iteration 2

Estimated o/p	Actual o/p
0.938	1
0.028	0
0.013	0
0.023	0

Loss = 0.027

Mean square error $\sum_{i=1}^{n} (y - \hat{y})^2$

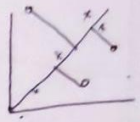
Mean absolute error $= \sum_{i=1}^{n} |y - \hat{y}|$

Huber loss

$$= \begin{cases} \frac{1}{2}(y - \hat{y})^2 \\ \delta |y - \hat{y}| - \frac{1}{2}\delta^2 \end{cases}$$

Drawback of mean²

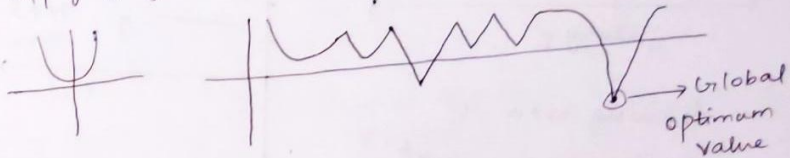
→ Very sensitive to outliers



(high residuals)² ↳ high

Advantages

- Applying global optimum value



Huber loss

δ — Hyperparameter value

To overcome Disadv of MSE

Regularisation

↑ Overfitting problem — Solve

- Low bias
- High variance

Underfitting

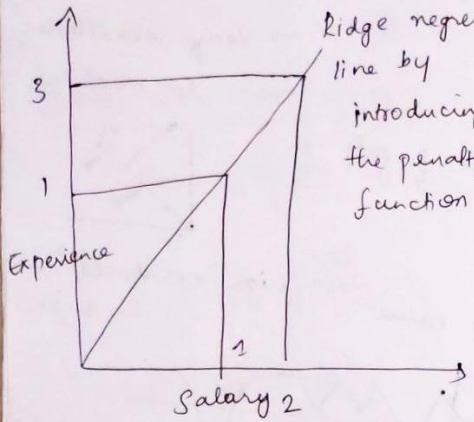
- High bias
- High variance

Prequisite - Bias and Variance

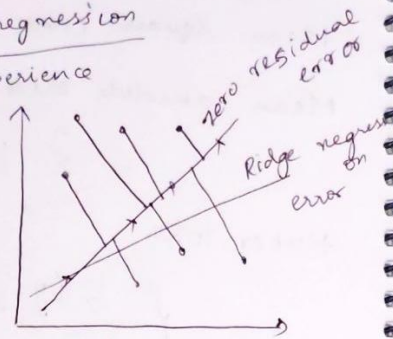
Generalised model

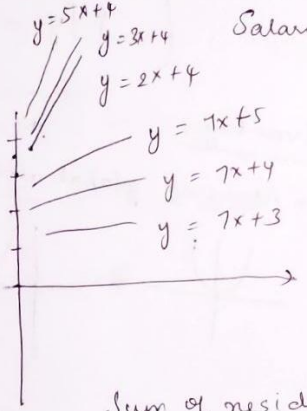
- Low bias
- Low variance

Ridge regression and Lasso regression



Ridge regression line by introducing the penalty function





Training error : 0

Testing error = high

Generalise by reducing the slop

Sum of residual = 0

Ridge estimator = Sum of residuals + λ * Slope^2

$= \sum_{i=1}^{n} (y - \hat{y})^2 + \lambda * \text{slope}^2$

Example : $y = 1.3x + 0.4$

$m = 1.3$

y intercept $c = 0.4$

λ - Penalty / Penal value

$\lambda = 0$ to ∞

$= 0 + 1 + (1.3)^2 = 2.69$

Slope = 0.8

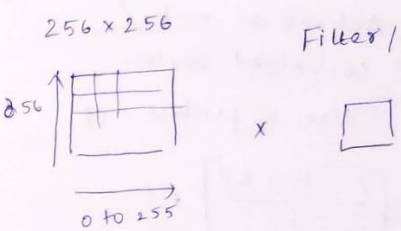
Lasso regression = Sum of residual + λ * |slope|

L2

Change of x = Change of y

Convolution Neural Network (CNN):

Kernels / Filters / Feature detectors are used to determine the features.



Building block of CNN

→ Convolution
- Kernel / Filter / Feature detector, Padding, Stride

→ Pooling
- Max pooling, Average pooling

→ Flatten ANN

5×5 Image convolution

Kernel/Filter

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
5	6	7	8	9
2	3	4	5	6

×

1	2	3
4	5	6
7	8	9

=

411	456	501
372	417	462

Stride = 1

Move filter by

Padding - Border preserving

Feature maps / Activation maps / convolved feature

n - size of image
f - filter size
p - padding
s - stride value

Size of feature map $= \left[\frac{n - f + 2p}{s}\right] + 1$

If padding = 0, n = 5, f = 3, s = 1

$$FM = \frac{5 - 3 + 2(0)}{1} + 1 = 3$$

One 5×5 → Overfitting
4×4 → Heavy computation

Max pooling, Average pooling, min pooling - Reduce dimensions

↓ Most common

After Pooling

11	22
33	44

→ Flatten

11
22
33
44

ANN - Classification

Image 256×256×3 × filter (3×3) = Feature map

$256 \times 256 \times 3$

$\frac{254}{2}$

Pooling 127×127×3

Convolution 127×127×32 × filter (3×3×32) → Convolution 62×62×64 → 30×30×64

51,600

-1	2	-1
+1	5	+1
-1	4	-1

Horizontal edge

-1	2	-1
+1	3	+1
-1	2	-1

Vertical edge

256×256×3

$$\frac{256 - 3 + 0}{1} + 1 = 254 - 3$$

Machine Learning

Image → Handcrafted feature extraction → Trained Classifier → Output

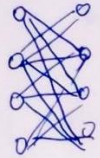
CNN

Image → Low level features → Middle Level features → High level features → Trained Concepts Classifier → Output

Key concepts

Input Image → Neurons → Weighted & bias parameter → Hidden units → Activation function / Smashing / Mapping / Transfer → Output units → Loss value / Cost value → Backpropogation

Hidden layer



Early stopping

Optimizers

- Gradient descent
- Learning rate
- Regularization
  - Ridge
  - Lasso
  - Dropout
- Early stopping

Hyperparameters

← ANN

- No. of hidden layers
- No. of neurons
- Learning rate
- Activation functions
- Optimisers
- Dropout values
- Weight initialization

Filter - Receptive field

CNN

- No. of convolution layers
- Stride
- Padding
- No. of kernels
- Pooling layer — Average / Global / Maximum
- Batch size
- Epochs

problems on CNN and ANN

Convolution Layers [1 - 10]
Pooling Operation [Max, Min, Globals]
Hidden layers [1 - 25]
Hidden units [32 - 512]
Activation functions [Relu, Leaky Relu, FRelu, Exponential Relu, tanh]

Optimizers = [Gradient descent
Stochastic Gradient Descent
SGD with momentum,
Adagrad
RMS prop
Adam]

Learning rate : [0.4 to 0.8]
Epochs : [200 to 500]
Dropout : [0.4 to 0.7)
Weight / Bias

Constrnet CNN

GridSearch = [RandomSearch / Genetic Algorithm / Particle Swarm Algorithm]