Subject: Machine Learning 28/07/2025

Reference Books :

1.) Pattern Recognition & Machine Learning by Christopher M. Bishop
2.) Deep Learning by Ian Good Fellow, Yoshua Bengio, and Aaron Courville
3.) Machine Learning By Andrew Ng (Standford)
4.) Machine Learning by Stanley chan (Purdue)

Mid Term – 30
Quizzes (2) – 20
End Term – 50

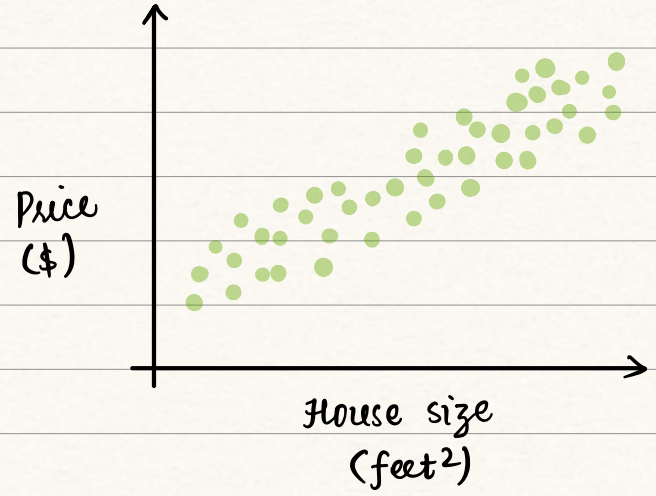
Theory class + Demonstration of Machine Learning (ML) Algorithms on Python Platform

→ Mathematically challeging – Knowledge of statistics, Basic Probability & Linear Algebra.

☆ Supervised Machine Learning ——

→ Labelled data set which is having both feature as well values of corresponding feature.

Example : House size and Price in $.



Q.) What is the goal based on this dataset ?

→ Predicting the house price close to the actual price.

Subject: / /

Q) Given the dataset like we seen before, how can we learn to predict the prices of the other houses which might or might not be present in the dataset?

$x^{(i)}$ → denotes the 'input' variable in the dataset.

↳ This superscript (i) denotes the i^{th} training example.

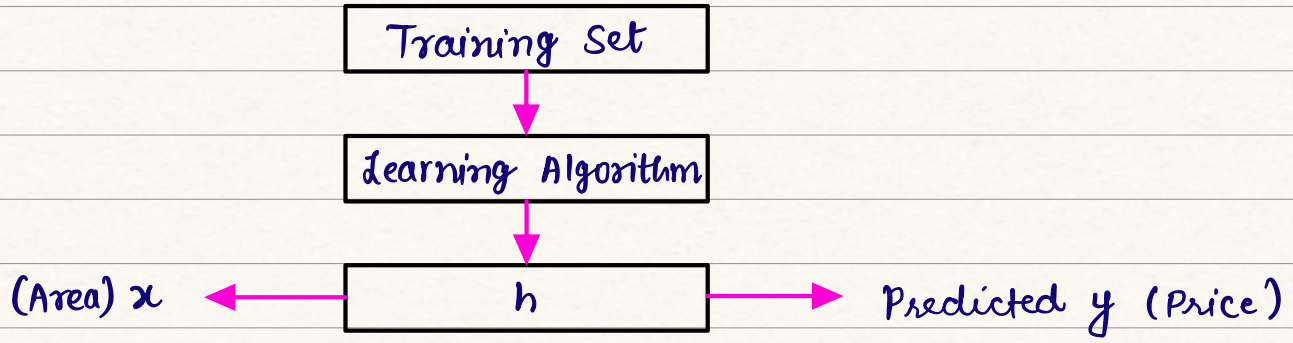
$y^{(i)}$ → denotes the 'output' variable that we are trying to predict.

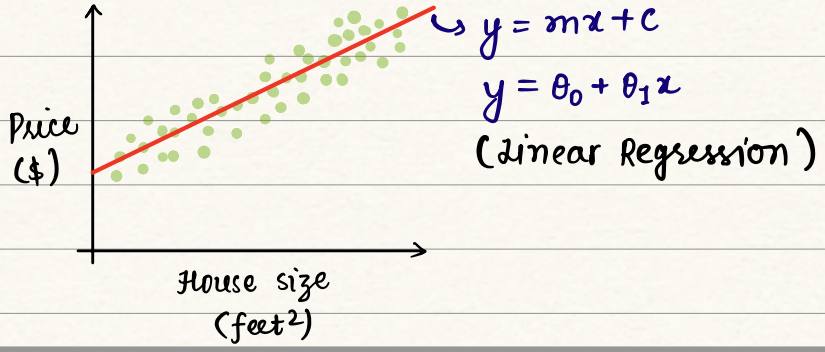
$(x^{(i)}, y^{(i)})$ → This pair is called i^{th} training example in the dataset.

$\{(x^{(i)}, y^{(i)})\ ;\ i = 1, 2, \ldots, N\}$

↳ This is the dataset.

Given a training set, we need to learn a function $h : X \to Y$ so that $h(x)$ is a 'good' predictor for corresponding values of y. {h → hypothesis function}





Subject: 29/07/2025

$$h(x) = \theta_0 + \theta_1(x) = \begin{bmatrix}\theta_0 & \theta_1\end{bmatrix}\begin{bmatrix}1\\ x\end{bmatrix} = \theta^T x$$

$$\theta = \begin{bmatrix}\theta_0\\ \theta_1\end{bmatrix} \qquad \theta^T = \begin{bmatrix}\theta_0 & \theta_1\end{bmatrix}$$

Q) Given a training dataset, choose the parameters θ_0 and θ_1?

We should choose the values of θ_0 and θ_1 such that $h(x^{(i)}) = \theta_0 + \theta_1 x^{(i)}$ is close to each $y^{(i)}$ (True value).

↳ Predicted $y^{(i)}$

$$\text{cost function } (J(\theta)) = \underbrace{\frac{1}{2n}\sum_{i=1}^{n}\left(h(x^{(i)}) - y^{(i)}\right)^2}_{\text{Mean squared Error (MSE)}}$$

Our aim to find a way to compute the optimum values of θ_0 and θ_1 that minimize MSE, $J(\theta)$.

Gradient Descent Algorithm

$$F = x_1 f_1 + x_2 f_2 + \ldots + x_n f_n$$

$$\nabla F = \begin{bmatrix}\frac{\partial F}{\partial f_1}\\ \frac{\partial F}{\partial f_2}\\ \vdots\\ \frac{\partial F}{\partial f_n}\end{bmatrix}$$

* starts with some initial guess of θ_0 & θ_1.
* Algorithm repeatedly changes the values of θ_0 and θ_1 till $J(\theta)$ converges to the minimum value.

$$\text{update } \theta_j \underset{\text{Assignment operator}}{:=} \theta_j - \underbrace{\alpha}_{\text{Learning rate}} \frac{\partial J(\theta)}{\partial \theta_j} \quad ; \quad j = 0, 1$$

This update is simultaneously done for θ_0 & θ_1.

Subject: 29/07/2025

* Learning rate should be optimum, it should not be too high and too low.

$n = 1$;

$$J(\theta) = \frac{1}{2}(h(x) - y)^2$$

$$= \frac{1}{2}(\theta_0 + \theta_1 x - y)^2$$

$$\frac{\partial J(\theta)}{\partial \theta_j} = (\theta_0 + \theta_1 x - y) \cdot \frac{\partial}{\partial \theta_j}(\theta_0 + \theta_1 x - y)$$

$$= (\theta_0 + \theta_1 x - y) \cdot x_j \qquad \{x_0^{(i)} = 1,\ x_j^{(i)} = x^{(i)}\}$$

$$\theta_j := \theta_j - \alpha \frac{\partial J(\theta)}{\partial \theta_j}$$

$$\frac{\partial J(\theta)}{\partial \theta_j} = (h(x) - y) \cdot x_j$$

$$\theta_j := \theta_j + \alpha (y - h(x)) \cdot x_j \quad \} \text{ for one training example}$$

$$\theta_j := \theta_j + \frac{\alpha}{n} \sum_{i=1}^{n} (y^{(i)} - h(x^{(i)})) \cdot x_j^{(i)}$$

Repeat unit convergence of $J(\theta)$.

$$\theta_j := \theta_j + \frac{\alpha}{n} \sum_{i=1}^{n} (y^{(i)} - h(x^{(i)})) \cdot x_j^{(i)}$$

for every $j = 0, 1$

$$\theta_0 := \theta_0 + \frac{\alpha}{n} \sum_{i=1}^{n} (y^{(i)} - h(x^{(i)})) \cdot x_0^{(i)}$$

$$\theta_1 := \theta_1 + \frac{\alpha}{n} \sum_{i=1}^{n} (y^{(i)} - h(x^{(i)})) \cdot x_1^{(i)}$$

$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix} \qquad x^{(i)} = \begin{bmatrix} x_0^{(i)} \\ x_1^{(i)} \end{bmatrix}$$

Subject:

30/07/2025

Batch Gradient Descent → Convex function

$$\theta_j := \theta_j + \frac{\alpha}{n} \sum_{i=1}^{n} \underbrace{\left(y^{(i)} - h(x^{(i)})\right)}_{\text{Error}} . x^{(i)} \quad \{j = 0, 1, 2\ ?\}$$

Q.) What will happen if the error is very small & very high?

Weights θ_0 & θ_1 → optimized (In case of very small)

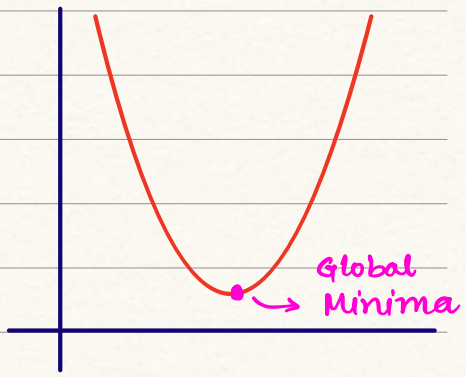
Weights θ_0 & θ_1 → not optimized (In case of very high)

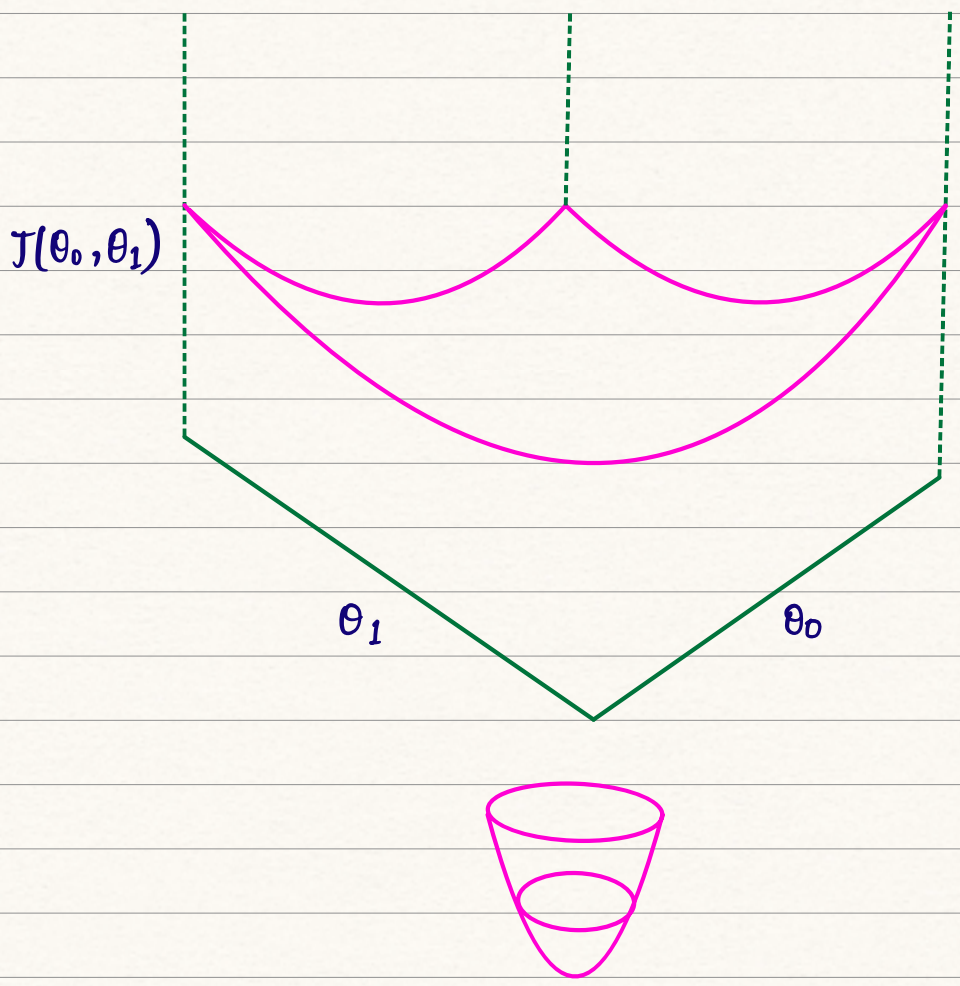
When we have a very large dataset, we don't use the batch gradient descent.

★ Convex function :—

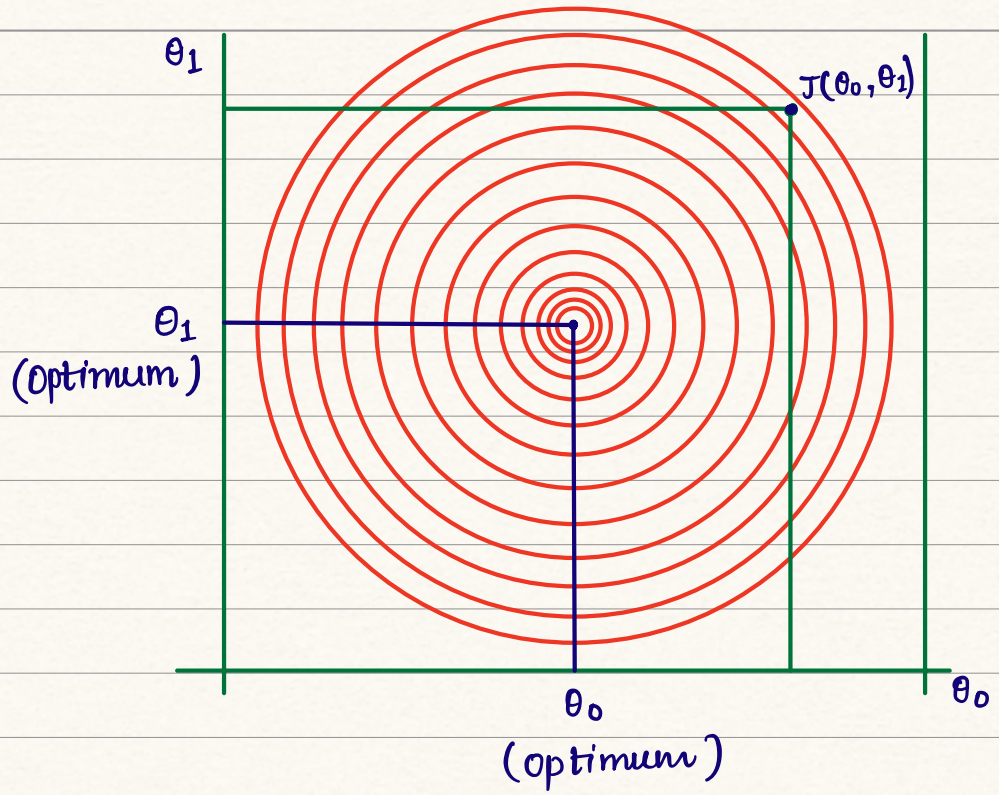
With GDA, $J(\theta)$ achieves the global minimum value.

$$J(\theta) = \frac{1}{2n} \sum_{i=1}^{n} \left(h(x^{(i)}) - y^{(i)}\right)^2$$





Subject: / /



✱ Stochastic Gradient Descent (Incremental Gradient Descent) :-

$$\theta_j := \theta_j + \alpha \left(y^{(i)} - h(x^{(i)})\right) . x_j^{(i)} \quad ; \quad j = 0, 1$$

for $i = 1$ to n.

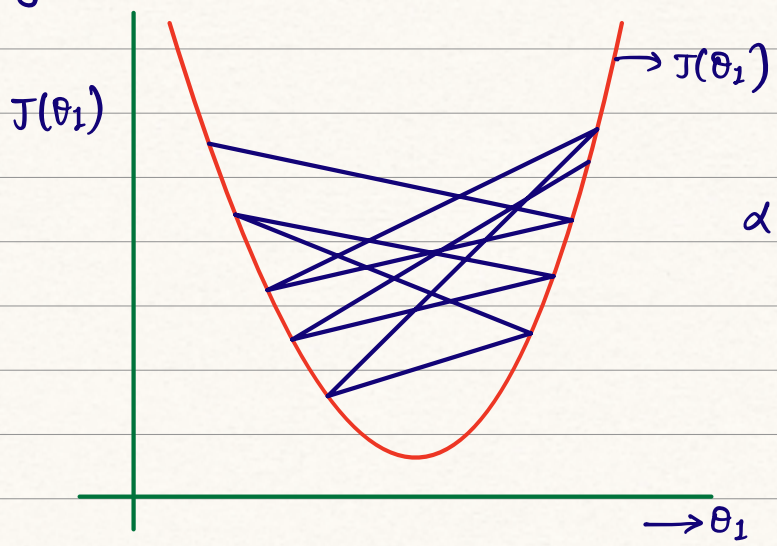
In the stochastic gradient descent algorithm, whenever we encounter the Training example, we update the parameters (θ_0, θ_1) according to the gradient of the error with respect to that single training example only.

* With the stochastic Gradient Descent Algorithm, we might never achieve the actual minimum value of the cost function and the achieved cost function with this algorithm might be oscillating around the global minimum value.
* Still, although it fails to achieve the global minimum value, it is found that the cost function value obtained is approximately very close to the global minimum value & hence we can still use that approximated achieved cost function value.

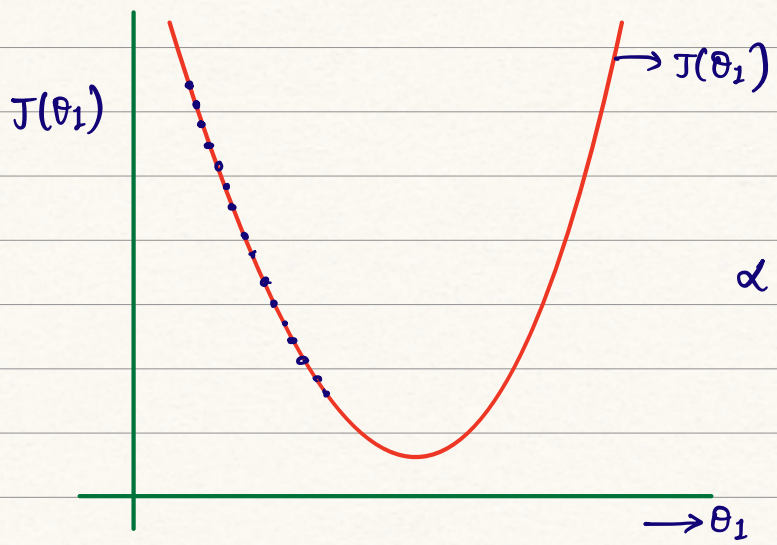
Subject: / /

* When the size of the training set is very large, stochastic gradient descent is prefessed over the Batch Gradient Descent.
* The parameter α (learning rate) should be chosen very carefully for the better convergence of the algorithm.
* If the α is chosen very large, Gradient Descent might fail to converge.

$y = \theta_1 x$



α is very large



α is very small

* If α is very small, GDA would take very much time to achieve the minimum value of $J(\theta_1)$.

Subject: 04/08/2025

$$y = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \theta_4 x_4$$

Multivariate Linear Regression

$x_1 \rightarrow$ size of house (ft^2)

$x_2 \rightarrow$ no. of bedrooms

$x_3 \rightarrow$ no. of gym

$x_4 \rightarrow$ garage or not

$$\theta_j := \theta_j + \frac{\alpha}{n} \sum_{i=1}^{n} \left(y^{(i)} - h(x^{(i)})\right) . x_j^{(i)}$$

$j = 0, 1, 2, 3, 4$

* In general, we can have 'd' number of features.

$$y = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_d x_d$$

$$\begin{bmatrix} \theta_0 & \theta_1 & \theta_2 & \ldots & \theta_4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix}$$

✯ Normal Equations :—

* GDA gives one way to find the parameters/weights (θ_0, θ_1) in order to minimize the mean squared error (MSE).

Let's study another way to find the optimum values of (θ_0, θ_1) so as to minimize $J(\theta)$.

Multi-features data set regarding the housing price

features (Living area (ft^2), No. of bedrooms, Price ($))

x_0	Living area ($feet^2$)	# of bedrooms	Price ($10,000)	
1	2104	3	400	→ Training Example 1
1	1600	3	330	
1	2400	3	369	

Subject: / /

$\theta_0 x_0^{(1)} + \theta_1 x_1^{(1)} + \theta_2 x_2^{(2)} \longrightarrow$ First training example

d input features

$x_0 \quad x_1 \quad x_2 \quad \dots \quad x_d$

no. of training examples $\begin{cases} 1 \\ 2 \\ \vdots \\ n \end{cases}$

$$\text{size} = n \times (d+1)$$

$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \\ \vdots \\ \theta_d \end{bmatrix}_{(d+1) \times 1} \qquad X = \begin{bmatrix} - \; x^{(1)T} \; - \\ - \; x^{(2)T} \; - \\ - \; x^{(3)T} \; - \\ - \; x^{(4)T} \; - \\ - \quad \vdots \quad - \\ - \; x^{(n)T} \; - \end{bmatrix}_{n \times (d+1)}$$

$$X\theta - \vec{y}$$

$$\begin{bmatrix} - \; x^{(1)T} \; - \\ - \; x^{(2)T} \; - \\ - \; x^{(3)T} \; - \\ - \; x^{(4)T} \; - \\ - \quad \vdots \quad - \\ - \; x^{(n)T} \; - \end{bmatrix}_{n \times (d+1)} \quad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \\ \vdots \\ \theta_d \end{bmatrix}_{(d+1) \times 1} - \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ \vdots \\ y_5 \end{bmatrix}_{n \times 1}$$

$\underbrace{\theta_0 + \theta_1 x_1^{(1)} + \theta_2 x_2^{(1)} + \dots + \theta_d x_d^{(1)}}_{h(x^{(1)})}$ } First training example

$\Rightarrow h(x^{(i)}) - y^{(i)}$

Subject: / /

$$\begin{bmatrix} h(x^{(1)}) \\ h(x^{(2)}) \\ \vdots \\ h(x^{(n)}) \end{bmatrix}_{n\times 1} - \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(n)} \end{bmatrix}_{n\times 1} = \begin{bmatrix} h(x^{(1)}) - y^1 \\ h(x^{(2)}) - y^2 \\ \vdots \\ h(x^{(n)}) - y^n \end{bmatrix}_{n\times 1}$$

$$J(\theta) = \frac{1}{2n} \sum_{i=1}^{n} \left(h(x^{(i)}) - y^{(i)} \right)^2 \longrightarrow \text{MSE}$$

$$= \frac{1}{2n} \Delta^T \Delta$$

where $\Delta = \begin{bmatrix} h(x^{(1)}) - y^1 \\ h(x^{(2)}) - y^2 \\ \vdots \\ h(x^{(n)}) - y^n \end{bmatrix}$

$$= \frac{1}{2n} (x\theta - \bar{y})^T (x\theta - \bar{y})$$

Given $c \in \mathbb{R}^n$, the differential function $f(x) = c^T x \; ; x \in \mathbb{R}^n$.

$$f(x) = c^T x$$
$$= c_1 x_1 + c_2 x_2 + \ldots + c_n x_n$$

$$c = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix}_{n\times 1}$$

$$c^T x = \begin{bmatrix} c_1 & c_2 & \cdots & c_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

$$c^T x = c_1 x_1 + c_2 x_2 + \ldots + c_n x_n$$

$$\text{Gradient } (\nabla_x f(x)) = \begin{bmatrix} \frac{\partial f(x)}{\partial x_1} \\ \frac{\partial f(x)}{\partial x_2} \\ \vdots \\ \frac{\partial f(x)}{\partial x_n} \end{bmatrix}$$

Subject: / /

For the sake of understanding, take $d = 2$;

$x_0 \quad x_1 \quad x_2$ → Suppose two training examples

$$X = \begin{bmatrix} 1 & x_1^{(1)} & x_2^{(1)} \\ 1 & x_1^{(2)} & x_2^{(2)} \end{bmatrix}_{2\times 3}$$

$$X^T = \begin{bmatrix} 1 & 1 \\ x_1^{(1)} & x_1^{(2)} \\ x_2^{(1)} & x_2^{(2)} \end{bmatrix}_{3\times 2}$$

$$X^TX = \begin{bmatrix} 1 & 1 \\ x_1^{(1)} & x_1^{(2)} \\ x_2^{(1)} & x_2^{(2)} \end{bmatrix}_{3\times 2} \begin{bmatrix} 1 & x_1^{(1)} & x_2^{(1)} \\ 1 & x_1^{(2)} & x_2^{(2)} \end{bmatrix}_{2\times 3}$$

$$X^TX = \begin{bmatrix} 2 & x_1^{(1)} + x_1^{(2)} & x_2^{(1)} + x_2^{(2)} \\ x_1^{(1)} + x_1^{(2)} & (x_1^{(1)} + x_1^{(2)})^2 & x_1^{(1)}x_2^{(1)} + x_1^{(2)}x_2^{(2)} \\ x_2^{(1)} + x_2^{(2)} & x_1^{(1)}x_2^{(1)} + x_1^{(2)}x_2^{(2)} & (x_2^{(1)} + x_2^{(2)})^2 \end{bmatrix}$$

$$X^TX = B = B^T$$

For n training examples and having 'd' no. of input features (excluding $x_0 = 1$). The matrix X^TX is always symmetric.

Suppose that $A = A^T$ (symmetric matrix).

$\nabla_x (x^TAx) =$ □

$$A = \begin{bmatrix} a_1 & a_2 \\ a_2 & a_1 \end{bmatrix}$$

$$x^TAx = \begin{bmatrix} x_1 & x_2 \end{bmatrix} \begin{bmatrix} a_1 & a_2 \\ a_2 & a_1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$x^TAx = a_1x_1^2 + a_2x_1x_2 + a_2x_1x_2 + a_1x_2^2$$

Subject: 04/08/2025

$$\nabla_x(x^TAx) = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 2a_1x_1 + a_2x_2 + a_2x_2 \\ a_2x_1 + a_2x_1 + 2a_1x_2 \end{bmatrix}$$

$$= 2\begin{bmatrix} a_1 & a_2 \\ a_2 & a_1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$\nabla_x(x^TAx) = 2Ax$$

$$J(\theta) = \frac{1}{2n}(x\theta - y)^T(x\theta - y)$$

$$= \frac{1}{2n}((x\theta)^T - y^T)(x\theta - y)$$

$$= \frac{1}{2n}\left[(x\theta)^Tx\theta - (x\theta)^Ty - y^Tx\theta + y^Ty\right]$$

$$= \frac{1}{2n}\left[\theta^Tx^Tx\theta - y^Tx\theta - y^Tx\theta + y^Ty\right] \quad \rightarrow \frac{\partial J(\theta)}{\partial \theta} = 0$$

$((x\theta)^Ty)^T = y^T((x\theta)^T)^T = y^Tx\theta$ (scalar)

$$= \frac{1}{2n}\left[\theta^Tx^Tx\theta - 2y^Tx\theta\right]$$

$[\ ]_{m\times n} \rightarrow$ matrix

$[\ ]_{m\times 1} \rightarrow$ vector

$$= \frac{1}{2n}\left[\theta^Tx^Tx\theta - 2(x^Ty)^T\theta\right]$$

$$\nabla_\theta J(\theta) = \frac{1}{2n}\left[2x^Tx\theta - 2x^Ty\right]$$

$\{\nabla_x\, x^TAx = 2Ax\}$

$b \in \mathbb{R}^n \quad x \in \mathbb{R}^n$

$f(x) = b^Tx$

$\nabla_x f(x) = b$

$\nabla_\theta f(\theta) = (x^Ty)^T\theta$

$= x^Ty$

$$\nabla_\theta J(\theta) = 0$$

$$\Rightarrow x^Tx\theta = x^Ty$$

$$\Rightarrow \theta = (x^Tx)^{-1}x^Ty$$

Subject: / /

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

↳ Pseudoinverse of X

$$\begin{cases} (X^T X)^{-1} X^T \\ = X^{-1} (X^T)^{-1} X^T \\ = X^{-1} \end{cases}$$

Q.) How dependent input features can be a problem ?

$$(X^T X)^{-1} X^T y$$

↳ can create a problem when the features in the data set are linearly dependent.

⇓

Determinant = 0

$$A^{-1} = \frac{\text{Adjoint of } A}{|A|}$$ → here it will create a problem

Gradient Descent Algorithm	Normal Equation
* Need to choose the learning rate α.	* No need to choose α.
* Need many iterations.	* Don't need to do iteration.
* Works well even the data set is very large.	* Need to compute $(X^T X)^{-1}$ $O(d^3)$ ←
	* Works slow when d is large.

* In general, GDA is the recommended method for finding the parameters $(\theta_0, \theta_1, \theta_2, \cdots, \theta_d)$.

☆ Gaussian Distribution ——

The PDF of a Gaussian Distributive random variable X having the parameters μ (mean) and σ^2 (variance) is

Subject: / /

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(x-\mu)^2}{2\sigma^2}} \quad ; \quad -\infty < x < \infty$$

$$X \sim N(\mu, \sigma^2)$$

read as X is modeled as Normal or Gaussian distribution function.

Standard Normal —

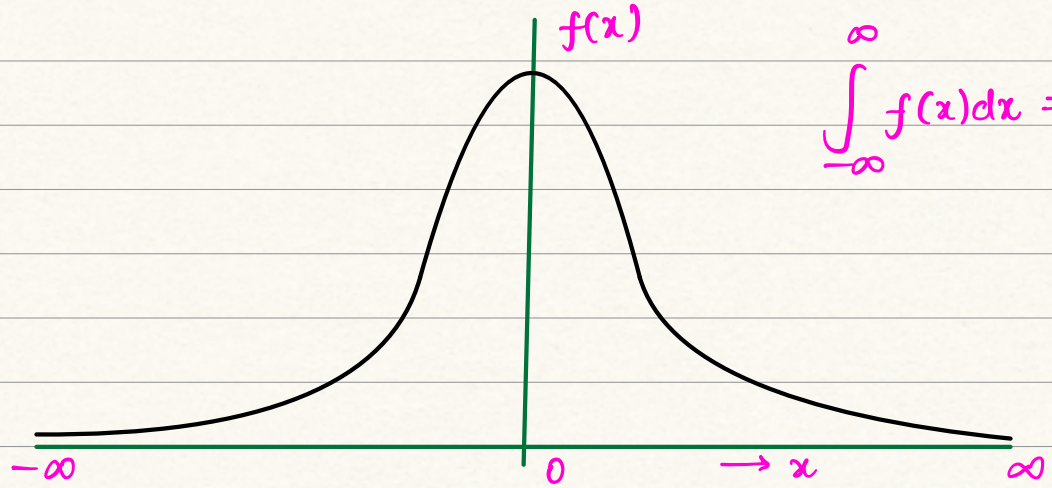
$\mu = 0$ and $\sigma^2 = 1$

$$X \sim N(0,1)$$

→ Expectation operator

$$\mu = E[X]$$

$$\sigma^2 = E[X^2] - (E[X])^2$$



$$\int_{-\infty}^{\infty} f(x)dx = 1$$

$$P(1 < X \leq 2) = \int_{1}^{2} f(x)dx$$

* Suppose that we have five random values from a normal distribution, but we don't know which parameters of normal distribution it belongs to i.e., we need to find the parameters μ and σ^2.

$$y = \{48.7, 50.3, 47, 51, 48\}$$

It's not coming from standard normal. $N(0,1)$.

Subject: / /

Q.) what is the likelihood of y generated from $N(0,1)$?

Almost zero.

Q.) What is the likelihood of y generated from $N(50,1)$?

Much much higher

* This is the sample example for the intuition of estimation of the parameters μ & σ^2. This approach in statistics is called maximum likelihood Estimation (MLE).

↳ Find the parameters that maximize the likelihood of generating the data we observe.

$$X \sim N(\mu, \sigma^2)$$

$$f_x(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right) \; ; \; -\infty < x < \infty$$

In this set up, we have a complete information about the parameters μ & σ^2. But what if the opposite is true - we know the outcome of the random variable X, but we do not know the parameters that generated it.

↳ Likelihood function

$$L = (\underbrace{\mu, \sigma^2}_{\text{Function of unknown parameters}} \mid \underbrace{X}_{\text{data}})$$

$$L = \underbrace{\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right)}_{\text{This pdf because we know } x \text{ is coming from Gaussian Distribution.}} \rightarrow \text{But we don't know } \mu \text{ & } \sigma^2 \text{ here.}$$

Subject: / /

$x_1, x_2, \ldots, x_n$ are independent & identically distributed (IID) – Random variables.

$$f(x_1, x_2, \ldots, x_n) = \underbrace{f(x_1)\, f(x_2) \cdots f(x_n)}_{\text{All are identical}} = \prod_{i=1}^{n} f(x_i)$$

Joint pdf

$$= \left(f(x)\right)^n$$

$x_1, x_2, \ldots, x_n \rightarrow$ IID samples from $N(\mu, \sigma^2)$

$$L(\mu, \sigma^2 \mid x) = \prod_{i=1}^{n} f(x_i \mid \mu, \sigma^2)$$

$$= \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right)$$

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

Now we will find the parameters μ & σ^2 which maximizes the likelihood function $L(\mu, \sigma^2 \mid x)$ → hence name is MLE.

$$L(\mu, \sigma^2 \mid x) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right)$$

Taking log on both the sides;

$\ln\left(L(\mu, \sigma^2 \mid x)\right) \rightarrow$ log likelihood function

$$\ln\left(L(\mu, \sigma^2 \mid x)\right) = \sum_{i=1}^{n} \ln \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right)$$

$$\ln\left(\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-(x_i-\mu)^2}{2\sigma^2}\right)\right) = \frac{-1}{2}\ln(2\pi) - \frac{1}{2}\ln(\sigma^2) - \frac{(x_i-\mu)^2}{2\sigma^2}$$

Subject: / /

$$\ln\left(L(\mu,\sigma^2|x)\right) =$$

$$\frac{\partial}{\partial\mu}\ln\left(L(\mu,\sigma^2|x)\right) = 0$$

$$\sum_{i=1}^{n}(x_i-\mu) = 0$$

$$\Rightarrow \hat{\mu}_{MLE} = \frac{1}{n}\sum_{i=1}^{n} x_i$$ → sample mean

$$\frac{\partial}{\partial\sigma^2}\ln\left(L(\mu,\sigma^2|x)\right) = 0$$

$$\Rightarrow \hat{\sigma}^2_{MLE} = \frac{1}{n}\sum_{i=1}^{n}(x_i-\mu)^2$$

$\{\mu = \hat{\mu}_{MLE}\}$

$$\hat{\sigma}^2_{MLE} = \frac{1}{n}\sum_{i=1}^{n}(x_i-\hat{\mu}_{MLE})^2$$

Q.) While doing the linear Regression, why did we chose the mean squared error (MSE) as a cost function?

$x_0^{(1)}$	$x_1^{(1)}$	$x_2^{(1)}$	$y^{(1)}$	$\theta^T x^{(1)}$
$x_0^{(2)}$	$x_1^{(2)}$	$x_2^{(2)}$	$y^{(2)}$	$\theta^T x^{(2)}$

$$y^{(1)} - \theta^T x^{(1)} = e^{(1)}$$

$$y^{(2)} - \theta^T x^{(2)} = e^{(2)}$$

} Error

Assumptions :

* Each $e^{(i)}$; $i \in \{1,2,3,\ldots,n\}$

n = # of training examples

Subject: 11/08/2025

$$e^{(i)} \sim N(0, \sigma^2)$$

Independent & Identically Distributed

$$f(e^{(i)}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(e^{(i)})^2}{2\sigma^2}\right) \quad ; -\infty < e^{(i)} < \infty$$
$$i \in \{1, 2, \ldots, n\}$$

$$\Rightarrow y^{(i)} = \theta^T x^{(i)} + e^{(i)}$$

For a given $x^{(i)}$ and parameterized by θ written as
$y^{(i)} \mid x^{(i)}; \theta$

$E[x+Y] = E[x] + E[Y]$

$x \rightarrow$ Deterministic

↳ is not random

$E[x] = x$

$$y^{(i)} \mid x^{(i)}; \theta \sim N(\theta^T x^{(i)}, \sigma^2)$$

$$\begin{aligned} E[y^{(i)}] &= E[\theta^T x^{(i)} + e^{(i)}] \\ &= E[\theta^T x^{(i)}] + E[e^{(i)}] \\ &= \theta^T x^{(i)} + 0 \\ &= \theta^T x^{(i)} \end{aligned}$$

$$\text{Var}(x) = E[x^2] - (E[x])^2$$
$$y^{(i)} = \theta^T x^{(i)} + e^{(i)}$$

$$\text{Var}(y^{(i)}) = E[(\theta^T x^{(i)} + e^{(i)})^2] - (E[\theta^T x^{(i)} + e^{(i)}])^2$$

$$\Rightarrow \text{Var}(y^{(i)}) = E[(\theta^T x^{(i)})^2 + (e^{(i)})^2 + 2\theta^T x^{(i)} e^{(i)}] - (E[\theta^T x^{(i)}] + E[e^{(i)}])^2$$

$$\Rightarrow \text{Var}(y^{(i)}) = (\theta^T x^{(i)})^2 + \sigma^2 + 0 - (\theta^T x^{(i)} + 0)^2$$

$$= \cancel{(\theta^T x^{(i)})^2} + \sigma^2 - \cancel{(\theta^T x^{(i)})^2}$$

$$= \sigma^2$$

$$y^{(i)} \sim N(\theta^T x^{(i)}, \sigma^2)$$

Subject: / /

$$f(y^{(i)} | x^{(i)}; \theta) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}\right)$$

Joint distribution of $y^{(i)}$: $f(y^{(1)}, y^{(2)}, \ldots, y^{(n)})$

$$f(y^{(1)}, y^{(2)}, \ldots, y^{(n)} | X; \theta) = f(y^{(1)} | x^{(1)}; \theta) f(y^{(2)} | x^{(2)}; \theta) \ldots f(y^{(n)} | x^{(n)}; \theta)$$

$$f(y^{(1)}, y^{(2)}, \ldots, y^{(n)} | X; \theta) = \prod_{i=1}^{n} f(y^{(i)} | x^{(i)}; \theta)$$

$$= \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}\right)$$

$$= L(\theta | X, y)$$ } likelihood function

$$\ln L(\theta | X, y) = \ln\left(\prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}\right)\right)$$

$$\log(abc) = \log(a) + \log(b) + \log(c)$$

$$\ln L(\theta | X, y) = n \log \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{2\sigma^2} \sum_{i=1}^{n} (y^{(i)} - \theta^T x^{(i)})^2$$

Maximizing $\ln L(\theta | X, y)$ $\longrightarrow$ minimizing $\sum_{i=1}^{n} (y^{(i)} - \theta^T x^{(i)})^2$

* Least square regression corresponds to the maximum likelihood estimate of θ.

☆ Polynomial Curve Fitting :—

Training set having N observations ; $x = [x_1, x_2, \ldots, x_N]^T$ and the corresponding outcomes to the observations are

Subject: 12/08/2025

$$t = [t_1, t_2, \ldots, t_N]^T$$

$N = 10$ observations ;

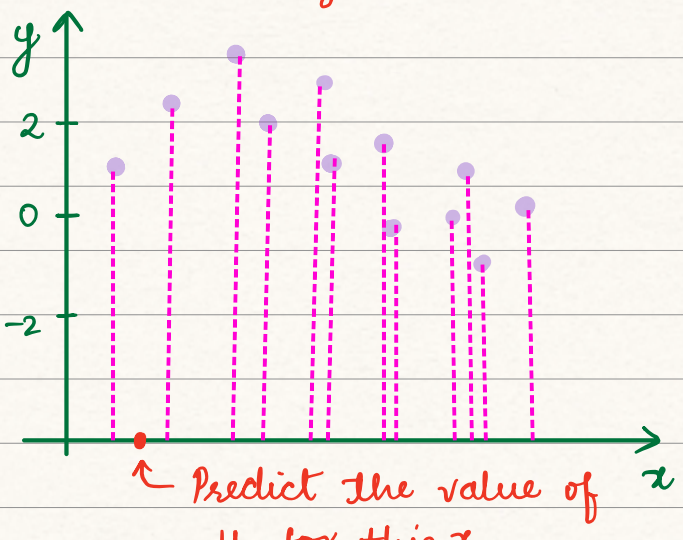
x_1 — 0.01
x_2 — 0.005
$\vdots$ — ...
x_{10} — 0.2

{ For each x_i, we are taking the value randomly lying between $[0,1]$.

$\sin(2\pi x_1) + \underbrace{n_1}_{\text{noise}} = y_1$ $\quad n_1 \sim N(0,1)$

$\Rightarrow \sin(2\pi x_i) + n_i = y_i$

This way we have artificially developed our own data set.



Non-linear ⟶ curve fitting Approach

$$y(x,\theta) = h(x,\theta) = \underbrace{\theta_0 + \theta_1 x + \theta_2 x^2 + \ldots + \theta_m x^m}_{m^{th} \text{ order polynomial}}$$

But I still have no idea regarding which value of m would best fit with the training data.

$$h(x,\theta) = \sum_{j=0}^{m} \theta_j x^j$$

$m \rightarrow$ order of the polynomial

Subject: / /

$h(x, \theta)$ is a linear function of the coefficients and non-linear function of x.

Our goal is to find the value of θ in such a way that MSE is minimized.

Suppose I fix $m = 5$;

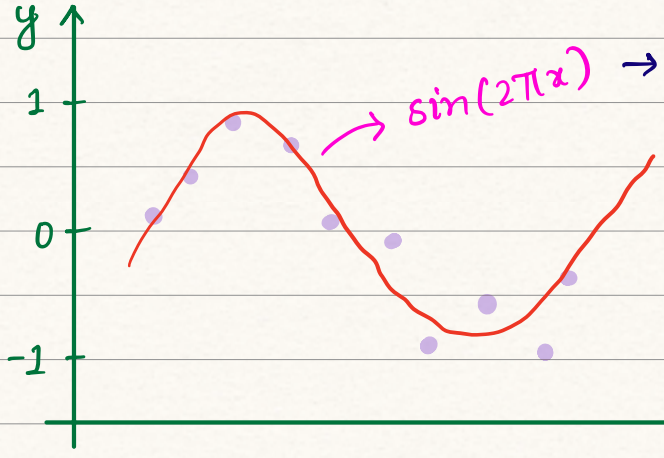
$$h(x, \theta) = \theta_0 + \theta_1 x^1 + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4 + \theta_5 x^5$$

$$\text{cost function, } J(\theta) = \frac{1}{2} \sum_{i=1}^{n} \left(y(x_i, \theta) - t_i \right)^2$$

$y(x_i, \theta)$ → predicted output corresponding to i^{th} training example

t_i → Actual output corresponding to i^{th} training example

Q.) What if we choose different values of m in the m^{th} order polynomial ?



→ original signal used to generate training data. But we don't have any idea about this.

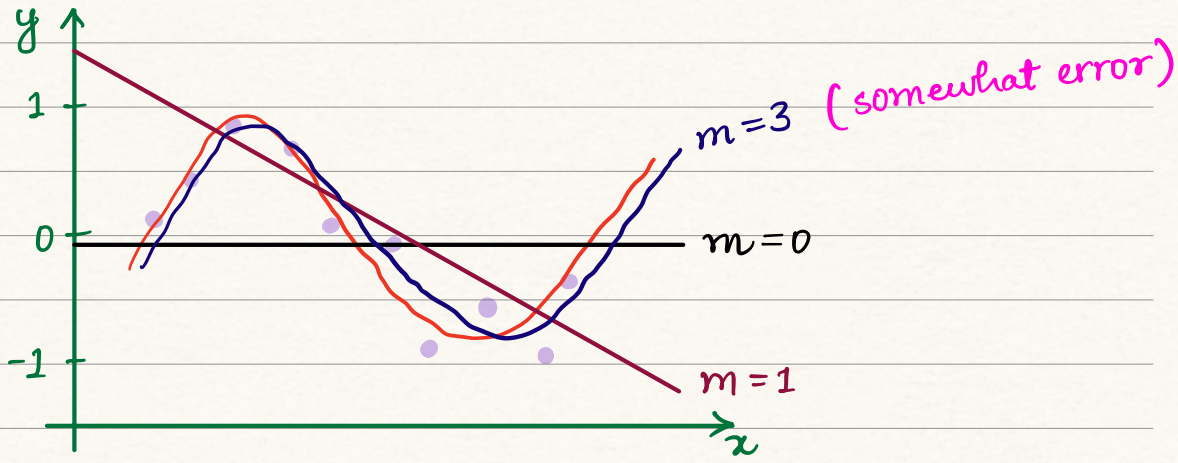
$m = 0$;

$$h(x, \theta) = \theta_0$$

$m = 1$;

$$h(x, \theta) = \theta_0 + \theta_1 x^1$$

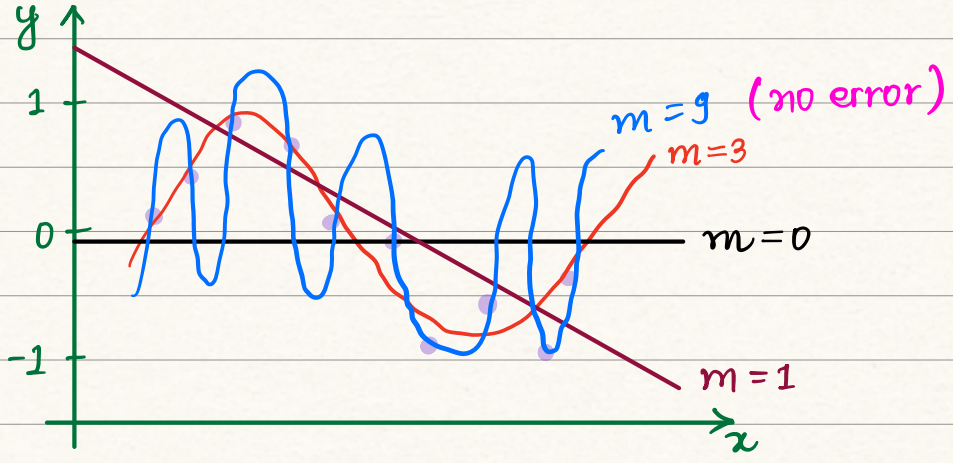
Subject: / /



(somewhat error)

$m = 3$;

$$h(x, \theta) = \theta_0 + \theta_1 x^1 + \theta_2 x^2 + \theta_3 x^3$$



(no error)

$m = 9$;

$$h(x, \theta) = \sum_{j=0}^{m} \theta_j x^j$$

For $m = 9$; there is not any training error but there is lot of fluctuations from $\sin(2\pi x)$.

This behaviour in the context of ML is called 'overfitting'.

1000 samples (Training)

800 → samples I will use to train the Algorithm.

200 → samples to test the performance of trained algorithm.

Subject: 13/08/2025

Training Error :- $E(\theta) = \frac{1}{2N} \sum_{n=1}^{N} (y(x_n, \theta) - t_n)^2$

we will compute this for $m = 0, 1, 2, \ldots, 9$

$y(x_n, \theta)$ → predicted value

$E(\theta)_{m=0}$
$E(\theta)_{m=1}$
$\vdots$
$E(\theta)_{m=9}$

$E_{RMS} = \sqrt{\frac{2E(\theta)}{N}}$ → Root mean square error

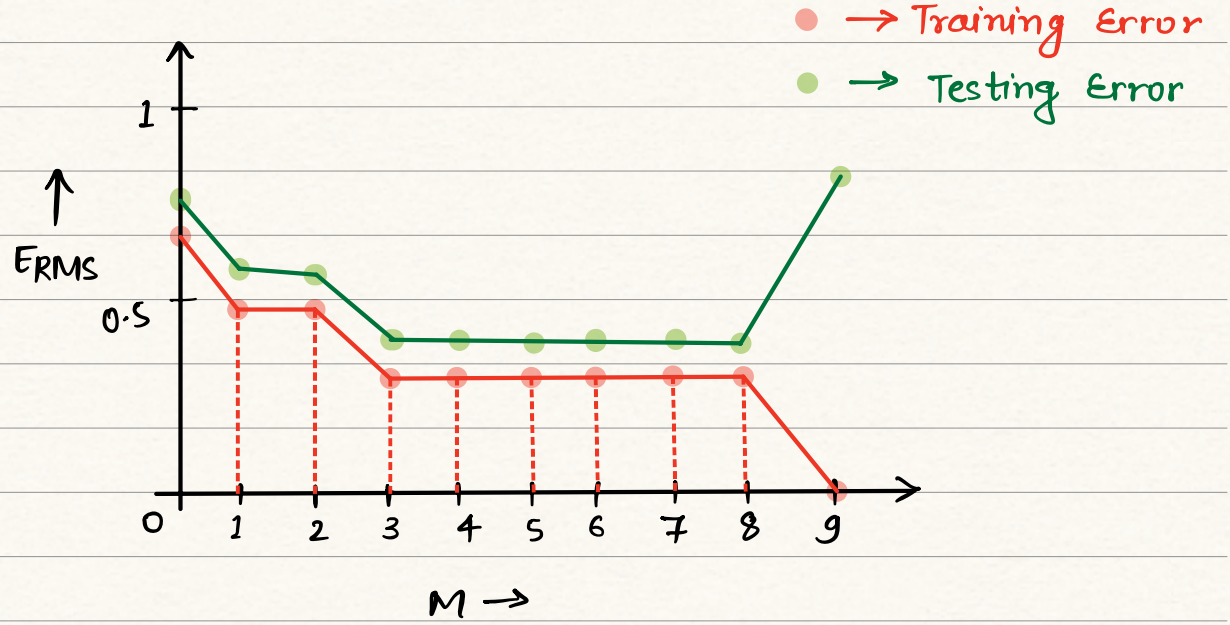
N = no. of training examples

* In this, E_{RMS}, N is included in order to compare different sizes of data sets on an equal footing and square root ensures that E_{RMS} is measured on the same scale as the target / output variable t.

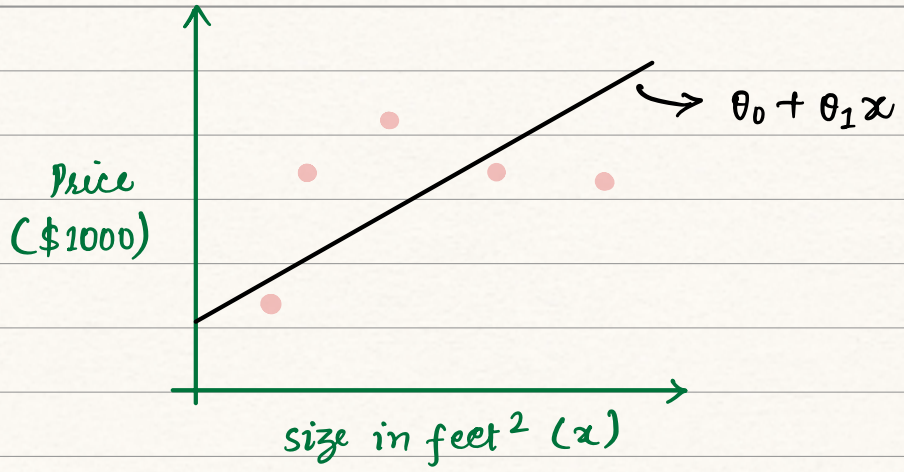
Testing Error :—

$$E(\theta) = \frac{1}{2N} \sum_{n=1}^{N} (y(x_n, \theta) - t_n)^2$$

Calculate error for value of x on trained example with trained weights of θ.

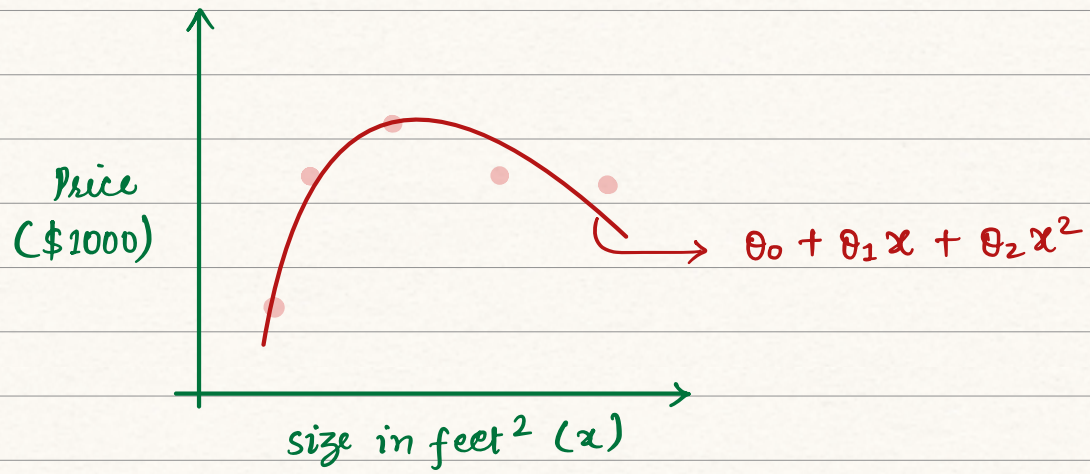


Subject: / /



This fit is very bad with the training data. This situation is called 'high bias' or 'underfit'.

Underfit or high bias → Algorithm is not able to fit the training data very well.



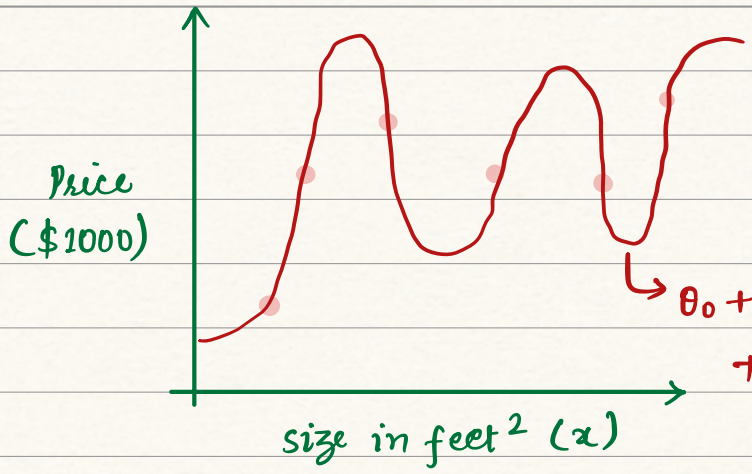
This fit with the training data is good.

→ We don't have high bias or underfit.

→ We don't have high variance or overfit.

* Our algorithm fits very well with the training data.
* Our algorithm would make very good predictions on the unseen data (or during testing phase).

Subject: / /



$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4 + \theta_5 x^5 + \theta_6 x^6 + \theta_7 x^7 + \theta_8 x^8 + \theta_9 x^9$

This fits the training set very well.

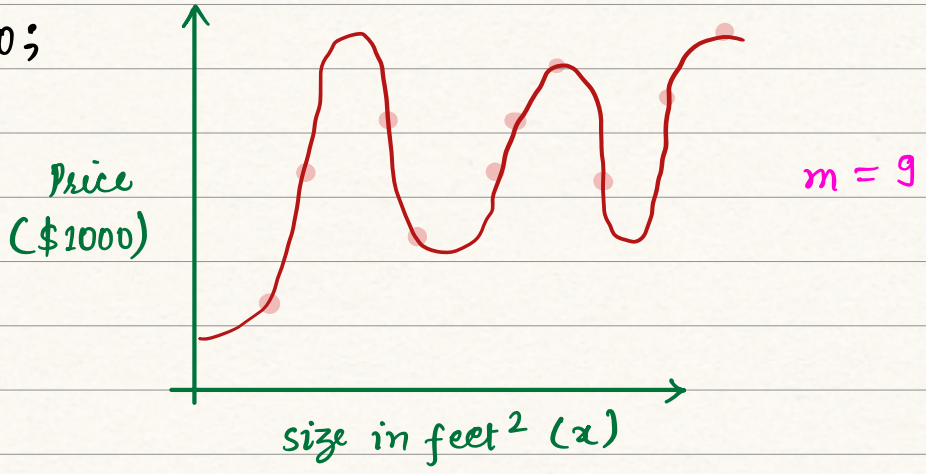
→ Overfitting problem ⇒ has high variance

⇒ leads to large testing error.

Q.) How can we overcome the overfitting problem?

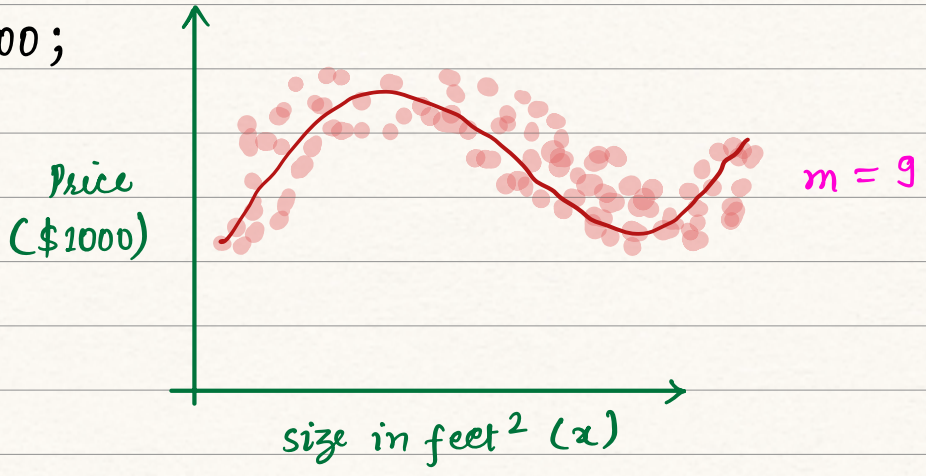
Way I: Collect more training examples.

$n = 10$;



$m = 9$

$n = 100$;

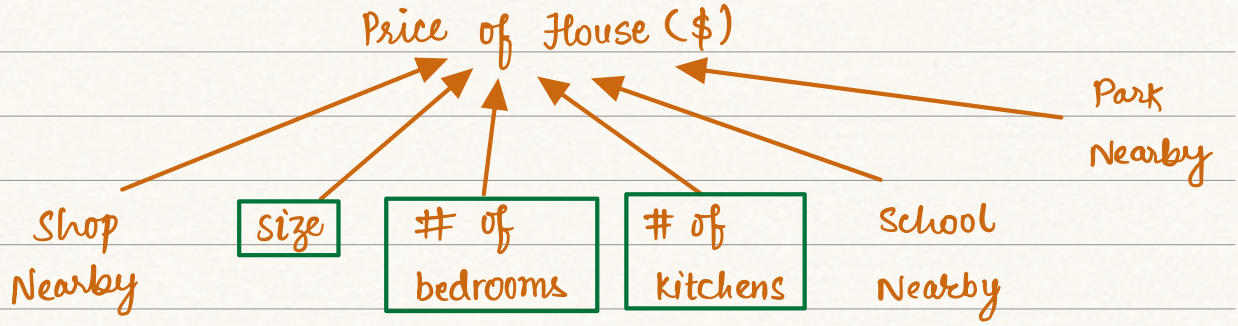


$m = 9$

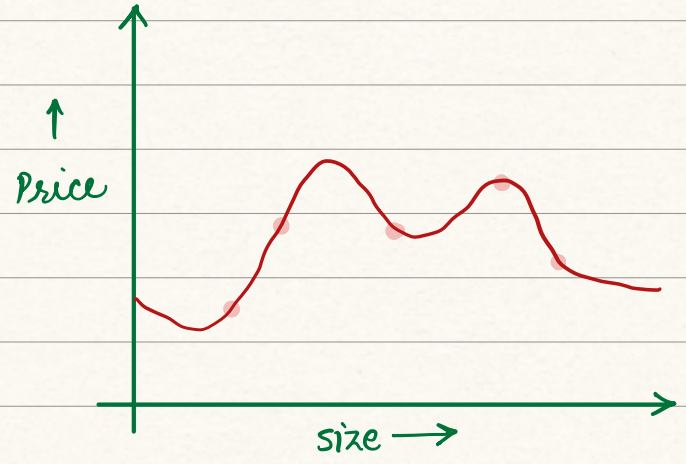
Subject: / /

Way II: Use few features

If there is a case of overfitting, only choose those features which affect the target value more than any other features.



✯ Regularization ——



$$h(x) = 100 + 28x - 385x^2 + 39x^3 - 174x^4$$

$m = 4$

Modified cost function :

$$J_{Reg}(\theta) = \frac{1}{2n}\sum_{i=1}^{n}(y_i - h(x_i))^2 + \frac{\lambda}{2}\sum_{j=1}^{d}\theta_j^2$$

↳ Also called Ridge Regression.

{ θ_0 is not included

λ = Regularization Parameter

$$h(x) = \theta_0 + \theta_1 x_i + \theta_2 x_i^2 + \dots + \theta_d x_i^d$$

Subject: / /

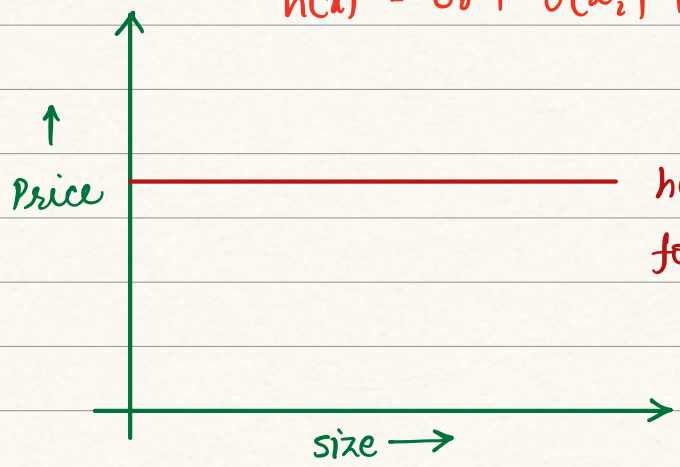
If $\lambda = 1000$;

We need to keep $\left[\frac{1000}{2} \sum_{j=1}^{d} \theta_j^2 \right]$ as small as possible.

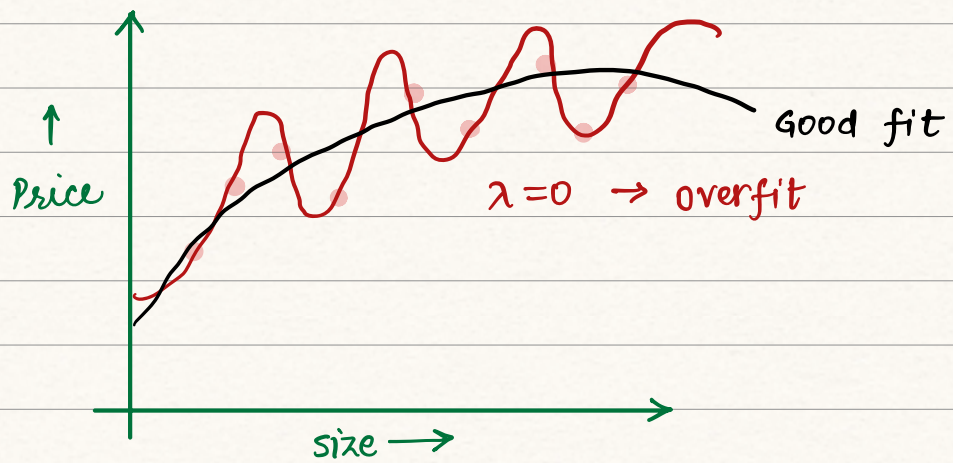
$h(x) \approx \theta_0 \longrightarrow \theta_i \approx 0 \quad \forall \quad i = 1 \text{ to } d$

case of high bias

$h(x) = \theta_0 + \theta(x_i) + \theta(x_i)^2 + \ldots + \theta(x_i^d)$



If $\lambda = 0$ and $d = 9$;

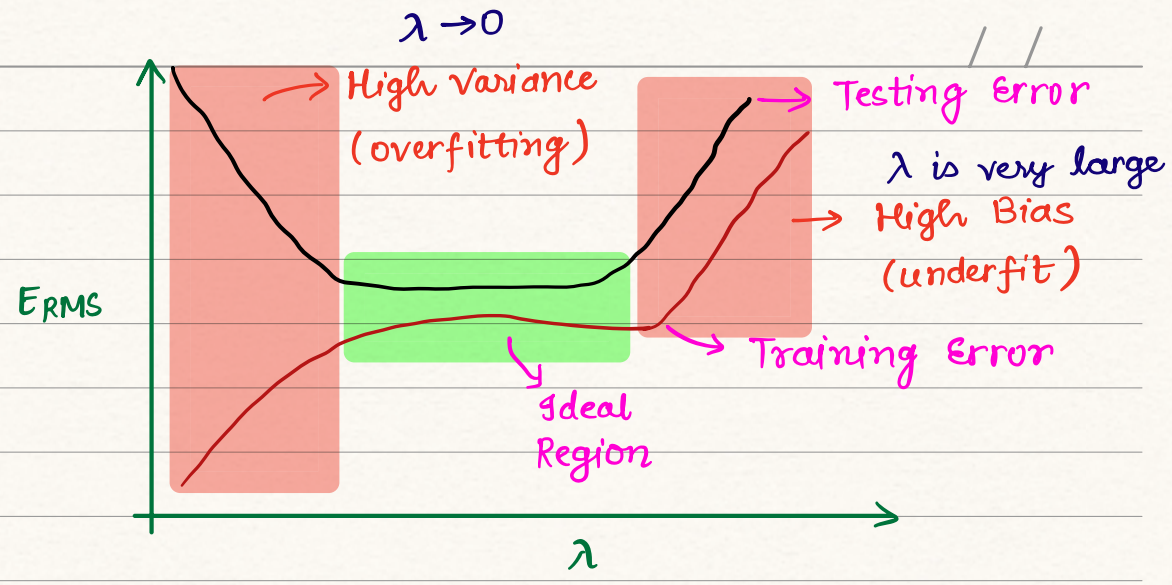


Regularized

$$\min_{\theta} \; J_{Reg}(\theta) = \frac{1}{2n} \sum_{i=1}^{n} \left(y^{(i)} - h(x^{(i)}) \right)^2 + \frac{\lambda}{2} \sum_{j=1}^{d} \theta_j^2$$

Regularized linear regression

Subject: / /

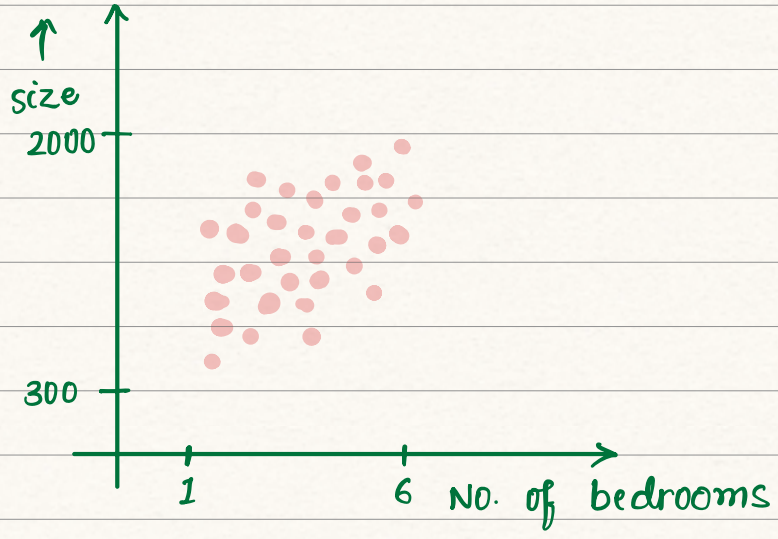


Feature Scaling (To speed up the Gradient Descent Algorithm)

Example –

Range of size of house = 300 ~ 2000 ft^2

Range of no. of bedrooms = 1 ~ 6



x_1 → size

x_2 → bedrooms

$$x_1 \text{ (scale)} = \frac{x_1}{2000 - 300}$$

$$x_2 \text{ (scale)} = \frac{x_2}{6 - 1}$$

Subject: 19/08/2025

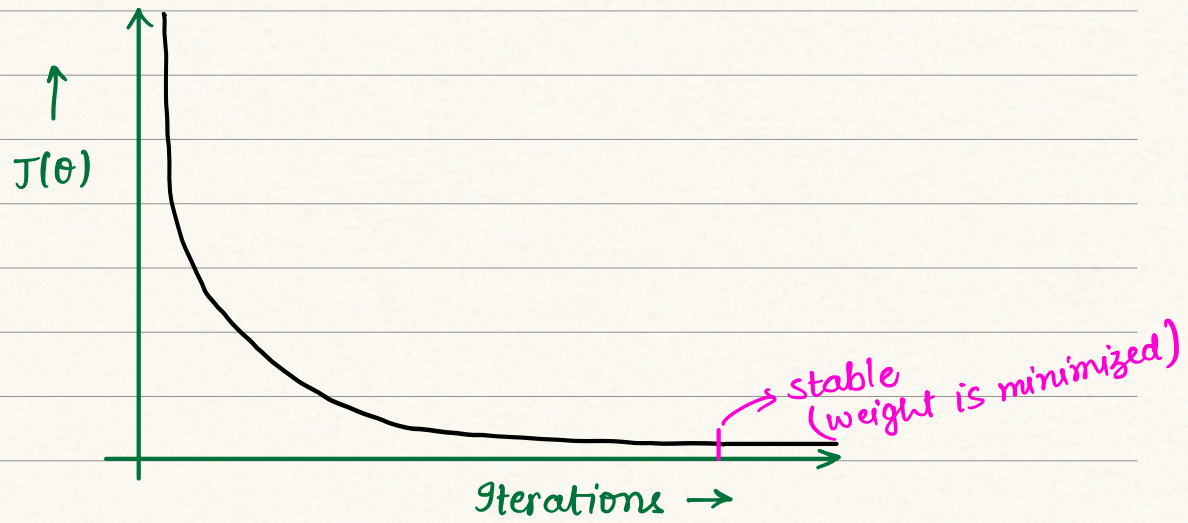
✯ Mean Normalization —

μ_1 = Average for feature x_1

μ_2 = Average for feature x_2

$$x_1 \text{ (Normalized)} = \frac{x_1 - \mu_1}{x_{1\,max} - x_{1\,min}}$$

$$x_2 \text{ (Normalized)} = \frac{x_2 - \mu_2}{x_{2\,max} - x_{1\,min}}$$



✯ Mean Squared Error (MSE) cost function : —

$$J(\theta) = \frac{1}{2n} \sum_{i=1}^{n} (y_i - h(x^{(i)}))^2$$

$$J(\theta) = \frac{1}{2n} (x\theta - y)^T (x\theta - y)$$

Theorem : A twice differentiable function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if the Hessian $\nabla^2 f$ is positive semi definite for all $x \in \mathbb{R}^n$.

Q) What is Hessian ?

Subject: / /

If $f(x)$ has continuous second-order partial derivatives, the Hessian of $f(x)$ is defined as

$$H(x) = \nabla\left((\nabla f(x))^T\right)$$

$$\nabla f(x) = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix} \begin{matrix} \rightarrow f_1 \\ \rightarrow f_2 \\ \\ \rightarrow f_n \end{matrix}$$

★ Jacobian ——

Gradient of vector valued function.

$$J = \nabla((\nabla f)^T) = \nabla\begin{bmatrix} f_1 & f_2 & \cdots & f_n \end{bmatrix}$$

$$= \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & & & \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_1 \partial x_2} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & & & \\ \frac{\partial^2 f}{\partial x_1 \partial x_n} & \frac{\partial^2 f}{\partial x_2 \partial x_n} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

Hessian matrix ↓ symmetric

$$\nabla^T = \begin{bmatrix} \frac{\partial}{\partial x_1} & \frac{\partial}{\partial x_2} & \cdots & \frac{\partial}{\partial x_n} \end{bmatrix}$$

Subject: / /

$$\nabla \nabla^T$$

$$\nabla = \begin{bmatrix} \frac{\partial}{\partial x_1} \\ \frac{\partial}{\partial x_2} \\ \vdots \\ \frac{\partial}{\partial x_n} \end{bmatrix} \qquad \nabla^T = \begin{bmatrix} \frac{\partial}{\partial x_1} & \frac{\partial}{\partial x_2} & \cdots & \frac{\partial}{\partial x_n} \end{bmatrix}$$

$$\nabla \nabla^T = \begin{bmatrix} \frac{\partial^2}{\partial x_1^2} & \frac{\partial^2}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2}{\partial x_1 \partial x_n} \\ \frac{\partial^2}{\partial x_1 \partial x_2} & \frac{\partial^2}{\partial x_2^2} & \cdots & \frac{\partial^2}{\partial x_2 \partial x_n} \\ \vdots & & & \\ \frac{\partial^2}{\partial x_1 \partial x_n} & \frac{\partial^2}{\partial x_2 \partial x_n} & \cdots & \frac{\partial^2}{\partial x_n^2} \end{bmatrix}$$

Example –

$$f(x,y) = x^2y + 3y^2$$

$$\nabla f(x,y) = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} = \begin{bmatrix} 2xy \\ 6y + x^2 \end{bmatrix} \begin{matrix} \} \rightarrow f_1 \\ \} \rightarrow f_2 \end{matrix}$$

$$J_{\nabla f(x,y)} = \begin{bmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} \end{bmatrix} = \begin{bmatrix} 2y & 2x \\ 2x & 6 \end{bmatrix}$$

Subject: / /

✯ Positive semi definite Matrix (PSD) ——

If the matrix (M) is symmetrix and $v^T M v \geq 0$ $\forall$ $v \in \mathbb{R}^n$. Then M is PSD.

$v_{n\times 1}$

$M_{n\times n}$

$v^T_{1\times n}$

M → Hessian Matrix

$M = H(x)$

$v^T H(x) v \geq 0 \quad \forall \; v \in \mathbb{R}^n$

Proving —

$$J(\theta) = \frac{1}{2n}(x\theta - y)^T (x\theta - y)$$

$$\nabla_\theta (J(\theta)) = \frac{1}{2n}(2x^T x\theta - 2x^T y)$$

$$= \frac{1}{n}(x^T x\theta - x^T y)$$

$$\nabla_\theta (J(\theta)) = \begin{bmatrix} \frac{\partial J(\theta)}{\partial \theta_0} \\ \frac{\partial J(\theta)}{\partial \theta_1} \\ \frac{\partial J(\theta)}{\partial \theta_2} \\ \vdots \\ \frac{\partial J(\theta)}{\partial \theta_d} \end{bmatrix}$$

$y \in \mathbb{R}^n$

$\theta \in \mathbb{R}^{(d+1)\times 1}$

$x \in \mathbb{R}^{n\times(d+1)}$

$x^T x\theta \in \mathbb{R}^{(d+1)\times 1}$

$x^T y \in \mathbb{R}^{(d+1)\times 1}$

Subject: / /

Hessian of $J(\theta) = \nabla_\theta \left(\nabla_\theta^T J(\theta) \right)$

$= \frac{1}{n} \nabla_\theta \left(\theta^T x^T x - y^T x \right)$

$d = 1$;

$$\theta^T x^T x = \begin{bmatrix} \theta_0 & \theta_1 \end{bmatrix} \begin{bmatrix} x_1 & x_2 \\ x_2 & x_1 \end{bmatrix}$$

$$= \begin{bmatrix} \theta_0 x_1 + \theta_1 x_2 & \theta_0 x_2 + \theta_1 x_1 \end{bmatrix}$$

$$= \begin{bmatrix} \frac{\partial f_1}{\partial \theta_0} & \frac{\partial f_1}{\partial \theta_1} \\ \frac{\partial f_2}{\partial \theta_0} & \frac{\partial f_2}{\partial \theta_1} \end{bmatrix}$$

$$= \begin{bmatrix} x_1 & x_2 \\ x_2 & x_1 \end{bmatrix} = x^T x$$

$\nabla^2 J(\theta) = x^T x \longrightarrow$ Hessian

$v^T x^T x v$ should be $\geq 0 \quad \forall \; v \in \mathbb{R}^{(d+1)}$

Concept :

suppose $x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^n$

Subject: 20/08/2025

L2 - Norm of vector x is

$$\|x\|_2 = \sqrt{x_1^2 + x_2^2 + \dots + x_n^2} \geq 0$$

$$\|x\|_2^2 = x_1^2 + x_2^2 + \dots + x_n^2 \geq 0$$

$$\|x\|_2^2 = x^T x$$

$$= \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

$$v^T x^T x v = (xv)^T (xv)$$

$x \in \mathbb{R}^{n \times (d+1)}$

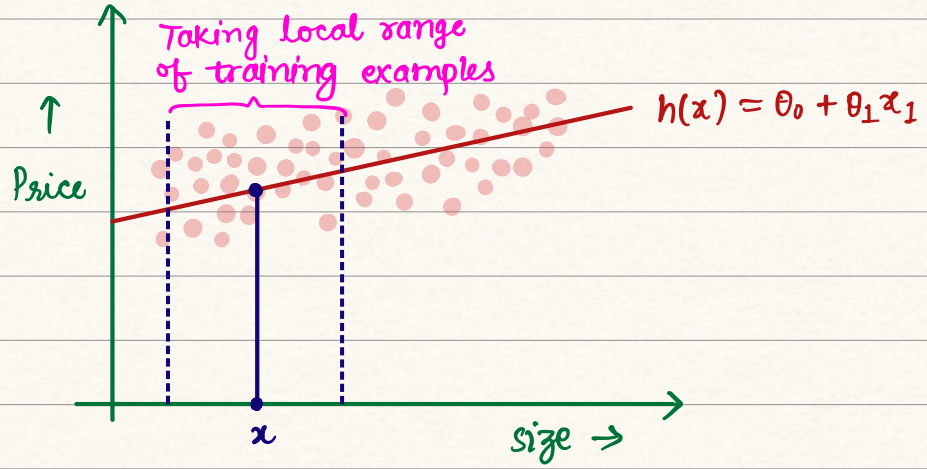
$v \in \mathbb{R}^{(d+1) \times 1}$

$(xv)^T \in \mathbb{R}^{n \times 1}$

$\|xv\|_2^2 \geq 0$

$\Rightarrow$ So cost function should be convex function.

★ Locally Weighted Linear Regression (LWLR) ——



To find the price for point x, consider only a range of local training examples.

Subject: / /

Modified cost function :

$$J(\theta) = \frac{1}{2n} \sum_{i=1}^{n} w_i \left(y^{(i)} - h(x^{(i)})\right)^2$$

where $w_i = \exp\left(-\frac{(x^{(i)} - x)^2}{2\tau^2}\right)$

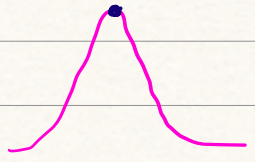
Range of local training examples

τ = hyperparameter

$\tau \rightarrow$ Defines how quickly the weight of a training example falls off with distance.

The problem here is that when we change the value of x, there will have a different range of training examples.

Decaying quickly



Needs less no. of training examples

Decaying slowly

Needs more no. of training examples

Locally weighted Linear Regression is also called non-parametric algorithm.

Unweighted (conventional) Linear Regression is called parametric Learning Algorithm.

Subject: 25/08/2025

☆ Logistic Regression

For classification

Example :-

Email spam / not spam

Online Transactions - Fraudulent (Yes/No)

Disease - Tumor (Yes/No)

Features		Output
x_1	x_2	y (Disease)
a	b	Yes
c	d	No
e	f	Yes
g	h	Yes

$y \in \{0, 1\}$

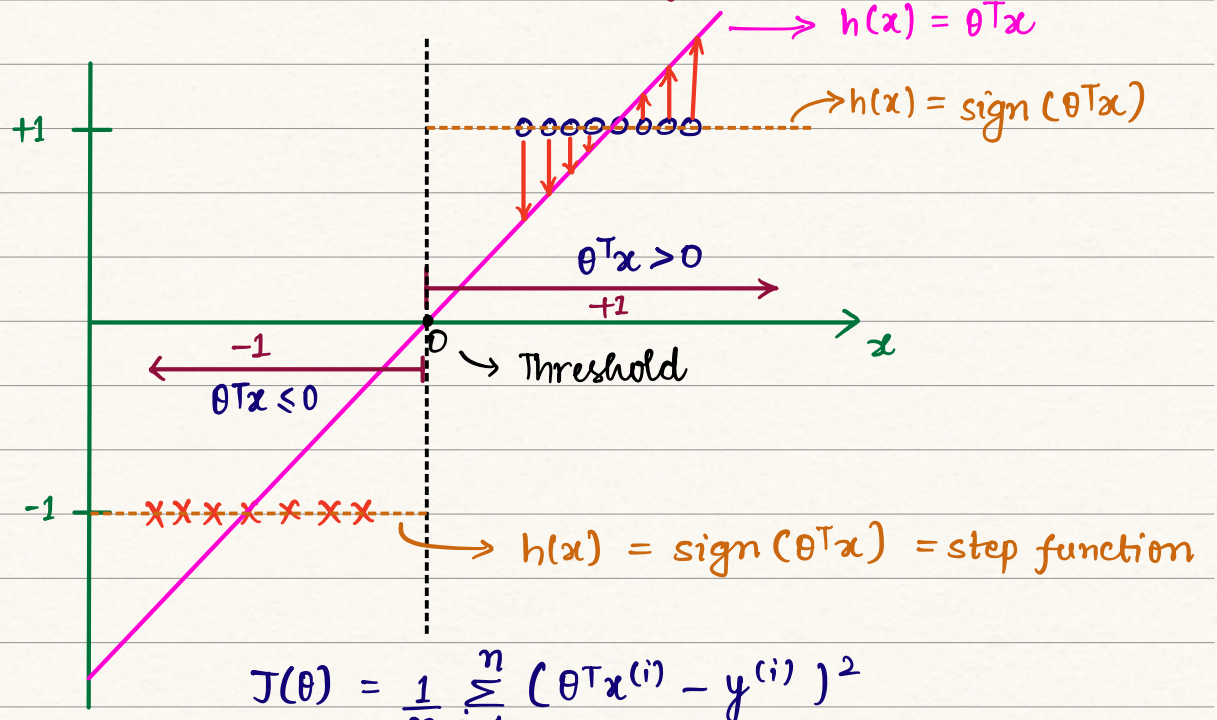
Binary classification

0 → Negative class

1 → Positive class

$y \in \{0, 1, 2, \dots\}$

Multi-class classification



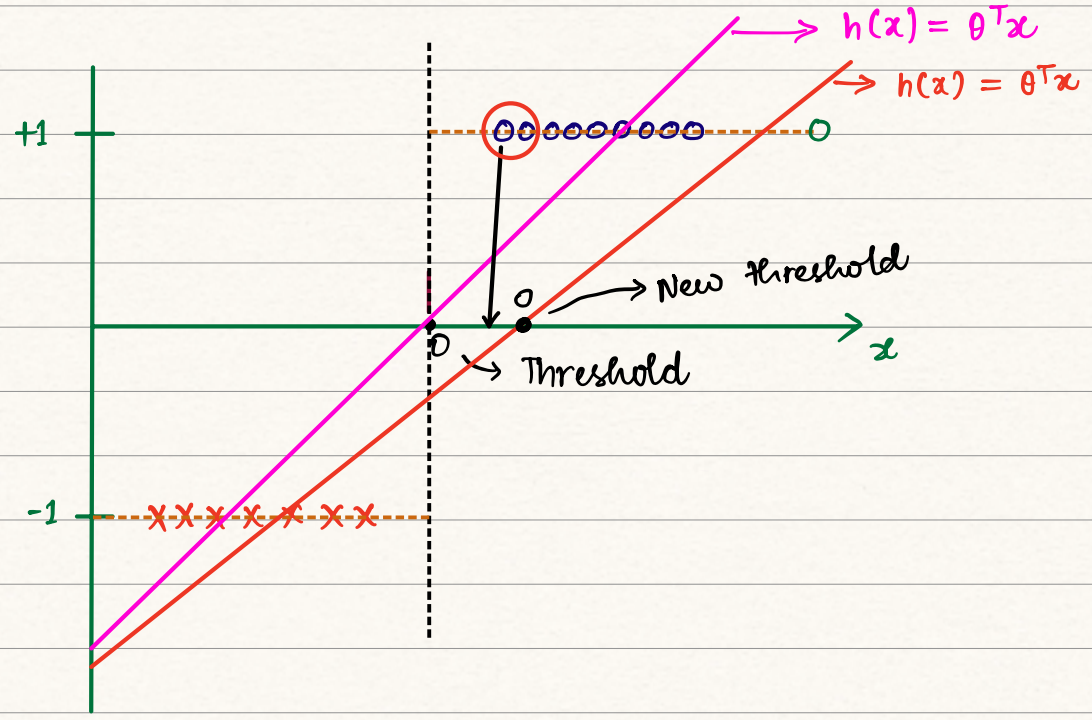
$$J(\theta) = \frac{1}{n} \sum_{i=1}^{n} (\theta^T x^{(i)} - y^{(i)})^2$$

Subject: / /

Prediction, $h(x) = \text{sign}(\theta^T x) = \begin{cases} +1 \; ; & \theta^T x > 0 \\ -1 \; ; & \theta^T x \leq 0 \end{cases}$

Q) what is the problem with this method of classification?

→ Sensitive to outlier



$$J(\theta) = \frac{1}{n} \sum_{i=1}^{n} (h(x^{(i)}) - y^{(i)})^2$$

$$h(x^{(i)}) = \theta_0 + \theta_1 x^{(i)}$$

For testing the house price based on the size of house, we were taking $h(x) = \theta^T x$ for predicting the house price (we were taking continuous range of values).

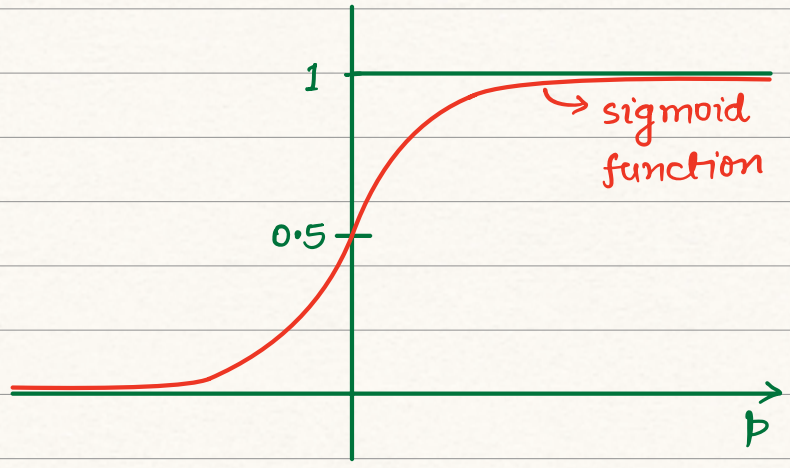
$$h(x) = \theta^T x$$

But for logistic regression, we want

$$0 \leq h(x) \leq 1$$

because $y \in \{0, 1\}$ by doing $h(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}}$

Subject: / /



$$g(p) = \frac{1}{1+e^{-p}}$$

$$p = \theta^T x$$

$$h(x) = g(\theta^T x) = \frac{1}{1+e^{-\theta^T x}} = P(y=1 \mid x;\theta)$$

Probability that $y=1$ given x & parameterized by θ

$$P(y=0 \mid x;\theta) = 1 - P(y=1 \mid x;\theta)$$

Predict "$y=1$" if $h(x) = \frac{1}{1+e^{-\theta^T x}} \geq 0.5$

Predict "$y=0$" if $h(x) = \frac{1}{1+e^{-\theta^T x}} < 0.5$

Testing ;

$g(h(x)) = \frac{1}{1+e^{-\theta^T x}} \geq 0.5 \Rightarrow x$ belongs to class 1

$g(h(x)) = \frac{1}{1+e^{-\theta^T x}} < 0.5 \Rightarrow x$ belongs to class 0

$$x = \begin{bmatrix} 1 \\ x_1 \end{bmatrix}$$

x_1 = Tumor size

$g(h(x)) = \frac{1}{1+e^{-\theta^T x}} = 0.7$ { The person has a tumor }

Subject: 26/08/2025

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{1}{1+e^{-\theta^T x^{(i)}}} - y^{(i)}\right)^2$$

Problem here is this function is no longer a convex function.

Training set :- $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})\}$

$$x^{(i)} \in \begin{bmatrix} x_0^{(i)} \\ x_1^{(i)} \\ \vdots \\ x_n^{(i)} \end{bmatrix} \qquad x_0^{(i)} = 1$$

$$\forall\ i \in \{1, 2, 3, \ldots, m\}$$

$$y^{(i)} \in \{0, 1\} \quad ; \quad i \in \{1, 2, 3, \ldots, m\}$$

$$h(x) = \frac{1}{1 + e^{-\theta^T x}}$$ → this is used for prediction purpose.

To find the optimum value of θ, we need to first find out the cost function which should be minimized w.r.t θ so as to find the optimum value of θ.

$$g(s) = \frac{1}{1+e^{-s}}$$ → sigmoid function

$$\frac{dg(s)}{ds} = \frac{d}{ds}\left(\frac{1}{1+e^{-s}}\right)$$

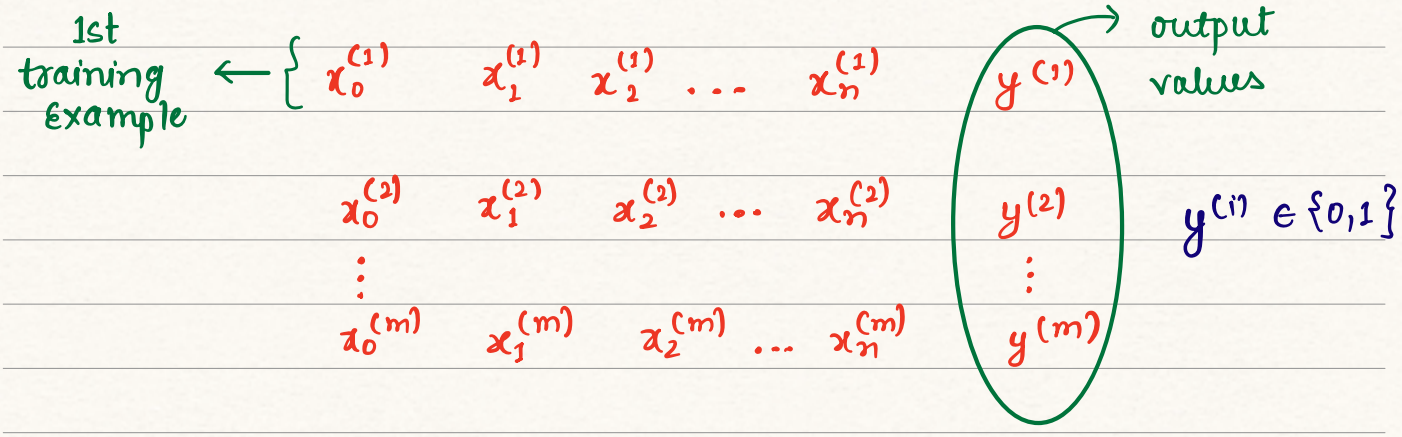
$$= \frac{e^{-s}}{(1+e^{-s})^2}$$

$$= \frac{1}{(1+e^{-s})}\left[1 - \frac{1}{1+e^{-s}}\right]$$

$$g'(s) = g(s)[1 - g(s)]$$

Subject: / /

Data set for logistic Regression



$$P(y=1 \mid x;\theta) = h(x) = \frac{1}{1+e^{-\theta^T x}}$$

$$P(y=0 \mid x;\theta) = 1 - \left[\frac{1}{1+e^{-\theta^T x}}\right]$$

output $y \in \{0,1\}$ is a Bernoulli Distributed Random variable.

Basics of Probability :-

If X is a Bernoulli distributed Random variable with $P(X=1) = p$, $P(X=0) = 1-p = q$. Then the probability mass function of Bernoulli distribution is

$$f(k,p) = p^k q^{(1-k)}$$

$$= p^k (1-p)^{(1-k)} \qquad k \in \{0,1\}$$

$$f(y \mid X;\theta) = (h(x))^y (1-h(x))^{1-y}$$

$$y \in \{0,1\}$$

X, Y are independent

Subject: / /

$$f_{XY}(x,y) = f_X(x) f_Y(y)$$

$$f(Y|X;\theta) = \prod_{i=1}^{m} \left(h(x^{(i)})\right)^{y^{(i)}} \left(1-h(x^{(i)})\right)^{1-y^{(i)}}$$

↓ considering all y

$$= L(\theta|X;Y)$$

log likelihood ;

$$l(\theta) = \log(L(\theta|X;Y))$$

$$= \sum_{i=1}^{m} y^{(i)} \log h(x^{(i)}) + (1-y^{(i)}) \log\left(1-h(x^{(i)})\right)$$

To maximize the log likelihood, we would consider the Gradient ascent

update Rule :

$$\theta := \theta + \alpha \nabla_\theta l(\theta)$$

For simplification, suppose $m = 1$ (i.e., only one training example)

$$l(\theta) = y \log(h(x)) + (1-y) \log(1-h(x))$$

$$\frac{\partial l(\theta)}{\partial \theta_j} = \frac{y}{h(x)} \frac{\partial}{\partial \theta_j}(h(x)) + \frac{(1-y)}{1-h(x)} \cdot (-1) \cdot \frac{\partial}{\partial \theta_j} h(x)$$

$$\frac{\partial h(x)}{\partial \theta_j} = h(x)(1-h(x)) \cdot \frac{\partial}{\partial \theta_j}(\theta^T x)$$

$$= h(x)(1-h(x)) \cdot x_j$$

$\theta_0 + \theta_1 x_1 + \theta_2 x_2$

$\frac{\partial}{\partial \theta_2} = x_2$

$$\frac{\partial}{\partial \theta_j} l(\theta) = \left[\frac{y}{h(x)} - \frac{(1-y)}{1-h(x)}\right] \left(h(x)(1-h(x)) \cdot x_j\right)$$

Subject: / /

$$\Rightarrow \theta_j := \theta_j + \alpha\,(y - h(x)).x_j$$

$$l(\theta) = \log(L(\theta))$$

$$= \sum_{i=1}^{m} y^{(i)} \log h(x^{(i)}) + (1-y^{(i)}) \log(1-h(x^{(i)}))$$

$$h(x) = \frac{1}{1+e^{-\theta^T x}}$$

$$l(\theta) = -\left[\sum_{i=1}^{m} y^{(i)} \log(1+\exp(-\theta^T x^{(i)})) + (1-y^{(i)}) \log(1+\exp(\theta^T x^{(i)}))\right]$$

Logistic loss, $L_{loss} = -l(\theta)$

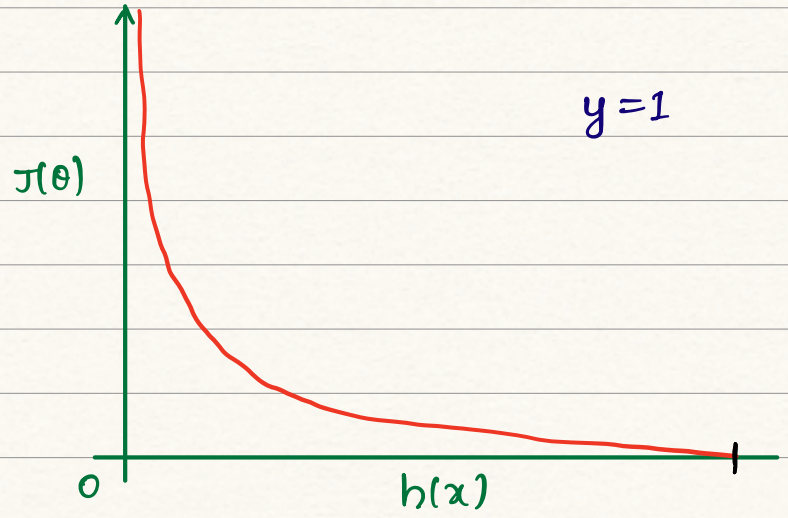
↳ Binary cross entropy loss function

for $m = 1$;

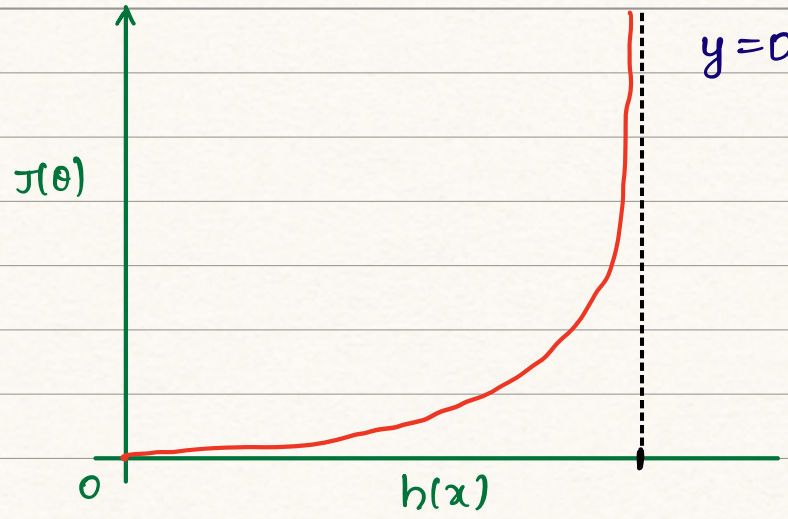
$$J(\theta) = -l(\theta) = -y \log(h(x)) - (1-y) \log(1-h(x))$$

$y = 1 \longrightarrow J(\theta) = -\log(h(x))$

$y = 0 \longrightarrow J(\theta) = -\log(1-h(x))$



Subject: 01/09/2025



★ Binary Cross Entropy Loss function :——

$$J(\theta) = -l(\theta) = -\left(\sum_{i=1}^{m} y^{(i)} \log h(x^{(i)}) + (1-y^{(i)}) \log (1-h(x^{(i)}))\right)$$

m = no. of training examples

$y^{(i)}$ = True label corresponding to the training example

$h(x^{(i)})$ = Predicted probability corresponding to ith training example

$$J(\theta) = -\left(\sum_{i=1}^{m} y^{(i)} \log h(x^{(i)}) + (1-y^{(i)}) \log (1-h(x^{(i)}))\right)$$

$$= -\left(\sum_{i=1}^{m} y^{(i)} \log h(x^{(i)}) + \log (1-h(x^{(i)})) - y^{(i)} \log (1-h(x^{(i)}))\right)$$

$$= -\left(\underbrace{\sum_{i=1}^{m} y^{(i)} \log \frac{h(x^{(i)})}{1-h(x^{(i)})}}_{\text{I}} + \underbrace{\log (1-h(x^{(i)}))}_{\text{II}}\right)$$

Solving I ;

$$\log\left(\frac{h(x^{(i)})}{1-h(x^{(i)})}\right) = \log\left(\frac{\frac{1}{1+e^{-\theta^T x^{(i)}}}}{1-\frac{1}{1+e^{-\theta^T x^{(i)}}}}\right)$$

$$h(x^{(i)}) = \frac{1}{1+e^{-\theta^T x^{(i)}}}$$

Subject: / /

$$\log\left(\frac{\frac{1}{1+e^{-\theta^T x^{(i)}}}}{1-\frac{1}{1+e^{-\theta^T x^{(i)}}}}\right) \equiv \log\left(e^{+\theta^T x^{(i)}}\right)$$

$$\Rightarrow \theta^T x^{(i)}$$

$$\theta^T x^{(i)} = \theta_0 + \theta_1 x_1^{(i)} + \theta_2 x_2^{(i)} + \ldots + \theta_d x_d^{(i)}$$

$$\nabla_\theta(x) = \begin{bmatrix} \frac{\partial x}{\partial \theta_0} \\ \frac{\partial x}{\partial \theta_1} \\ \vdots \\ \frac{\partial x}{\partial \theta_d} \end{bmatrix} = \begin{bmatrix} 1 \\ x_1^{(i)} \\ \vdots \\ x_d^{(i)} \end{bmatrix} = x^{(i)}$$

$$v^T H v \geq 0$$

For function to be convex

Hessian is positive semi definite so function is convex for $\forall \quad v \in \mathbb{R}^{(d+1)}$.

$\Rightarrow \theta^T x^{(i)}$ is convex

→ (-ve) comes from the whole expression.

$F = -\log(1-h(x^{(i)}))$ is convex.

I want to prove this. How can we prove this?

$$F = -\log\left(1-h(x^{(i)})\right)$$

$$= -\log\left(1-\frac{1}{1+e^{-\theta^T x^{(i)}}}\right)$$

$$= -\log\left(\frac{e^{-\theta^T x^{(i)}}}{1+e^{-\theta^T x^{(i)}}}\right)$$

$$= -\log\left(e^{-\theta^T x^{(i)}}\right) - \log\left(1+e^{-\theta^T x^{(i)}}\right)$$

Subject:

$$= \theta^T x^{(i)} + \log\left(1 + e^{-\theta^T x^{(i)}}\right)$$

$$\nabla_\theta(F) = x^{(i)} + \nabla_\theta\left(\log\left(1 + e^{-\theta^T x^{(i)}}\right)\right)$$

$$= x^{(i)} + \frac{-e^{-\theta^T x^{(i)}}}{1 + e^{-\theta^T x^{(i)}}} \cdot x^{(i)}$$

$$= x^{(i)}\left[1 - \frac{e^{-\theta^T x^{(i)}}}{1 + e^{-\theta^T x^{(i)}}}\right]$$

$$= \frac{1}{1 + e^{-\theta^T x^{(i)}}} \cdot x^{(i)}$$ → scalar

$$= h(x^{(i)}) \cdot x^{(i)}$$

Hessian is $\nabla_\theta(\nabla_\theta^T(F))$

$$\nabla_\theta(F) = h(x^{(i)}) \cdot x^{(i)}$$

$$\nabla_\theta^T(F) = h(x^{(i)}) \cdot (x^{(i)})^T$$

$$\nabla_\theta(\nabla_\theta^T(F)) = \nabla_\theta\left(h(x^{(i)}) \cdot x^{(i)T}\right)$$

$$= \nabla_\theta\left(\frac{1}{1 + e^{-\theta^T x^{(i)}}}\right) \cdot x^{(i)T}$$

$$= \frac{e^{-\theta^T x^{(i)}}}{\left(1 + e^{-\theta^T x^{(i)}}\right)^2} \cdot x^{(i)} \cdot x^{(i)T}$$

$$\equiv \left(\frac{1}{e^{-\theta^T x^{(i)}}}\right)\left(1 - \frac{1}{1 + e^{-\theta^T x^{(i)}}}\right) \cdot x^{(i)} \cdot x^{(i)T}$$

Subject: / /

$$= \boxed{h(x^{(i)})\left(1-h(x^{(i)})\right)} . x^{(i)} . x^{(i)T}$$

↓ Scalar

$x^{(i)} \in \mathbb{R}^{(d+1)\times 1}$

$x^{(i)T} \in \mathbb{R}^{1\times(d+1)}$

$x^{(i)} . x^{(i)T}$ is symmetric

$$= h(x^{(i)})\left(1-h(x^{(i)})\right) . x^{(i)} x^{(i)T}$$

$$= P$$

Next task is to prove that $v^T P v \geq 0$; $v \in \mathbb{R}^{(d+1)\times 1}$

{ $v^T P v$ = Scalar }

$$v^T x^{(i)} . x^{(i)T} v \equiv \| v^T x^{(i)} \|_2^2 \geq 0$$

So F is convex.

$$= -\left(\sum_{i=1}^{m} y^{(i)} \log \frac{h(x^{(i)})}{1-h(x^{(i)})} + \log\left(1-h(x^{(i)})\right) \right)$$

Convex Convex

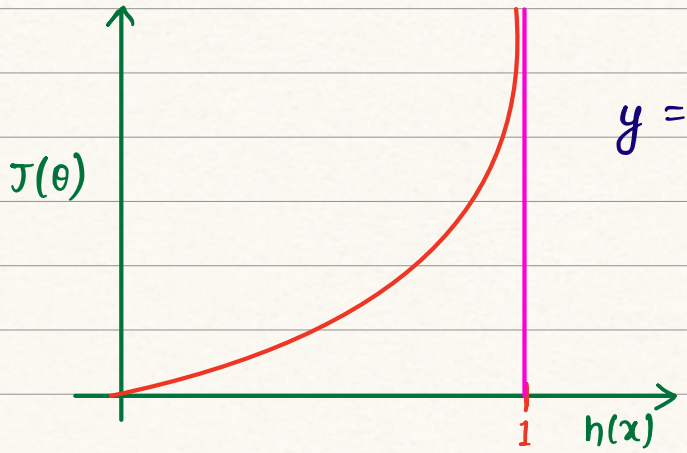
$$x^{(i)} = \begin{bmatrix} 1 \\ x_1^{(i)} \\ x_2^{(i)} \\ \vdots \\ x_d^{(i)} \end{bmatrix}$$

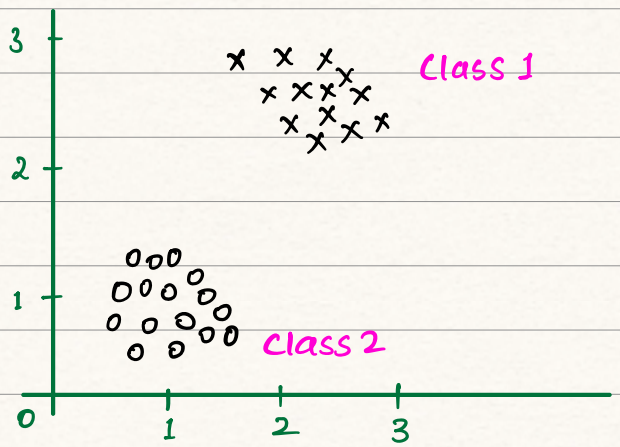
$$\| x^{(i)} \|_2^2 = 1 + (x_1^{(i)})^2 + (x_2^{(i)})^2 + \dots + (x_d^{(i)})^2$$

$$J(\theta) = -y \log h(x) - (1-y) \log\left(1-h(x)\right)$$

$$J(\theta) = \begin{cases} -\log h(x) \text{ ; if } y = 1 \\ -\log\left(1-h(x)\right) \text{ ; if } y = 0 \end{cases}$$

Subject: / /



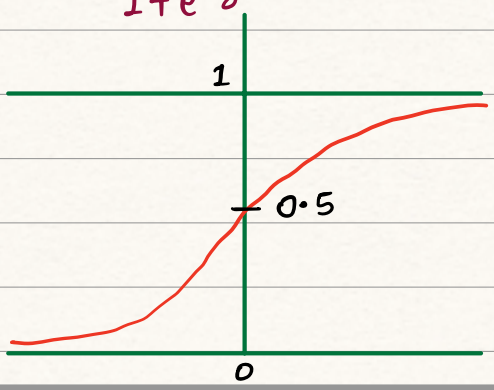


Decision Boundary

X → Class 1

O → Class 2

$$g(z) = \frac{1}{1+e^{-z}}$$



Subject: / /

Predict '1' when $g(z) \geq 0.5$

Predict '0' when $g(z) < 0.5$

$$h(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

$$= \frac{1}{1 + e^{-\theta^T x}} = \frac{1}{1 + e^{-(\theta_0 + \theta_1 x_1 + \theta_2 x_2)}}$$

Suppose you get $\theta_0 = -3$, $\theta_1 = 1$, $\theta_2 = 1$

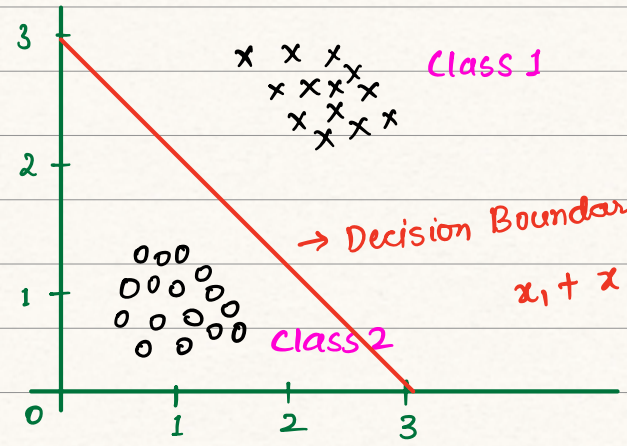
$$\theta = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix}$$

Predict '1' when

$\theta_0 + \theta_1 x_1 + \theta_2 x_2 \geq 0$

$-3 + 1 x_1 + 1 x_2 \geq 0$

$x_1 + x_2 \geq 3$



→ Decision Boundary ($\theta_0 + \theta_1 x_1 + \theta_2 x_2 = 0$)

$x_1 + x_2 = 3$

Testing

Suppose you get $x^{(i)}$

Then $h_\theta(x^{(i)}) = \frac{1}{1 + e^{-\theta^T x^{(i)}}} \geq 0.5 \rightarrow y = 1$

↑ we know θ now after training

$x_1 + x_2 = 3$

$x_1^{(i)} + x_2^{(i)} > 3$

Subject: 02/09/2025

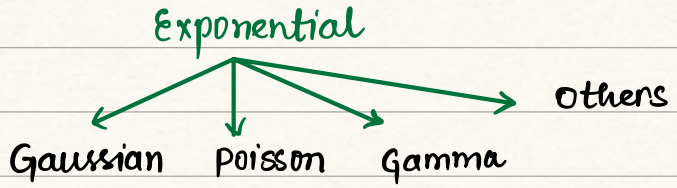
✯ Exponential Family ——

* Distribution falls under the exponential family if it can be written of the form :—

$$f(y;x) = b(y)\exp(\eta^T T(y) - a(\eta))$$

$\eta \rightarrow$ Natural parameter

$T(y) = y$ (sufficient statistic)



Q.) Gaussian and Bernoulli are the family of exponential distribution. Prove it.

Bernoulli Distribution

$$y \in \{0,1\}$$

$$P(y=1 \mid x;\theta) = P(y=1) = \phi$$

$$P(y=0 \mid x;\theta) = 1 - P(y=1) = 1-\phi$$

$$f(y;\phi) = \phi^y (1-\phi)^{1-y} \quad ; \quad y \in \{0,1\}$$

We want this expression to be in the form of

$$f(y;x) = b(y)\exp(\eta^T T(y) - a(\eta))$$

$$\begin{aligned} f(y;\phi) &= \phi^y (1-\phi)^{1-y} \\ &= \exp\left[\log\left(\phi^y (1-\phi)^{1-y}\right)\right] \\ &= \exp\left[y\log\phi + (1-y)\log(1-\phi)\right] \\ &= \exp\left[y\log\phi - y\log(1-\phi) + \log(1-\phi)\right] \\ &= \exp\left[y\log\left(\frac{\phi}{1-\phi}\right) + \log(1-\phi)\right] \end{aligned}$$

$$b(y) = 1 \qquad \eta = \log\left(\frac{\phi}{1-\phi}\right)$$

Subject: / /

$T(y) = y \qquad a(\eta) = \log(1-\phi)$

$\phi = \frac{1}{1+e^{-x}} \} P(y=1)$

sigmoid function

Gaussian Distribution ($\sigma^2 = 1$)

$$f(y; \mu, \sigma^2) = f(y; \mu)$$
$$= \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(y-\mu)^2}{2}\right)$$
$$= \frac{1}{\sqrt{2\pi}} \exp\left(\frac{-(y^2+\mu^2-2y\mu)}{2}\right)$$
$$= \frac{1}{\sqrt{2\pi}} \exp\left(\frac{-y^2}{2}\right) \exp\left(\mu y - \frac{\mu^2}{2}\right)$$

$$b(y) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{y^2}{2}\right)$$

$T(y) = y$

$\eta = \mu$

$a(\eta) = \frac{\mu^2}{2} = \frac{\eta^2}{2}$

☆ Generalized Linear Models ——

$h(x) = \theta^T x \longrightarrow$ Linear Regression

$h(x) = \frac{1}{1+e^{-\theta^T x}} \longrightarrow$ Logistic Regression

Belong to the class of Generalized linear models.

Subject: / /

In Regression ;

$$y|x;\theta \sim \mathcal{N}(\mu, \sigma^2)$$
$$\mu = \theta^T x$$

In Logistic ;

$$y|x;\theta \sim \text{Bernoulli}(p)$$

$$p = P(y=1|x;\theta) = \frac{1}{1+e^{-\theta^T x}}$$

Constructing Generalized Linear Models —

* Following assumptions are made :

1.) $y|x;\theta \sim$ Exponential family (η)

2.) For a given x, prediction $h(x) = E[y|x]$

$X = \{x_1, x_2\}$ ← Discrete Random Variable

p_1 p_2

$$E[X] = p_1 x_1 + p_2 x_2$$
$$E[y] = 1 \cdot \phi + 0(1-\phi) \qquad y \in \{0, 1\}$$
$$\left\{\phi = \frac{1}{1+e^{-\theta^T x}}\right\}$$

3.) The natural parameter η and the input vector x is related as $\eta = \theta^T x$.

* Linear Regression is a special case of Generalized Linear models.

1.) $y|x;\theta \sim$ Exponential (η)
$\sim$ Gaussian (μ)
$\Rightarrow \mu = \eta$

Subject: / /

2.) $h(x) = E[y|x] = \mu = \eta$

3.) $\eta = \theta^T x$

* Logistic Regression ——

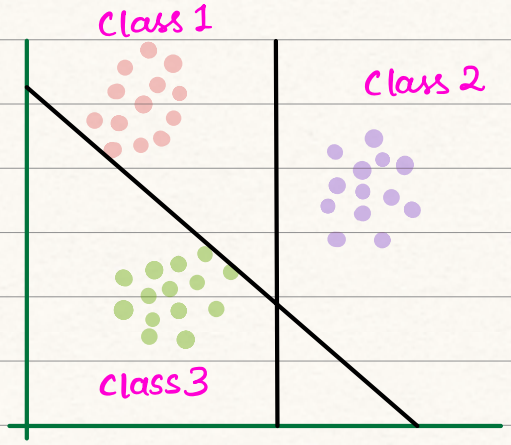
1.) $y|x;\theta \sim$ Exponential (η)
$\sim$ Bernoulli (ϕ)

2.) $h(x) = E[y|x] = \phi = \frac{1}{1+e^{-\eta}}$

3.) $\eta = \theta^T x$

$\Rightarrow h(x) = \frac{1}{1+e^{-\theta^T x}}$

☆ Multi-class classification ——



Softmax function

	$P(y\|x;\theta)$
Class 1	0.8
Class 2	0.15
Class 3	0.05

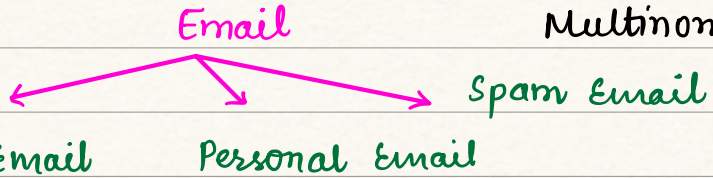
☆ Softmax Regression ——

Multi-class classification

$y \in \{1, 2, \ldots, k\}$ → Discrete Value

Example :-

This can be modeled with Multinomial Distribution.



Subject: 03/09/2025

Task 1 : Let's represent multinomial distribution as a class of exponential distribution.

* Multinomial distribution over k possible outcomes.

Let $\phi_1, \phi_2, \cdots, \phi_k$

$P(y=1)$ $P(y=2)$ $P(y=k)$

$$\begin{cases} \sum_{i=1}^{k} \phi_i = 1 \end{cases}$$

$$\phi_i = P(y=i ; \phi)$$
$$\phi_k = P(y=k ; \phi)$$

$$\Rightarrow \phi_k = 1 - \sum_{i=1}^{k-1} \phi_i$$

Exponential Distribution :

$$f(y ; \eta) = b(y) \exp\left(\eta^T T(y) - a(\eta)\right)$$

(η: vector, $T(y)$: vector)

To express the multinomial as an Exponential family distribution, we will define ;

$$T(y) \in \mathbb{R}^{(k-1)}$$

$$T(1) = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \rightarrow \text{First Entry} \qquad T(2) = \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \rightarrow \text{second Entry}$$

$$T(k-1) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix} \rightarrow (k-1)^{th} \text{ entry}$$

Subject: / /

$T(y)_i$ → denotes the i^{th} element of vector $T(y)$.

$$T(y)_i = \underbrace{1\{y=i\}}_{\text{Indicator function}}$$

Indicator Function :—

An indicator function $1\{\cdot\}$ takes on the value 1 if its argument is true & 0 otherwise.

$1\{\text{True}\} = 1$ $\qquad$ $1\{\text{False}\} = 0$

$1\{2=3\} = 0$ $\qquad$ $1\{3=3\} = 1$

$$1\{y,i\} = \begin{cases} 1 & \text{if } y=i \\ 0 & \text{if } y\neq i \end{cases}$$

$$\begin{aligned} E[1\{y,i\}] &= 1\cdot P(y=i) + 0\cdot P(y\neq i) \\ &= P(y=i) \\ &= \phi_i \end{aligned}$$

$$f(y,\phi) = \phi_1^{1\{y=1\}} \cdot \phi_2^{1\{y=2\}} \cdot \phi_3^{1\{y=3\}} \cdots \phi_k^{1\{y=k\}}$$

For $k=2$;

$$f(y,\phi) = \phi^{1\{y=1\}} \cdot \phi^{1\{y=2\}}$$

$$1\{y=k\} = 1 - \sum_{i=1}^{k-1} 1\{y=i\}$$

$$= \phi_1^{1\{y=1\}} \cdot \phi_2^{1\{y=2\}} \cdots \phi_k^{1-\sum_{i=1}^{k-1} 1\{y=i\}}$$

$$= \phi_1^{(T(y))_1} \cdot \phi_2^{(T(y))_2} \cdots \phi_k^{1-\sum_{i=1}^{k-1}(T(y))_i}$$

$$= \exp\Big[(T(y))_1 \log\phi_1 + (T(y))_2 \log\phi_2 + \cdots + (T(y))_{k-1}\log\phi_{k-1} + \log\phi_k - \log\phi_k \sum_{i=1}^{k-1}(T(y))_i\Big]$$

Subject: / /

$$= \exp\Big[(T(y))_1 \log \frac{\phi_1}{\phi_k} + (T(y))_2 \log \frac{\phi_2}{\phi_k} + \cdots + (T(y))_{k-1} \log \frac{\phi_{k-1}}{\phi_k} + \log \phi_k \Big]$$

$$= b(y) \exp(\eta^T T(y) - a(\eta))$$

$$\eta = \begin{bmatrix} \log\left(\frac{\phi_1}{\phi_k}\right) \\ \log\left(\frac{\phi_2}{\phi_k}\right) \\ \vdots \\ \log\left(\frac{\phi_{k-1}}{\phi_k}\right) \end{bmatrix} \qquad T(y) = \begin{bmatrix} (T(y))_1 \\ (T(y))_2 \\ \vdots \\ (T(y))_{k-1} \end{bmatrix}$$

$$\eta_i = \log \frac{\phi_i}{\phi_k} \qquad i = 1, 2, \ldots, k-1$$

$$\eta_k = \log \frac{\phi_k}{\phi_k}$$

$$a(\eta) = -\log(\phi_k)$$

$$b(y) = 1$$

$$\phi_i = \phi_k \exp(\eta_i)$$

$$\Rightarrow \phi_k = \frac{\phi_i}{\exp(\eta_i)}$$

$$\sum_{i=1}^{k} \phi_i = 1 \qquad \Rightarrow \phi_k \sum_{i=1}^{k} \exp(\eta_i) = 1$$

$$\phi_k = \frac{1}{\sum_{i=1}^{k} \exp(\eta_i)}$$

Subject: / /

$$\phi_i = \frac{\exp(\eta_i)}{\sum_{i=1}^{k} \exp(\eta_i)}$$ } Softmax Function

$\eta_i = \theta_i^T x \quad ; \; i = 1, 2, \ldots, k-1$

$\eta_k = \theta_k^T x = 0$

$\Rightarrow \theta_k = 0$ (zero vector)

$$P(y = i \mid x; \theta) = \phi_i = \frac{\exp(\eta_i)}{\sum_{j=1}^{k} \exp(\eta_j)} = \frac{\exp(\theta_i^T x)}{\sum_{j=1}^{k} \exp(\theta_j^T x)}$$

$h(x) = E[T(y) \mid x; \theta)$

$$E\left[\begin{matrix} 1\{y=1\} \\ 1\{y=2\} \\ \vdots \\ 1\{y=k-1\} \end{matrix} \;\middle|\; x; \theta\right] \Rightarrow \begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_{k-1} \end{bmatrix}$$

$$\begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_{k-1} \end{bmatrix} = \frac{1}{\sum_{j=1}^{k} \exp(\theta_j^T x)} \begin{bmatrix} \exp(\theta_1^T x) \\ \exp(\theta_2^T x) \\ \vdots \\ \exp(\theta_{k-1}^T x) \end{bmatrix}$$

$$P(y = k \mid x; \theta) = 1 - \sum_{i=1}^{k-1} P(y = i \mid x; \theta)$$

$$P(y = 1 \mid x; \theta) = \frac{\exp(\theta^T x)}{1 + \exp(\theta_1^T x)} = \frac{1}{1 + \exp(-\theta_1^T x)} = h(x)$$

$\exp(0) = 1$

$$P(y = 0 \mid x; \theta) = \frac{1}{1 + \exp(\theta^T x)} = 1 - P(y = 1 \mid x; \theta)$$

Subject: 10 / 09 / 2025

Multiclass Classification

Likelihood function

$\{x^{(i)}, y^{(i)}\}$; $i = 1, 2, \ldots, n$

Each $y^{(i)}$ is IID.

$$\begin{bmatrix} \text{cat} \\ \text{dog} \\ \text{horse} \end{bmatrix} \qquad \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$ → This corresponds to cat

$x_0^{(1)}$ $x_1^{(1)}$ $x_2^{(1)}$ → $y^{(1)}$ $\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$ {cat}

$x_0^{(2)}$ $x_1^{(2)}$ $x_2^{(2)}$ → $y^{(2)}$ $\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$ {dog}

$$f(y^{(i)}; \phi) = \phi_1^{1\{y^{(i)}=1\}} \cdot \phi_2^{1\{y^{(i)}=2\}} \cdots \phi_k^{1\{y^{(i)}=k\}}$$

$$= \prod_{l=1}^{k} \phi_l^{1\{y^{(i)}=l\}}$$ } Multinomial Distribution

Joint Probability

$$f(y, \phi) = f(y^{(1)}, \phi) \cdot f(y^{(2)}, \phi) \cdots f(y^{(n)}, \phi)$$

$$= \prod_{i=1}^{n} f(y^{(i)}, \phi)$$

$$= \prod_{i=1}^{n} \prod_{l=1}^{k} \phi_l^{1\{y^{(i)}=l\}}$$

$$\phi_j = P(y^{(i)} = j) = P(y^{(i)} = j \mid x; \theta) = \frac{e^{\theta_j^T x^{(i)}}}{\sum_{s=1}^{k} e^{\theta_s^T x^{(i)}}}$$

Subject:

$$f(y,\phi) = \prod_{i=1}^{n}\prod_{l=1}^{k}\left(\frac{e^{\theta_l^T x^{(i)}}}{\sum_{j=1}^{k} e^{\theta_j^T x^{(i)}}}\right)^{1\{y^{(i)}=l\}}$$

$L(\theta)$ → Likelihood function

log likelihood function = $l(\theta)$

$$\log(L(\theta)) = l(\theta)$$

$$= \log\left(\prod_{i=1}^{n}\prod_{l=1}^{k}\left(\frac{e^{\theta_l^T x^{(i)}}}{\sum_{j=1}^{k} e^{\theta_j^T x^{(i)}}}\right)^{1\{y^{(i)}=l\}}\right)$$

$$= \sum_{i=1}^{n}\log\prod_{l=1}^{k}\left(\frac{e^{\theta_l^T x^{(i)}}}{\sum_{j=1}^{k} e^{\theta_j^T x^{(i)}}}\right)^{1\{y^{(i)}=l\}}$$

$$= \sum_{i=1}^{n}\sum_{l=1}^{k} 1\{y^{(i)}=l\}\log\left(\frac{e^{\theta_l^T x^{(i)}}}{\sum_{j=1}^{k} e^{\theta_j^T x^{(i)}}}\right)$$

Cost function, $J(\theta) = -l(\theta)$

$$= -\sum_{i=1}^{n}\sum_{l=1}^{k} 1\{y^{(i)}=l\}\log p(y^{(i)}=l \mid x^{(i)};\theta)$$

↓

Cross entropy loss function which is a convex function

Subject: 15/09/2025

$$h_\theta(x) = \begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \end{bmatrix} = \frac{1}{\sum_{j=1}^{3} e^{\theta_j^T x}} \begin{bmatrix} \exp(\theta_1^T x) \\ \exp(\theta_2^T x) \\ \exp(\theta_3^T x) \end{bmatrix}$$

Till now we have considered the Learning Algorithms that modeled as $f(y|x;\theta)$ i.e., conditional distribution of the output y given the input feature vector x and parameterized by the weight vector θ.

$f(y|x;\theta) \sim$ Normal (Linear Regression)

$f(y|x;\theta) \sim$ Bernoulli (Logistic Regression)

$f(y|x;\theta) \sim$ Multinomial (Multiclass Regression)

$f(y|x;\theta) \sim$ Exponential Distribution (GLMs)

These algorithms are called Discriminative Learning Algorithms.

We would now consider the algorithms that also consider/model the input feature vector x i.e., we would define $f(x|y;\theta)$. These algorithms are called Generative Learning Algorithms.

$f(x|y;\theta)$

↓

Conditional Probability density function (pdf) of input feature vector x for given y and θ.

Subject: / /

$P(y)$ → Prior probability of y

Using Bayes' Theorem;

$$P(y|x;\theta) = \frac{f(x|y;\theta)\,P(y)}{f(x)}$$

$P(y|x;\theta)$ → Posterior distribution of y given x and θ

$f(x|y;\theta)$ → Descrete Random Variable

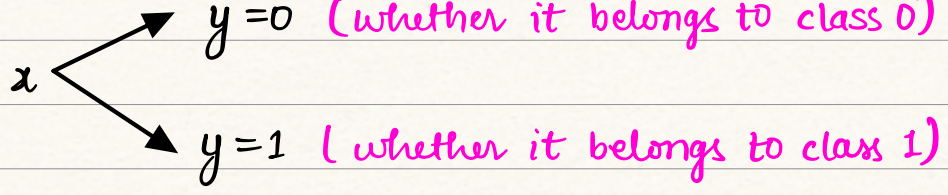
$f(x)$ → Continuous Random Variable

$$P(y|x;\theta) = \frac{f(x|y;\theta)\,P(y)}{f(x|y=0;\theta)\,P(y=0) + f(x|y=1;\theta)\,P(y=1)}$$

θ → Known

Now if I give x as input then how will you decide either $y=0$ or $y=1$?

New training example $x = [x_0 \;\; x_1 \;\; x_2 \;\ldots, x_N]$



To make decision, we should consider;

$$\arg\max_y P(y|x;\theta) = \arg\max_y f(x|y;\theta)\,P(y)$$

$y \in \{0, 1\}$

Generative Learning Algorithms :—

→ First is Gaussian Distribution Analysis (GDA)

Subject: / /

$f(x|y;\theta)$ is distributed according to a multivariate normal distribution.

↳ Input feature is vector.

★ Multivariate Normal Distribution :—

$$x \in \mathbb{R}^d$$

$$E[X] = E\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix} = \begin{bmatrix} E[x_1] \\ E[x_2] \\ \vdots \\ E[x_d] \end{bmatrix} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_d \end{bmatrix} \in \mathbb{R}^d$$

(Mean Vector, μ)

Covariance Matrix :

$$\text{Covariance } \Sigma = E[\underbrace{(X - E[X])}_{\mathbb{R}^{d\times 1}}\underbrace{(X - E[X])^T}_{\mathbb{R}^{1\times d}}]$$

$$f(x;\mu,\Sigma) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left(\frac{-1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)$$

For $d = 1$;

$$\begin{aligned}\Sigma &= E[(x - E[x])(x - E[x])^T] \\ &= E[(x-\mu)(x-\mu)^T] \\ &= E[(x-\mu)(x-\mu)] \\ &= E[(x-\mu)^2] \\ &= \sigma^2\end{aligned}$$

$$f(x;\mu,\Sigma) = \frac{1}{(2\pi)^{1/2}|\Sigma|^{1/2}} \exp\left(\frac{-1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)$$

Subject: / /

Suppose we have a classification problem in which the input features x are continuous valued and output y is discrete $\{0, 1\}$.

$$y \sim \text{Bernoulli}(\phi)$$
$$x \mid y=0 \sim \text{Multivariate Normal Distribution}$$
$$\sim \mathcal{N}(\mu_0, \Sigma)$$

$$x = [x_0 \quad x_1 \quad x_2 \quad \dots \quad x_N]$$

$$P(y) = \phi^y (1-\phi)^{(1-y)}$$

$$P(y=1) = \phi \qquad y \in \{0,1\}$$
$$P(y=0) = 1-\phi$$

$$f(x \mid y=0) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} \exp\left(\frac{-1}{2}(x-\mu_0)^T \Sigma^{-1} (x-\mu_0)\right)$$

$$f(x \mid y=1) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} \exp\left(\frac{-1}{2}(x-\mu_1)^T \Sigma^{-1} (x-\mu_1)\right)$$

Decision Rule;

$$\arg\max_y f(x \mid y; \theta) P(y)$$

$$f(x \mid y=0; \theta) P(y=0)$$
$$f(x \mid y=1; \theta) P(y=1)$$

Parameters in Multivariate Normal distribution

$$f(x \mid y=0; \mu_0, \Sigma) P(y=0)$$
$$f(x \mid y=1; \mu_1, \Sigma) P(y=1)$$

Subject: / /

$$P(y=0) = 1-\phi$$
$$P(y=1) = \phi$$

$$\mu_{0\,MLE} = \frac{\sum_{i=1}^{n} 1\{y^{(i)}=0\} \cdot x^{(i)}}{\sum_{i=1}^{n} 1\{y^{(i)}=0\}}$$

Maximum likelihood Estimator of μ_0

$$\mu_{1\,MLE} = \frac{\sum_{i=1}^{n} 1\{y^{(i)}=1\} \cdot x^{(i)}}{\sum_{i=1}^{n} 1\{y^{(i)}=1\}}$$

$$\phi = P(y=1)$$

$$\phi_{MLE} = \frac{\sum_{i=1}^{n} 1\{y^{(i)}=1\}}{n} = P(y=1)$$

Covariance Matrix $\Sigma = E[(x-\mu)(x-\mu)^T]$

$$= \frac{1}{n} \sum_{i=1}^{n} (x^{(i)} - \mu_{y^{(i)}})(x^{(i)} - \mu_{y^{(i)}})^T$$

$$\mu_{1\,MLE} = \frac{\sum_{i=1}^{n} 1\{y^{(i)}=1\} \cdot x^{(i)}}{\sum_{i=1}^{n} 1\{y^{(i)}=1\}} \quad \text{if } y^{(i)}=1$$

$$\mu_{0\,MLE} = \frac{\sum_{i=1}^{n} 1\{y^{(i)}=0\} \cdot x^{(i)}}{\sum_{i=1}^{n} 1\{y^{(i)}=0\}} \quad \text{if } y^{(i)}=0$$

Q.) Which would perform better out of Gaussian Distribution Analysis (GDA) and Logistic Regression ?

Subject: 22/09/2025

$$\Sigma_{MLE} = \frac{1}{n}\sum_{i=1}^{n}(x^{(i)} - \mu_{y^{(i)}})(x^{(i)} - \mu_{y^{(i)}})^T$$

Relation between Gaussian Discriminant Analysis (GDA) and Logistic Regression

$$f(x|y) \sim \mathcal{N}(\text{Multivariate Gaussian})$$

$$P(y=1 \mid x, \phi, \mu_0, \mu_1, \Sigma) = \frac{f(x; \mu_1, \Sigma)\, P(y=1|x,\phi)}{f(x)}$$

$$f(x) = P(y=1|x;\phi)\, f(x, \mu_1, \Sigma) + P(y=0|x;\phi) f(x, \mu_0, \Sigma)$$

$$= \frac{P(y=1|x;\phi) f(x;\mu_1,\Sigma)}{P(y=1|x;\phi) f(x,\mu_1,\Sigma) + P(y=0|x;\phi) + f(x,\mu_0,\Sigma)}$$

$$= \frac{\phi\, \mathcal{N}(\mu_1,\Sigma)}{\phi\, \mathcal{N}(\mu_1,\Sigma) + (1-\phi)\mathcal{N}(\mu_0,\Sigma)}$$

$$= \frac{1}{1 + \frac{(1-\phi)}{\phi}\frac{\mathcal{N}(\mu_0,\Sigma)}{\mathcal{N}(\mu_1,\Sigma)}}$$

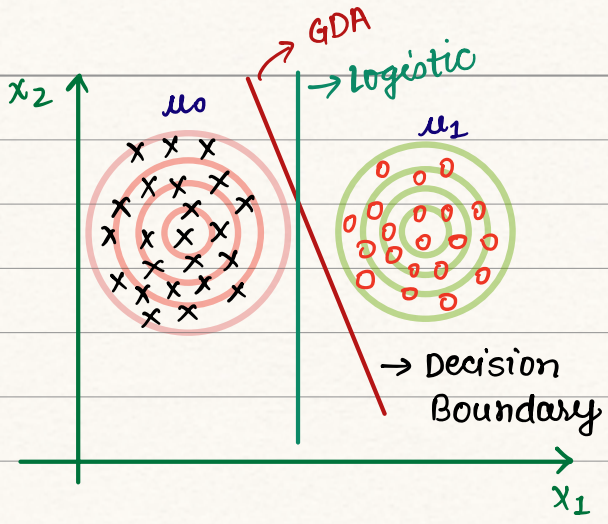
$$\mathcal{N}(\mu_0,\Sigma) = \frac{1}{2\pi^{d/2}|\Sigma|^{1/2}} \exp\left(\frac{-1}{2}(x-\mu_0)^T \Sigma^{-1}(x-\mu_0)\right)$$

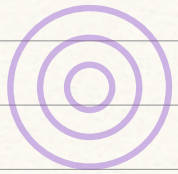
$$\mathcal{N}(\mu_1,\Sigma) = \frac{1}{2\pi^{d/2}|\Sigma|^{1/2}} \exp\left(\frac{-1}{2}(x-\mu_1)^T \Sigma^{-1}(x-\mu_1)\right)$$

When we assume x to be multivariate Gaussian, then

$$P(y=1|x, \mu_0, \mu_1, \phi, \Sigma) = \text{sigmoid}$$

Subject: / /





3D Gaussian Curve

$x_1^{(1)}$	$x_2^{(1)}$	$y = 0$
$x_1^{(2)}$	$x_2^{(2)}$	$y = 1$
$x_1^{(3)}$	$x_2^{(3)}$	$y = 0$

Q.) When would we prefer GDA or logistic regression?

$f(x|y) \sim$ Multivariate Gaussian

$P(y=1|x)$ = sigmoid function

But converse is not true.

$P(y=1|x)$ → sigmoid function does not necessarily imply $P(x|y)$ is multivariate gaussian.

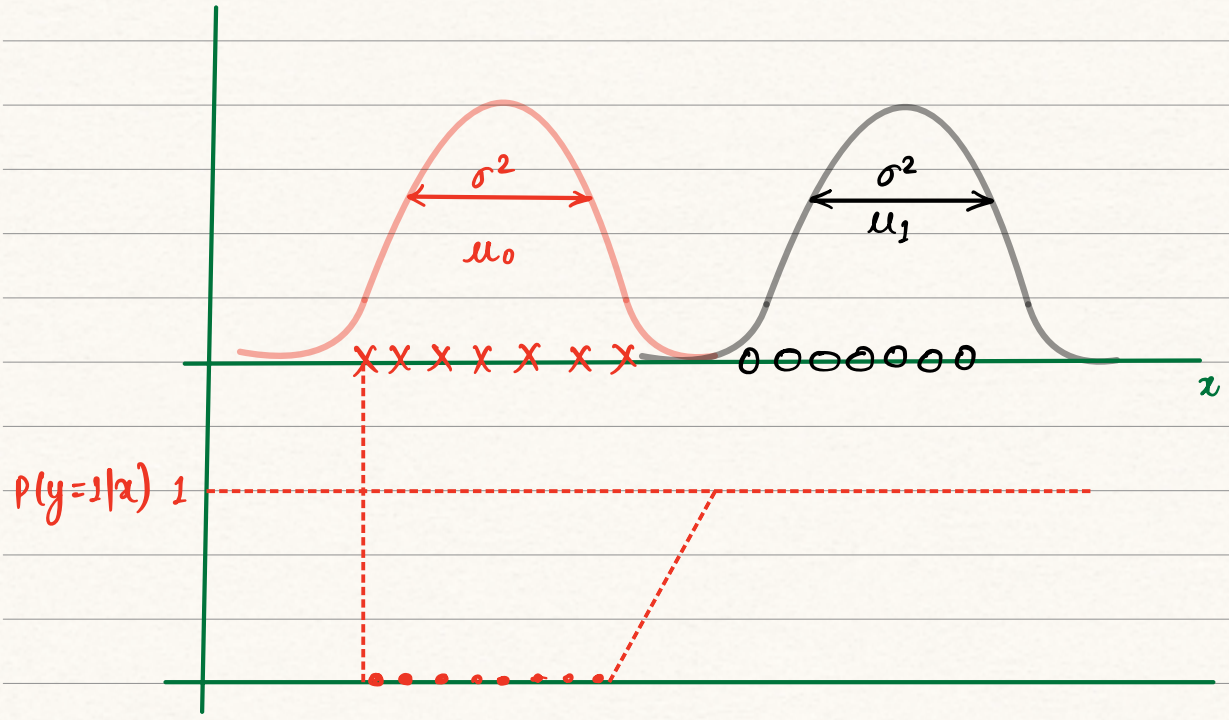
⇒ GDA has stronger modelling assumptions as compared to logistic regression.

Data is Gaussian

If the Gaussian modelling assumptions are correct, then GDA will find better performance as compared to logistic regression. Else GDA would perform poor.

$P(y=1|x)$ = Sigmoid

Subject: 24/09/2025



✯ Kernal Method

$$y = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3$$

$$\phi : \mathbb{R} \rightarrow \mathbb{R}^4$$

$$\phi(x) = \begin{bmatrix} 1 \\ x \\ x^2 \\ x^3 \end{bmatrix} \qquad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix}$$

$$y = \theta^T \phi(x)$$

↳ Complexity increased

that's why we are using kernel

In the context of kernels, the original input 'x' is called Input attribute.

$\phi(x)$ → Feature map which maps the input attribute (x) to features.

Subject: / /

update rule with $\phi(x)$;

$$h(x^{(i)}) = \theta^T \phi(x^{(i)})$$

$$\theta_j := \theta_j + \frac{\alpha}{n} \sum_{i=1}^{n} \left(y^{(i)} - h(x^{(i)}) \right) . \phi(x_j^{(i)})$$

Generalized ;

$$\phi : \mathbb{R}^d \to \mathbb{R}^p$$

ϕ takes the input vector x in $\mathbb{R}^d$.

$$x = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ \vdots \\ x_{d-1} \end{bmatrix} \quad (\mathbb{R}^d) \qquad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \\ \theta_{p-1} \end{bmatrix} \quad (\mathbb{R}^p)$$

Suppose that $d = 10^3$ then $p = d^3 = 10^9$. Since 10^9 is huge, as it will take lot of time to update 10^9 parameters. Instead of 10^3, the algorithm is now more complex.

★ Least Mean Squared (LMS) update Rule with kernels :—

Q.) How will we update θ with kernels ?

Our claim: At any time, θ can be expressed as a linear combination of $\phi(x^{(1)})$, $\phi(x^{(2)})$, ... $\phi(x^{(n)})$

$$\theta = a_1 \phi(x^{(1)}) + a_2 \phi(x^{(2)}) + \ldots + a_n \phi(x^{(n)})$$

$$\theta := \theta + \alpha \sum_{i=1}^{n} \left(y^{(i)} - \theta^T \phi(x^{(i)}) \right) . \phi(x^{(i)})$$

Subject: 06/10/2025

Suppose we initially set $\theta = 0$ (zero vector).

$$\theta := 0 + \alpha \sum_{i=1}^{n} (y^{(i)} - 0).\phi(x^{(i)})$$

New updated weight, $\theta = \sum_{i=1}^{n} \alpha . y^{(i)} . \phi(x^{(i)})$ } 2nd Iteration

$$\theta = \sum_{i=1}^{n} \beta^{(i)} . \phi(x^{(i)})$$

where $\beta^{(i)} = \alpha . y^{(i)}$

$$\theta := \theta + \alpha \sum_{i=1}^{n} (y^{(i)} - \theta^T \phi(x^{(i)})).\phi(x^{(i)})$$

$$\theta = \sum_{i=1}^{n} \beta^{(i)}.\phi(x^{(i)}) + \alpha \sum_{i=1}^{n} \left[y^{(i)} - \left(\sum_{j=1}^{n} \beta_j . \phi(x^{(j)}) \right)^T . \phi(x^{(i)} \right] . \phi(x^{(i)})$$

$$= \sum_{i=1}^{n} \beta_i . \phi(x^{(i)}) + \alpha \sum_{i=1}^{n} \left(y^{(i)} - \sum_{j=1}^{n} \beta_j \phi(x^{(j)})^T \phi(x^{(i)}) \right) . \phi(x^{(i)})$$

$$= \sum_{i=1}^{n} \left(\beta_i + \alpha \left(y^{(i)} - \sum_{j=1}^{n} \beta_j . \phi(x^{(j)})^T \phi(x^{(i)}) \right) \right) . \phi(x^{(i)})$$

$$\beta_i^{(new)} := \beta_i + \alpha \left(y^{(i)} - \underbrace{\sum_{j=1}^{n} \beta_j . \phi(x^{(j)})^T . \phi(x^{(i)})}_{O(np)} \right)$$

Complexity does not improved

Example :

$x \in \mathbb{R}^d$

$\phi(x)$ which contains the monomials of x with degree ≤ 3.

Subject: / /

$$\underbrace{\langle \phi(x), \phi(z) \rangle}_{\text{Dot product}} = 1 + \sum_{i=1}^{d} x_i z_i + \sum_{i,j} x_i x_j z_i z_j + \sum_{i,j,k} x_i \cdot x_j \cdot x_k \, z_i \cdot z_j \cdot z_k$$

$i, j \in [1, \ldots, d]$

$i, j, k \in [1, \ldots, d]$

$$= \underbrace{1 + \langle x, z \rangle + (\langle x, z \rangle)^2 + (\langle x, z \rangle)^3}$$

$$= K(x, z) \qquad O(d) \qquad \boxed{d << p}$$

This is Kernel.

$$\langle \phi(x), \phi(z) \rangle = K(x, z) = \text{Kernel}$$

$$\beta_i^{new} := \beta_i + \alpha \left(y^{(i)} - \sum_{j=1}^{n} \beta_j \, \phi(x^{(j)})^T \phi(x^{(i)}) \right)$$

This is fixed for each training example.

$$= \beta_i + \alpha \left(y^{(i)} - \sum_{j=1}^{n} \beta_j \cdot \overbrace{K(x^{(j)} \cdot x^{(i)})}^{O(d)} \right)$$

$O(nd)$

$$K(x^{(j)} \cdot x^{(i)}) = \langle \phi(x^{(j)}), \phi(x^{(i)}) \rangle$$

$x^{(j)} = \phi(x^{(j)})$

$x^{(i)} = \phi(x^{(i)})$

$i, j = 1, 2, \ldots, n$

As this is fixed for each training example we will store its value separately so that algorithm becomes much more faster.

Let suppose

Subject: 07/10/2025

Q) What kind of function $K(.)$ can corresponds to some feature map, ϕ ? In other words, can we tell if there is some feature mapping ϕ so that $K(x, z) = \phi(x)^T$

Example :—

$$x, z \in \mathbb{R}^d$$

$$K(x, z) = (x^T z)^2 \rightarrow \text{polynomial kernel of degree 2}$$

$$x = [x_1 \quad x_2 \quad \dots \quad x_d]$$

$$z = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_d \end{bmatrix}$$

$$K(x, z) = (x^T z)^2$$

$$= (x^T z)(x^T z)$$

$$= \left(\sum_{i=1}^{d} x_i z_i \right) \left(\sum_{j=1}^{d} x_j z_j \right)$$

$$= \sum_{i=1}^{d} \sum_{j=1}^{d} x_i x_j z_i z_j$$

$$= \langle \phi(x), \phi(z) \rangle$$

$$\phi(x) = \begin{bmatrix} x_1 x_1 \\ x_1 x_2 \\ x_2 x_1 \\ x_2 x_2 \end{bmatrix} \qquad \phi(z) = \begin{bmatrix} z_1 z_1 \\ z_1 z_2 \\ z_2 z_1 \\ z_2 z_2 \end{bmatrix}$$

Subject: / /

Suppose $d = 3$;

$$\phi(x) = \begin{bmatrix} x_1 x_1 \\ x_1 x_2 \\ x_1 x_3 \\ x_2 x_1 \\ x_2 x_2 \\ x_2 x_3 \\ x_3 x_1 \\ x_3 x_2 \\ x_3 x_3 \end{bmatrix} \qquad \phi(z) = \begin{bmatrix} z_1 z_1 \\ z_1 z_2 \\ z_1 z_3 \\ z_2 z_1 \\ z_2 z_2 \\ z_2 z_3 \\ z_3 z_1 \\ z_3 z_2 \\ z_3 z_3 \end{bmatrix}$$

$$K(x, z) = (x^T z + c)^2$$

↳ Polynomial kernel of degree 2

$$= \sum_{i,j=1}^{d} (x_i x_j).(z_i z_j) + \sum_{i=1}^{d} (\sqrt{2c}\, x_i)(\sqrt{2c}\, z_i) + c^2$$

$$= \langle \phi(x), \phi(z) \rangle$$

$d = 3$

$$\phi(x) = \begin{bmatrix} x_1 x_1 \\ x_1 x_2 \\ x_1 x_3 \\ x_2 x_1 \\ \vdots \\ x_3 x_3 \\ \sqrt{2c}.x_1 \\ \sqrt{2c}.x_2 \\ \sqrt{2c}.x_3 \\ c \end{bmatrix} \qquad \phi(z) = \begin{bmatrix} z_1 z_1 \\ z_1 z_2 \\ z_1 z_3 \\ z_2 z_1 \\ \vdots \\ z_3 z_3 \\ \sqrt{2c}.z_1 \\ \sqrt{2c}.z_2 \\ \sqrt{2c}.z_3 \\ c \end{bmatrix}$$

Subject: / /

✯ Necessary condition for a valid kernel —

Actual Inputs $= [x^{(1)}, x^{(2)}, \dots x^{(n)}]$

$K_{ij} = K(x^{(i)}, x^{(j)})$

$i, j \in [1, 2, \dots n]$

K = Kernel matrix $= \begin{bmatrix} K_{11} & K_{12} & \dots & K_{1n} \\ K_{21} & K_{22} & \dots & K_{2n} \\ \vdots & \vdots & \dots & \vdots \\ K_{n1} & K_{n2} & \dots & K_{nn} \end{bmatrix}_{n \times n}$

For a kernel $K(\cdot)$ to be valid,

1.) $K_{ij} = K_{ji}$ → Symmetric matrix

2.) For any $z \in \mathbb{R}^n$,

$z^T K z \geq 0$

K → Positive Semi Definite

✯ Gaussian Kernel or Radial basis function (RBF) Kernel :—

$x, z \in \mathbb{R}^d$

$K(x, z) = \langle \phi(x), \phi(z) \rangle$

Very large when x and z are very close to each other

Very small when x and z are far apart or orthogal to each other

$$K(x, z) = \exp\left(\frac{-\|x - z\|^2}{2}\right) = \langle \phi(x), \phi(z) \rangle$$

Subject: / /

* It has infinite dimensions feature map.

Proof : Gaussian Kernel : $K(u, v) = \exp\left(-\frac{\|u-v\|^2}{2}\right)$

$$\begin{aligned}\|u-v\|^2 &= (u-v)^T(u-v) \\ &= (u^T - v^T)(u-v) \\ &= u^Tu - v^Tu - v^Tu + v^Tv \\ &= \|u\|^2 - 2v^Tu + \|v\|^2\end{aligned}$$

$$\exp\left(-\frac{\|u-v\|^2}{2}\right) = \exp\left(-\frac{1}{2}\left(\|u\|^2 - 2u^Tv + \|v\|^2\right)\right)$$

$$= \exp\left(-\frac{\|u\|^2}{2}\right)\exp\left(-\frac{\|v\|^2}{2}\right)\exp\left(\langle u, v\rangle\right)$$

$$\exp(x) = \sum_{k=0}^{\infty}\frac{x^k}{k!}$$

$$= \exp\left(-\frac{\|u\|^2}{2}\right)\exp\left(-\frac{\|v\|^2}{2}\right)\sum_{k=0}^{\infty}\left(\frac{\langle u, v\rangle}{k!}\right)^k$$

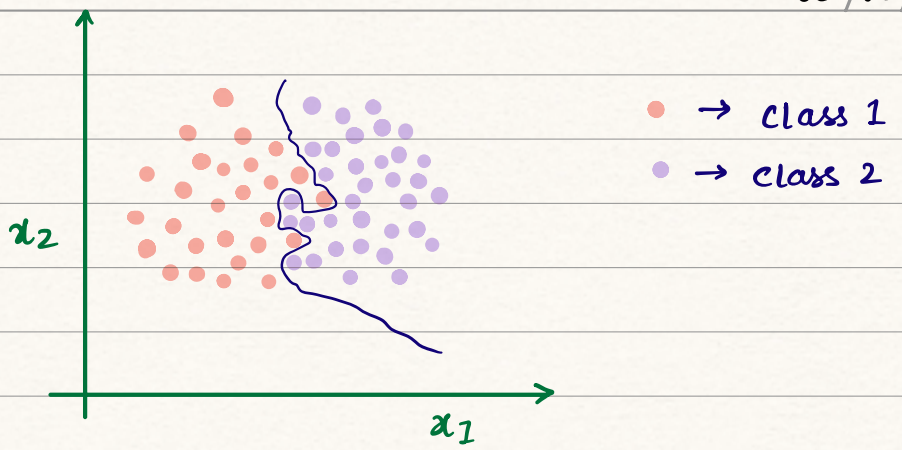
★ Neural Network ——

Q.) Why do we need Neural Network when we already have the algorithms such as Linear Regression and Logistic Regression ?

Suppose we have a training set corresponding to Logistic Regression when we need to learn complex non-linear decision-boundary.

$$\text{Logistic} : h(x) = \frac{1}{1+e^{-\theta^Tx}}$$

Subject: 08/10/2025



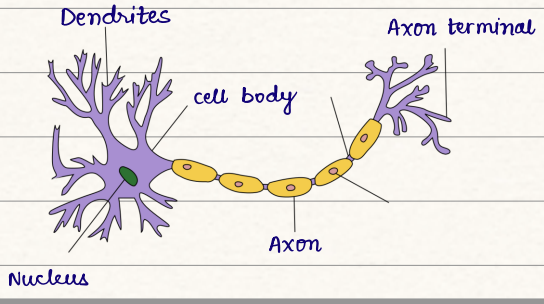
In real-life, there are more than 30 input features and to train on the dataset, the complexity will increase exponentially. So that is why we use neural networks to solve this problem.

* Simple Linear/Logistic Regression together with a lot of quadratic or cubic features is not a good way to learn the complex non-linear decision boundary as it will lead to huge complexity.

* We should consider the neural networks which turn out to be a much better way to learn complex non-linear decision boundaries with less complexity even when the number of inputs are very large.

Neural Networks :— Algorithm that try to mimic the human brain

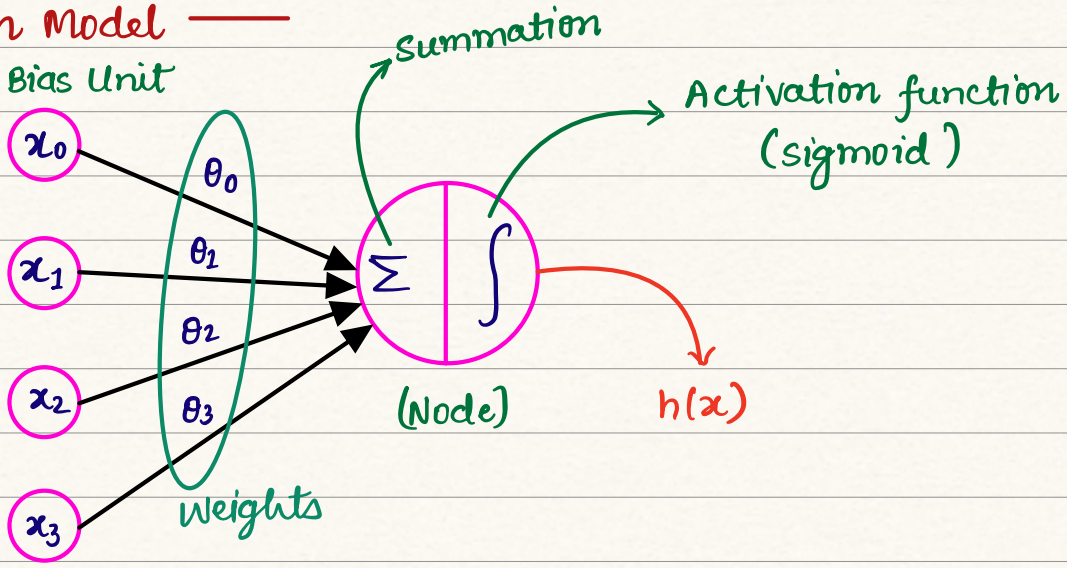
Biological Neuron ——



Subject: / /

Biological Neuron	Artificial Neuron
Dendrites	Inputs
Cell Nucleus	Node
Synapsis	Weights
Axon	Outputs

Neuron Model

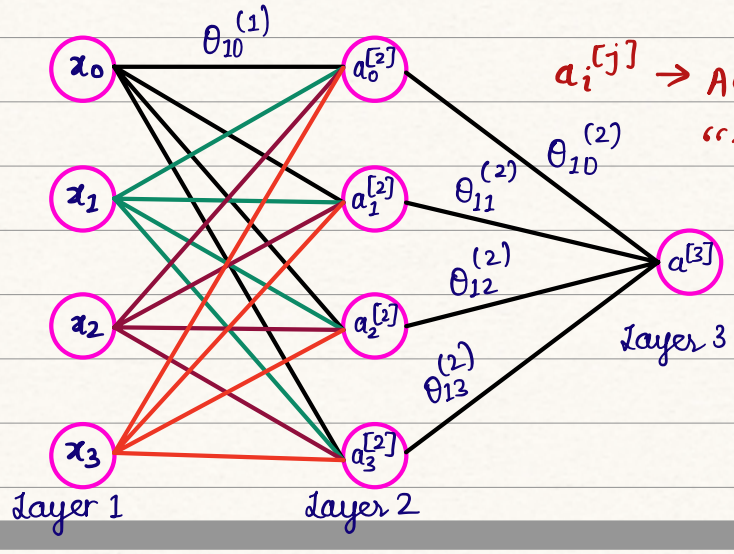


$$\Sigma = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3$$

$$\int = \frac{1}{e^{-\theta^T x}} = \frac{1}{e^{-(\theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3)}}$$

Neural Network

It is the group of the different neurons stuck together.



$a_i^{[j]}$ → Activation of neuron "i" in layer "j".

Subject: / /

Feed Forward Propagation

Layer 1 :- Input Layer

Layer 2 :- Hidden Layer

Layer 3 :- Output Layer

Backward Propagation

$\theta^{(j)}$ Matrix/Vector of weights controlling function mapping from layer 'j' to layer 'j+1'.

$$a_i^{[j]} = \Sigma \int$$

$$a_1^{[2]} = g\left(\theta_{10}^{(i)} x_0 + \theta_{11}^{(i)} x_1 + \theta_{12}^{(i)} x_2 + \theta_{13}^{(i)} x_3\right)$$

$g \rightarrow$ sigmoid activation function

$$a_1^{[2]} = g\left(\theta_{10}^{(1)} x_0 + \theta_{11}^{(1)} x_1 + \theta_{12}^{(1)} x_2 + \theta_{13}^{(1)} x_3\right)$$

$$a_2^{[2]} = g\left(\theta_{20}^{(1)} x_0 + \theta_{21}^{(1)} x_1 + \theta_{22}^{(1)} x_2 + \theta_{23}^{(1)} x_3\right)$$

$$a_3^{[2]} = g\left(\theta_{30}^{(1)} x_0 + \theta_{31}^{(1)} x_1 + \theta_{32}^{(1)} x_2 + \theta_{33}^{(1)} x_3\right)$$

$a^{[2]} = g(z^{[2]})$ } Vector

$$\theta^{(1)} = \begin{bmatrix} \theta_{10}^{(1)} & \theta_{11}^{(1)} & \theta_{12}^{(1)} & \theta_{13}^{(1)} \\ \theta_{20}^{(1)} & \theta_{21}^{(1)} & \theta_{22}^{(1)} & \theta_{23}^{(1)} \\ \theta_{30}^{(1)} & \theta_{31}^{(1)} & \theta_{32}^{(1)} & \theta_{33}^{(1)} \end{bmatrix} \qquad x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

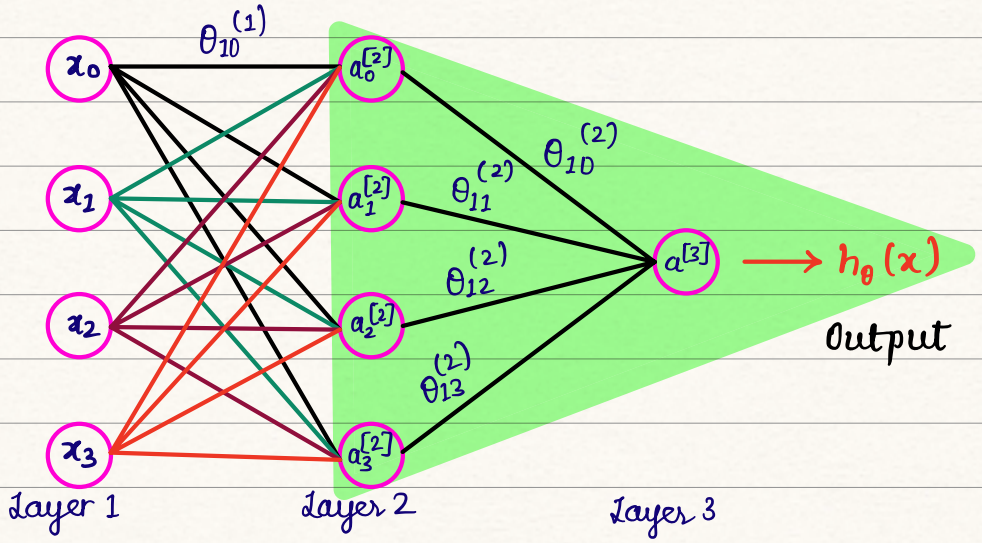
$$\underset{(3\times4)}{\theta^{(1)}} \quad \underset{(4\times1)}{x} = \underset{(3\times1)}{z^{[2]}}$$

$\theta^{(1)} \in \mathbb{R}^{3\times4}$

Subject: 13/10/2025

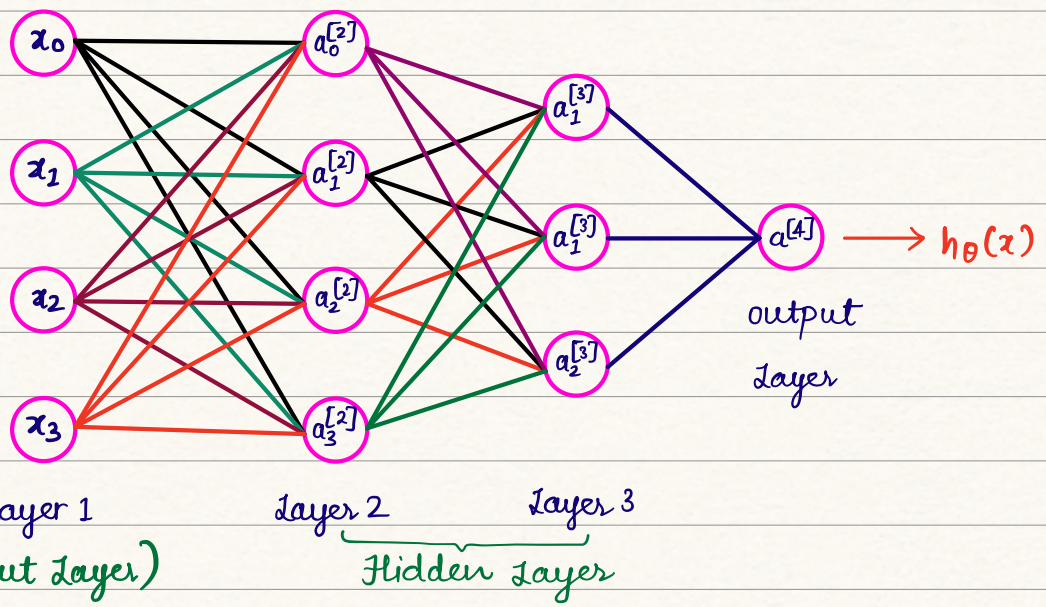
$$a^{[3]} = g\left(a_0^{[2]}\theta_{10}^{(2)} + a_1^{[2]}\theta_{11}^{(2)} + a_2^{[2]}\theta_{12}^{(2)} + a_3^{[2]}\theta_{13}^{(2)}\right)$$

★ Neural Network learning its own features ——

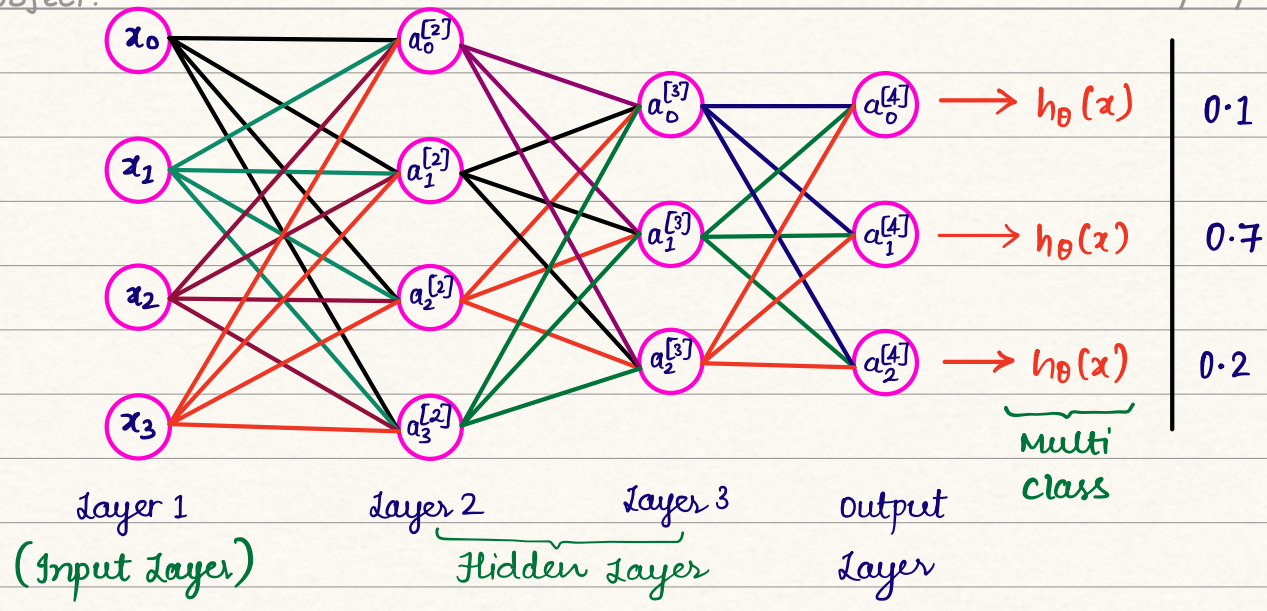


$$h_\theta(x) = g\left(\theta_{10}^{(2)}a_0^{[2]} + \theta_{11}^{(2)}a_1^{[2]} + \theta_{12}^{(2)}a_2^{[2]} + \theta_{13}^{(2)}a_3^{[2]}\right)$$

* In this Neural network Architecture, it gets to learn its own features $a_1^{[2]}$, $a_2^{[2]}$, $a_3^{[2]}$ to feed into the logistic regression. With this, with the help of Neural Networks, we can easily learn complex non-linear decision boundaries.



Subject: / /



Output Layer : $a^{[j]}$ → Activation function

House Price

$h_\theta(x) = \theta^T x$

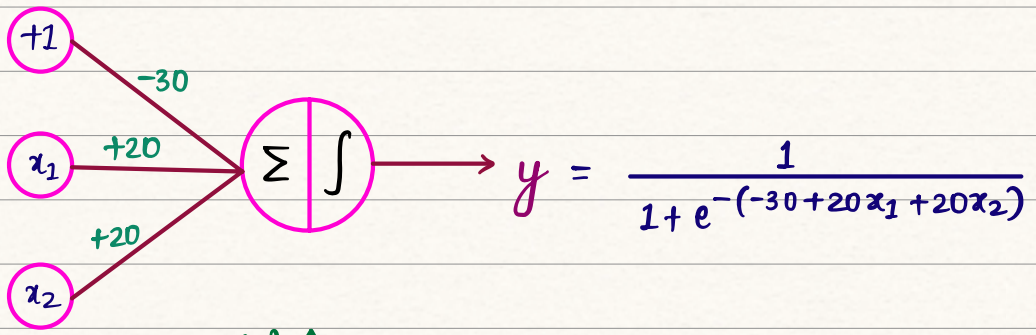
Cancer Prediction

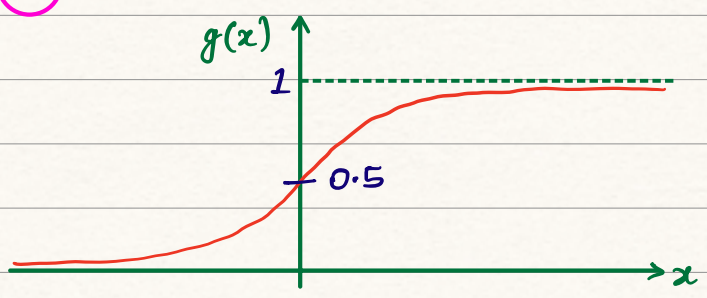
$h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$

☆ Logic Gates using Neural Networks ——

$x_1, x_2 \in \{0, 1\}$ → Binary Inputs

$y = x_1 x_2$ = AND operation of x_1 and x_2





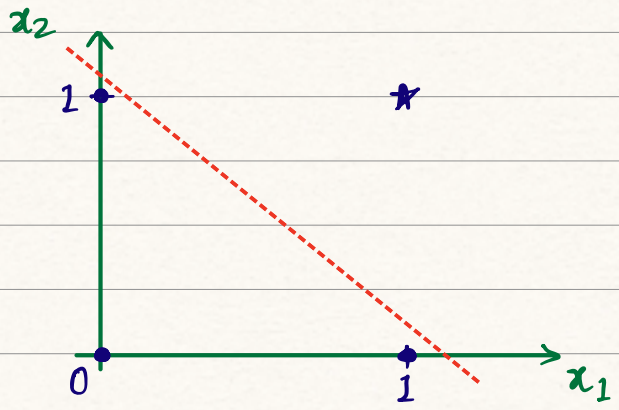
Subject: / /

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

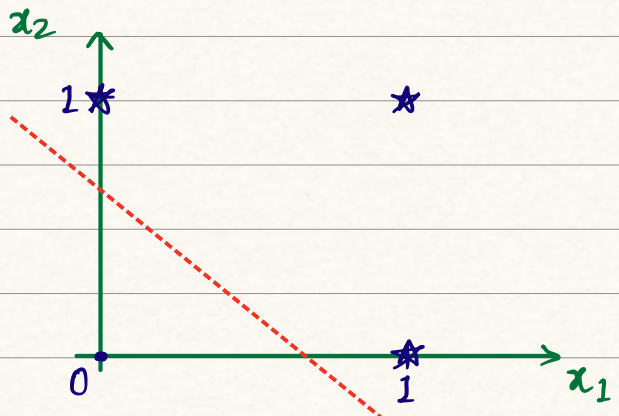
AND Gate

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

OR Gate



AND Gate

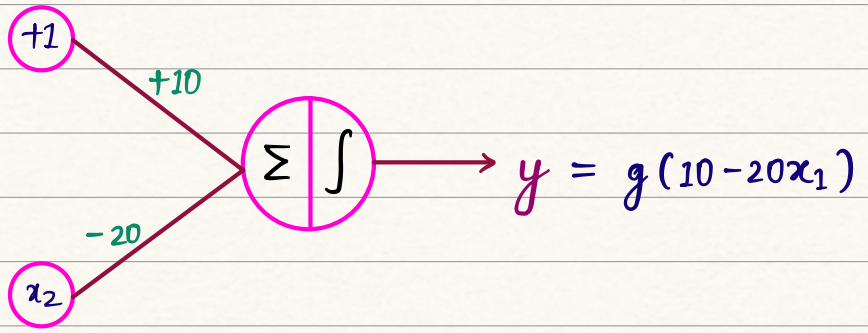


● → 0

★ → 1

OR Gate

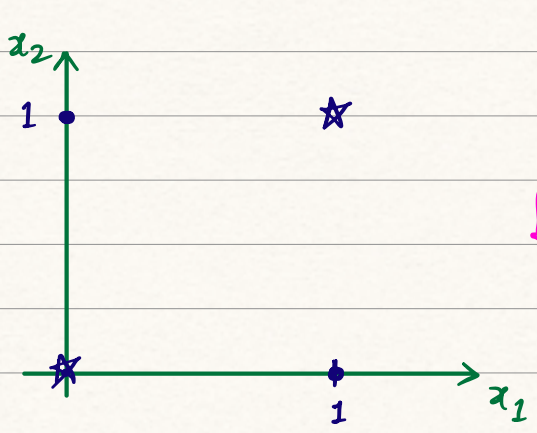
NOT Gate :



Subject: / /

★ Non-Linear classification example —

$$\text{XNOR} = x_1 x_2 + \bar{x}_1 \bar{x}_2$$

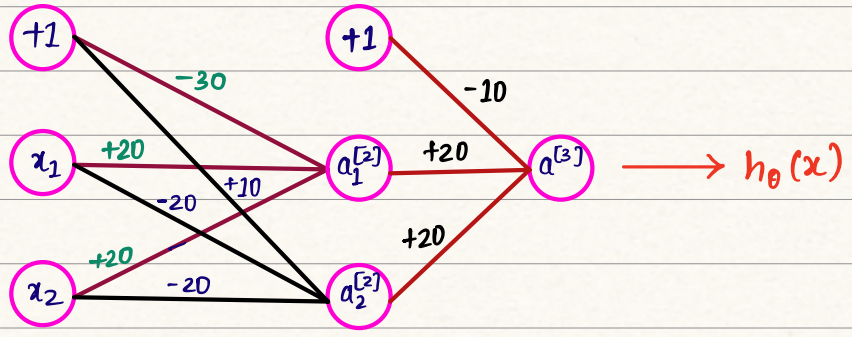


● → 0

★ → 1

{ Linear Decision boundary won't work

$x_1 x_2 + \bar{x}_1 \bar{x}_2$

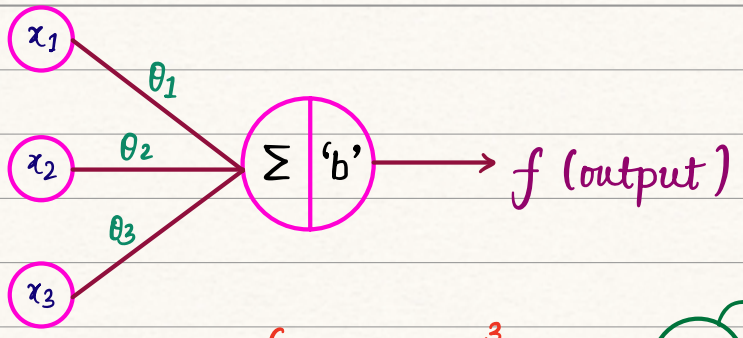


x_1	x_2	$a_1^{[2]}$	$a_2^{[2]}$	$h_\theta(x)$
0	0	0	1	1
0	1	0	0	0
1	0	0	0	0
1	1	1	0	1

★ Perceptron ——

Take several inputs and gives a single output (binary).

Subject: 14/10/2025



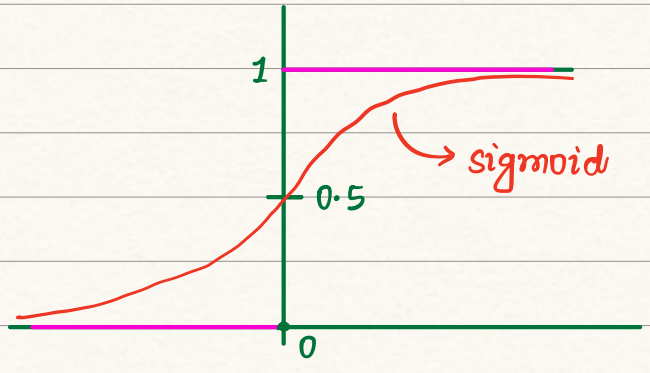
Threshold

$$f = \begin{cases} 0 & \text{if } \sum_{i=1}^{3} x_i \theta_i \leq -b \\ 1 & \text{if } \sum_{i=1}^{3} x_i \theta_i > -b \end{cases}$$

$$f = \begin{cases} 0 & \text{if } \theta^T x + b \leq 0 \\ 1 & \text{if } \theta^T x + b > 0 \end{cases}$$

Let $z = \theta^T x + b$

$$\text{Activation, } a = f(z) = \begin{cases} 0 & \text{if } z \leq 0 \\ 1 & \text{if } z > 0 \end{cases}$$



* Perceptron Example —

Q.) Will you go to the picnic ?

x_1 : Is the weather good ?

x_2 : Do you have any company with friends ?

x_3 : Is there any good garden nearby ?

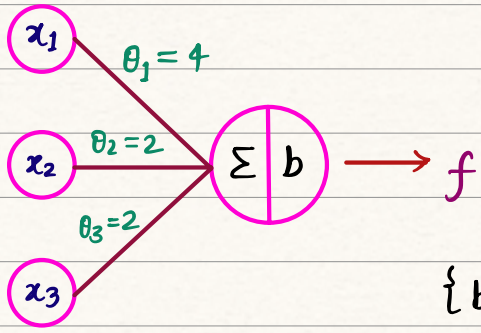
Subject: / /

We decide that we will go to picnic when the weather is good and if any one of the rest of the factors is in my favour.

"Weather"

"Good company"

"Good Garden nearby"



$\{b = -5\}$

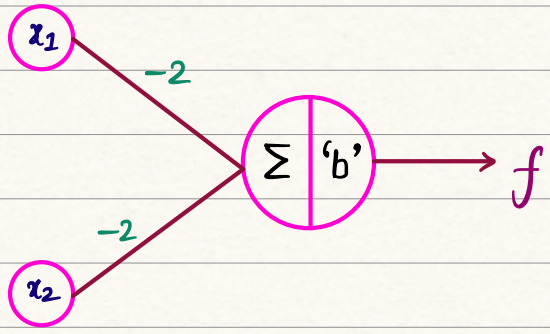
$$f = 4x_1 + 2x_2 + 2x_3 - 5$$

Suppose that $x_1 = 0$ (weather is bad)

$$x_1 = 0 \;,\; x_2 = 1 \;,\; x_3 = 1$$

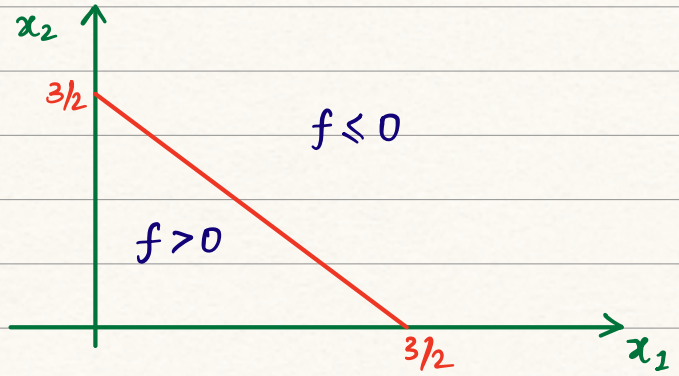
$$f = -1 < 0 \longrightarrow 0$$

☆ Perceptron also work on linear classifier ——



$\{b = 3\}$

$$f = -2x_1 - 2x_2 + 3$$

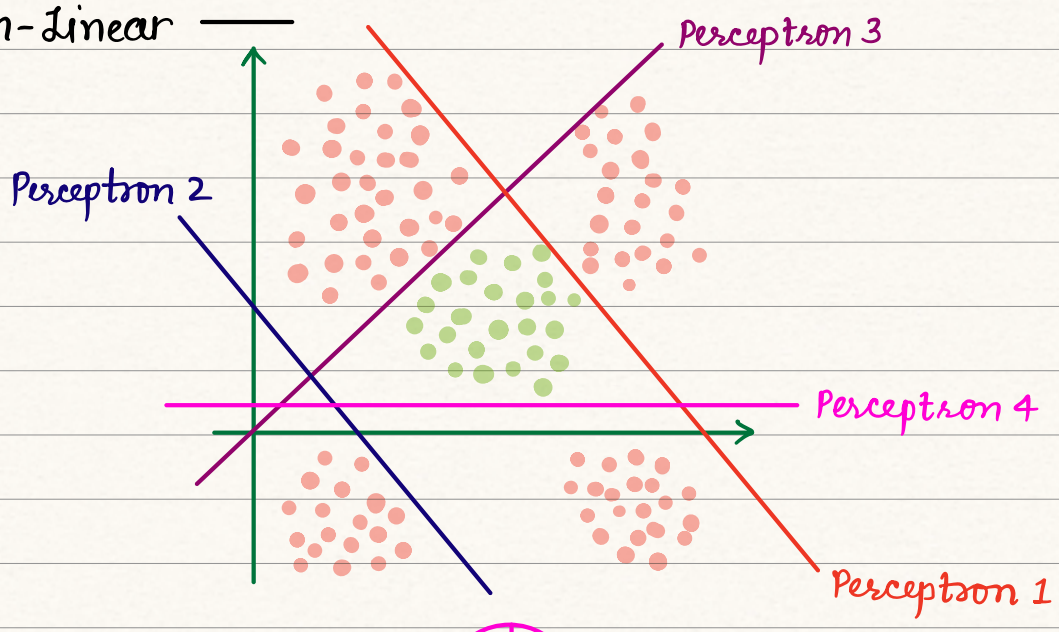


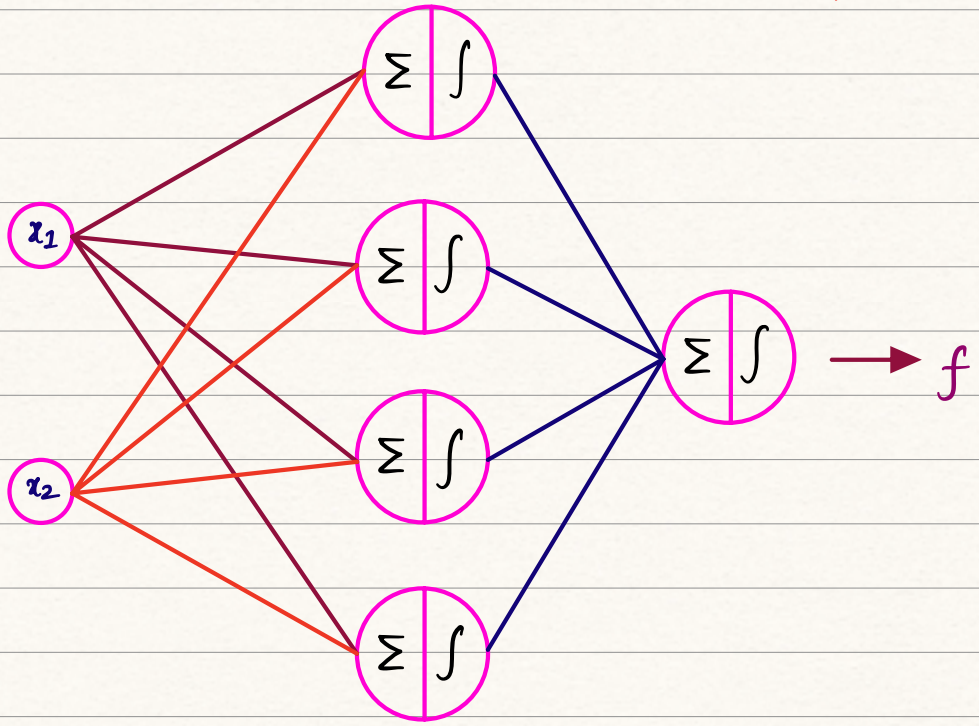
Subject: / /

x_1	x_2	f	a(f)
0	0	3	1
0	1	1	1
1	0	1	1
1	1	-1	0

NAND Gate

* Non-Linear ——





* Drawbacks of perceptron ——

(i) Step function

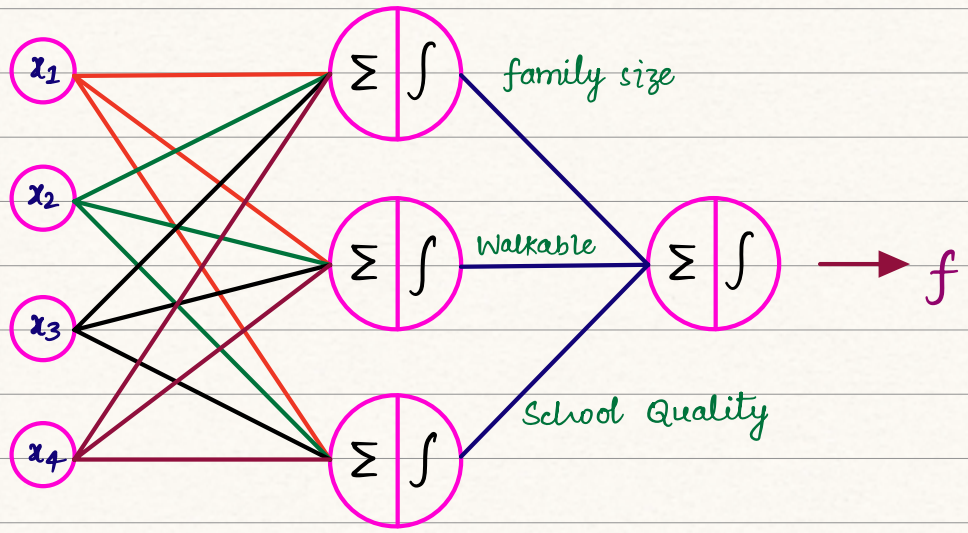
(ii) Not differentiable

Subject: / /

Housing Price Prediction Problem :—

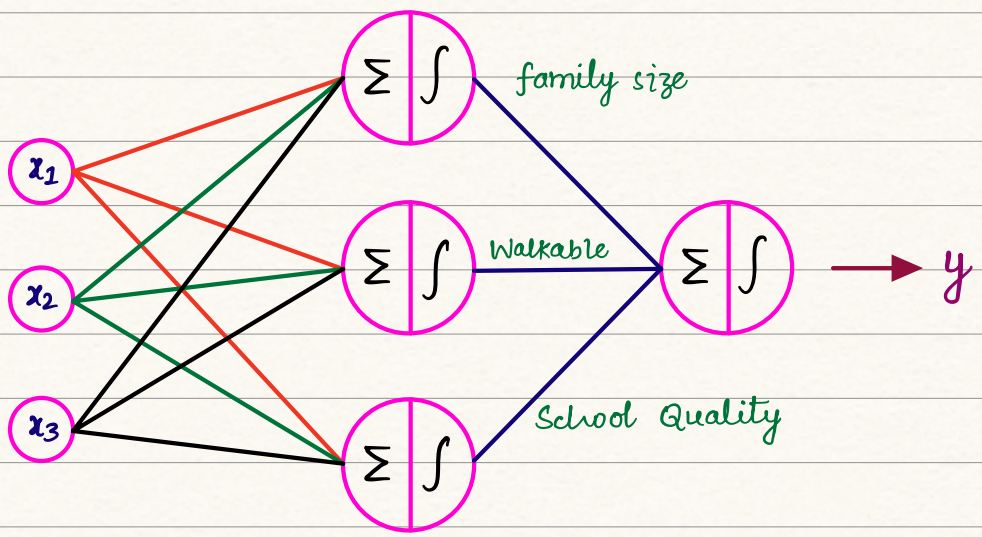
Suppose the inputs :

- x_1 : size of house
- x_2 : number of bedrooms
- x_3 : zip code
- x_4 : wealth



Fully Connected Neural Network

Q.) Why we connect each input to each neuron ?



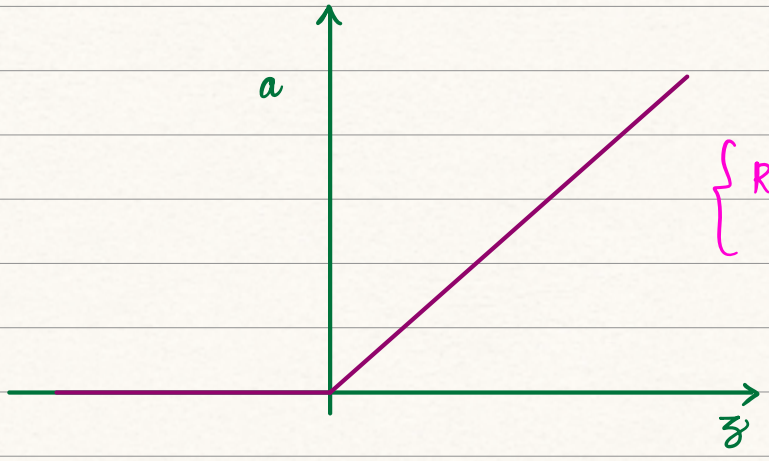
$0 \leq y \leq 1$ → Sigmoid Activation Function

Subject: / /

$-1 \leq y \leq 1$ → tanh Activation Function

Q.) What is problem associated with sigmoid function for higher value of 'z'?

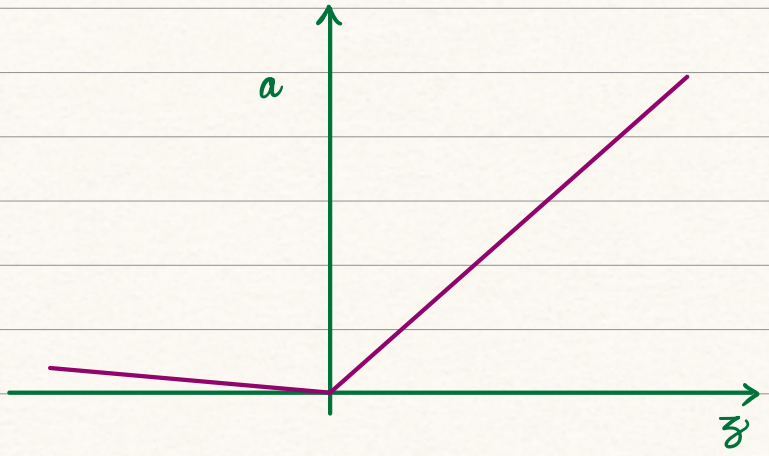
Solution to this problem : ReLU Activation Function



{ ReLU – Rectified Linear Unit }

$a = \max(z, 0)$

Leaky ReLU :—



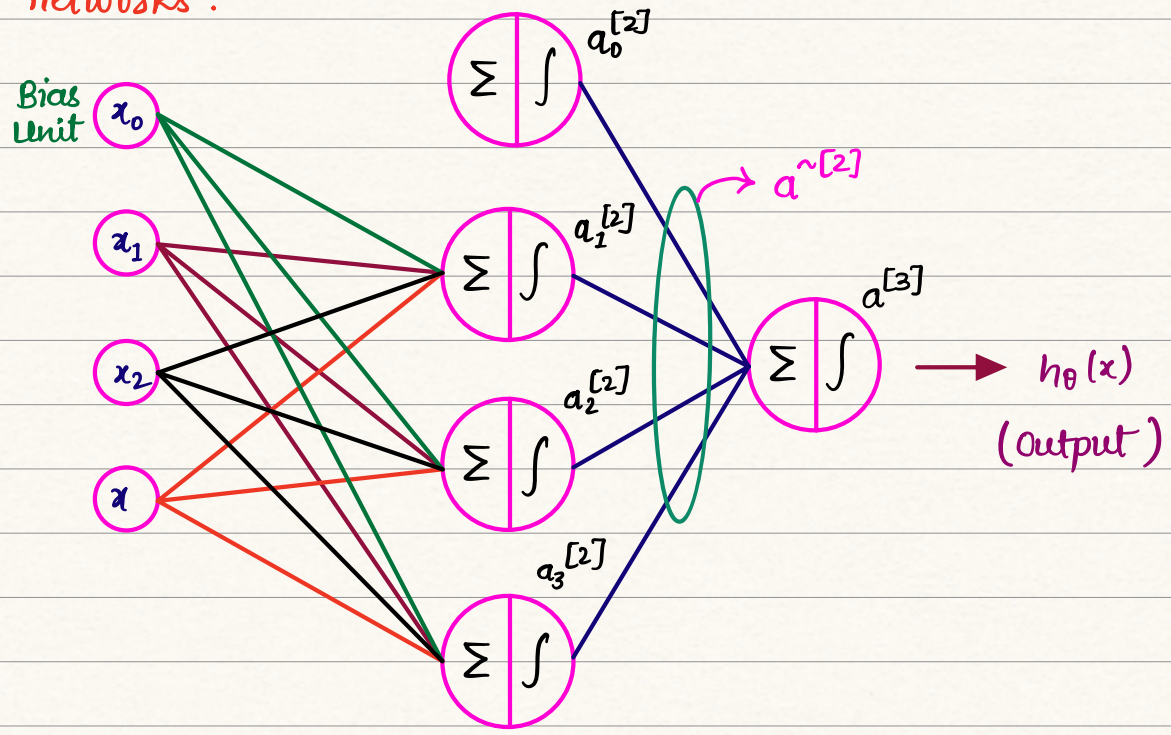
$a = \max(0.01z, z)$

$g \rightarrow$ Identity function

$f(a) = a$

Subject: 15/10/2025

Q.) Why we need non-linear activation functions for neural networks ?



Non-linear Activation Functions :—

(i) Sigmoid
(ii) tanh
(iii) ReLU
(iv) Leaky ReLU

$$z^{[2]} = \theta^{(1)} \underline{x} \quad \in \mathbb{R}^{3\times 1}$$

Matrix of weights while going from layer 1 to layer 2.

$x \in \mathbb{R}^{4\times 1}$

$\theta \in \mathbb{R}^{3\times 4}$

$a^{[2]} = g(z^{[2]})$

$$a^{\sim[2]} = \begin{bmatrix} a_0^{[2]} \\ \underline{a}^{[2]} \end{bmatrix}_{4\times 1}$$

$$a^{\sim[2]} = \begin{bmatrix} 1 \\ a_1^{[2]} \\ a_2^{[2]} \\ a_3^{[2]} \end{bmatrix} \rightarrow \underline{a}^{[2]}$$

$$z^{[3]} = \theta^{(2)} a^{\sim[2]}$$
$$(1\times 4)\ (4\times 1)$$

Subject: / /

$$a^{[3]} = g(z^{[3]})$$

Instead of any Non-linear Activation Function ;

$g(z) = z$ {Identity function}

$$g(z) = \frac{1}{1+e^{-z}}$$

$$z^{[2]} = \theta^{(1)} x$$

$$a^{[2]} = g(z^{[2]}) = z^{[2]} = \theta^{(1)} x$$

$$z^{[3]} = \theta^{(2)} a^{[2]}$$

$$a^{[3]} = g(z^{[3]}) = z^{[3]} = \theta^{(2)} a^{[2]}$$

$$= \theta^{(2)} \begin{bmatrix} a_0^{[2]} \\ a^{[2]} \end{bmatrix}$$

$$\begin{cases} a_0^{[2]} = 1 \\ a^{[2]} = z^{[2]} = \theta^{(1)} x \end{cases}$$

$$= \theta^{(2)} \theta^{(1)} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

$$a^{[2]} = \begin{bmatrix} a_1^{[2]} \\ a_2^{[2]} \\ a_3^{[2]} \end{bmatrix}_{3\times 1}$$

$$\theta^{(1)} = \begin{bmatrix} \theta_{10}^{(1)} & \theta_{11}^{(1)} & \theta_{12}^{(1)} & \theta_{13}^{(1)} \\ \theta_{20}^{(1)} & \theta_{21}^{(1)} & \theta_{22}^{(1)} & \theta_{23}^{(1)} \\ \theta_{30}^{(1)} & \theta_{31}^{(1)} & \theta_{32}^{(1)} & \theta_{33}^{(1)} \end{bmatrix}_{3\times 4} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}_{4\times 1}$$

$$\theta^{(2)} = \begin{bmatrix} \theta_{10}^{(2)} & \theta_{11}^{(2)} & \theta_{12}^{(2)} & \theta_{13}^{(2)} \end{bmatrix}$$

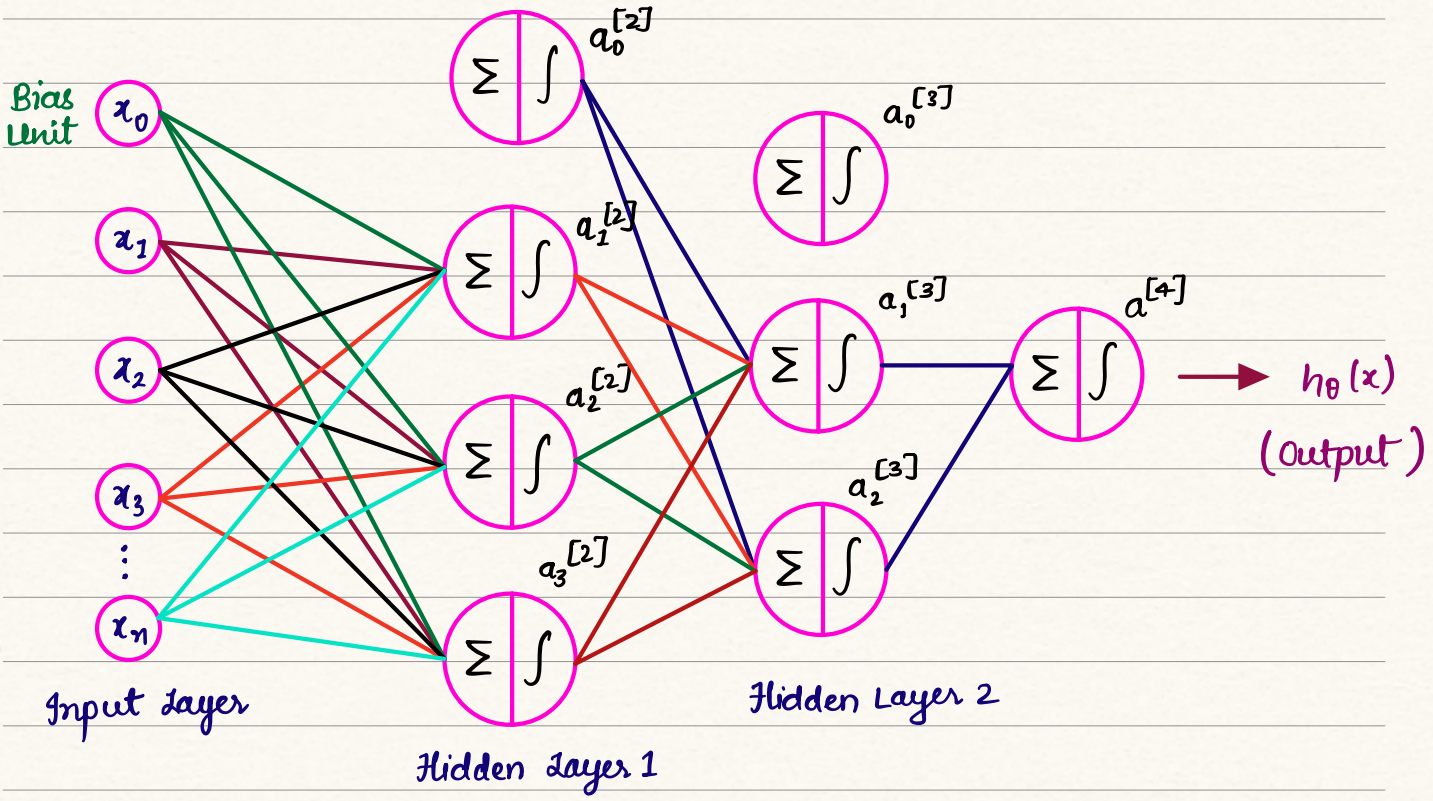
Subject: / /

$$z^{[2]} = \begin{bmatrix} \theta_{10}^{(1)} x_0 + \theta_{11}^{(1)} x_1 + \theta_{12}^{(1)} x_2 + \theta_{13}^{(1)} x_3 \\ \theta_{20}^{(1)} x_0 + \theta_{21}^{(1)} x_1 + \theta_{22}^{(1)} x_2 + \theta_{23}^{(1)} x_3 \\ \theta_{30}^{(1)} x_0 + \theta_{31}^{(1)} x_1 + \theta_{32}^{(1)} x_2 + \theta_{33}^{(1)} x_3 \end{bmatrix}$$

$$\begin{bmatrix} \theta_{10}^{(2)} & \theta_{11}^{(2)} & \theta_{12}^{(2)} & \theta_{13}^{(2)} \end{bmatrix} \begin{bmatrix} \theta_{10}^{(1)} x_0 + \theta_{11}^{(1)} x_1 + \theta_{12}^{(1)} x_2 + \theta_{13}^{(1)} x_3 \rightarrow a \\ \theta_{20}^{(1)} x_0 + \theta_{21}^{(1)} x_1 + \theta_{22}^{(1)} x_2 + \theta_{23}^{(1)} x_3 \rightarrow b \\ \theta_{30}^{(1)} x_0 + \theta_{31}^{(1)} x_1 + \theta_{32}^{(1)} x_2 + \theta_{33}^{(1)} x_3 \rightarrow c \end{bmatrix}$$

$$= \theta_{10}^{(2)} + \theta_{11}^{(2)} a + \theta_{12}^{(2)} b + \theta_{13}^{(2)} c$$

★ Backpropogation Algorithm —



$$z^{[1]} = \begin{bmatrix} \theta_{10}^{(1)} x_0 + \theta_{11}^{(1)} x_1 + \theta_{12}^{(1)} x_2 + \theta_{13}^{(1)} x_3 \\ \theta_{20}^{(1)} x_0 + \theta_{21}^{(1)} x_1 + \theta_{22}^{(1)} x_2 + \theta_{23}^{(1)} x_3 \\ \theta_{30}^{(1)} x_0 + \theta_{31}^{(1)} x_1 + \theta_{32}^{(1)} x_2 + \theta_{33}^{(1)} x_3 \end{bmatrix}$$

Subject: 27 /10 /2025

"$x_0 = 1$";

$$z^{[1]} = \begin{bmatrix} \theta_{10}^{(1)} + \theta_{11}^{(1)} x_1 + \theta_{12}^{(1)} x_2 + \theta_{13}^{(1)} x_3 \\ \theta_{20}^{(1)} + \theta_{21}^{(1)} x_1 + \theta_{22}^{(1)} x_2 + \theta_{23}^{(1)} x_3 \\ \theta_{30}^{(1)} + \theta_{31}^{(1)} x_1 + \theta_{32}^{(1)} x_2 + \theta_{33}^{(1)} x_3 \end{bmatrix}$$

$$= \begin{bmatrix} \theta_{11}^{(1)} & \theta_{12}^{(1)} & \theta_{13}^{(1)} \\ \theta_{21}^{(1)} & \theta_{22}^{(1)} & \theta_{23}^{(1)} \\ \theta_{31}^{(1)} & \theta_{32}^{(1)} & \theta_{33}^{(1)} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} \theta_{10}^{(1)} \\ \theta_{20}^{(1)} \\ \theta_{30}^{(1)} \end{bmatrix}$$

↗ Bias Vector

$$z^{(1)} = w^{(1)} x + b^{(1)}$$

(3×1) (3×4) (4×1)

$$a^{(1)} = \sigma(z^{(1)}) = \sigma(w^{(1)} x + b^{(1)})$$

{$\sigma \rightarrow$ sigmoid Activation function}

$$z^{(2)} = w^{(2)} a^{(1)} + b^{(2)}$$

(2×1) (2×3) (3×1)

$$\begin{bmatrix} \theta_{10}^{(2)} & \theta_{11}^{(2)} & \theta_{12}^{(2)} & \theta_{13}^{(2)} \\ \theta_{20}^{(2)} & \theta_{21}^{(2)} & \theta_{22}^{(2)} & \theta_{23}^{(2)} \end{bmatrix} \begin{bmatrix} 1 \\ a_1^{(1)} \\ a_2^{(1)} \\ a_3^{(1)} \end{bmatrix}$$

$$a^{(2)} = \sigma(z^{(2)}) = \sigma(w^{(2)} a^{(1)} + b^{(2)})$$

$$z^{(3)} = w^{(3)} a^{(2)} + b^{(3)}$$

(1×1) (1×3) (3×1)

$$a^{(3)} = \sigma(z^{(3)}) = \sigma(w^{(3)} a^{(2)} + b^{(3)})$$

Subject: / /

$$z^{(1)} = W^{(1)} x + b^{(1)}$$
$$a^{(1)} = \sigma(z^{(1)}) = \sigma(W^{(1)} x + b^{(1)}) \quad (3 \times 1)$$

$$z^{(2)} = W^{(2)} a^{(1)} + b^{(2)}$$
$$a^{(2)} = \sigma(z^{(2)}) = \sigma(W^{(2)} a^{(1)} + b^{(2)}) \quad (2 \times 1)$$

$$z^{(3)} = W^{(3)} a^{(2)} + b^{(3)}$$
$$a^{(3)} = \sigma(z^{(3)}) = \sigma(W^{(3)} a^{(2)} + b^{(3)}) \quad (1 \times 1)$$

To train any Neural Network Model, we need to update the following parameters :

1.) $W^{(1)}$
2.) $b^{(1)}$
3.) $W^{(2)}$
4.) $b^{(2)}$
5.) $W^{(3)}$
6.) $b^{(3)}$

Q.) How can I update these parameters ?

We need to define the loss function.

→ Optimize these parameters such that loss function is minimized.

Suppose that I have a dataset (training) that deals with the binary classification problems.
Suppose that I have "m" training examples.

Loss function = Binary cross Entropy loss function

$$= \sum_{i=1}^{m} -\left(y^{(i)} \log \hat{y}^{(i)} + (1-y^{(i)}) \log (1-\hat{y}^{(i)})\right)$$

For $n = 1$;

$$L = -y \log \hat{y} - (1-y) \log (1-\hat{y})$$

Subject: / /

$$W^{(l)} := W^{(l)} - \alpha \frac{\partial L}{\partial W^{(l)}} \qquad l = \{1,2,3\}$$

$$b^{(l)} := b^{(l)} - \alpha \frac{\partial L}{\partial b^{(l)}}$$

$$L = -y \log \hat{y} - (1-y) \log (1-\hat{y})$$

$$\hat{y} = \sigma(z^{(3)}) = \sigma(W^{(3)} a^{(2)} + b^{(3)})$$

$(1\times 2)\ (2\times 1)$

$$\frac{\partial L}{\partial W^{(3)}} = -y \frac{\partial}{\partial W^{(3)}} \log(\sigma(z^{[3]})) - (1-y) \frac{\partial}{\partial W^{(3)}} \log(1-\sigma(z^{[3]}))$$

$$= \frac{-y}{\sigma(z^{(3)})} \sigma(z^{(3)})(1-\sigma(z^{(3)}))(a^{(2)})^T - \frac{(1-y)}{(1-\sigma(z^{(3)}))} (1-\sigma(z^{[3]}))(-\sigma(z^{[3]}))(a^{(2)})^T$$

$$X = \begin{bmatrix} w_1 & w_2 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = w_1 a_1 + w_2 a_2$$

$$\frac{\partial X}{\partial W} = \begin{bmatrix} a_1 & a_2 \end{bmatrix}$$

$$= -(y(1-a^{(3)}) - (1-y)a^{(3)})(a^{(2)})^T$$

$$= -(y - a^{(3)}y - a^3 + ya^{(3)})(a^{(2)})^T$$

$(a^{(2)})^T \in \mathbb{R}^{1\times 2}$

$$\frac{\partial L}{\partial W^{(3)}} = -\underbrace{(y - a^{(3)})}_{\frac{\partial L}{\partial z^{(3)}}} \underbrace{(a^{(2)})^T}_{\frac{\partial z^{(3)}}{\partial W^{(3)}}} \qquad \left\{ (a^{(2)})^T = \frac{\partial z^{(3)}}{\partial W^{(3)}} \right.$$

$$L = -y \log(\sigma(z^{(3)})) - (1-y) \log(1-\sigma(z^{(3)}))$$

Subject: 28/10/2025

$$\frac{\partial L}{\partial z^{(3)}} = \frac{-y}{\sigma(z^{(3)})}\,\sigma(z^{(3)})(1-\sigma(z^{(3)})) - \frac{(1-y)}{(1-\sigma(z^{(3)}))}\,(-\sigma(z^{(3)})(1-\sigma(z^{(3)})))$$

$$= -y(1-\sigma(z^{(3)})) + (1-y)(\sigma(z^{(3)}))$$

$$= -y + y\sigma(z^{(3)}) + \sigma(z^{(3)}) - y\sigma(z^{(3)})$$

$$= -(y - \sigma(z^{(3)}))$$

$$= -(y - a^{(3)})$$

$$\underbrace{\frac{\partial L}{\partial W^{(2)}}}_{(2\times 3)} = \frac{\partial L}{\partial a^{(3)}} \times \frac{\partial a^{(3)}}{\partial z^{(3)}} \times \frac{\partial z^{(3)}}{\partial a^{(2)}} \times \frac{\partial a^{(2)}}{\partial z^{(2)}} \times \frac{\partial z^{(2)}}{\partial W^{(2)}}$$

We are using chain rule because one layer has dependency on the previous layers.

$$L = -y \log a^{(3)} - (1-y)\log(1-a^{(3)})$$

$$a^{(3)} = \sigma(z^{(3)})$$
$$z^{(3)} = W^{(3)} a^{(2)} + b^{(3)}$$

$$a^{(2)} = \sigma(z^{(2)})$$
$$z^{(2)} = W^{(2)} a^{(1)} + b^{(2)}$$

$$\frac{\partial L}{\partial W^{(2)}} = -\underbrace{(y^{(i)} - a^{(3)})}_{(1\times 1)}\underbrace{(W^{(3)})^T}_{(2\times 1)}\underbrace{a^{(2)}(1-a^{(2)})}_{(2\times 1)}\underbrace{(a^{(1)})^T}_{(1\times 3)}$$

$$\frac{\partial L}{\partial a^{(3)}} \times \frac{\partial a^{(3)}}{\partial z^{(3)}} \quad \frac{\partial z^{(3)}}{\partial a^{(2)}} \quad \frac{\partial a^{(2)}}{\partial z^{(2)}} \quad \frac{\partial z^{(2)}}{\partial W^{(2)}}$$

$$z^{(2)} = W^{(2)} a^{(1)} + b^{(2)}$$

$$W^{(2)} = \begin{bmatrix} W_{11}^{(2)} & W_{12}^{(2)} & W_{13}^{(2)} \\ W_{21}^{(2)} & W_{22}^{(2)} & W_{23}^{(2)} \end{bmatrix}$$

Subject:

$$\frac{\partial z^{(2)}}{\partial W^{(2)}} = \begin{bmatrix} \frac{\partial z^{(2)}}{\partial w_{11}^{(2)}} & \frac{\partial z^{(2)}}{\partial w_{12}^{(2)}} & \frac{\partial z^{(2)}}{\partial w_{13}^{(2)}} \\ \frac{\partial z^{(2)}}{\partial w_{21}^{(2)}} & \frac{\partial z^{(2)}}{\partial w_{22}^{(2)}} & \frac{\partial z^{(2)}}{\partial w_{23}^{(2)}} \end{bmatrix}$$

$$z^{(2)} = W^{(2)} a^{(1)} + b^{(2)}$$

$$= \begin{bmatrix} w_{11}^{(2)} a_1^{(1)} + w_{12}^{(2)} a_2^{(1)} + w_{13}^{(2)} a_3^{(1)} + b^{(2)} \\ w_{21}^{(2)} a_1^{(1)} + w_{22}^{(2)} a_2^{(1)} + w_{23}^{(2)} a_3^{(1)} \end{bmatrix}$$

$$\frac{\partial L}{\partial W^{(2)}} = -(y^{(i)} - a^{(3)})(W^{(3)})^T a^{(2)}(1-a^{(2)})(a^{(1)})^T$$

$$= (W^{(3)})^T . a^{(2)}(1-a^{(2)})\left(-(y^{(i)} - a^{(3)})(a^{(1)})^T\right)$$

Component wise product ←

$a^{(2)} = \sigma(z^{(2)})$

$$\begin{bmatrix} a_1^{(2)} \\ a_2^{(2)} \end{bmatrix} = \sigma \begin{bmatrix} z_1^{(2)} \\ z_2^{(2)} \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} \frac{\partial a_1^{(2)}}{\partial z_1^{(2)}} \\ \frac{\partial a_2^{(2)}}{\partial z_2^{(2)}} \end{bmatrix} = \begin{bmatrix} a_1^{(2)}(1-a_1^{(2)}) \\ a_2^{(2)}(1-a_2^{(2)}) \end{bmatrix}$$

Component-wise product :- $\begin{bmatrix} a \\ b \end{bmatrix}_{2\times1} \begin{bmatrix} c \\ d \end{bmatrix}_{2\times1} = \begin{bmatrix} a.c \\ b.d \end{bmatrix}_{2\times1}$

$$\frac{\partial L}{\partial W^{(2)}} = \underbrace{(W^{(3)})^T . a^{(2)}(1-a^{(2)})}_{(2\times1)}\left(-\underbrace{(y^{(i)} - a^{(3)})}_{(1\times1)}\underbrace{(a^{(1)})^T}_{(1\times3)}\right)$$

Subject: 03/11/2025

☆ Support Vector Machines (SVMs) :—

$$h(\theta^T x) = \frac{1}{1+e^{-\theta^T x}}$$

$\theta^T x >> 0 \longrightarrow y = 1$

$\theta^T x << 0 \longrightarrow y = 0$

Functional Margin ——

In SVM, such insights fall under the notion of functional margins.

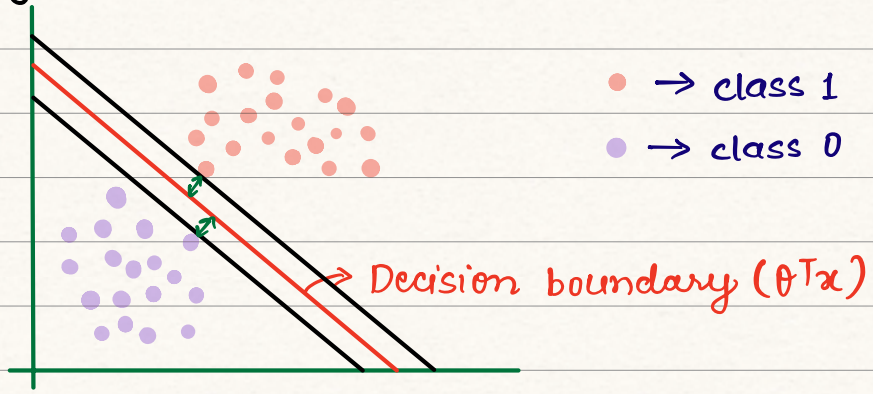
To avoid $g(\theta^T x) \approx 0.5$

We will train the parameters (θ) using gradient descent algorithm to get ;

(i) $g(\theta^T x) >> 0.5$ for $y = 1$

(ii) $g(\theta^T x) << 0.5$ for $y = 0$

Geometric Margins ——



Notations :—

$y \in \{-1, 1\}$ (SVM)

$y \in \{0, 1\}$ (Logistic Regression)

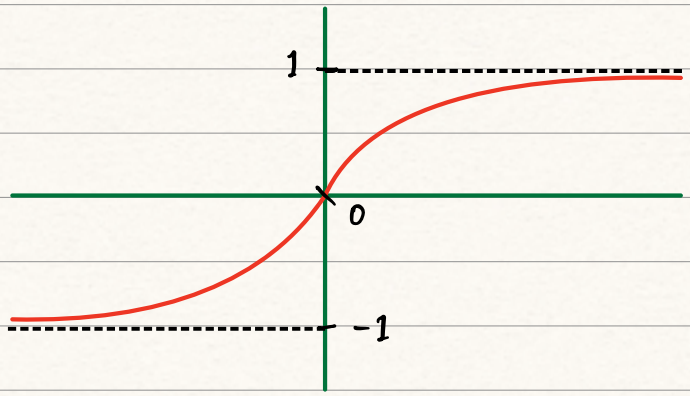
$h_{w,b}(x) = g(w^T x + b)$ (SVM)

$h_\theta(x) = g(\theta^T x)$ (Logistic Regression)

Subject: / /

$$g(w^T x + b) = 1 \quad \text{if } w^T x + b \geq 0$$

$$g(w^T x + b) = -1 \quad \text{if } w^T x + b < 0$$



✯ Functional Margin in context of SVM :—

Suppose I have a training example $(x^{(i)}, y^{(i)})$ $\hookrightarrow \{-1, 1\}$

$x^{(i)}$ → Input feature vector for i^{th} training example

Functional margin of (w, b) w.r.t this training example is ;

$$\hat{\gamma}^{(i)} = y^{(i)}(w^T x^{(i)} + b) \qquad \left[y^{(i)} = \{-1, +1\}\right]$$

If $\hat{\gamma}^{(i)} > 0 \rightarrow$ We are not making any training error.

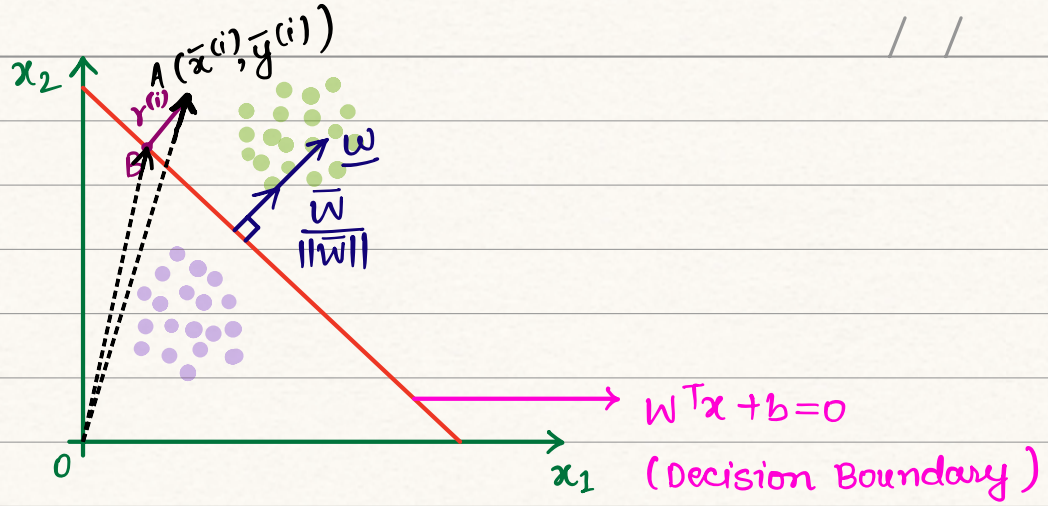
$$h_{w,b}(x^{(i)}) = y^{(i)}$$

$$S = \left[x^{(i)}, y^{(i)}\right] \; ; \quad i = 1, 2, \ldots, n$$

$$\hat{\gamma} = \min_{1,2,\ldots,n} \hat{\gamma}^{(i)}$$

✯ Geometric Margin in context of SVM :—

Subject: / /



$$\vec{A} - \vec{B} = \gamma^{(i)} \frac{\bar{w}}{\|\bar{w}\|}$$

$$\vec{B} = \vec{A} - \gamma^{(i)} \frac{\bar{w}}{\|\bar{w}\|}$$

For point A;

$$\vec{B} = \bar{x}^{(i)} - \gamma^{(i)} \frac{\bar{w}}{\|\bar{w}\|}$$

For point B;

$$w^T \vec{B} + b = 0$$

$$w^T \left(x^{(i)} - \gamma^{(i)} \frac{\bar{w}}{\|\bar{w}\|} \right) + b = 0$$

$$\Rightarrow w^T x^{(i)} - \gamma^{(i)} \frac{w^T w}{\|\bar{w}\|} + b = 0$$

$\{ w^T w = \|w\|^2 \}$

$$\Rightarrow w^T x^{(i)} - \gamma^{(i)} \|w\| + b = 0$$

$$\gamma^{(i)} = \frac{w^T x^{(i)} + b}{\|w\|}$$

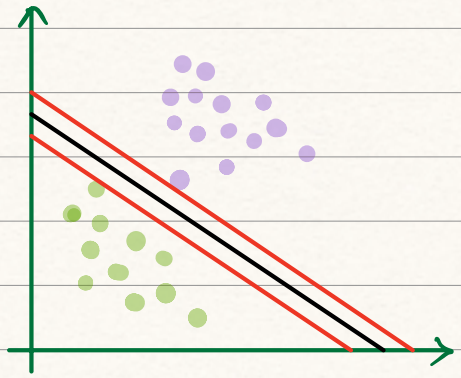
In this case, the Geometric Margin for the $(x^{(i)}, y^{(i)})$ training example is;

$$\gamma^{(i)} = y^{(i)} \left(\frac{w^T x^{(i)}}{\|w\|} + \frac{b}{\|w\|} \right)$$

Subject: 04/11/2025

If $\|w\| = 1$;

→ Geometric Margin ≡ Functional Margin



Hard SVM

Clear separation between two datasets

Optimization problem:

$$\max_{w,b} \gamma \quad \text{(Geometric Margin)}$$

Subject to $y^{(i)}(w^T x^{(i)} + b) \geq \gamma \quad \{i = 1, 2, \ldots, n\}$

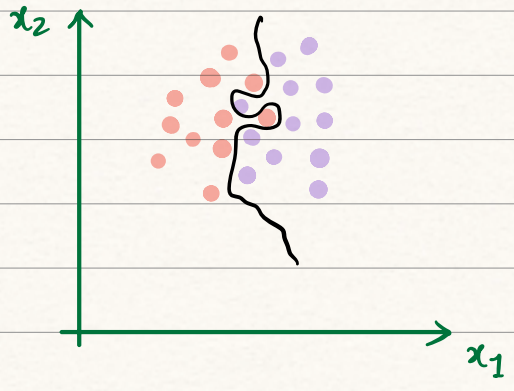
$\|w\| = 1$

Q.) Why should I go beyond the hard SVMs?

→ We should opt for soft SVMs.

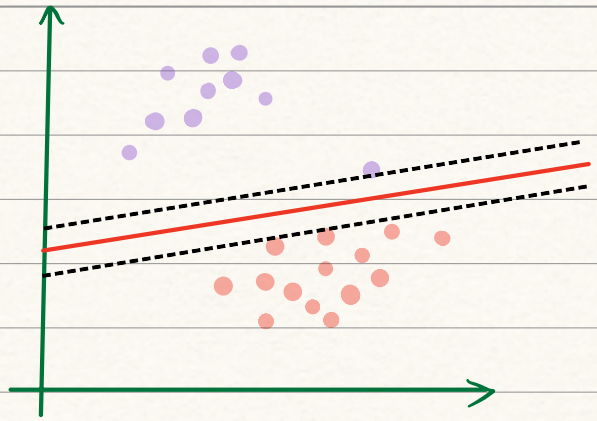
→ We should relax some constraints.

(i) In general, the dataset in realistic applications is non-separable.

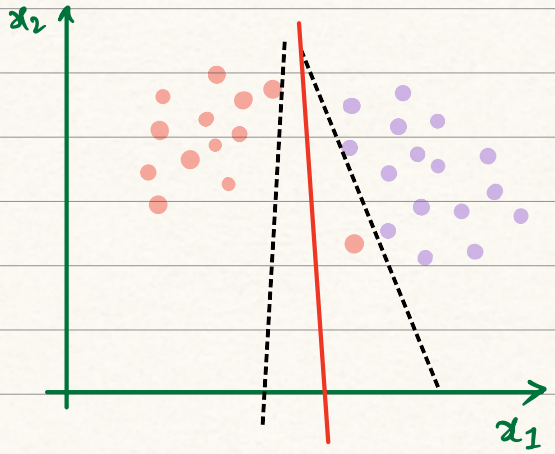


(ii) Hard SVMs would not work when there are some outliers in the dataset.

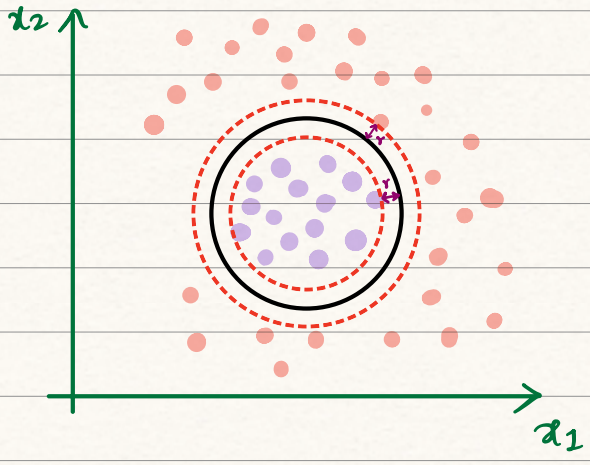
Subject: / /



Geometric margin is minimized due to outlier.



Soft Margin



Transformation using kernels (Combine SVMs with kernels)

■ Unsupervised Learning Algorithms :

✯ Clustering :—

Suppose we have a training set ;

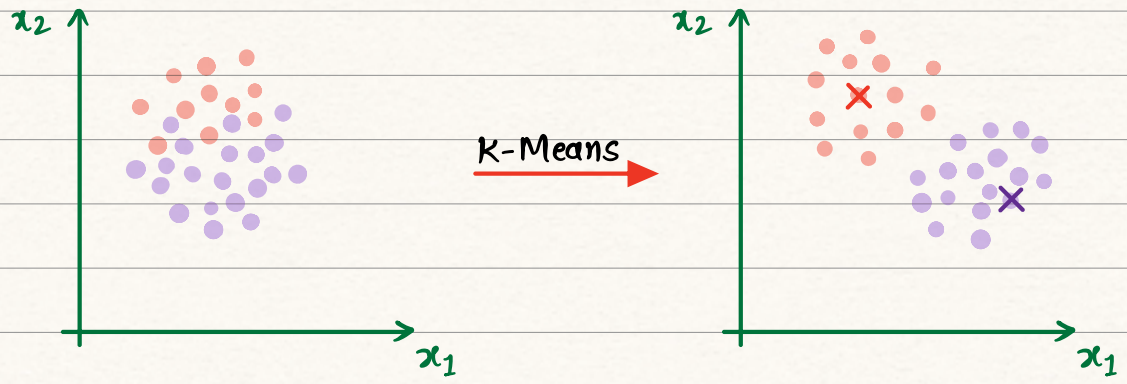
$$\{x^{(1)}, x^{(2)}, x^{(3)}, \ldots, x^{(n)}\}$$

and we want to group the data into a few classes.

Here, $x^{(i)} \in \mathbb{R}^d$ and there is no labelling.

Subject: / /

So this is unsupervised learning problem.



☆ K-Means clustering Algorithm :-

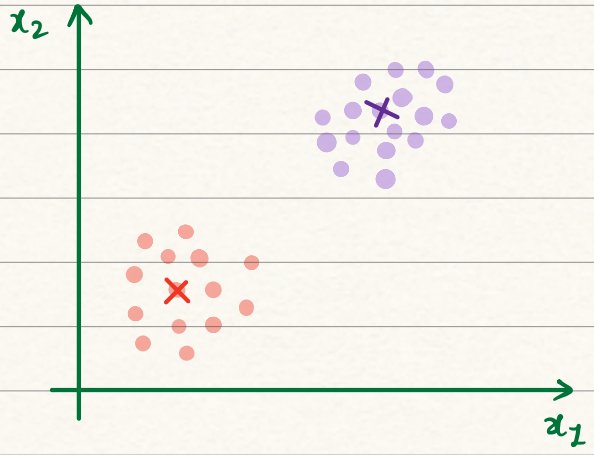
1.) Initialize the cluster centroids $\bar{u}_1, \bar{u}_2, \ldots, \bar{u}_K \in \mathbb{R}^d$ randomly.

2.) Repeat until convergence for every i, set ;

$$c^{(i)} := \arg\min_j \| \bar{x}^{(i)} - \bar{u}_j \|^2$$

For every j (clusters), set ;

$$u_j := \frac{\sum_{i=1}^{n} 1\{c^{(i)} = j\} . x^{(i)}}{\sum_{i=1}^{n} 1\{c^{(i)} = j\}}$$



Subject: 10/11/2025

☆ Expectation - Maximization (EM) Algorithm for Mixture of Gaussians

$\{ \underline{x}^{(1)}, \underline{x}^{(2)}, \ldots, \underline{x}^{(n)} \}$ → Data do not have any labels

$\underbrace{(\{ \underline{x}^{(1)}, \underline{x}^{(2)}, \underline{x}^{(3)} \})}_{(\mu_1, \Sigma_1)}$

$\underbrace{(\{ \underline{x}^{(4)}, \underline{x}^{(5)}, \underline{x}^{(6)} \})}_{(\mu_2, \Sigma_2)}$

We first model the input data by specifying a Joint distribution ;

$$f(\underline{x}^{(i)}, z^{(i)}) = f(\underline{x}^{(i)} | z^{(i)}) \, P(z^{(i)})$$

$z^{(i)} \sim$ Multinomial distribution (ϕ)

$$\phi_j \geq 0, \quad \sum_{j=1}^{k} \phi_j = 1, \quad \phi_j = P(z^{(i)} = j)$$

$$(\underline{x}^{(i)} | z^{(i)} = j) \sim \mathcal{N}(\underline{\mu}_j, \Sigma_j)$$

Q.) How $\underline{x}^{(i)}$ was generated ?

$i = \{ 1, 2, 3, \ldots, k \}$ (k Gaussians)

Randomly choose Gaussian from 'i'

$\underline{x}^{(1)}$ dataset coming Gaussian 'i'

$$\{ \underbrace{\underline{x}^{(1)}, \underline{x}^{(2)}}_{(\underline{\mu}_1, \Sigma_1)}, \underbrace{\underline{x}^{(3)}, \underline{x}^{(4)}}_{(\underline{\mu}_2, \Sigma_2)}, \underbrace{\underline{x}^{(5)}, \underline{x}^{(6)}}_{(\underline{\mu}_3, \Sigma_3)} \}$$

Subject: / /

$$f(\underline{x}^{(1)}, \underline{x}^{(2)}, \cdots, \underline{x}^{(n)}) = f(\underline{x}^{(11)}) \cdot f(\underline{x}^{(21)}) \cdots f(\underline{x}^{(n1)})$$

$$= \prod_{i=1}^{n} f(\underline{x}^{(i)})$$

$$L(\underline{\mu}, \Sigma, \phi) = \prod_{i=1}^{n} f(\underline{x}^{(i)}, \underline{\mu}, \Sigma, \phi)$$

↑ Likelihood

$$\log(L(\underline{\mu}, \Sigma, \phi)) = \sum_{i=1}^{n} \log f(\underline{x}^{(i)}, \underline{\mu}, \Sigma, \phi)$$

$$= \sum_{i=1}^{n} \log \sum_{z^{(i)}=1}^{k} f(\underline{x}^{(i)} | z^{(i)}, \mu, \Sigma) P(z^{(i)}, \phi)$$

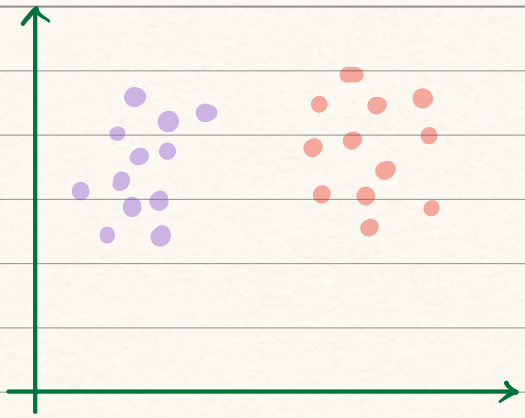
If we know $\underline{x}^{(i)}$ is coming from $\mu^{(i)}$ Gaussian.
(Labelled data)

$$\phi_j = \frac{\sum_{i=1}^{n} 1\{z^{(i)} = j\}}{n}$$

$$\mu_j = \frac{\sum_{i=1}^{n} 1\{z^{(i)} = j\} \cdot x^{(i)}}{\sum_{i=1}^{n} 1\{z^{(i)} = j\}}$$

$$\Sigma_j = \frac{\sum_{i=1}^{n} 1\{z^{(i)} = j\} (\underline{x}^{(i)} - \mu_j)(\underline{x}^{(i)} - \mu_j)^T}{\sum_{i=1}^{n} 1\{z^{(i)} = j\}}$$

Subject: / /



K = 2 Gaussians
(Labelled data)

K = 2 Gaussians
(Unlabelled data)

So for unlabelled data we will use EM Algorithm for mixture of Gaussians.

It has two steps :—

1.) E-step : It tries to guess the values of $z^{(i)}$'s.

2.) M-step : It updates the parameters.

Repeat until it converges { (E-step). For each i, j, set $w_j^{(i)} := P(z^{(i)} = j \mid x^{(i)}; \phi, \Sigma, \mu)$

$w_j^{(i)}$ ↓ what is the probability that the ith training example is drawn from jth Gaussian once I observed it.

weights ← $w_j^{(i)}$
(0, 1)

Soft Clustering ↓

Clusters coming from two or more Gaussians.

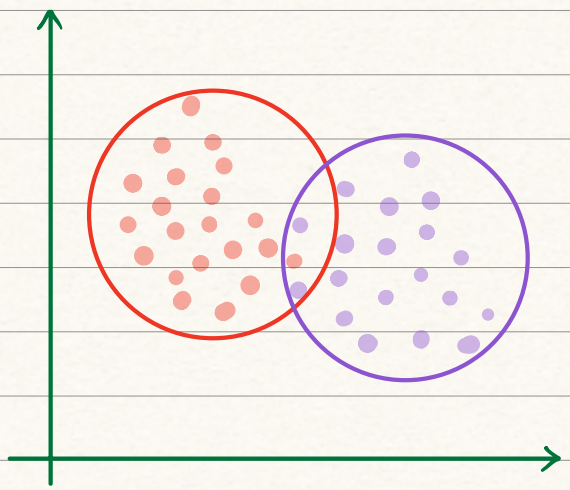
Subject: / /

(M-step);

$$\phi_j := \frac{\sum_{i=1}^{n} w_j^{(i)}}{n}$$

$$\mu_j := \frac{\sum_{i=1}^{n} w_j^{(i)} \, \underline{x}^{(i)}}{\sum_{i=1}^{n} w_j^{(i)}}$$

$$\Sigma_j := \frac{\sum_{i=1}^{n} w_j^{(i)} \, (\underline{x}^{(i)} - \underline{\mu}_j)(\underline{x}^{(i)} - \underline{\mu}_j)^T}{\sum_{i=1}^{n} w_j^{(i)}}$$



$$p(z^{(i)} = j \mid \underline{x}^{(i)}, \phi, \mu, \Sigma) = \frac{f(\underline{x}^{(i)} \mid z^{(i)} = j; \mu, \Sigma) \, p(z^{(i)} = j, \phi)}{\sum_{l=1}^{K} f(\underline{x}^{(i)} \mid z^{(i)} = l; \mu, \Sigma) \, p(z^{(i)} = l, \phi)}$$