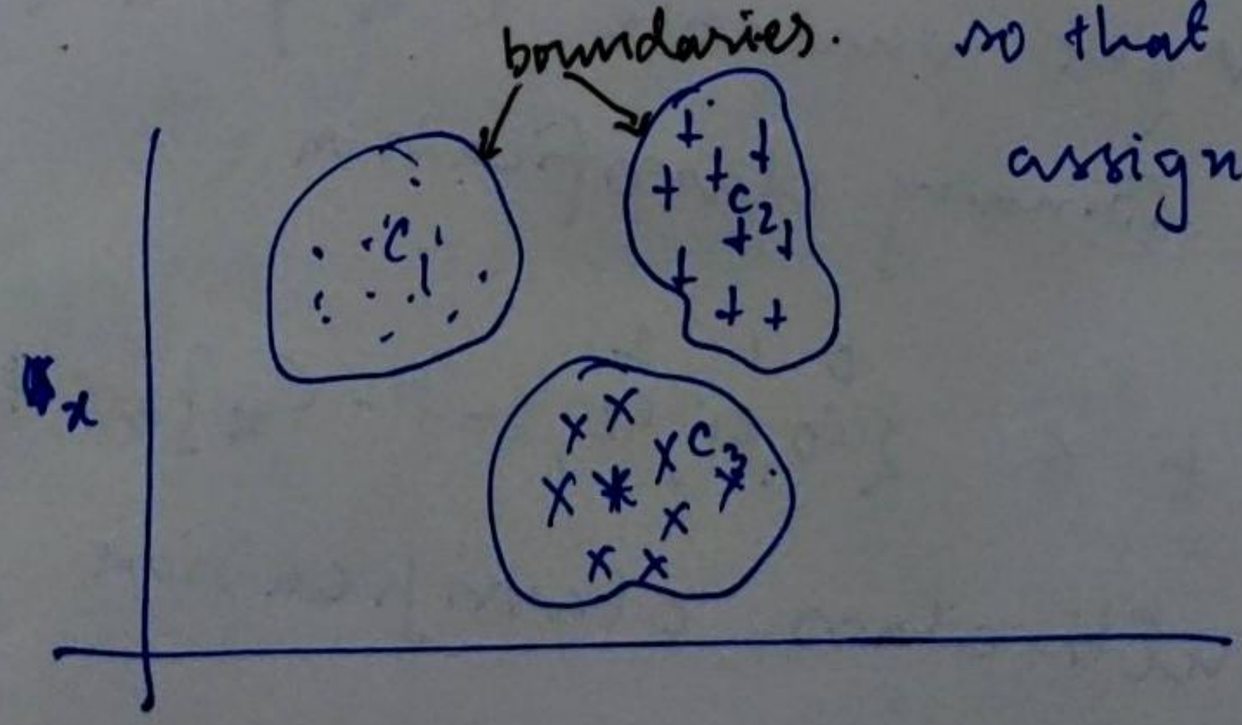
"Classification and logistic regression"

— Classification probs are just like regression prob, except that the values of target variable y, which we want to predict takes only a small number of discrete values. (Ex: $K = \{1, 2, \cdots K\}$)

→ The goal in classification prob is to take an input vector $\underline{x}$ and to assign it to one K discrete classes C_K, where

$$K = \{1, 2, \cdots K\}$$

→ Common ~~since~~ scenario classes are disjoint so that each input is assigned only one class.



— ~~Input space is thereby divi~~ Decision surface/ decision boundary.

Input space is divided into decision spaces, whose boundaries are decision boundaries.

— In general. is $\underset{\sim}{x} \in \mathbb{R}^{D \times n}$ (D dimensional)

$$\underset{\sim}{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{Dm} \end{pmatrix} \in \mathbb{R}^{D \times n} \qquad n: \dim(\text{features})$$

If linear models of classification. is. adopted, which means. "decision surfaces" are linear functions of the input vector $\underset{\sim}{x}$ and hence are defined by $(n-1)$ - dimensional hyperplane.

— . Binary & Multiclass classification.

The target variable t or $y \in \{0, 1\}$ is called Binary classification

If $t \in \{C_0, C_1, C_2, \cdots C_k\}$ then it is multi-class classification.

C_0
$t = \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$, C_1 $t = \begin{pmatrix} 0 \\ 1 \\ \vdots \\ \vdots \end{pmatrix}$, C_K $t = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \text{ kth} \end{pmatrix}$

Three distinct approachs for clarrification prob.

Deterministic: This involves constructing a "discriment function", which assigns each input vector $\underline{x}$ a class.

probabilistic approach.

- Is to model the conditional probability $p(C_k|\underline{x})$

- two different approaches to model $p(C_k|\underline{x})$.

A) Model them directly. Represent them as parametric models and then optimizing the parameters using a training set.

B) Second, is generative approach. in which we model the class-conditional densities, given by $p(\underline{x}|C_k)$, together with prior probabilities $p(C_k)$ for the classes. and.

using Baye's therm. $= P(\underline{x}, C_k) / P(\underline{x})$

$$P(C_k|\underline{x}) = \frac{P(\underline{x}|C_k)\, P(C_k)}{P(\underline{x})}$$

C_k is the classes.

$$P(x) = \sum_k P(x|C_k)\, P(C_k)$$

We model the joint distribution in generative model.

Classification: Binary ($k=2$) $\begin{cases} C_0 \\ C_1 \end{cases}$

e.g, Email : Spam / Not Spam

Online transaction: fraudulent (Yes/No)

Tumor : Malignant / Benign

$\underset{\sim}{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$ ⟶ $y \in \{0, 1\}$

Feature vector

0: negative class

1: +ive class.

example: lets take one feature x_1: "tumor size".

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

"Not malignant / Benign"

"Yes" Malignant.

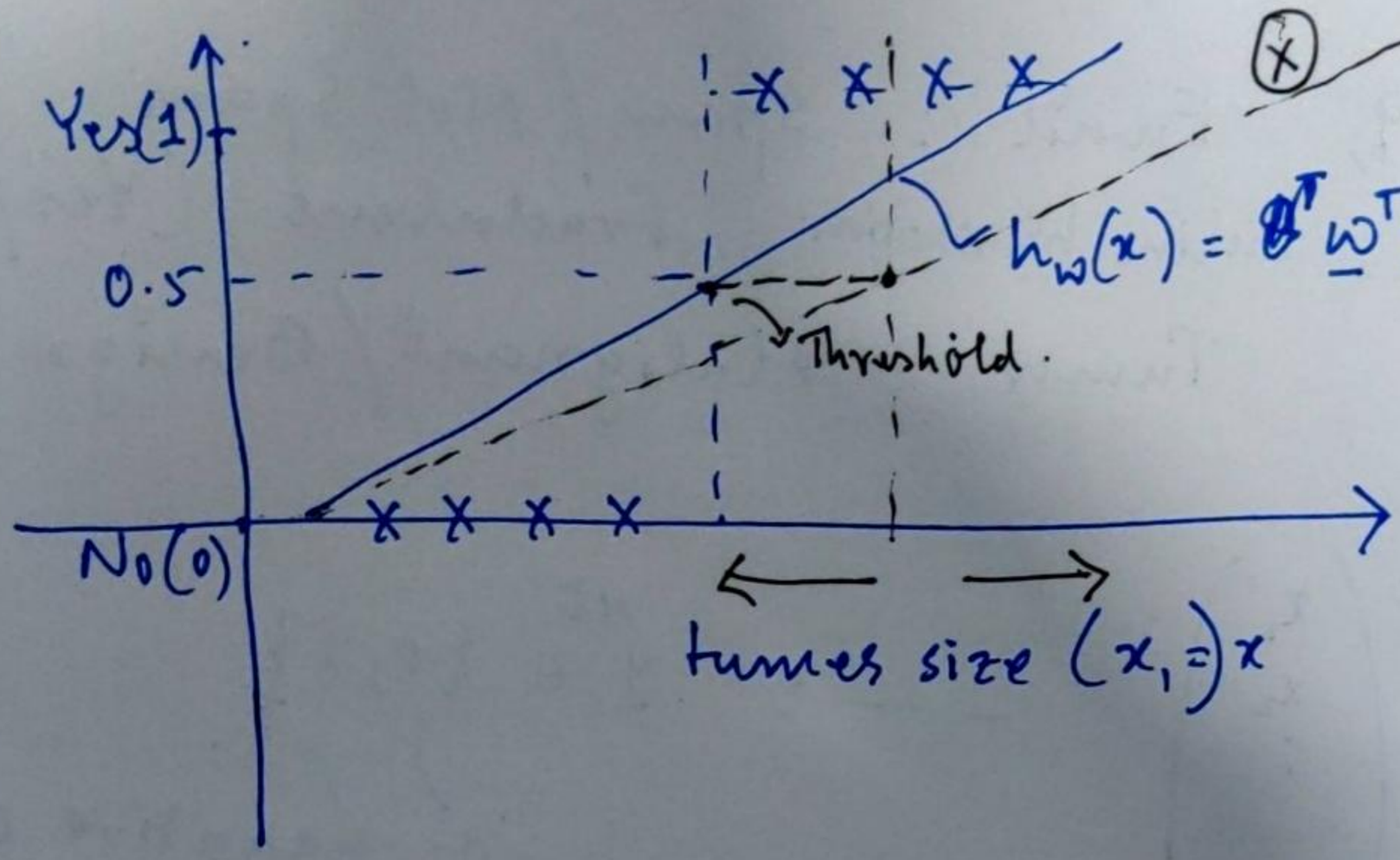
usual approach may be.

i) Form the hypothesis function, and.

$h_w(x) = \underline{w}^T \underline{x} + w_0 = \underline{w}^T \underline{x} + w_0$

Find the parameters., assuming $h_{\underline{w}}(x)$ is a continuous function.

how would we classify them.



define a threshhold classifier output $h_w(x) = 0.5$, ($h_w(x)$ is found by fitting the data)

Data

$(x_1^{(1)}, 1) \Leftrightarrow (x_1^{(1)}, Y)$ If $h_w(x) \geq 0.5$, assign $y = 1$

$(x_1^{(2)}, 0) \Leftarrow (x_1^{(2)}, N)$ If $h_w(x) < 0.5$, assign $y = 0$.

$(x_1^{(3)}, 0) \Leftarrow (x_1^{(3)}, N)$

$\vdots$

$(x_1^{(m)}, 1) \Leftarrow (x_1^{(m)}, Y)$ Please Note x is a continuous variable.

& we are assuming $h_w(x) = \underline{w}^T \underline{x} + w_0$ is Conti

Issues: Although final target variable

y w r t $= 0$ or 1

but, $h_w(x)$ can be > 1 or < 0.

which DOES NOT make sense (Intuitively)

Tbnc. Introduce, nonlinear boundedness:

As y is a discrete variable (for class identification) we change the hypothesis function defination by introducing a non-linear functional map.

logistic regression. $\rightarrow h_w(x) = g(\tilde{w}^T \tilde{x}')$ $\quad \tilde{x}' = \begin{pmatrix} 1 \\ \tilde{x} \end{pmatrix}$

$= g(w^T x + w_0)$ $\quad g$ ← Bounded hypothesis.

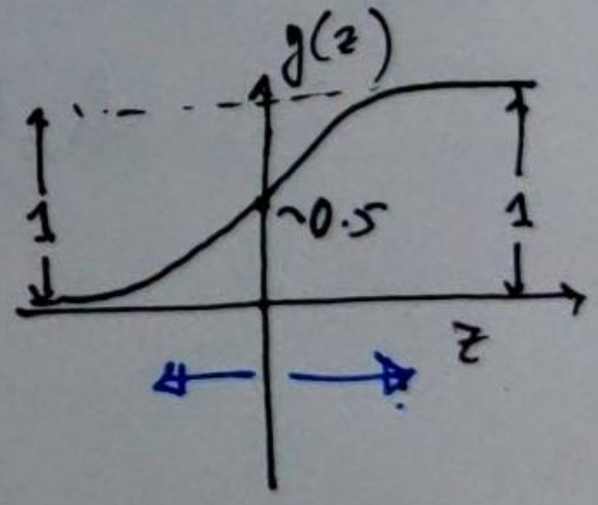
so that $0 \leq g(\tilde{w}^T \tilde{x}') \leq 1$ $\quad 0 \leq h_w(x) \leq 1$

$g(\cdot)$: is called activation function.

g^{-1} : is called Link function. $\quad \tilde{x} = \begin{pmatrix} x_0 \\ x_1 \end{pmatrix}$ $\quad \tilde{w} = \begin{pmatrix} w_0 \\ w_1 \end{pmatrix}$

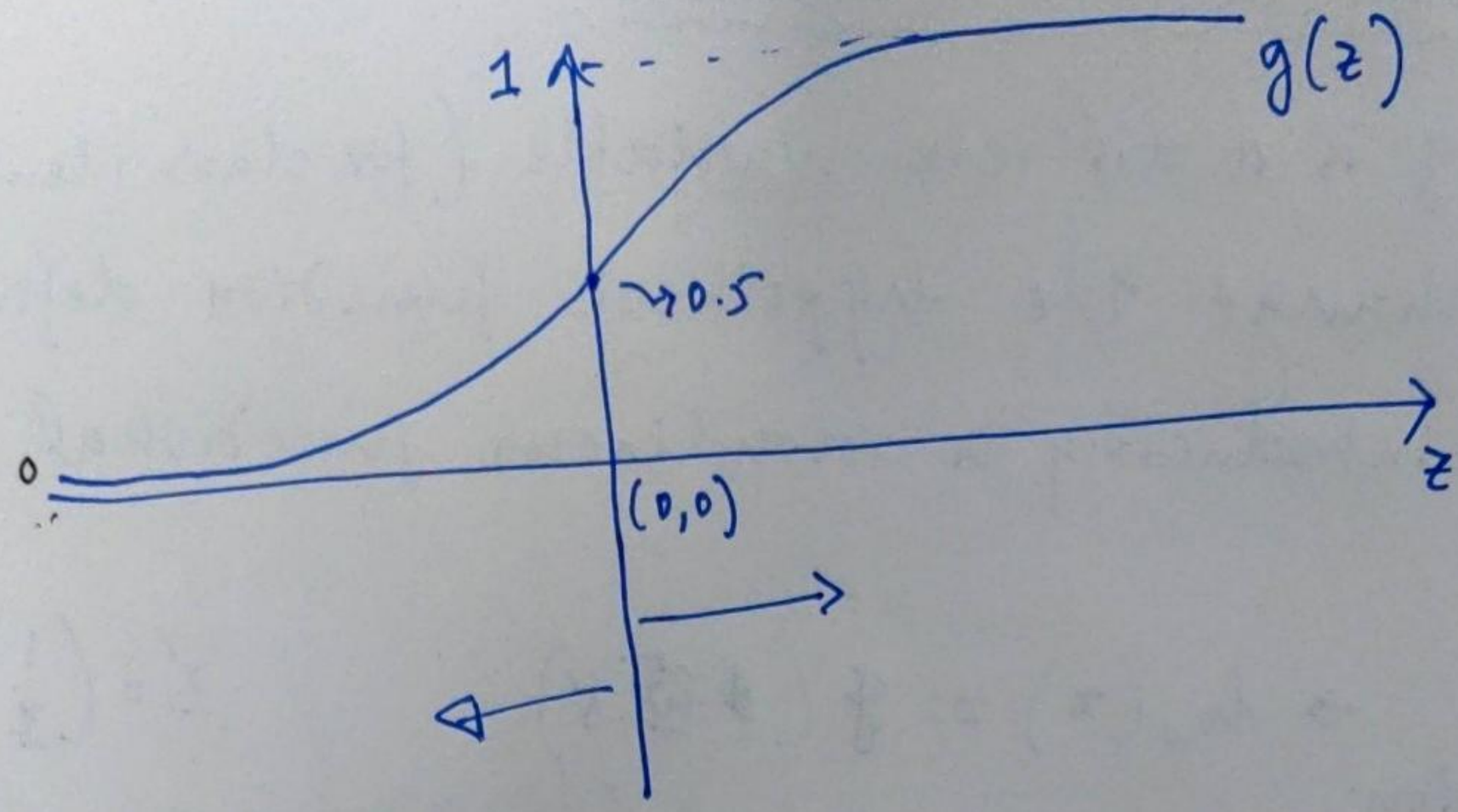
One type of activation function is. "Sigmoid Function"

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



hence, for logistic regression, the hypothesis function

$$h_w(x) = g(w^T x) = \left(\frac{1}{1 + e^{-w^T x}} \right)$$



Notice: that $g(z) \to 1$ as $z \to \infty$

and $g(z) \to 0$ as $z \to -\infty$ hence.

$h_w(x)$ is always bounded between 0 to 1

Interpretation of hypothesis outcome. / A probabilistic viewpoint. (**)

$h_w(x)$ = estimated probability that $y = 1$ on input x

$$= p(y = 1 \mid x; w)$$

Example:

$$x = \begin{pmatrix} x_0 \\ x_1 \end{pmatrix} = \begin{pmatrix} 1 \\ \text{tumor size} \end{pmatrix}$$

so if $h_w(x) = 0.7 \Rightarrow$ tell patient 70% chance of tumor being malignant.

$0 \leq h_w(\underline{x}) \leq 1$ $\Rightarrow$ can be assinged to a prob.

$$h_w(\underline{x}) = p(y = C_k \mid \underline{x}; \underline{w}).$$

↑ general. ~~posterior~~ probability.

probability that $y=1$, given x, parameterized by $\underline{w}$

$$\Rightarrow p(y=0) + p(y=1) = 1.$$

or, $$p(y=0 \mid x; \underline{w}) + p(y=1 \mid x; w_*) = 1 \quad \Big\} *.$$

Decision boundary.

The goal in classification is to take an input vector $\underline{x}$ and to assign it to one of K discrete classes C_k where

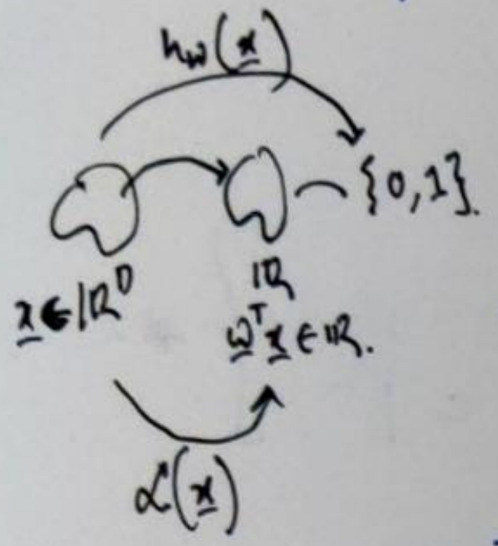
$K = 1, \ldots K$

In most common scenario, the classes are taken to be disjoint, so that each input is assigned to only one and only one class.

Input space is thereby divided into decision regions whose boundaries are called <u>decision boundaries</u>

or decision surfaces.

Linear models of classification, involves. decision surfaces. are linear functions of input variable $\underline{x}$



$$h_w(\underline{x}) : \mathbb{R}^D \longrightarrow \{C_1, C_2\}$$

$$\Rightarrow g(\underline{w}^T\underline{x}) : \mathbb{R}^D \longrightarrow \{C_1, C_2\}.$$

$$\Rightarrow g(\underbrace{w_0 + w_1 x_1 + \cdots w_D x_D}_{\text{linear function.}}) : \mathbb{R}^D \longrightarrow \{C_1, C_2\}.$$

$w_0 + w_1 x_1 + \cdots w_D x_D$ is a D-1 dimensional hyperplane. $\in \mathbb{R}^D$.

The descision surfaces are defined as.

$$g(w_0 + w_1 x_1 + \cdots w_D x_D) = \text{const.}$$

decision surface.

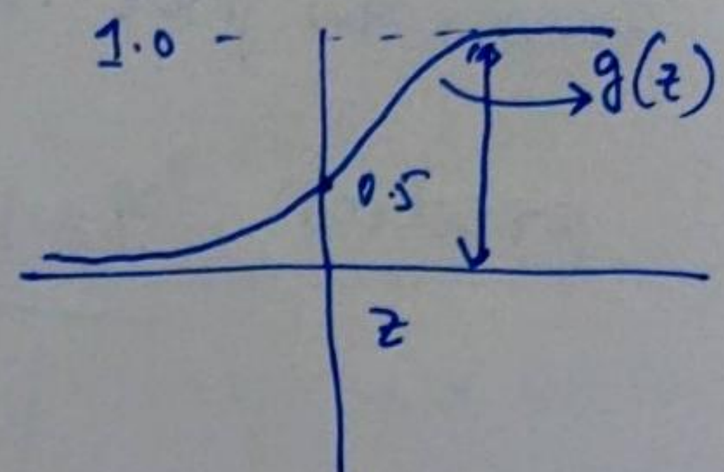
$$\Rightarrow w_0 + w_1 x_1 + \cdots w_D x_D = \text{const}$$

hence, decision surfaces are linear functions of $\underline{x}$, even if the function g is non-linear.

$h_w(\underline{x}) = g(\underline{w}^T\underline{x})$: called generalized linear model

Example: linear case: Decision boundary

$$h_w(x) = g(\underline{w}^T \underline{x})$$
$$= g(w_0 + w_1 x_1 + w_2 x_2)$$



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

Case I: If $y = 1$ for $h_w(x) \geq 0.5$ ← # Threshold definition # logical choice.

$$\Rightarrow g(\underline{w}^T \underline{x}) \geq 0.5$$

$$\Rightarrow \underline{w}^T \underline{x} \geq 0. \quad \text{[from the graph]}$$

$\Rightarrow$ ↑ decision surface / boundary

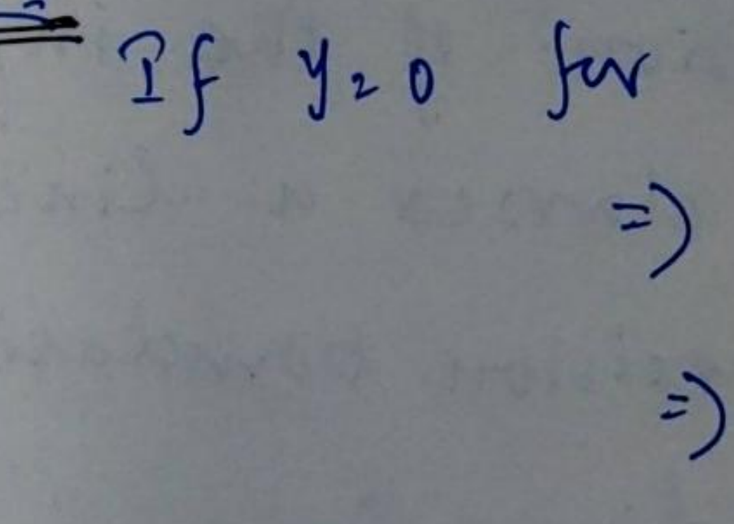
Case II: If $y = 0$ for $h_w(x) < 0.5$

$$\Rightarrow g(\underline{w}^T \underline{x}) < 0.5$$

$$\Rightarrow z = \underline{w}^T \underline{x} < 0. \quad \text{[From graph]}.$$

↑ decision surface / boundary in 2D

Example



$$\underline{w} = \begin{pmatrix} -3 \\ 1 \\ 1 \end{pmatrix}$$ ← lets assume weights.

$$h_w(\underline{x}) = g(\overset{-3}{w_0} + \overset{1}{w_1} x_1 + \overset{1}{w_2} x_2)$$

decision boundary (line in 2D).

predict As per the linear model of log classification.

predict "$y = 1$" if $\underbrace{-3 + x_1 + x_2 \geq 0}_{\theta^T x}$

→ $x_1 + x_2 \geq 3$: Eq of decision Boundary

$y = 0$ if $x_1 + x_2 < 3$.

Till now. we talked about linear transformation of input variable. $\underset{\sim}{x}$.

If the classes can be seperated by. linear transformation of the input variables $\underset{\sim}{x}$, then these classes a linearly seperable. & decision boundaries are. hyperplanes. of dim = D-1

previous example $\underset{\sim}{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \in \mathbb{R}^2$.

. hence $D = 2$.

$\dim(\text{decision boundary}) = 2 - 1 = 1$ (D-1)

line

Non-linear decision boundary

The "decision boundary" to seperate classes can be done if non-linear transformations are done on the input variables, using a a vector of basis functions $\phi(\underline{x})$.

$$h_w(\underline{x}) = g\left(w_0 + w_1 x_1 + w_2 x_2 + w_3 x_1^2 + w_4 x_2^2\right)$$

$\phi_1(x_1)$ $\phi_2(x_2)$ $\phi_3(x_3)$ ϕ_4

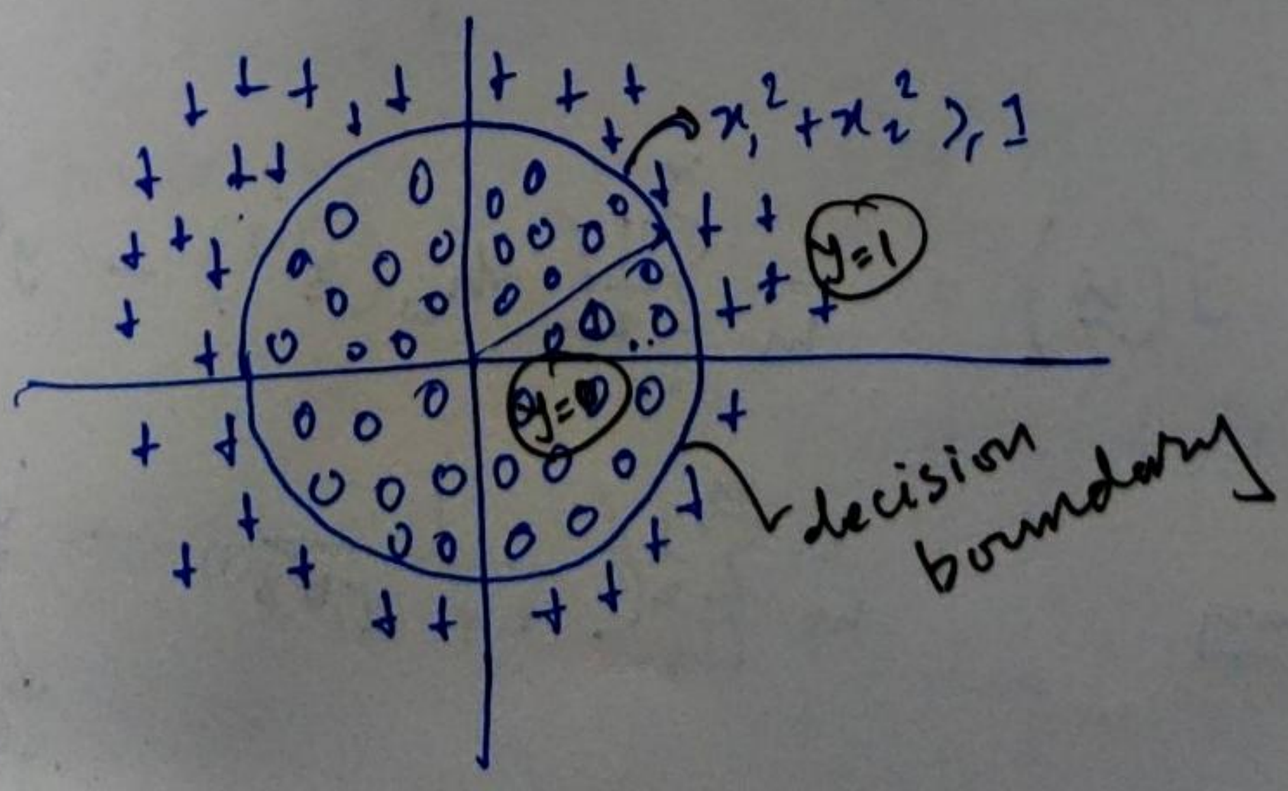
If weight vector is $w = \begin{pmatrix} -1 \\ 0 \\ \vdots \end{pmatrix}$ then.

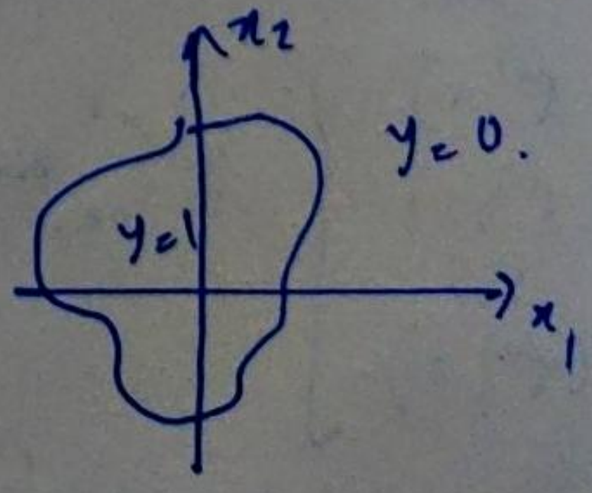
known weights.

$$h_w(\underline{x}) = g\left(-1 + 0.x_1 + 0.x_2 + 1.x_1^2 + 1.x_2^2\right)$$

predict "$y=1$" if $g(-1 + x_1^2 + x_2^2) \geq 0 \Rightarrow -1 + x_1^2 + x_2^2 \geq 0$

$$\Rightarrow \quad x_1^2 + x_2^2 \geq 1$$





Logistic regression : Cost function : How to find the weights.??

Given the training set (m - data set)

$$\{(x', y'), (x^{(2)}, y^{(2)}), (x^{(3)}, y^{(3)}), \ldots (x^{(m)}, y^{(m)})\}$$

and,

$$\underline{x} = \begin{pmatrix} x^{(1)} \\ x^{(2)} \\ x^{(3)} \\ \vdots \\ x^{(n)} \end{pmatrix} \in \mathbb{R}^{n+1} \qquad \underline{x} = \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{pmatrix} \in \mathbb{R}^{n+1}$$

, $x_0 = 1$

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

$$h_w(\underline{x}) = \left(\frac{1}{1 + e^{-\underline{w}^T \underline{x}}} \right).$$

how to choose the parameters of the model?

Now cost function can not be defined as. least squeare (like) for regression; for discriminant function $h_{\underline{w}}(\underline{x})$ for ~~finite~~ practical reason.

$$J(\underline{w}) = \frac{1}{m} \sum_{i=1}^{m} \frac{1}{2} \left(h_w(\underline{x}^{(i)}) - y^{(i)} \right)^2$$

Data.

$(\underline{x}^{(1)}, 1)$
$(\underline{x}^{(2)}, 1)$
$\vdots$
$(\underline{x}^{(m)}, 0)$

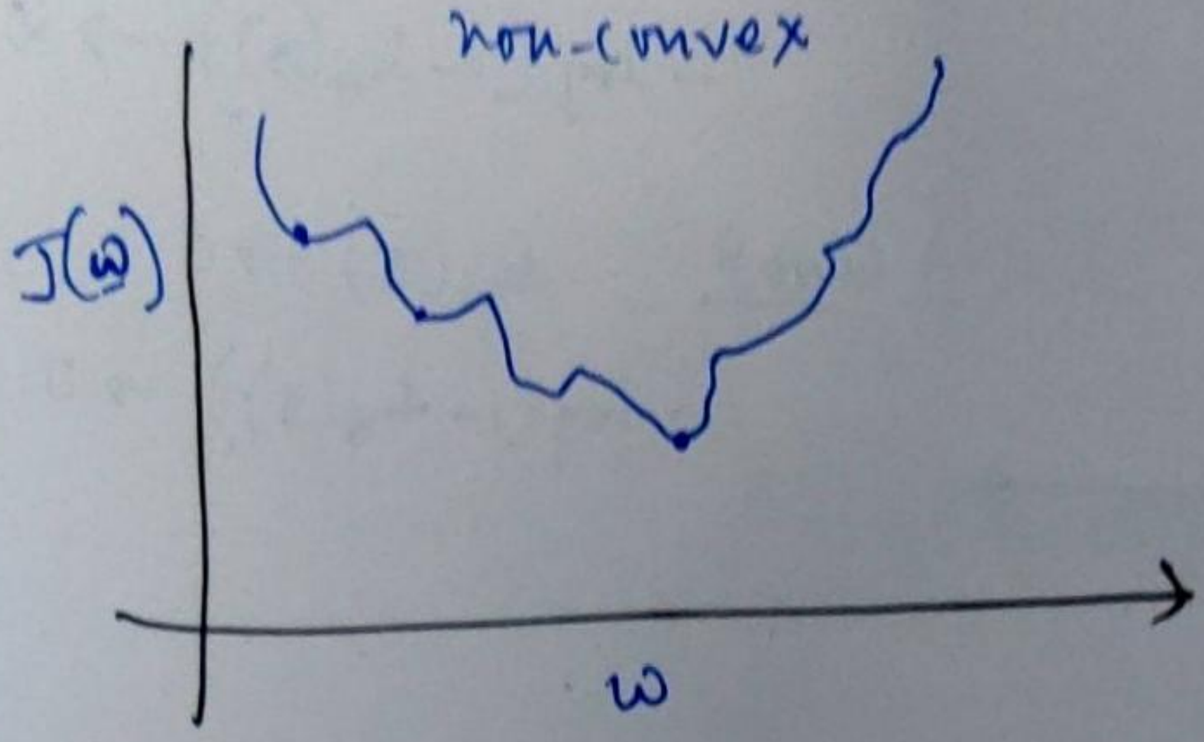
$y \begin{cases} 1 \\ 0 \end{cases}$ Binary

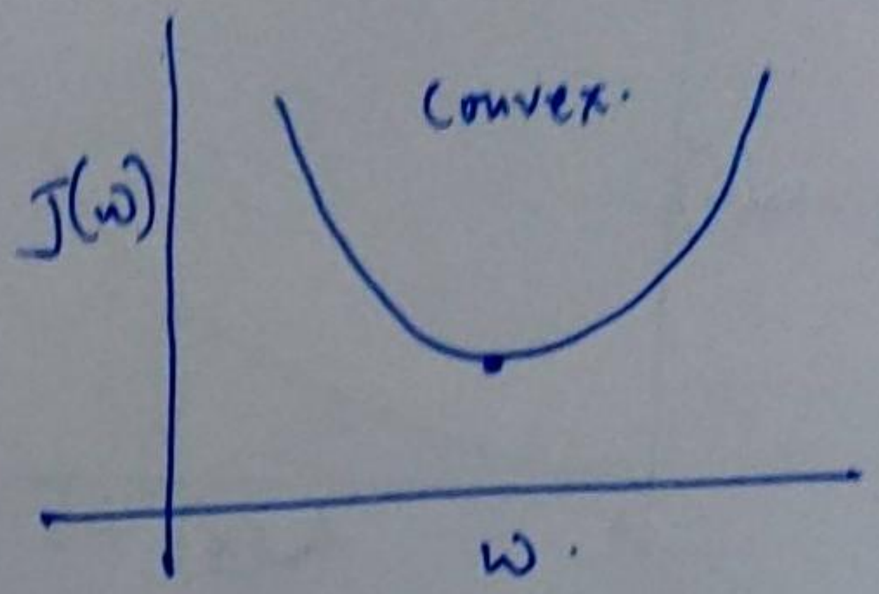
$$= \frac{1}{m} \sum_{i=1}^{m} \frac{1}{2} \left(\frac{1}{1 + e^{-\underline{w}^T \underline{x}^{(i)}}} - y^{(i)} \right)^2$$

$y^{(i)} \in \{0, 1\}$.

generates a non-convex function.

– Difficult to optimize (Non-convex)





multiple local minimum

(working at the individual term in the loss function $J(w)$, for a particular (i))

$$\underline{\underline{Cost(h_w(\underline{x}) - y)}} = \frac{1}{2}(h_w(\underline{x}) - y)^2.$$

Not working.

Need to look for an alternative cost/loss function.

– An alternative is to convert it into a monotonic function. – "log" of the squred loss.

$y - h_w(x)$.

$$Cost(h_w(\underline{x}) - y) = \begin{cases} -\log(h_{\underline{w}}(\underline{x})) & \text{if } y = 0 \\ -\log(h_{\underline{w}}(\underline{x}) - 1) & \text{if } y = 1 \end{cases}$$

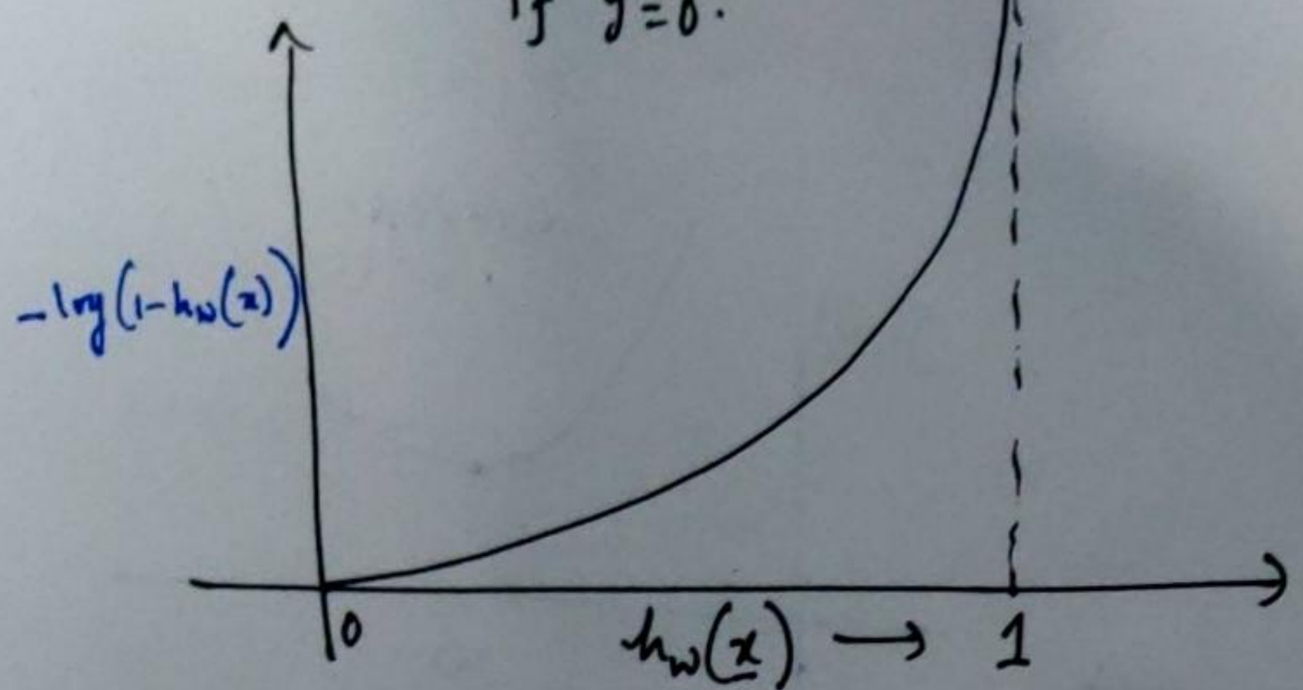
$h_w(x)$ ↗

$1 - h_w(x)$

Analyzing The cost

$$\text{Cost} = -\log\left(\frac{1-h_w(x)}{h_w(x)}\right)$$

If $y=0$.



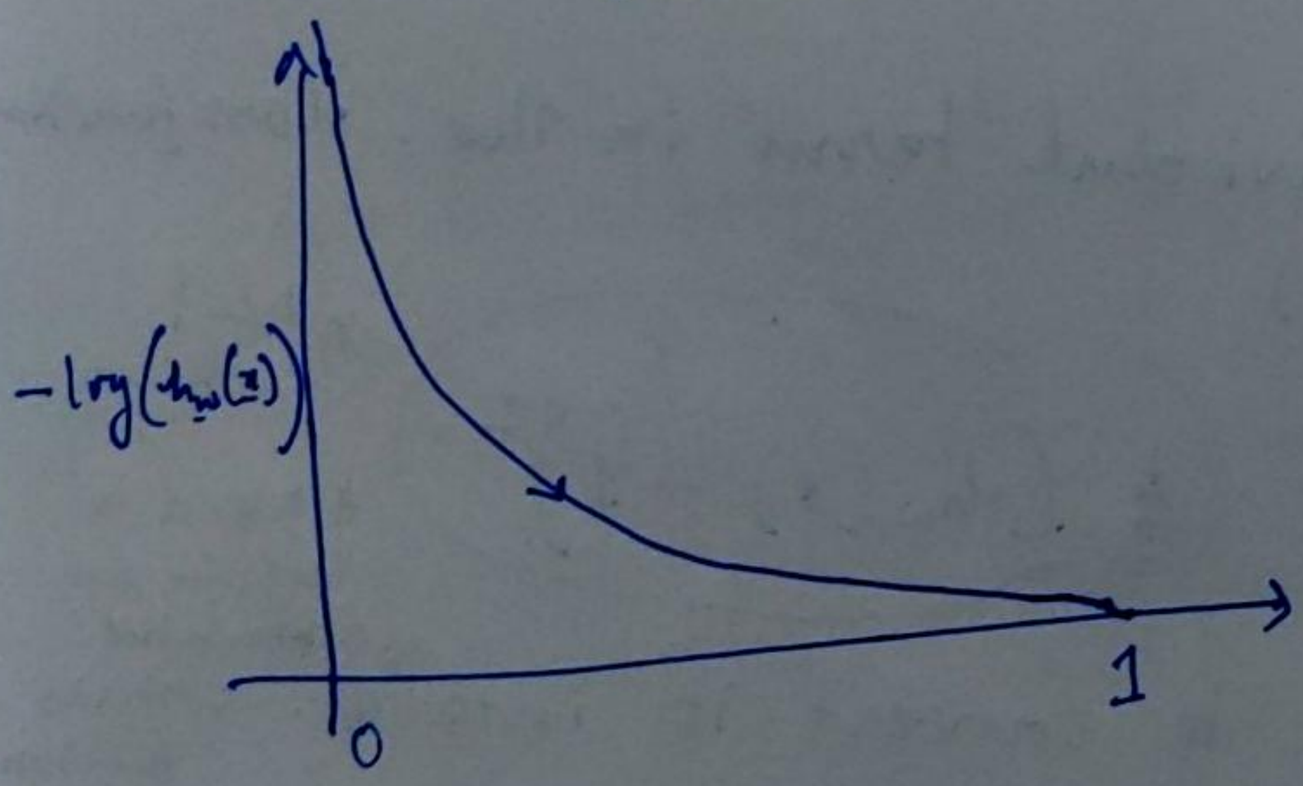
Case I: $h_w(x) \Rightarrow 1$

$-\log(1-h_w(x)) \rightarrow \infty$

Case II: $h_w(x) \rightarrow 0$

$-\log(1-h_w(x)) \rightarrow 0$

If $y=1$



Case I $h_w(x) = 1$

$-\log(h_w(x)) = 0$

Case II $h_w(x) \rightarrow 0$

$-\log(h_w(x)) \rightarrow \infty$.

We can combine the two for ~~binary classification~~ cost function for binary classification. ← Convex function.

$$\text{cost}(h_w(x), y) = -y\log(h_w(x)) - (1-y)\log(1-h_w(x))$$

This gives the same cost for $y=0$ & $y=1$ as earlier.

logistic regression : cost function

→ Binary classification

$y^{(i)}$ — $h_\theta(x^i)$

$$J(\underline{w}) = \frac{1}{m} \sum_{i=1}^{m} \text{Cos}\left(h_w(\underline{x}^{(i)}) - y^{(i)}\right)$$

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

(**)

The optimization prob. then boisl down to.

minimize $J(\underline{w})$ or $\underline{w}^* = \text{argmin}\left(J(\underline{w})\right)$.

please note $J : \mathbb{R}^n \longrightarrow \mathbb{R}$.

output $h_w(\underline{x}) = \left(\frac{1}{1+e^{-\underline{w}^T\underline{x}}}\right)$ or $\underline{p(y=c_k \mid x; \underline{w})}$ $k=2$.

Algorithms for optimization:

Type I: Gradient descent. (Batch).

$$\underline{w}^{k+1} = \underline{w}^k - \alpha \underline{d}^k.$$

where $\underline{d}^k = -\nabla_{\underline{w}} J(\underline{w})$

$$= -\begin{pmatrix} \frac{\partial J}{\partial w_1} \\ \frac{\partial J}{\partial w_2} \\ \frac{\partial J}{\partial w_j} \\ \frac{\partial J}{\partial w_n} \end{pmatrix} = -\frac{\partial J}{\partial w_j} \quad \forall\ j = 1, 2, \cdots n.$$

$$= \frac{1}{2}\sum_{i=1}^{m}\left(h_w(x^i) - y^i\right)x_j^i$$

hence Loop over k

$$\underline{w}^{k+1} = \underline{w}^{k} - \alpha \left[\frac{1}{m} \sum_{i=1}^{m} \left(h_w(\underline{x}^i) - y^i \right) \underline{x}^i \right]$$

Total sum over data set

does this change anything?

Stop until converges.

Stochastic gradient descent:

Loop over k

for $i = 1, m$ (loop over data).

$$\underline{w}^{k+1} = \underline{w}^{k} - \alpha \left(h_w(\underline{x}^i) - y^i \right) \underline{x}^i$$

end.

Stop until converges.

If not written in vectorized form $\underline{w}$ etc.

Loop over k

for $i = 1, m$ (loop over data)

for $j = 1, n$ (loop over # of features) + 1)

$$w_j^{k+1} = w_j^{k} - \alpha \left(y^i - h_w(\underline{x}^i) \right) x_j^i$$

end

end

stop until converges.

Other optimization algorithms.

Discuss. i) Gradient descent

ii) Conjugate gradient

iii) BFGS

iv) L-BFGS.

Derivation of cost function (~~Max likelihood approach~~)

(**) Another look of deriving the cost function from "maximum likelihood approach"

lets assume: $p(y=1 \mid \underline{x}; w) = h_w(\underline{x})$

$p(y=0 \mid \underline{x}; w) = 1 - h_w(\underline{x})$

we can assume a probability distribution.

for bernoulli's

$$\boxed{p(y \mid \underline{x}; \underline{w}) = (h_w(\underline{x}))^y (1 - h_w(\underline{x}))^{1-y}}$$

Now, from this probability distribution the Likelihood function is defined as.

$$L(\underline{w}) = p(\underline{y} \mid \underline{X}; \underline{w})$$

$$= \prod_{i=1}^{m} p(y^i \mid \underline{x}^i; \underline{w}) \quad \text{[assuming all the data pts are indep]}$$

$$= \prod_{i=1}^{m} (h_w(\underline{x}^i))^y (1 - h_w(\underline{x}^i))^{1-y}$$

To maximize the likelihood. (or log of the likelihood

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

or minimize the $\underline{-\log(L(\underline{w}))}$

Our optimization problem is posed as.

$$\min_{\underline{w}} J(\underline{w}) = \min_{\underline{w}} -\log(L(\underline{w}))$$

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

Now. to find $\nabla_{\underline{w}} J(\underline{w})$ we need. (for i)

$$\frac{\partial J}{\partial w_j} = \frac{\partial}{\partial w_j}\left[y^{(i)} \log g(\underline{w}^T \underline{x}^{(i)}) + (1-y^{(i)}) \log(1-g(\underline{w}^T \underline{x}^i)) \right]$$

$$= \left[y^{(i)} \frac{1}{g(w_j^T \underline{x}_j)} g'(\underline{w}^T \underline{x}) + (1-y^{(i)}) \frac{1}{1-g(\underline{w}^T \underline{x})} (-g'(\underline{w}^T \underline{x})) \right] x_j$$

now we know $g'(z) = g(z)(1-g(z))$

Substituting we get

$$\frac{\partial J}{\partial w_j} = \left[y^{(i)}(1-g(\underline{w}^T \underline{x})) - (1-y^{(i)}) g(\underline{w}^T \underline{x}) \right] x_j$$

$$= \left(y^{(i)} - h_w(\underline{x}^i) \right) x_j \qquad \forall\ j = 1, \cdots m$$

The update rule turns out to be same as in "regression prob"

$h_w(\underline{x}^{(i)})$ is now a nonlinear function of $\underline{w}^T \underline{x}^i$

GLM: generalized linear models. discusses the models which are linear with respect to the "features" x_j

Note: If we combine all $j = 1, 2, \ldots m$ and vectrize. the expression

$$\begin{pmatrix} \frac{\partial J}{\partial w_1} \\ \frac{\partial J}{\partial w_2} \\ \vdots \\ \frac{\partial J}{\partial w_m} \end{pmatrix} = \left(y^{(i)} - h_w(\underline{x}^{(i)}) \right) \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{pmatrix}^i$$

$$\Rightarrow \boxed{\nabla_w J = \left(y^{(i)} - h_w(\underline{x}^{(i)}) \right) \underline{x}^{(i)}}$$

i is the data point.

Perceptron learning Algorithm.

~ Perceptron algorithm has some historical significance.

~ The logistic regression method can be modified to change the nonlinear mapping $g(\underline{w}^T \underline{x} + w_0)$ as

$$g(\underline{w}^T \underline{x} + w_0) = g(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{if } z < 0 \end{cases}$$

~ If we then let $h_w(\underline{x}) = g(\underline{w}^T \underline{x} + w_0)$ as before, but using this modified definition of g, we can use the update rule.

$$\boxed{w_j := w_j + \alpha \left(y^{(i)} - h_w(\underline{x}^{(i)}) \right) x_j^{(i)}}$$

This is "perceptron learning algorithm"

Note: In 1960, this perceptron was argued for how individual neurons in the brain work. Inspite of its simplicity, "perceptron algorithm" is difficult to endow a "meaningful probabilistic interpretations" OR derive "perceptron" as a "maximum likelihood estimation algo".

(3A) Week.

Logistic Regression

"Multiclass classification"

"One-vs-all" / On-vs-rest. (Ref Andrew NG).

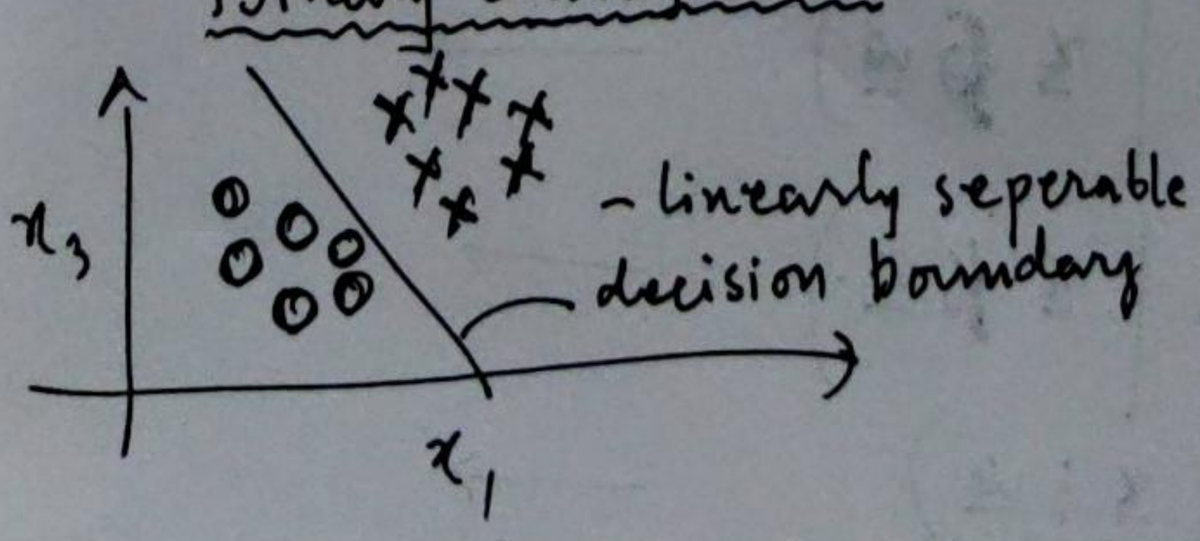
Email foldering/tagging: Work, family, Friends, Hobby.
$y=C_1$ $y=C_2$ $y=C_3$ $y=C_4$

Medical diagram: Not ill, cold, Flu
$y=C_1$ $y=C_2$ $y=C_3$

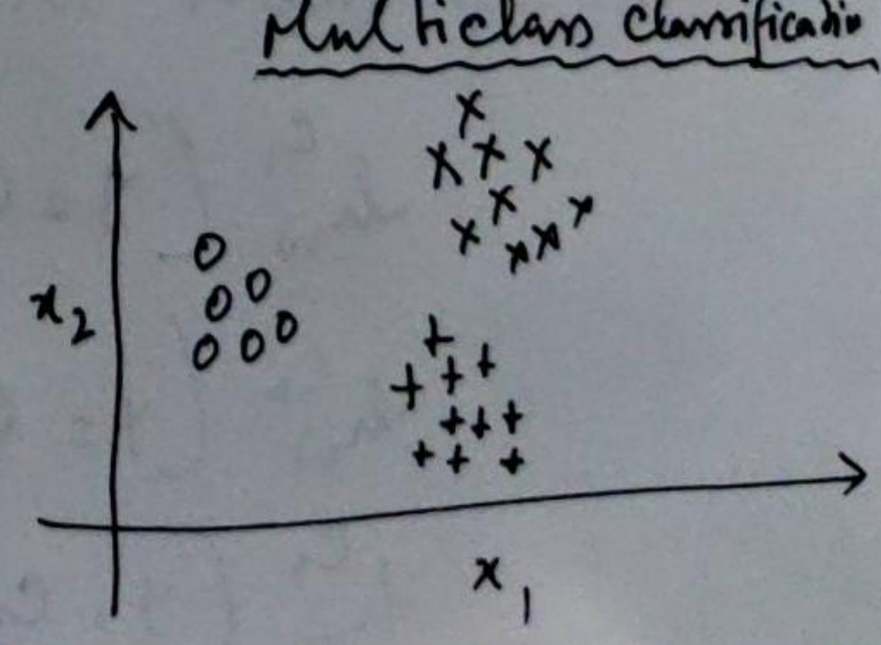
Weather: Sunny, Cloudy, Rain, Snow.
$y=C_1$, $y=C_2$, $y=C_3$, $y=C_4$

When "data" can be labeled into multiple independent class, we need multi-class classification algorithm.

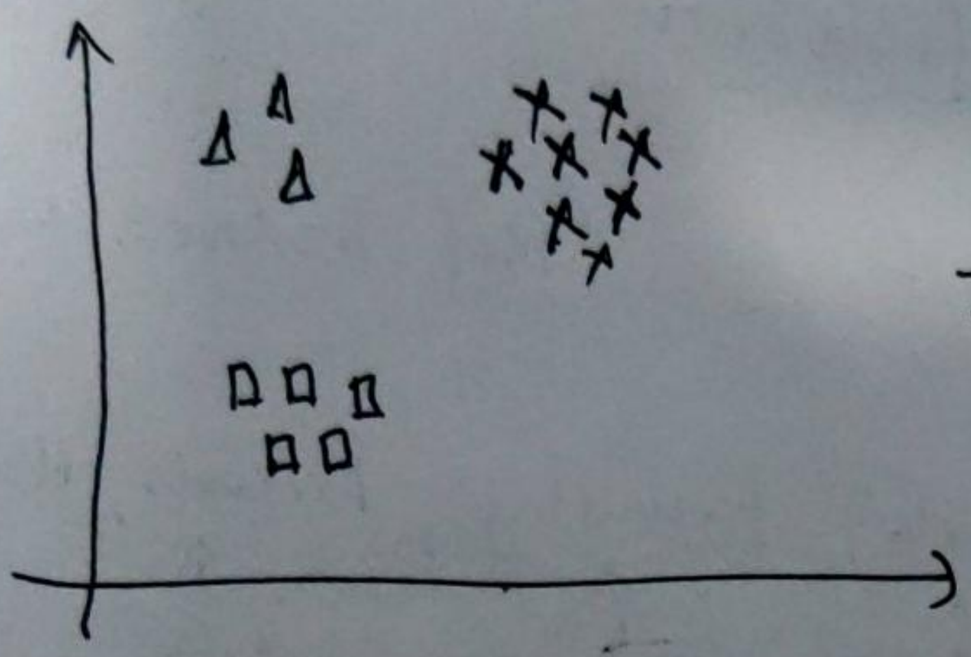
Binary classification

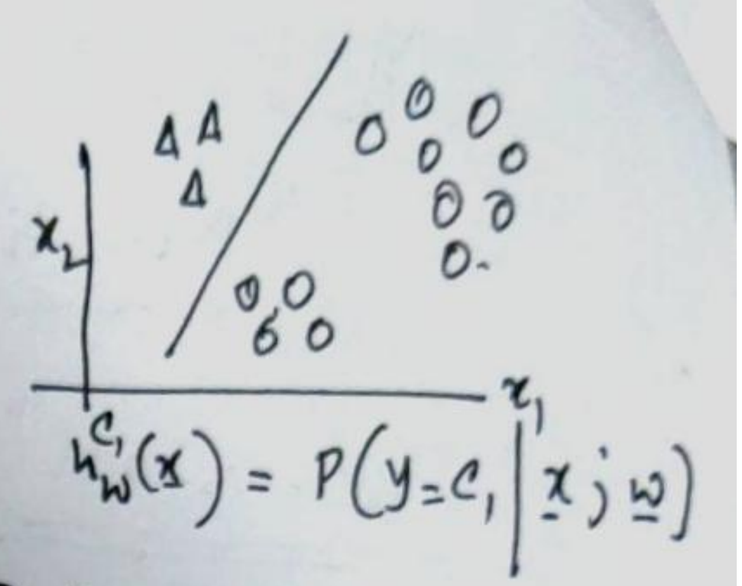


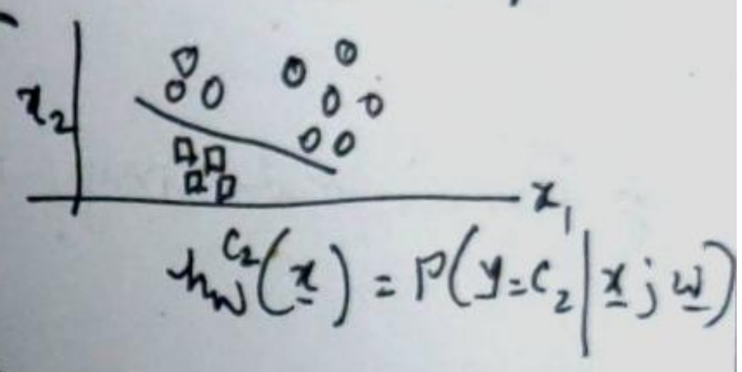
Multiclass classification

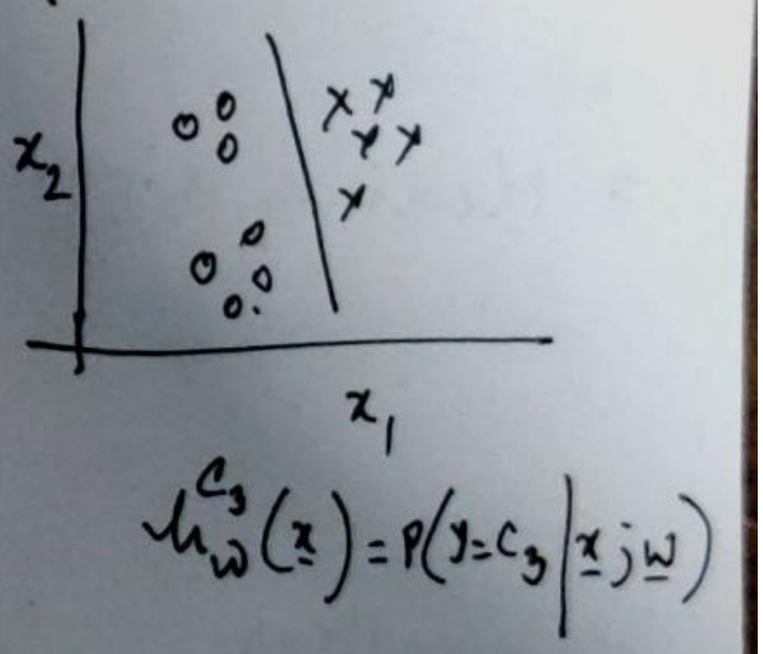


One-vs-all (One-vs-rest)









During the training phase

- Define binary classification probability density for each class C_K, $K \in \{0, 1, 2, \cdots K\}$.
- Find the optimized weights.
- to define probdensity by class
- conditional prob posterior prob

$$h_w^{C_K}(x) = P(y = C_K \mid \underline{x}; w) = \text{[illegible]}$$

Training

- So if there are 3-classes we have

$$h_w^{C_1}(y = C_1 \mid \underline{x}; \underline{w}) \longrightarrow$$

$$h_w^{C_2}(y = C_2 \mid \underline{x}; \underline{w}) \longrightarrow$$

$$h_w^{C_3}(y = C_3 \mid \underline{x}; \underline{w}) \longrightarrow$$

Prediction

For prediction purpose., for a new input & feature, $\underline{x}$, pick the class C_k, that maximize

$$\max_{C_k} \; h_w^{C_k}(\underline{x})$$

Summary: - Binary classification applied one at a time to different classes.

- Training set is ~~organized~~ arranged in such a way to. train the posterior prob. function. of the classes.