STATISTICAL METHODS IN A.I

Reference Books:

- Pattern Classification – Duda Hart and Stork
- Machine Learning – A Probabilistic Perspective by Kevin Murphy
- Neural Networks – Simon Haykin
- AI – A Modern Approach – Russell and Norwig.

Grading Schemes –

Mini Project – 20%
Homeworks – 30%
Mid Terms – 15% + 15%
Final – 20%

Tasks in AI →
- Sensing (Perception)
  - Cameras
  - Microphones
  - Touch, promixity
- Recognition – where am I?, what do I see around
- Reasoning
- Acting

Recognition ("Re" + "Cognize") – Identification of a pattern as a member of a category we all know, or familiar with.
→ associating it with a class of objects } "examples"

Pattern – an entity vaguely defined, that could be given a "name"

→ Same sample might be classified as different classes.

"Class" : Collection of "similar" objects
→ defined by class samples

Pattern Recognition – inferring a generality from a few examples

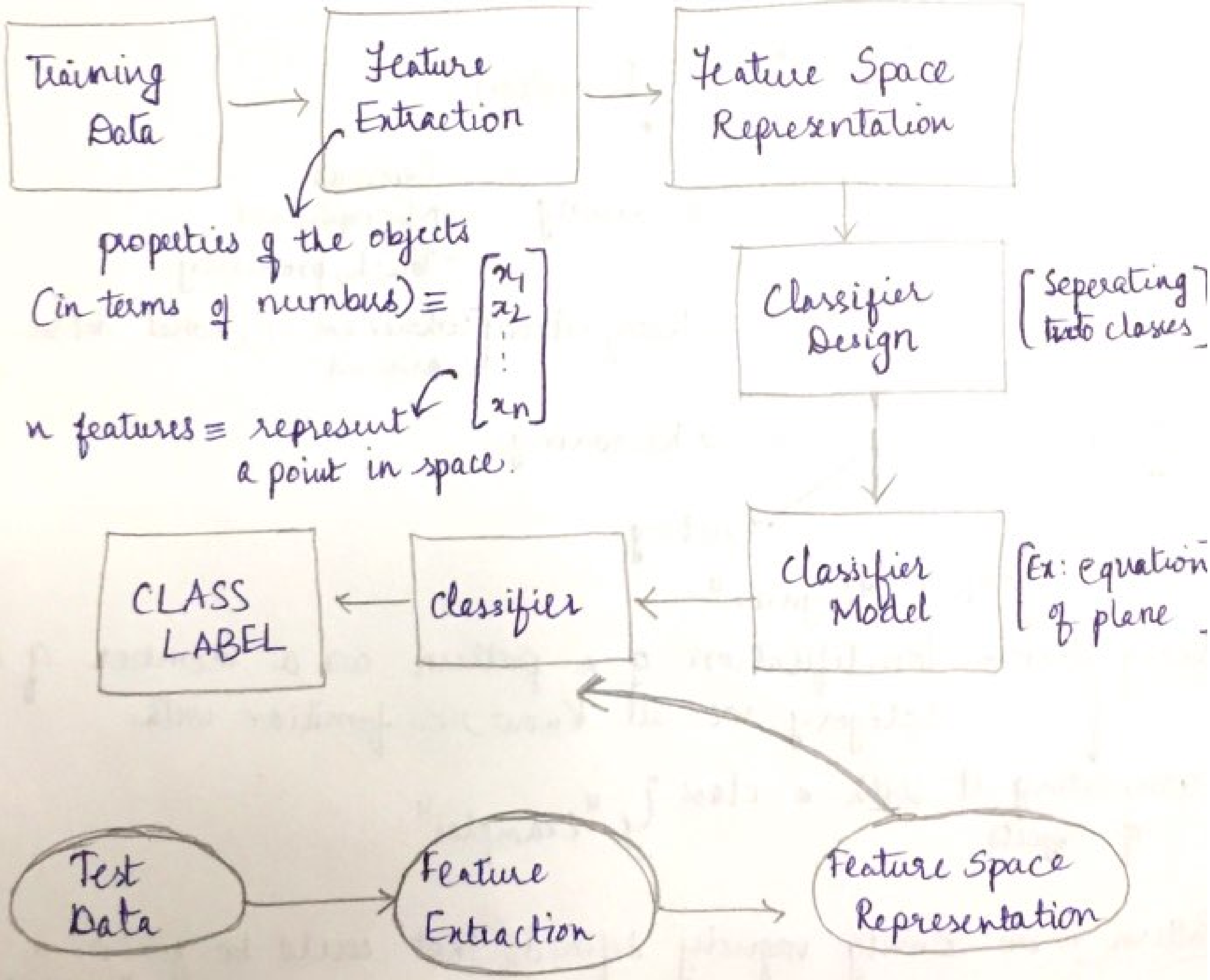
↳ system "learns" to tell whether or not an object belongs to a class.

* class
  - ↗ Inter-class variability
  - ↘ Intra-class variability

  ↓ represented as $[\omega_1, \omega_2, \ldots \omega_n]$

"train/teach the machine with a "large dataset"."

Pattern Recognition Process :-



* Class – modeled by a $\underbrace{\text{probability density function}}$, $P(x)$

↓ class-conditional $\equiv P(x/\omega_1)$

Gaussian Distribution $\equiv$ $P(x) = \frac{1}{\sqrt{2\pi}\,\sigma} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \sim N(\mu, \sigma^{2})$

Mathematics :- Revision

→ **Linear Algebra :**

$f(x) \equiv f(\alpha x + \beta y) = \alpha f(x) + \beta f(y)$ for a function to be linear

$$a_{11} x_1 + a_{12} x_2 + a_{13} x_3 = b_1$$
$$a_{21} x_1 + a_{22} x_2 + a_{23} x_3 = b_2$$
$$a_{31} x_1 + a_{32} x_2 + a_{33} x_3 = b_3$$

Basic Representation $\equiv$ matrices and vector form

$$\underset{A}{\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}} \underset{X}{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}} = \underset{B}{\begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}} \qquad AX = B$$

(3D-vector)

$|x|$ absolute value.

(Norm/Length)
Size of Vector $\equiv$ $X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$

$\|x\|$

$$\|x\|_2 = \sqrt[2]{x_1^2 + x_2^2 + x_3^2}$$

$$\|x\|_p = \sqrt[p]{|x_1|^p + |x_2|^p + |x_3|^p + \dots |x_k|^p} \equiv L_p\text{-norm}.$$

L_p-norm : $\|x\|_p = \sqrt[p]{\left(\sum_{i=1}^{n} |x_i|^p\right)}$

doesn't have any physical significance.

Total distance covered $\equiv$ L_1-norm $\Rightarrow \|x\|_1 = \sum_{i=1}^{n} |x_i|$

L_0-norm $\equiv \|x\|_0 = \sum_{i=1}^{n} |x_i|^0$ number of non-zero dimensions/entries in the vector.

Scanned by CamScanner

L_∞- norm $\equiv$ $||x||_\infty = |$ Maximum value of the entries $|$

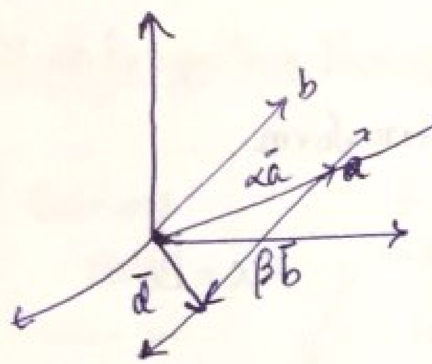
for Ex: $\begin{bmatrix} 1 \\ 2 \end{bmatrix}$

$L_1 = 3$
$L_2 = 2.24$
$L_3 = 2.08$
$L_4 = 2.03$
$\vdots$
$L_{10} = 2.00019$

$\Rightarrow$ Span of a set of vectors -

linearly independent $\equiv$ when any combination of two vectors donot result in the third vector

for Ex: $c = \alpha a + \beta b$ (c is dependent on a, b)



$\bar{d} = \bar{a}\alpha + \bar{b}.\beta$

If the vectors can be used in expressing the space

are linearly independent, then they can span the whole space.

$$M = \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{bmatrix}$$

Det of $M = |M| \neq 0$, if rows are linearly independent

form a basis

if Rank = 2, then out of 3, only 2 vectors are linearly independent & the other one is dependent on the first two.

dimensionality of the subspace spanned.

as in feature $\Rightarrow$ typically, vectors are "column vectors".

$\Rightarrow$ Orthonormal Basis — Normal + Perpendicular basis (linearly independent)

$|$Norm$| = 1$

perpendicular (90°) [Dot product of orthogonal vectors = 0]

$a \perp b \equiv a.b = ab\cos\theta = 0$

Scanned by CamScanner

⇒ <u>Dot Product</u> ≡ $\begin{array}{cc} X & Y \\ \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} & \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} \end{array} \Rightarrow \overset{X.Y}{x_1 y_1 + x_2 y_2 + x_3 y_3} = X^T.Y$ (Inner Product)

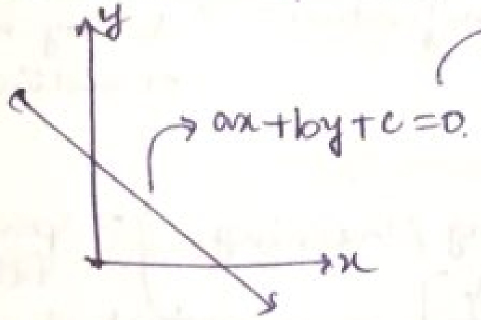
$$XY^T = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} [y_1\ y_2\ y_3] = \begin{bmatrix} x_1 y_1 & x_1 y_2 & x_1 y_3 \\ x_2 y_1 & x_2 y_2 & x_2 y_3 \\ x_3 y_1 & x_3 y_2 & x_3 y_3 \end{bmatrix} \text{(outer Product)}$$

⇒ <u>Cross Product</u> ≡ $(X,Y) = X \times Y = \begin{vmatrix} i & j & k \\ x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{vmatrix}$

$\alpha = \bar{X} \times \bar{Y}$

perpendiculer to both X & Y

⇒ <u>Representation of Line</u> :-



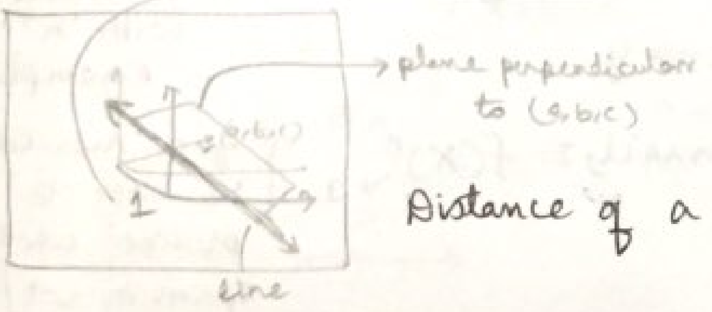
is it a vector or a Line?

[3D vector]

$$\underset{\omega}{\begin{bmatrix} a \\ b \\ c \end{bmatrix}} \quad \underset{x}{\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}} \qquad \omega^T x = 0$$

how do you relate both?

Space (augmented vector)



If ω is scaled, the plane doesnt change and so does the line.

Distance of a point from a plane

Normalise ω and take the dot product.

Linear Transformation ⇔ Matrix Multiplication

by inverting the matrix, we can invert the transformation

⇒ <u>Eigen Values and Eigen Vectors</u>

$M = u D v^T$ [λ] eigen values

orthonormal vectors

rows = eigen vectors

Scanned by CamScanner

physical entity — represented as a vector

$X_1 \ldots \quad X_n \Rightarrow$ vectors

with $d \equiv$ dimension

where $x_i \in \mathbb{R}^d$

$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$ d — dimension

observations

physical entities are represented differently in different domains.

<u>Main Problem</u>

<u>For Ex:</u> $\boxed{x_i \longrightarrow y_i} \in \{0,1\}$ (Say, spam or not)

$\{-1,+1\}$

$\{1,2,\ldots k\}$ (say, professional, personal, ... etc)

K-class.

$y_i \in R \equiv$ Real, continuous measure

It is called as "<u>Regression</u>"

classification $\longrightarrow$ Can be a binary one or multiclass.

representation of an email $\in \mathbb{R}$

spam or not

Given, (x_i, y_i), $i = 1, \ldots N$.

examples for learning/training

[where $\underline{x_i \in \mathbb{R}^d}$, $y_i \in \{0,1\}$]

$\equiv$ Spam filter to be built with "n" examples

Find a function, $f: x \rightarrow y$

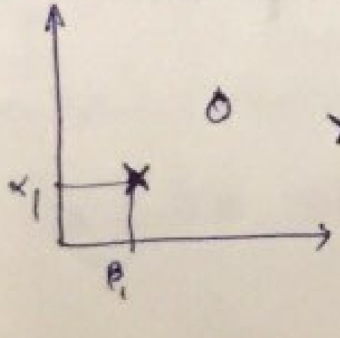
So when given a new email $\equiv f(X)$ $\nearrow 0$ $\searrow 1$

X

[you should be able to predict whether spam or not]

x_1	x_2	x_3	
$\begin{bmatrix} \alpha_1 \\ \beta_1 \end{bmatrix}$	$\begin{bmatrix} \alpha_2 \\ \beta_2 \end{bmatrix}$	$\begin{bmatrix} \alpha_3 \\ \beta_3 \end{bmatrix}$	$\rightarrow$ 2 features with different values
0	1	0	(spam / not spam)

can be represented on a plane.



$\longrightarrow f() \equiv f(x) = w^T x + b$

$\begin{bmatrix} x^1 \\ x^2 \end{bmatrix}$

$f(x) = w_1 x^1 + w_2 x^2 + b$

Now the problem is reduced to "finding the value of w_1, w_2, b".

→ Find w_1, w_2, b ?

↓

OPTIMIZATION problem ≡ Find w that best solves the problem.

↓

we need an "objective function"

Data (x_i, y_i) pairs

↓

Problem ≡ Find best w] which minimizes

↓

Objective Function ≡ Error/Loss

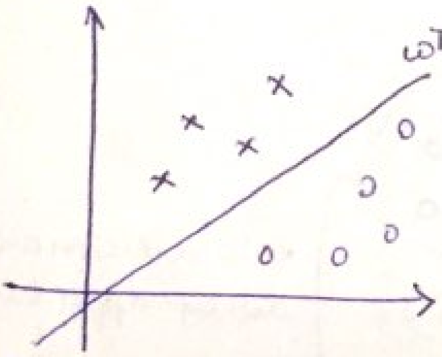
changing the loss function can also changes the optimization Algorithm. ↑ because $N >>$

$f(x) = w^Tx + b$.

→ Regression

→ Classification.

Representation ≡



x — Spam

o — Not Spam

Case ①

$f(x) = w^Tx + b = 0$.

Spam if $f(x) < 0$

Not Spam if $f(x) \geq 0$

↓

Discriminent function

Case ②:

$$f_1(x) = w_1^T x + b_1$$
$$f_2(x) = w_2^T x + b_2$$

$f_1 - f_2 > 0$.

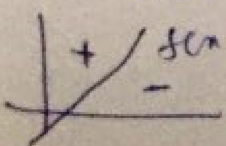
Spam if $f_1(x) > f_2(x)$

Not Spam if $f_1(x) \leq f_2(x)$

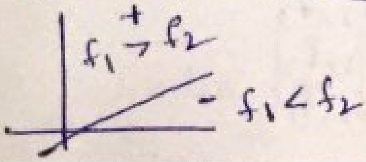
$$f_1 - f_2 = (w_1 - w_2)^T x + b_1 - b_2$$
$$= w_3^T x + b_3.$$

In the 1st case ≡

$f(x)$ ≡ function discriminates spam from not spam



In the 2nd case ≡



? (when to use this approach)

Scanned by CamScanner

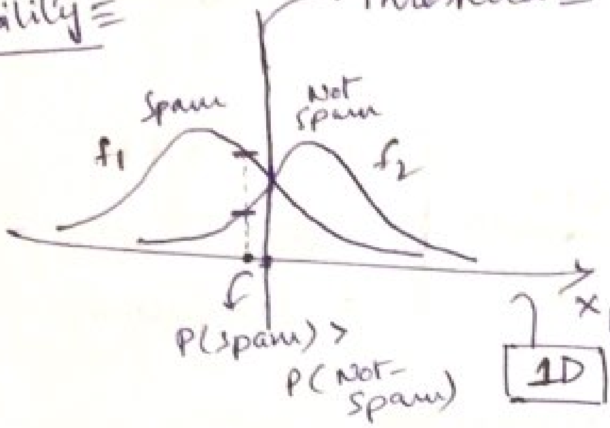
the Second case: Assume $f_1(x) \equiv$ Probability of being spam.

$f_2(x) \equiv$ Probability of being not spam

both the viewpoints are equivalent in this case.

Probability $\equiv$

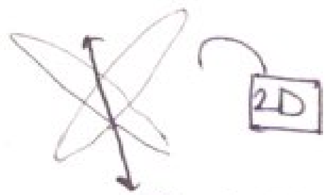
Threshold $\equiv$ P(spam) = P(Not spam)



So $\equiv$ Spam

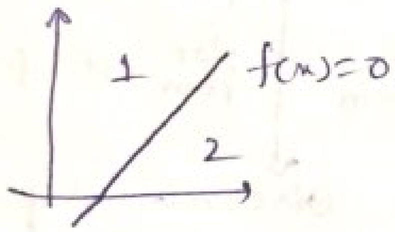
if $x < T \equiv$ Spam

$\geq T \equiv$ Not Spam.



These both viewpoints go in parallel. ✓ (probabilistic and Non probabilistic Approach)

2-Class Classification Problem:

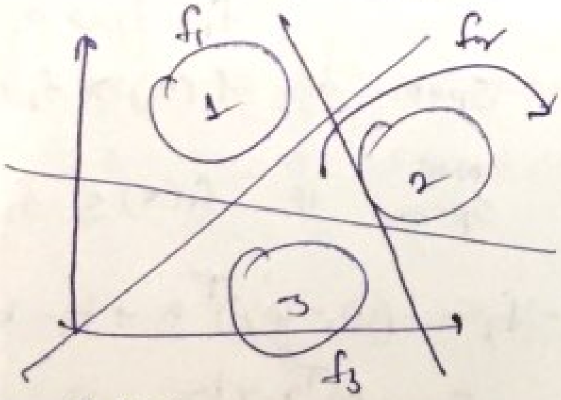


1 if $f(x) > 0$

2 if $f(x) \leq 0$

this decision rule doesnt apply here.

What about a 3-class classification?



what about this sample space.

"No-man"

"multi-class classification problem"

Decision Rule:

1: $f_1 > f_2, f_3$

2: $f_2 > f_1, f_3$

3: $f_3 > f_2, f_1$

Output – Discrete (Not Continuous)

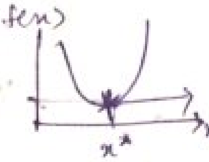
↓

becomes an discrete optimization problem.

Loss function → Optimization Problem

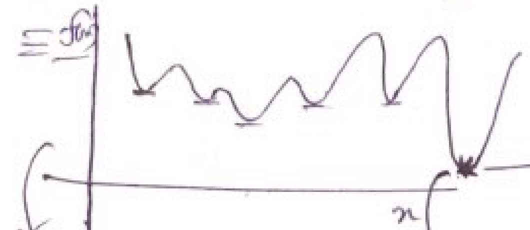
Optimization

→ Convex ≡



single unique minima.

→ Non-convex ≡



maybe the best minima to get

we can solve it.

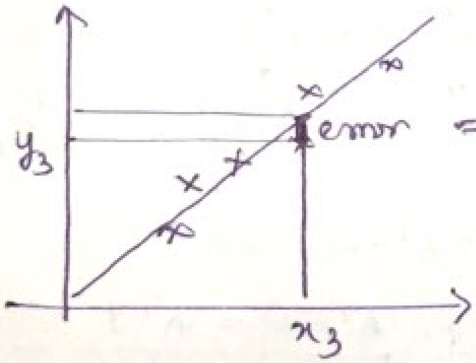
how to find?

— brute force search.

Regression ≡

$(x_1, y_1) \dots (x_n, y_n)$ where $y_i \in R$.

$f(x) = w^T x + b$



error ⇒ $\varepsilon_3 = w^T x_3 + b - y_3$

$$\text{Error} \equiv \Sigma = \sum_{i=1}^{N} \Big[y_i - \underbrace{(w^T x_i + b)}_{\text{predicted value by the function}} \Big]^2$$

⇓

$\min\limits_{w,b} \Sigma$

⇓

$$\min_{w} \Big(\sum_{i=1}^{N} (y_i - w^T x_i)^2 \Big)$$

can be +ve / -ve [above or below the line]

Minimum Square Error (MSE)

$$\overset{w}{\begin{bmatrix} o \\ o \\ o \end{bmatrix}}^T \overset{x}{\begin{bmatrix} o \\ o \\ o \end{bmatrix}} + b$$

How to remove b ⇓

$w^T x$

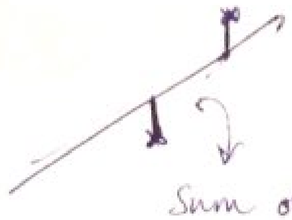
⇓

$$\overset{w'}{\begin{bmatrix} o \\ o \\ o \\ b \end{bmatrix}}^T \overset{x'}{\begin{bmatrix} o \\ o \\ o \\ 1 \end{bmatrix}}$$

x is augmented with 1.

Sigma

$$x_i = \begin{bmatrix} x_{i1} \\ x_{i2} \\ x_{i3} \\ \vdots \\ x_{id} \end{bmatrix} \qquad \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_n^T \end{bmatrix} \begin{bmatrix} w \end{bmatrix}$$



Sum of Square error.

$N \times 1$ $\qquad$ $N \times d$ $\qquad$ w $d \times 1$

$(N \times (d+1))$ with the augmentation.

(Predicted Y) $Y = AW$

$N >>> d$

$$\sum_{i=1}^{N} (y_i - w^T x_i)^2 \equiv \left(\begin{matrix}\text{in metrix}\\ \text{form}\end{matrix}\right) \equiv [Y - AW]^T [Y - AW]$$

Sigma is gone

$(N \times 1)^T \quad (N \times 1) \equiv (1 \times 1)$

To minimize this equation $\Longleftarrow Y^T Y + (AW)^T AW - 2(AW)^T Y$

[Differentiate and equate = 0.] $\Longrightarrow \frac{\partial}{\partial w}\left[Y^T Y + (AW)^T AW - 2(AW)^T Y\right] = 0.$

$$= \frac{\partial}{\partial w}\left[Y^T Y + w^T A^T A w - 2 w^T A^T Y\right] = 0.$$

$0 \qquad 2A^T A w \qquad 2A^T Y$

$$2A^T A w - 2A^T Y = 0.$$

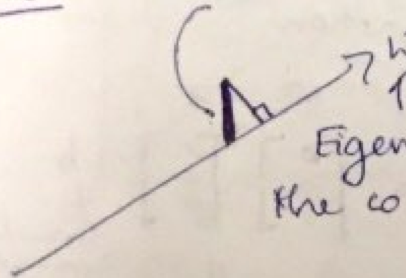
$$A^T A w = A^T Y$$

Tom Minka ≡ Metrix Differentiation.

$$\Longrightarrow \boxed{w = (A^T A)^{-1} A^T Y}$$

Issues with Error:

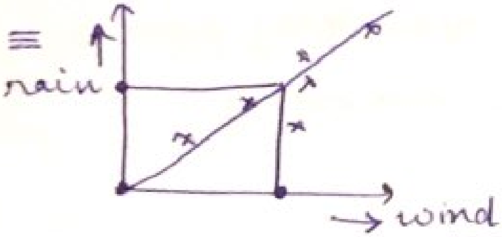
we took this as error



> Eigen vector of the co-variance metrix } for minimizing the perpendicular distance

> A Size $\equiv N \times d$ $\longrightarrow$ could be extremely large $\equiv$ "Gradient Descent"

Examples of Regression ≡



$Y = f(x)$
$= w^T z$ $\begin{bmatrix} x \\ 1 \end{bmatrix}$
$\begin{bmatrix} x^2 \\ x \\ 1 \end{bmatrix} = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix}$

— not restrictive $w_1 x^2 + w_2 x + w_3 = 0$
— due to augmentation
(not passing thru origin ✓)
— New vectors = Non linear functions embedded in $w^T x = 0$.

representation of x

$Y = ax + b$

$\Rightarrow X = \begin{bmatrix} x \\ 1 \end{bmatrix}$ $\begin{bmatrix} a \\ b \end{bmatrix}^T \begin{bmatrix} x \\ 1 \end{bmatrix}$

(augmented vector)

$y = w_1 x_1 + w_2 x_2 + w_3$

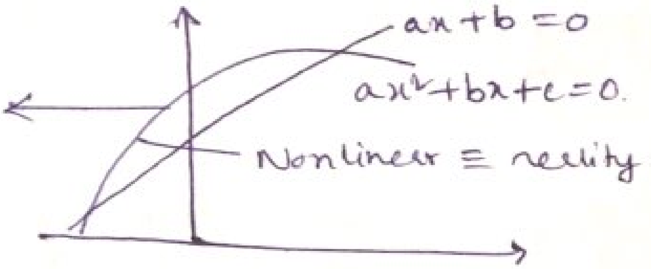
$\Rightarrow X = \begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix}$ $\begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix}^T \begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix}$ ⟶ this even works for non-linear

line fitting

can be represented as ≡

$X = \begin{bmatrix} x^2 \\ x \\ 1 \end{bmatrix}$ $w = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$

↳ $w^T X = 0$.



* Main goal:- Find "$\underline{w}$", for a function of the form ≡ $\underline{w}^T x$

Not restrictive

$\Rightarrow Y = A w$ ⇒ To find w ≡

$N \times 1$, $N \times d$, $d \times 1$

$w = A^{-1} Y$

(doesnt work.

as long as A is not singular and square matrix ($N = d$)

→ $x \to z$ can be mapped

$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}_x \Rightarrow \begin{bmatrix} x_1^2 \\ x_1 x_2 \\ x_2^2 \\ x_1 x_3^2 \\ \vdots \end{bmatrix}_z$ these are not independent.

$w = \boxed{(A^T A)^{-1} A^T} Y$

pseudo inverse of A.

approach ≡ Minimum Square Error

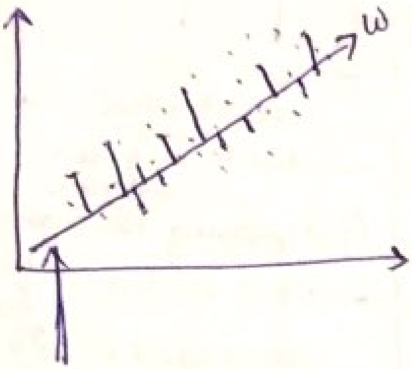
→ Doesnt minimize perpendicular (orthogonal) dist.

* $(A^T A)^{-1}$ ≡ computational storage

— Data comes incrementally (data is always not offline)

w — needs to be incrementally updated.

⇒ Orthogonal Distance Minimizing → find P, a point
→ find w, the direction.



$x_i \Rightarrow i = 1 \ldots N$ (set of points)

Let a point P be the point ~~such~~ which has the least square error distance from all the points.

$$E = \sum_{i=1}^{N} |x_i - P|^2 = \sum_{i=1}^{N} [x_i - P]^T [x_i - P]$$

eigenvector from the covariance matrix. (passing thru the mean).

For minimum, $\frac{\partial E}{\partial P} = 0$

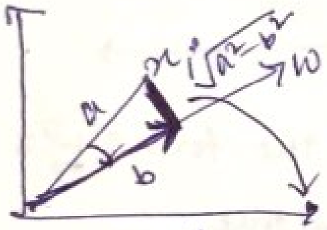
$$\frac{\partial}{\partial P} \sum_{i=1}^{N} \left(\underbrace{x^T x}_{0} + \underbrace{P^T P}_{2P} - \underbrace{2 P^T x_i}_{2x_i} \right)$$

$$= \sum_{i=1}^{N} 2P - 2x_i = 0$$

$$\sum_{i=1}^{N} P = \sum_{i=1}^{N} x_i$$

$$P \cdot N = \sum_{i=1}^{N} x_i$$

$$P = \frac{\sum_{i=1}^{N} x_i}{N}$$



Normalized.

$\boxed{w^T w = 1}$

Norm of x's are fixed.

$$\sum_{i=1}^{N} \left(\|x_i\|^2 - (w^T x_i)^2 \right)$$

independent of w.

$$\text{Minimize} \equiv \sum_{i=1}^{N} -(w^T x_i)^2, \quad \text{s.t. } w^T w = 1$$

$$\text{Maximize} \equiv \sum_{i=1}^{N} (w^T x_i)^2$$

$(Xw)^T X w$

$w^T X^T X w$

$(w^T x)^T (w^T x)$

$x^T w w^T x$?

wrong.

Max: $w^T X^T X w$.

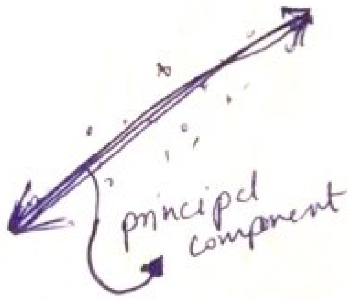
look at regression.

Maximize $\equiv w^T X^T X w$ st $w^T w = 1$

$\downarrow$ w is eigen vector of $(X^T X) \equiv$ PCA

(largest) eigen value

$\Downarrow$ why? — covariance matrix



$Ax = \lambda x$

Max ($w^T \underbrace{X^T X}_{} w$) — λw

To Maximize this quantity $w^T \lambda w$, we have to take largest value of λ

$\Downarrow$

largest eigenvalue

$w^T \lambda w$.

max($w^T A w$)

$$w^T \lambda w \equiv \lambda \underbrace{w^T w}_{1} = \boxed{\lambda}$$

λ $\downarrow$ eigen value.

Direction is given by w, where w is the eigen vector corresponding to the largest value of $X^T X$

$\downarrow$

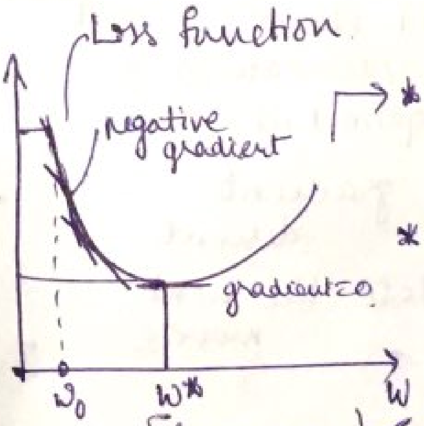
Mainly used in PCA

Gradient Descent $\Rightarrow$

(x_i, y_i), $i = 1 \ldots N$.

$\int (y_i - f(x_i))$

loss function = need to be minimized



* Randomly initialize w
* Move on the negative grad

$w^{n+1} \leftarrow w^n - \eta \nabla J$

{learning rate}

* Repeat steps until you reach to minⁿ

* Goal
  - objective
* Data (x, y)
* Optimization
  - closed form Algo
  - Iterative Algo $\longrightarrow$ Gradient Descent

Scanned by CamScanner

Given (x_i, y_i), $i = 1, \ldots n$.

Find $y = f(x)$ such that it fits the existing (and future) data or minimize a loss/error function.

$\rightarrow y_i \sim f(x_i) \; \forall i$

$\swarrow \; \searrow$

R $\quad \{0,1\}$

$\quad\quad$ C

$f(x) = w^T x$ } Assumed it to be a linear function.

$$\begin{bmatrix} x^1 \\ x^2 \\ x^3 \\ x^4 \end{bmatrix} = w_1 x^1 + w_2 x^2 + w_3 x^3 + w_4 x^4$$

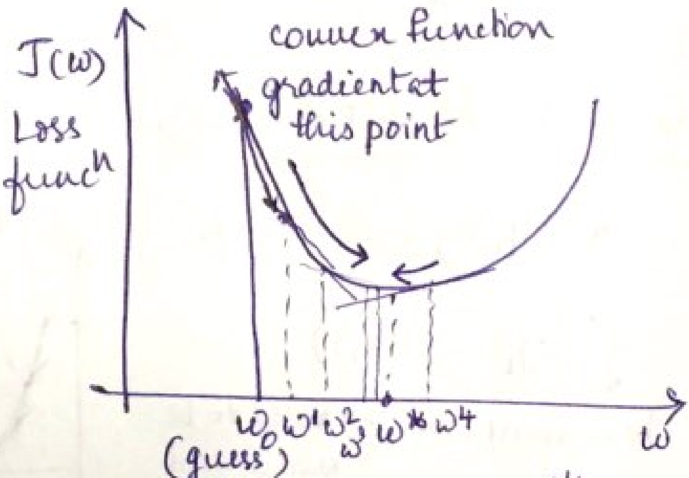
$w^T x \equiv$ Not restrictive

$$\underset{2d}{\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}} \rightarrow \underset{5d}{\begin{bmatrix} x_1^2 \\ x_2^2 \\ x_1 \\ x_2 \\ 1 \end{bmatrix}}$$

$\rightarrow$ Find the w suiting those dimensions/features.

5d $\rightarrow$ In five dimension, it is linear.

Gradient Descent:



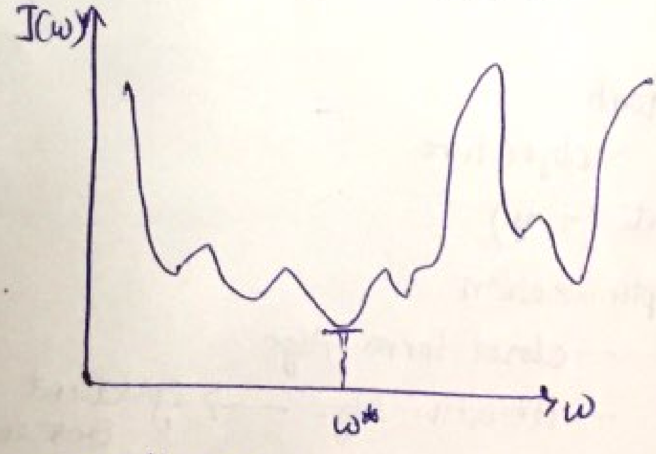
> Make a random guess.

$\downarrow$

you should either increase w/decrease w to reach optimal w.

> check the gradient descent and calculate the next move $\equiv$

Non-convex f^n



$$w = w^0 - \eta \nabla J(w^0)$$

η = amount of movement.

At the minimum;

$\quad \Delta J(w) = 0$ and

$\quad\quad w$, doesnt change $\longrightarrow$ Algorithm converges

May not reach $\nabla J(w^0) = 0$, we can put an error constant threshold

In nonconvex function; the (local) minimum that we get maynot be the best one by using the same equation

$\nabla J(w_0) < 0.001$

changing it doesnt give much of an advantage

What is η(eta)? — Adaptive over time / iterations based on the behavior of algo.

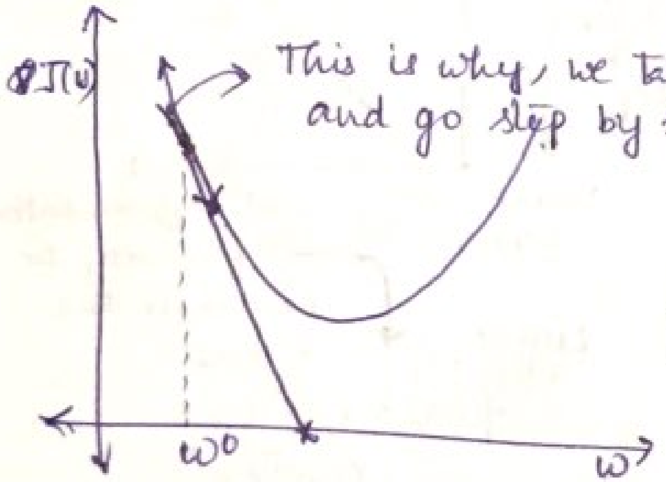
Learning Rate

$$w^{n+1} \leftarrow w^n - \eta \nabla J(w^n)$$

when we make a move into the negative gradient

→ Given is data (x_i, y_i) not the loss function

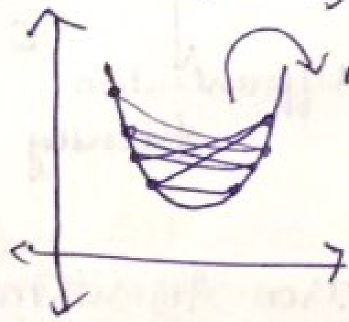
— Assume the function to be a line at the initial stage. and compute the slope.



This is why, we take η and go step by step. ← (if function nature is known, it would have been easy).

⇒ Start with a very small η

Reason: (with large η, you take huge leaps (to and fro into positive and negative gradient regions))



⇒ η is increased, if direction of w in the ~~past~~ prev step and ~~next~~ present step are same.

⇒ and if direction changes between steps, $\eta \equiv$ decreased

for Ex:

$\eta \leftarrow 1.5 \times \eta$, if prev dr = present dr

$\eta \leftarrow 0.5 \times \eta$, if direct change

Taylor series expansion ≡

$x = w^{n+1}$
$a = w^n$

$$f(x) = f(a) + f'(a).(x-a) + f''(a)(x-a)^2 + \ldots$$

$$f(w^{n+1}) = f(w^n) + [w^{n+1} - w^n] f'(w^n)$$

negative quantity. $w^{n+1} \leftarrow w^n - \eta \Delta J(w^n)$

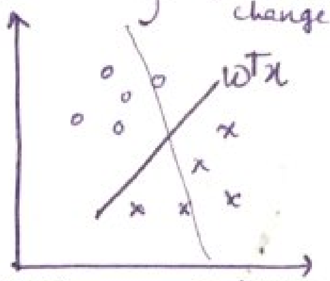
→ GD for Classification: New function value (Less than the old one) = Old function value − (Positive quantity).

» Logistic Regression ≡

» ~~Perceptron Algorithm~~ ≡

(instead of step) → search for smooth step

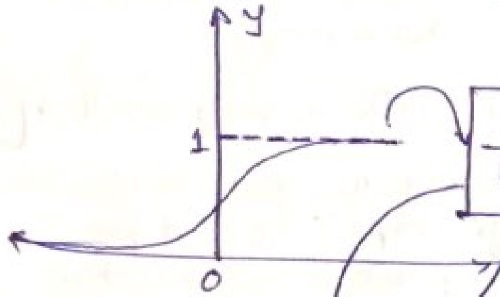
GD approach ≡ w^0 (take a line and acc^h to change the dir)



minimum error. ≡ not differentiation friendly to eliminate the errors

Linear classifier: $f(x) = 1, w^T x > 0$
$0, w^T x \leq 0$.

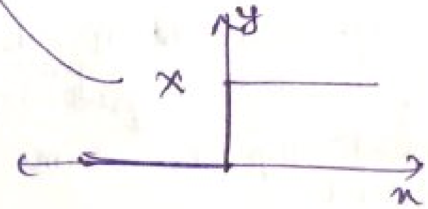
But instead;



$\boxed{\frac{1}{1+e^{-w^T x}}}$

$w^T x \to -\infty$: 0 $w^T x \to \infty$: 1

differentiation friendly.



» Perceptron Algorithm ≡ (created a new objective func^n)

$$J = \sum_{x \in \text{Misclassified samples}} w^T x$$

} Minimize the no. of such (error) classifications.

Sum over misclassified samples. (instead of over all the samples)

when, Misclassified samples = {ϕ} ⤳ we found the "line".

Logistic Regression ≡ step → smooth step

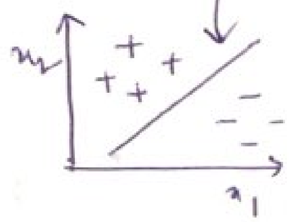
Perceptron Algorithm ≡ creating a new objective function

every sample that gets misclassified is added to it

$w^{n+1} \leftarrow w^n - x$?

If the samples are "linearly seperable", ($\exists w$ which can seperate +ve and -ve examples) → perceptron will find that w

Classification ≡ Model of classifier is ≡ Line. [Given this model, we cannot generate an apple]

1. 

→ we model the line in the feature space
Boundary line between class A and class B.

→ Discriminative classifier/model

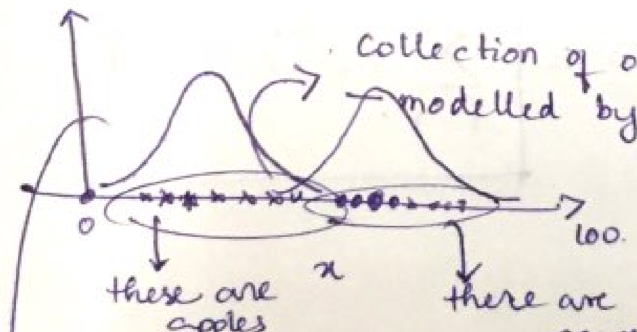
2. Generative Model (we donot model the boundary) ≡ reduced representation

instead model the data [as in "a point"]

[Ex: model oranges and apples independently]

ForEx: given a fruit, how apple-like or orange-like is it?

Model the classes?



→ Collection of objects modelled by probability density.

[If I see 10.5, how likely is to classify it as apple]

likelihood functions = Independently Learning about apples/oranges.

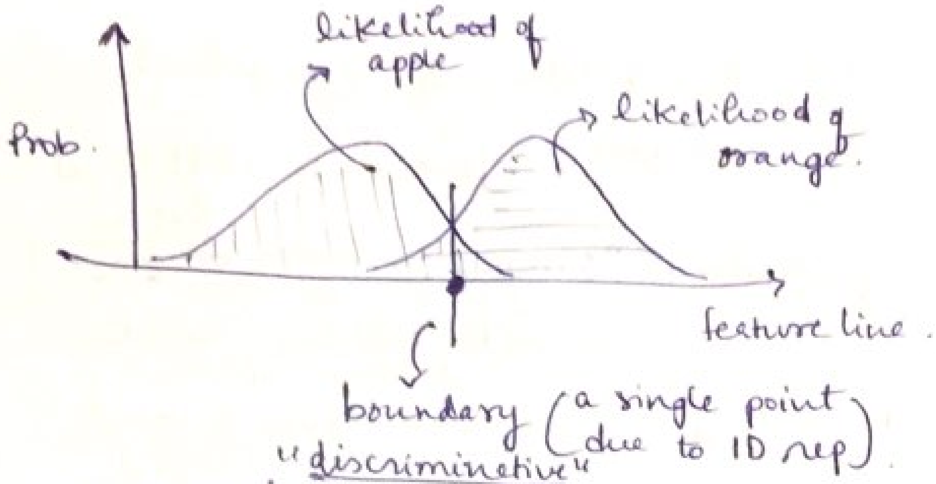
⇒ given density function } → we can generate new points [that is why ≡ generative model]

Discriminative Classifier $\}\equiv$ we cannot generate samples as it can lie anywhere over the apple side of the line.

↓ Not rich enough to capture the details

↓ cannot explicitly model apples.

[can only say that anything that lies on one side is apple]



$\omega \equiv$ class

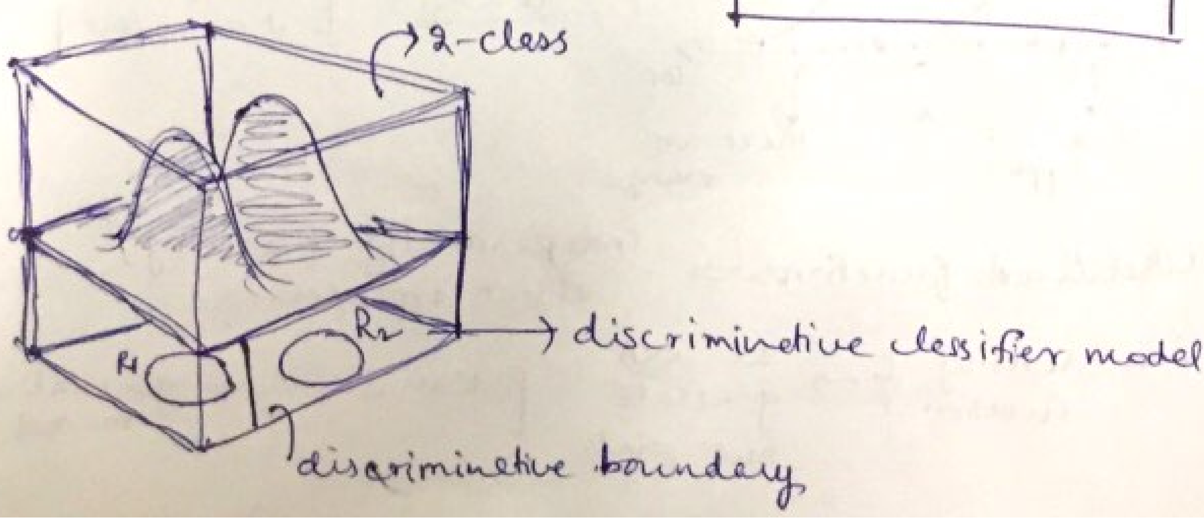
$P(\omega_1/x) \equiv$ "Probability" that given feature vector, x belongs to the ω_1 class. $[0 < P < 1]$

↓ feature vector

$P(\omega_1/x)$, $P(\omega_2/x)$ can be compared to determine the class.

[in the case of 2-class problems $\} \equiv P(\omega_1/x) + P(\omega_2/x) = 1$]

Whereas, if there are c classes $\equiv$ $\boxed{\sum_{i=1}^{c} P(\omega_i/x) = 1}$



P [Probability] ; $P(\omega_1/x)$, $P(\omega_2/x)$.

p [pdf]

$P(A,B) = P(A/B) \cdot P(B)$

(independent events)

$= P(A) \cdot P(B)$

$P(\omega_1, x) = P(\omega_1/x) \cdot P(x)$ (or) $p(x/\omega_1) \cdot P(\omega_1)$.

Joint Probability

$$\boxed{\therefore P(\omega_1/x) = \frac{p(x/\omega_1) \cdot P(\omega_1)}{p(x)}}$$

Posterior prob. → Bayes Theorem.

⇒ $P(\omega_1) \equiv$ Prob. of class irrespective of given sample.

→ Prior probability / belief

⇒ $p(x/\omega_i)$ ⇒ likelihood.

⇒ $p(x)$ → how likely is that we'll observe x. "evidence".

Posterior prob $\equiv$ Post observing x, what is the probability $P(\omega_1/x)$

Ex: Disease and Test

$P(D) = 0.001$

$P(+/D) = 0.99$ If you have the disease, tests will say +, 99% (Accuracy of test)

$P(D/+) = \dfrac{0.99 \times 0.001}{P(+)}$

Prob of having disease, given test came out +ve

Evidence = likelihood that the test come out +ve. (irrespective of disease or not) on random person

$P(+) = P(+/D) \cdot P(D) + P(+/ND) \cdot P(ND)$

person with D gets +ve result

person with ND

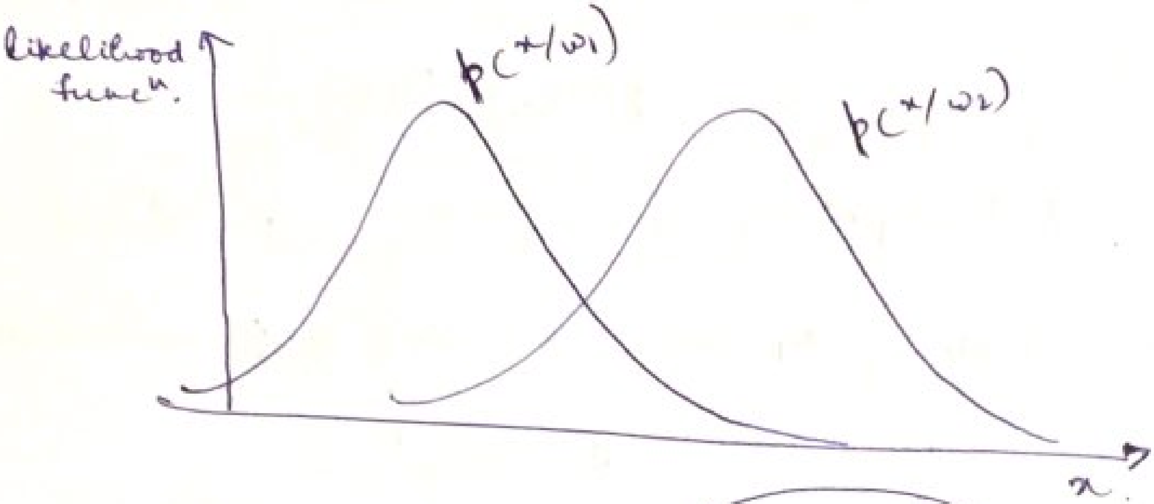
$= 0.99 \times 0.001 + 0.01 \times 0.999$

$= 0.01 (0.099 + 0.999) = 1.098 \times 0.01$

Scanned by CamScanner

$$P(D/+) = \frac{0.99 \times 0.001}{0.01098} = 0.09 \; (9\%).$$

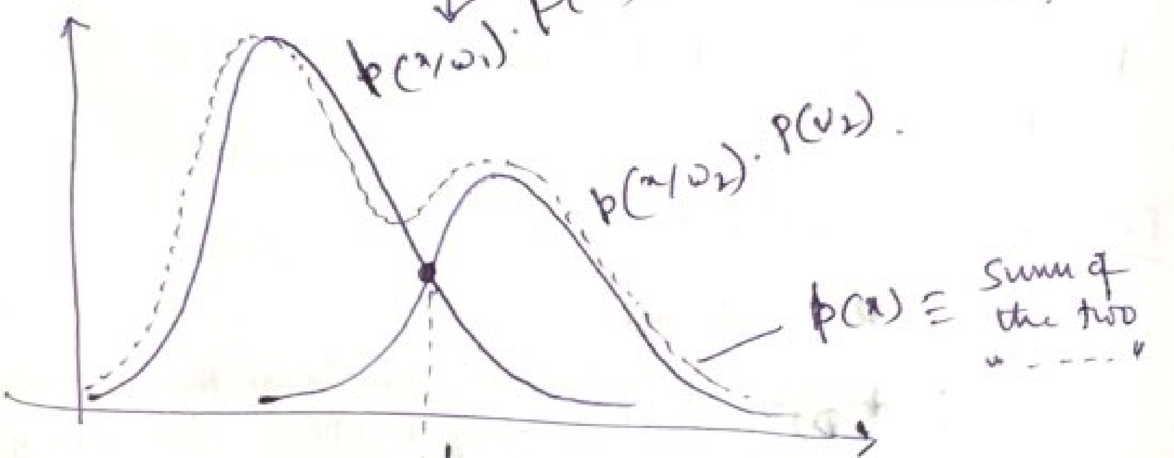
↑ chance of having the disease.



$$P(\omega_1/x) = \frac{p(x/\omega_1)\cdot P(\omega_1)}{p(x)}$$

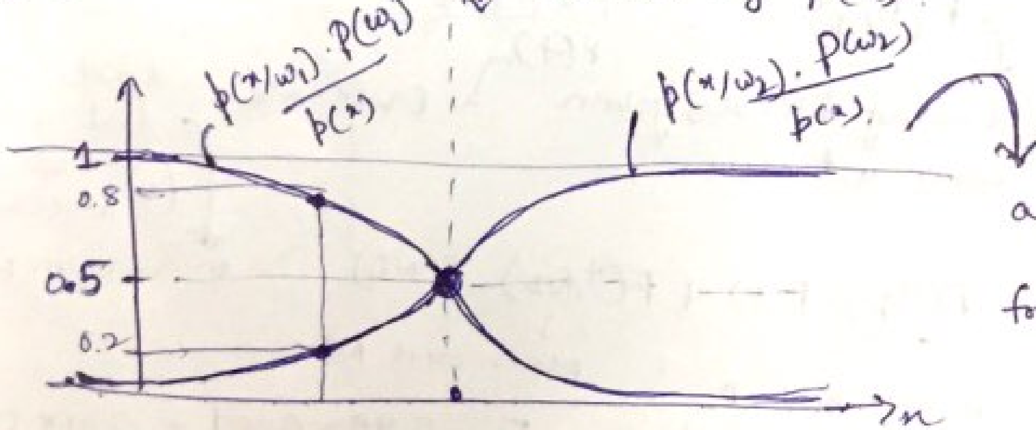
— constant ↓ one of the curves will be larger than the other.

↓



↓ divide by p(x)

Normalize ≡



add upto 1 always for any x

Bayes Theorem $\equiv$

$$P(\omega_j/x) = \frac{p(x/\omega_j) . P(\omega_j)}{p(x)}$$

after observing x, state quature $\equiv \omega_j$

probability that state of nature, ω_j is true. Pick ω_j that has maximum prob.

Are all errors equally costly? No

so we define a loss function $\equiv \lambda(\alpha_i/\omega_j)$

action, α_i

state of nature $\equiv$ class

Loss is not symmetrical,

E_1 if class A is misclassified as class B or

E_2 if class B is misclassified as class A.

$E_1 \neq E_2$

$\rightarrow$ there can be many actions

Risk, $R(\alpha_i/x) \rightarrow$ observe x, risk of taking α_i action.

(of taking action α_i) $R(\alpha_i/x) = \sum_{j=1}^{c} \lambda(\alpha_i/\omega_j) . P(\omega_j/x)$

$\rightarrow$ Prob that ω_j is true

$\rightarrow$ loss incurred because of taking α_i ~~when~~ in ω_j state of nature

c classes.

Strategy to pick the best action $\equiv$ based on action plan

Evaluate the action plan? $\alpha(x)$ — if i see this observation, i'll do this.

$$R(\alpha(x)/x) . p(x)$$

prob of observing x

Risk of following action plan, $\alpha(x)$ on observing x.

given an x, what action needs to be taken.

"Action plans can be eval. based on risks"

— Integrate over all possible x,

Overall Risk $\equiv$ $R_{\alpha(x)} = \int_x R(\alpha(x)/x) . p(x)\, dx$

To Minimize overall risk, by minimize $R(\alpha(x)/x)$ for ~~all~~ x

choosing appropriate / best "$\alpha(x)$"

From Bayes —
$$R(\alpha_i/x) = \sum_{j=1}^{c} \lambda(\alpha_i/\omega_j) \cdot P(\omega_j/x)$$

$R(\alpha(x)/x) = \underset{\alpha_i}{\operatorname{argmin}} \left(R(\alpha_i/x)\right)$

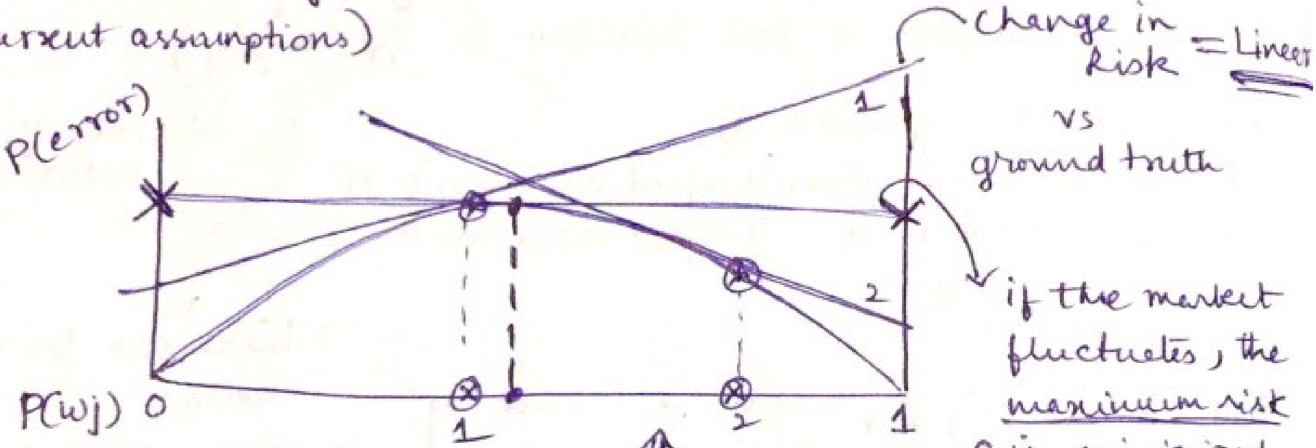
$R^* =$ Bayes Risk.

Bayes Action Plan.

Risk $\propto$ P(error).

*

$P(\omega_j) \rightarrow$ can change
(current assumptions)



(Mix of the classes ≡ Ex: Mix of Kashmir and Washington apples in the market)

"Neyman-Pearson criterion"

if the market fluctuates, the maximum risk is minimized
choose such action plan.

$$P(\omega_i/x) = \frac{p(x/\omega_j)\ P(\omega_j)}{p(x)}$$

$p(x) \longrightarrow$ need not be computed to decide / compare [independent of ω_j]

$$g_i(x) = p(x/\omega_i) \cdot P(\omega_i)$$

(this is different g) $g_i(x) = \ln p(x/\omega_i) + \ln P(\omega_i)$

To classify, you need to compute $P(\omega_i/x)$, you can either compute $g_i(x)$ or $g_i(x)$ (latter one)

Equation of Normal density $\Rightarrow p(x) = \frac{1}{(2\pi)^{d/2}|\Sigma_i|^{1/2}} e^{-\frac{1}{2}(x-\mu_i)^t \Sigma_i^{-1}(x-\mu_i)} = P(x/\omega_i)$

(can be ignored) constant

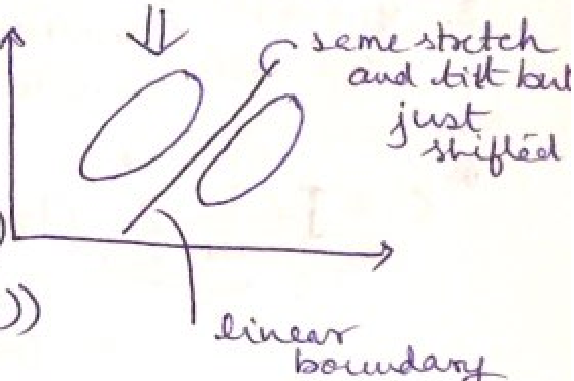
$$g_i(x) = -\frac{d}{2}\ln 2\pi - \frac{1}{2}\ln|\Sigma_i| - \frac{1}{2}(x-\mu_i)^t \Sigma_i^{-1}(x-\mu_i) + \ln P(\omega_i)$$

$g_1(x), g_2(x)$ can be computed for the normal densities

* $$g_i(x) = -\frac{1}{2}\ln|\Sigma_i| - \frac{1}{2}(x-\mu_i)^t \Sigma_i^{-1}(x-\mu_i) + \ln P(\omega_i)$$

Case 2: if $\Sigma_i = \Sigma$ (both the classes), if Kashmir and Washington, variance may be same, same covariance matrix

$$g_i(x) = -\frac{1}{2}(x-\mu_i)^t \Sigma_i^{-1}(x-\mu_i) + \ln P(\omega_i)$$



$$= -\frac{1}{2}(\underline{x^t\Sigma^{-1}x} - 2\Sigma^{-1}\mu_i^t x + \mu_i^t \Sigma^{-1}\mu_i) + \ln(P(\omega_i))$$

independent of i, class

$$g_i(x) = \underbrace{\Sigma^{-1}\mu_i^t x}\ \underbrace{- \frac{1}{2}\mu_i^t \Sigma^{-1}\mu_i + \ln(P(\omega_i))}$$

tends to be "linear"

Discriminant function.

$$= w_{i1}^T x + w_{i0}$$ } linear function.

In 2-D, decision boundary would be a linear function (a line)

In 3-D, it'll be a plane

* $\Sigma_i \neq \Sigma$

$\Rightarrow$ if covariance matrix is not the same, then the decision boundary will no more be a linear function but a complex function. (it could be an elipse, hyperbole ...)

Case 3: $\Sigma_i = \Sigma_i$

$$g_i(x) = -\frac{1}{2}(x^t\Sigma^{-1}x - 2\Sigma^{-1}\mu_i^t x + \mu_i^t \Sigma^{-1}\mu_i) + \ln P(\omega_i)$$

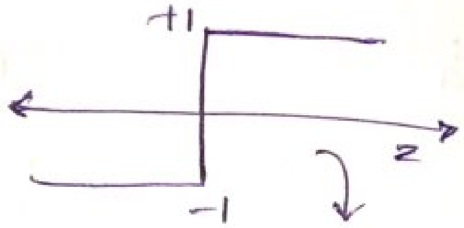
Loss :

In Regression — $R \equiv J = \sum_{i=1}^{N} (y_i - w^T x_i)^2$

$g(z) = +1 \; ; z \geq 0$

$\quad\quad\;\; -1 \; ; z < 0$

In Classification — $C \equiv$ 0% misclassifications.



Classification

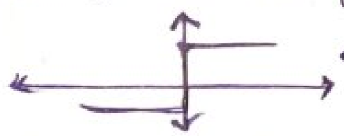
Ex: $y_i * g(\omega^T x)$.

1. $y_i = +1$ ✓ correct $= 1 = y_i (g(\omega^T x))$
   $\omega^T x = 10$.

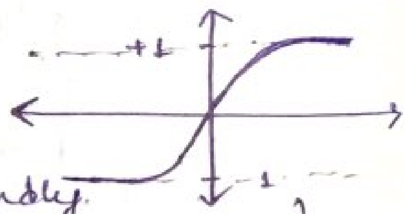
2. $y_i = -1$ ✓ correct $= 1 = y_i (g(\omega^T x))$
   $\omega^T x = -10$.

3. $y_i = -1$ × incorrect
   $\omega^T x = 10$ (misclassified).

For this, instead of using



its better to use

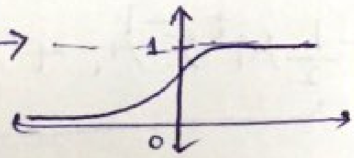


this is (Not Logistic Regression)

$* \; C \equiv J = \underbrace{\frac{1}{2N} \sum_{i=1}^{N} (1 - y_i \cdot g(\omega^T x_i))}$

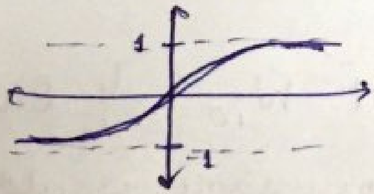
— Good
— Not optimisation friendly.

optimize it by changing the $\boxed{g}$ →



[Logistic Regression]

(or)



Gradient Descent:

1. start with w^0, $n = 0$
2. $w^{n+1} \leftarrow w^n - \eta \nabla J$
   - ∇J ↳ changes, when loss function changes
   - η → varies over time (iteration) [usually constant]
3. $n \leftarrow n + 1$
4. Repeat 2-4 until some convergence criteria is met.

?

— Change in w is very very small.
— change in loss is very small.

→ Loss function is not linear but the discriminant func^n is linear.

$$f(y) = f(x) + (y-x) f'(x) + \frac{(y-x)^2}{2} f''(x)$$

approximated $\equiv f(Y) = f(X) + (y-x)^T \nabla f(x) + \frac{1}{2}(y-x)^T H [y-x]$ $\equiv$ H — operator — d×d matrix — Hessian matrix

$$\boxed{f(w^{n+1}) = f(w^n) + [w^{n+1} - w^n]^T \nabla f(w^n)} + \frac{1}{2}(w^{n+1} - w^n)^T H [w^{n+1} - w^n]$$

$$w^{n+1} \leftarrow w^n - \eta \nabla f$$

$$w^{n+1} - w^n = -\eta \nabla f$$

only proves that

$$f(w^{n+1}) = f(w^n) + (w^{n+1} - w^n)^T \nabla f$$

$\boxed{\nabla f^T . \nabla f}$

$$f(w^{n+1}) = f(w^n) - \eta\,(\text{+ve quantity})$$

$\Rightarrow$ GD reduces the loss in each iteration if η is +ve and small

but can we go faster?

$$f(w^{n+1}) = f(w^n) - \eta \|\nabla f\|^2 + \frac{\eta^2}{2} \nabla f^T H [\nabla f]$$

Since we want minimum, diff $f(w^{n+1})$ wrt η.

$$\frac{\partial}{\partial \eta} f(w^{n+1}) = 0 - \|\nabla f\|^2 + \frac{2\eta}{2} . \nabla f^T H [\nabla f] = 0$$

$$\eta . \nabla f^T . H(\nabla f) = \|\nabla f\|^2$$

$$\eta = \frac{\nabla f . \nabla f^T}{\nabla f^T . H [\nabla f]} = \frac{\nabla f^T \nabla f}{\nabla f^T H \nabla f}$$

$w^{n+1} - w^n \equiv S$

Better update rule than this: $w^{n+1} \leftarrow w^n - \eta \nabla f$?

$$f(w^{n+1}) = f(w^n) + S^T \nabla f + \frac{1}{2} S^T H [S]$$

$$= f(w^n) + S^T \nabla f + \frac{1}{2} S^T H . S$$

H – matrix
S – vector

$$\frac{\partial}{\partial S} = 0 \Rightarrow \nabla f + HS = 0$$

$$S = -H^{-1} \nabla f$$

Better update rule, – Newton iteration Descent (* but not used)

↳ Not used → maybe H might be singular. ✓
(inverse doesnt exist)

if |H| is very less ↘ inverse can be very sensitive
↓
Not computationally attractive

First of all, How is this better than the first one? GD → In the first case, approximating to the first order [GD] gives a smaller value (higher error)

quadratic and cubic factors give less error but computational load ↑

For now, constant η. ✓ but we have to handle the loss, J

Perceptron ≡ Sum the loss only over the misclassified samples. ↘ optimizing wrong objective funcⁿ

objective function.

$w^Tx \geq 0$, $y_i = +1$
$w^Tx < 0$, $y_i = -1$

$$\sum_{x_i \in \Sigma} -y_i \, w^T x_i$$

↳ misclassified

Total Error (Amount of misclassification)
Not number, because we are taking w^Tx instead of $g(w^Tx)$.

↓

update rule changes to ≡ $\boxed{w^{n+1} \leftarrow w^n + \eta \sum_{x_i \in \Sigma} y_i x_i}$

Changes with every step.
so
↓

Not computationally different ← alternative ≡

$$\boxed{w^{n+1} \leftarrow w^n + \eta \sum_{i=1}^{N} (t_i - o_i) x_i}$$

target output

if target = 1, output = 1
$t - o = 0$ (correctly classified)

doesnt need to ~~the~~ ~~[illegible]~~ go over misclassified set.

Convergence criteria $\Rightarrow$ until no class is misclassified

How does this work? (Intuitive Approach)

① Assume, $\underbrace{t_i = +1, \; o_i = -1}_{\text{Error is present.}}$ and $x_i = +ve$

$w^T x \geq 0$, $+1$
$w^T x < 0$, -1

initially $w < 0$ as $x > 0$ & $w^T x < 0$

$\downarrow$

with these values, w's going to increase $\Rightarrow$ increase w

↳ w becomes $+ve$

② $t_i = +1$, $o_i = -1$ and $x_i = -ve$

$w^T x < 0$

$x < 0$ $\quad w > 0$

$(t - o)x \Rightarrow -ve \longrightarrow w$ decreases.

↳ w becomes $-ve$

11^{ly} for diff values of t & o.

$\rightarrow$ if a sample is misclassified, add/subtract it. (based on values of t and o)

Binary (2-class) classification :- Loss/obj/Error

optimised using gradient descent

* Perception — $\underset{w}{\text{argmin}} \sum_{x \in \epsilon} - y_i w^T x \equiv$ OBJ Func^n. (misclassified samples)
* Logistic Regression

$w^{k+1} \leftarrow w^k - \eta \nabla J$

Algo depends on ∇J

Algo :-

* Initialize w^0, $k = 0$.

change is done per sample?

$w^{k+1} \leftarrow w^k + \eta y_k x_k$, if $x_k \in \Sigma$

$k \leftarrow k+1$

Repeat until Σ is empty.

3 variations = $\longrightarrow$ sol^n changes with every sample (update for every sample)

* Online (imp: data streaming (continuously flowing samples))
* Batch $\longrightarrow$ "compute derivative over all the samples"
* Stochastic

Online (samples one at a time)

- start with the first sample.
- if error, update. else go to next sample
- Repeat step 2.

⇒ * If Batch goes into oscillations, you never come out of it

Stochastic (Not deterministic approach)

↳ randomly picked up.

↓

Why? , because there might be patterns leading to oscillations.

MiniBatch

- Instead of taking all the samples, consider a set of samples!

For Ex: out of 100 samples, sample at set of 10.

$\boxed{\text{mini batch stochastic batch gradient}}$ * mini SGD.

sample has to be replaced back

> online is a form of stochastic.

* $\boxed{\text{Pseudocodes of all the algos}}$ *

online | stochastic | miniSGD.

Perceptron (Recap) ≡

$$w^{k+1} \leftarrow w^k + \eta \sum_{i=1}^{N} (t_i - o_i) x_i$$

$$w^{k+1} = w^k + \eta \sum y_i x_i$$

this misclassification.

t_i	o_i	x_i	w	(to remove this error)
+1	−1	+ve	⇒ −ve	w has to ↑
−1	+1	+ve	⇒ +ve	w has to ↓

* Separable — $\exists$ "$\underline{w}$" such that data can be correctly classified.

Cannot have a linear separating hyperplane for every problem

→ Problem is linearly separable, there exists a solution weight vector.

Termination criteria of perceptron — $\mathcal{E}$ should be empty

$\mathcal{E}$: misclassification sample set.

* If data/samples is linearly separable;

$$\boxed{\underbrace{y_i \cdot w^T x_i}_{\text{represents the Classification}} \geq \gamma}$$

⟹ this means, they are correctly classified

γ: small positive quantity (margin)

* Bounded ⟹ $\|x_i\| \leq R$

$y=1 \Rightarrow \quad w^{k+1} = w^k + y_i x_i$

Let w^* be the separating plane. (solution = optima)

$$w^{*T} \cdot w^{k+1} = w^{*T} w^k + \underbrace{y_i w^{*T} x_i}_{(>\gamma)}$$

$$w^{*T} w^{k+1} > w^{*T} w^k + \gamma$$

previous term

(∴ let $w^0 = 0$)

$$w^{*T} w^k > w^{*T} w^{k-1} + \gamma$$

$$\vdots$$

$$w^{*T} w^1 > w^{*T} w^0 + \gamma$$

$$\Rightarrow w^{*T} w^{k+1} > (k+1)\gamma$$

$\swarrow$ $w^{k+1} = w^k + y_i\, w_i^T x_i$

Squaring on both the sides. $\equiv$

$$\|w^{k+1}\|^2 = \|w^k + y_i\, w_i^T x_i\|^2$$

$\|x_i\|^2 \longrightarrow$ Bounded

$$\downarrow$$

$$k^2 \gamma^2 < k R^2$$

$\rightarrow$ (Perceptron algo guarantees that it converges in finite no. of steps)

$$k < \frac{R^2}{\gamma^2}$$

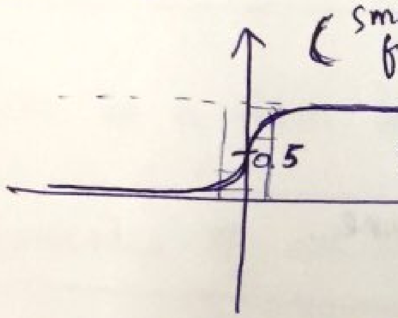
"(Converge in these many finite no of steps)"

* $\gamma \rightarrow$ large ; $K \rightarrow$ converge faster

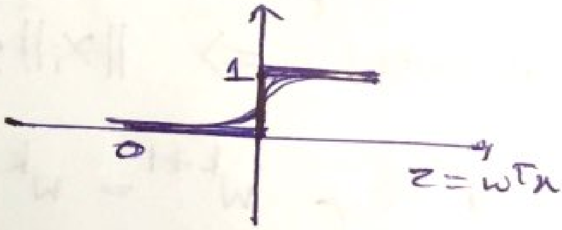
(data sets are nicely separated) (upper bound on number of iterations)

Logistic Regression $\Rightarrow$

(smooth function)



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



$\searrow$ can be $\frac{1}{1+e^{-kz}}$ (gets more closer to the axis)

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

$\downarrow$

$P(y=1/x)$

If $w^T x \gg$ large, belongs to class-1
else, belongs to class-0

$$g(w^T x) = P(y=1/x)$$

$$\underbrace{1 - g(w^T x) = P(y=0/x)}$$

$$\downarrow$$

$\Rightarrow$ Classify as 1, if $P(y=1/x) > 0.5$

Classify as 0, " ≤ 0.5

$w^T x = 0$ is the decision boundary.

though both the curves (smooth & step) infer the same thing $\leftarrow$

$J=\sum_{i=1}^{N}\left(y_i - g(w^Tx_i)\right)^2$ X Not a good function (due to local minimas)

y_i: either 1/0

$g(w^Tx_i)$: smooth curve $= \frac{1}{1+e^{-z}}$

differentiable but it is not convex?

but linear regression $\Big\} \equiv \sum^{N}(y_i - w^Tx_i)^2$ ✓

$$J = \sum_{i=1}^{N} \log\left(1+e^{-y_i\, g(w^Tx_i)}\right)$$

$$= \sum_{i=1}^{N} y_i \log(w^Tx_i) + (1-y_i)\log\left(1-g(w^Tx_i)\right)$$

→ (1) * $J =$ objective function + $\boxed{\text{Regularizer}}$?

*usually a function of w

additional constraints beyond objective function.

diff(J) = diff(obj f^n) + diff(Regularizer)

→ (2) Extending algo to multi-class (k-classes)

* [combining many binary to make a multi-class]