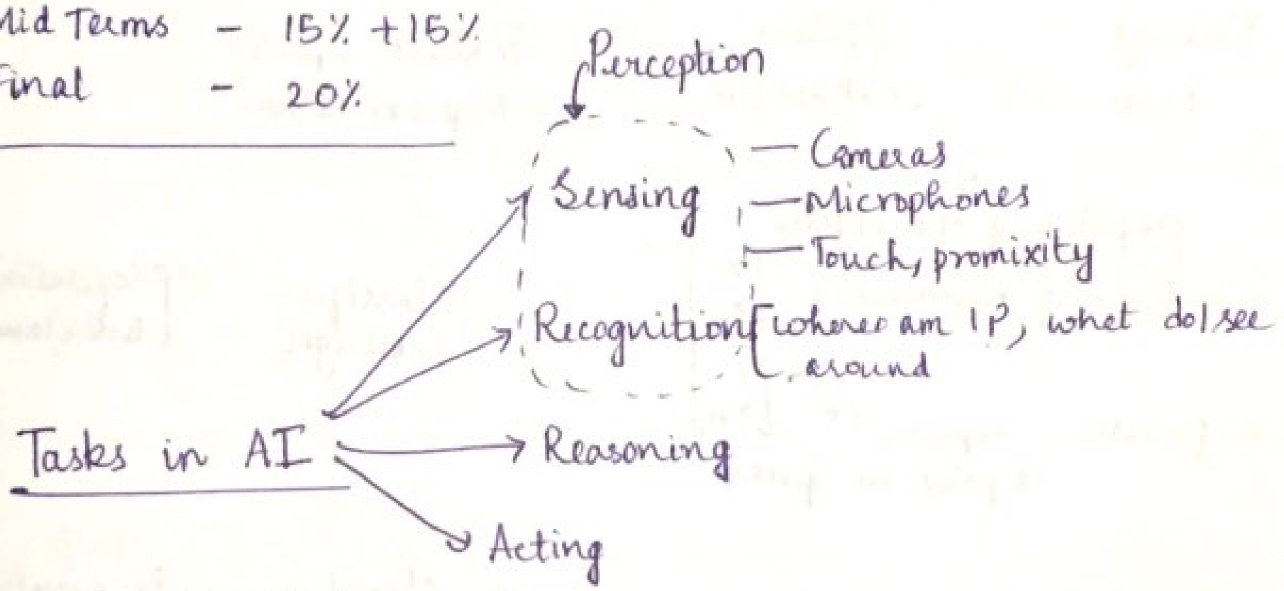
STATISTICAL METHODS IN A.I

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%



Perception

Sensing — Cameras, — Microphones, — Touch, promixity

Recognition { where am I?, what do I see around

Tasks in AI → Sensing, Recognition, Reasoning, Acting

"Re" + "Cognize"

Recognition - 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

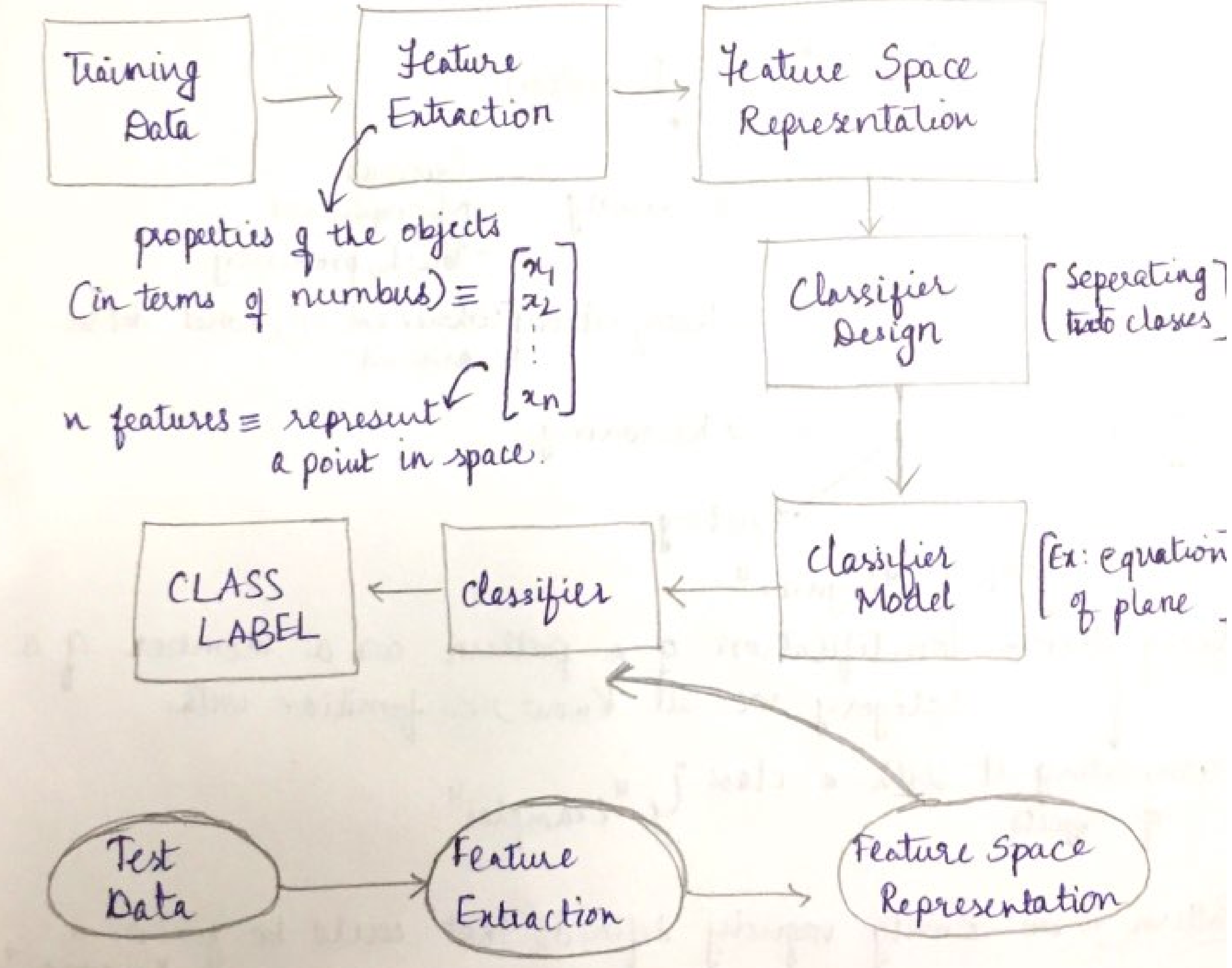
Pattern Recognition – infering a genarality from a few examples

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

$$\begin{tikzcd} & \text{Inter-class variability} \\ * \text{ class} \arrow[ur] \arrow[dr] \arrow[d] & \\ \text{represented as } [\omega_1, \omega_2, \ldots \omega_n] & \text{Intra-class variability} \end{tikzcd}$$

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

Pattern Recognition Process :-



properties of the objects (in terms of numbers) $\equiv \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$

n features ≡ represent a point in space.

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

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

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

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 ≡ matrices and vector form

$$\underset{A}{\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}} \underset{\substack{X \\ \text{(3D-vector)}}}{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}} = \underset{B}{\begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}} \qquad AX = B$$

$|x|$ ↷ absolute value.

(Norm/Length)

Size of Vector ≡ $X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$

$||x||$

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

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

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 ≡ L_1-norm ⇒ $||x||_1 = \sum_{i=1}^{n} |x_i|$

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

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

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

$L_1 = 3$

$L_2 = 2.24$

$L_3 = 2.08$

$L_4 = 2.03$

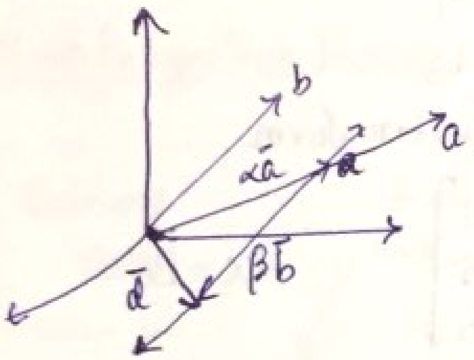
$\vdots$

$L_{10} = 2.00019$

⇒ Span of a set of vectors -

linearly independent ≡ when any combination of two vectors donot result in the third vector

for En: $c = \alpha a + \beta b$ (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.

Rank → dimensionality of the subspace spanned.

as in feature ⇒ typically, vectors are "column vectors".

⇒ Orthonormal Basis — Normal + Perpendicular basis (linearly independent)

Normal → |Norm| = 1

Ortho → perpendiculer (90°) [Dot product of orthogonal vectors = 0]

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

⇒ Dot Product ≡

$$\overset{X}{\begin{bmatrix}x_1\\x_2\\x_3\end{bmatrix}}\overset{Y}{\begin{bmatrix}y_1\\y_2\\y_3\end{bmatrix}} \Rightarrow \overset{X.Y}{x_1y_1+x_2y_2+x_3y_3} = X^T.Y$$

(Inner Product)

$$XY^T = \begin{bmatrix}x_1\\x_2\\x_3\end{bmatrix}\begin{bmatrix}y_1 & y_2 & y_3\end{bmatrix} = \begin{bmatrix}x_1y_1 & x_1y_2 & x_1y_3\\ x_2y_1 & x_2y_2 & x_2y_3\\ x_3y_1 & x_3y_2 & x_3y_3\end{bmatrix}$$ (outer Product)

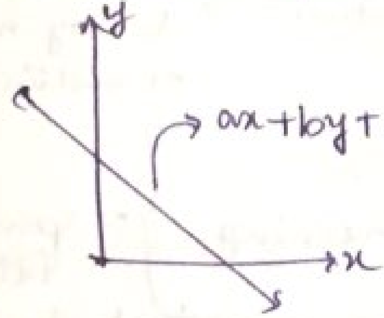
⇒ Cross Product ≡ $(X,Y) = X \times Y = \begin{vmatrix} i & j & k\\ x_1 & x_2 & x_3\\ y_1 & y_2 & y_3\end{vmatrix}$

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

↓

perpendiculer to both X & Y

⇒ Representation of Line :-



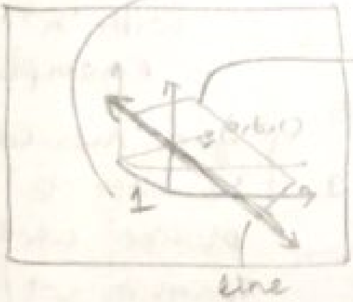
is it a vector or a Line?

[3D vector]

$$\underset{w}{\begin{bmatrix}a\\b\\c\end{bmatrix}}\ \underset{x}{\begin{bmatrix}x\\y\\1\end{bmatrix}} \qquad w^Tx = 0$$

how do you relate both?

↑ Space (augmented vector)



plane perpendicular to (a,b,c)

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

Distance of a point from a plane

Normalise w and take the dot product.

Linear Transformation ⇔ Matrix Multiplication

↓

by inverting the matrix, we can invert the transformation

⇒ Eigen Values and Eigen Vectors

$M = u\,D\,v^T$ $[\lambda]$ eigen values

orthonormal vectors

rows = eigen vectors

↓ physical entity —— represented as a vector

$X_1 \dots \quad X_n \Rightarrow$ vectors

with $d \equiv$ dimension

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

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

observations

physical entities are represented differently in different domains.

Main Problem

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

$\{-1,+1\}$

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

K-class.

classification → Can be a binary one or multiclass.

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

It is called as "Regression"

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

spam or not

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

examples for learning/training

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

≡ Spam filter to be built with "n" examples

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

So when given a new email $\equiv f(x)$ ↗ 0 ↘ 1

x

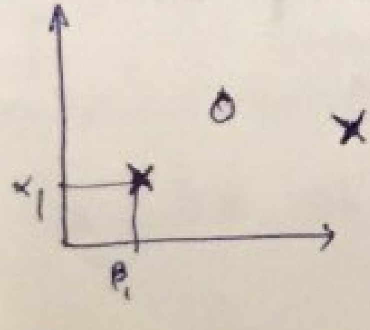
[you should be able to predict whether spam or not]

$$\begin{array}{ccc} x_1 & x_2 & x_3 \\ \begin{bmatrix} \alpha_1 \\ \beta_1 \end{bmatrix} & \begin{bmatrix} \alpha_2 \\ \beta_2 \end{bmatrix} & \begin{bmatrix} \alpha_3 \\ \beta_3 \end{bmatrix} \\ 0 & 1 & 0 \end{array}$$

→ 2 features with different values

(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

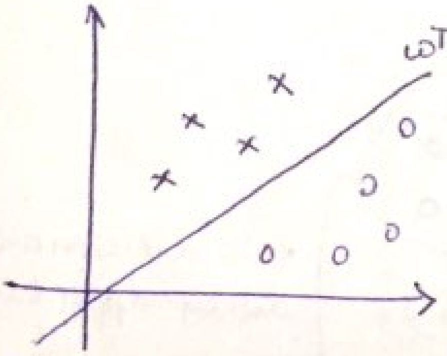
changing the loss function can also changes the optimization Algorithm. ↑ because N>>

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

→ Regression

→ Classification.

Representation ≡



x — Spam

o — Not Spam

Case ①

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

Spam if $f(x) < 0$

Not spam if $f(x) > 0$

↓

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 $f_1(x) > f_2(x)$

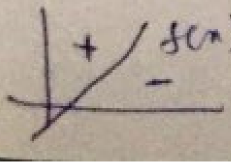
Not Spam if $f_1(x) \leq f_2(x)$

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

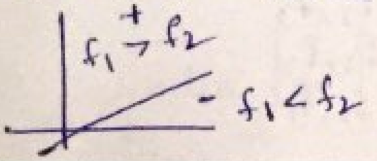
$= w_3^T x + b_3.$

In the 1st case ≡

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



In the 2nd case ≡



? (when to use this approach)

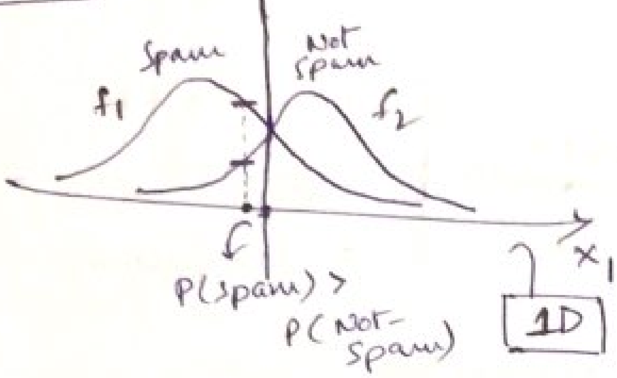
the Second case: Assume $f_1(x) \equiv$ Probability of being spam.

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

both the viewpoints are equivalent in this case.

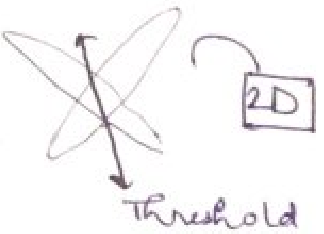
Probability ≡

Threshold ≡ P(spam) = P(not spam)



if $x < T \equiv$ spam

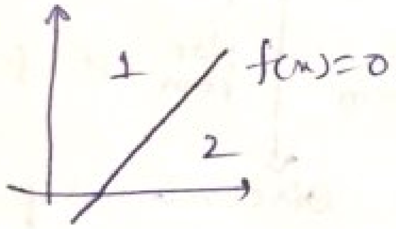
$\geq T \equiv$ Not spam.



So ≡ Spam

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

2-Class Classification Problem:

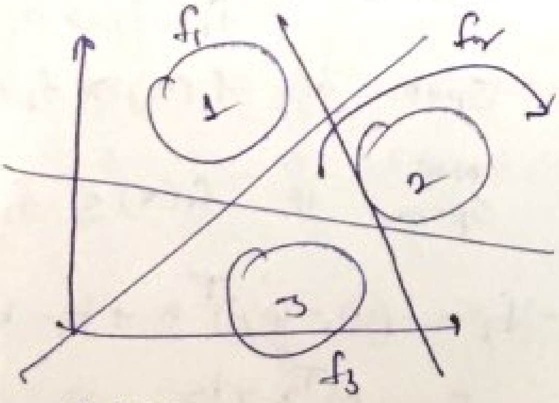


1 if f(x) > 0

2 if f(x) ≤ 0

this decision rule doesn't 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: $f_3 > f_2, f_1$

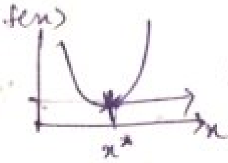
Output – Discrete (Not Continuous)

↓

becomes an discrete optimization problem.

Loss function → Optimization Problem

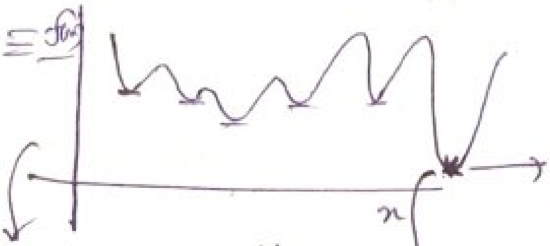
Optimization → Convex ≡



single unique minima.

Optimization → Non - convex

≡



→ maybe the best minima to get

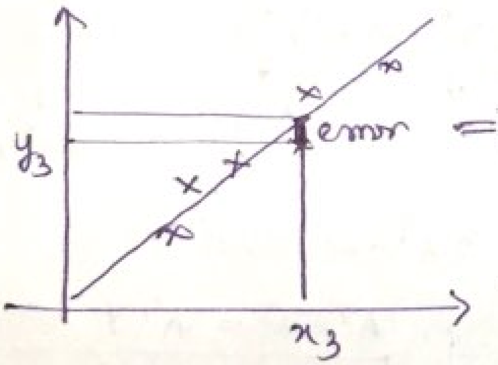
↓ we can solve it.

↓ 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 ⟹ $\varepsilon_3 = w^T x_3 + b - y_3$

↓

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

⇓

$\min_{w,b} \Sigma$

⇓

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

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

↓ Minimum Square Error (M.S.E)

$$\begin{array}{cc} w & x \\ \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}^T & \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \end{array} + b$$

↓ $w^T x$

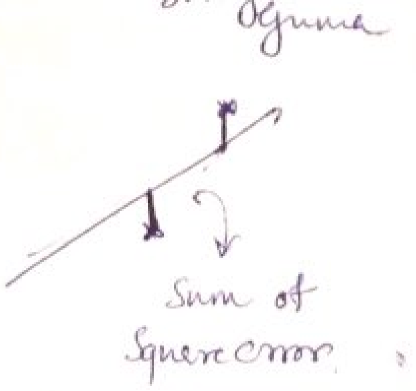
↓ How to remove b ⇓

⇓

$$\begin{array}{cc} w' & x' \\ \begin{bmatrix} 0 \\ 0 \\ 0 \\ b \end{bmatrix}^T & \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \end{array}$$

x is augmented with 1.

$$x_1 = \begin{bmatrix} x_1^1 \\ x_1^2 \\ x_1^3 \\ \vdots \\ x_1^d \end{bmatrix} \qquad \underset{N\times 1}{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}} = \underset{\substack{N\times d \\ (N\times(d+1))}}{\begin{bmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_n^T \end{bmatrix}} \underset{\substack{w \\ d\times 1}}{\begin{bmatrix} w \end{bmatrix}}$$



$N >>> d$

$$\left(\substack{\text{Predicted} \\ Y}\right)\ \hat{Y} = AW$$

with the augmentation.

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

sigma is gone

$(N\times 1)^T \quad (N\times 1) \equiv \boxed{1\times 1}$

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

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

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

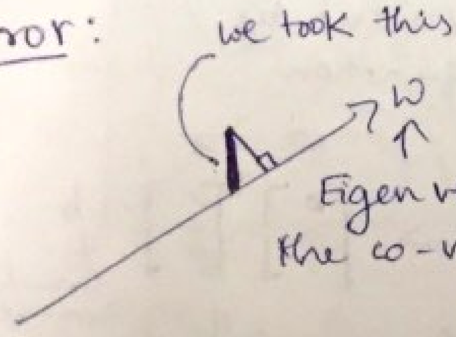
$$2A^T A w - 2A^T Y = 0.$$

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

Tom Minka ≡ Metrix Differentiation.

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

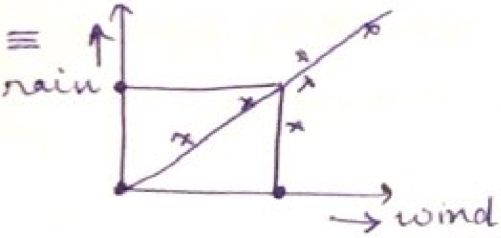
Issues with Error:



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

- A size ≡ N×d → could be extremely large ≡ "Gradient Descent"

Examples of Regression ≡



$Y = f(x)$

$= w^T Z \curvearrowright \begin{bmatrix} x \\ 1 \end{bmatrix}$

$\begin{bmatrix} x^2 \\ x \\ 1 \end{bmatrix} \equiv \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix}$

- 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

$Y = ax + b$

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

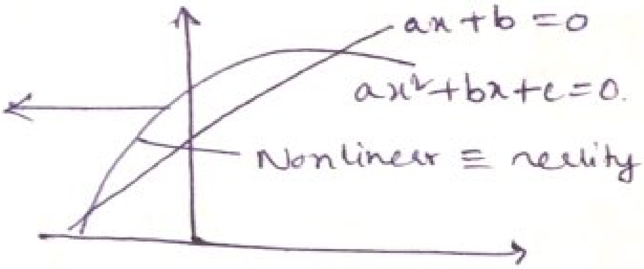
$y = w_1 x_1 + w_2 x_2 + w_3$

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

can be represented as ≡

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

↳ $w^T X = 0$.



* Main goal:- Find "$\underline{\underline{w}}$", for a $\underbrace{\text{function}}_{\text{Not restrictive}}$ of the form ≡ $\underline{\underline{w^T x}}$.

$\Rightarrow Y = Aw \Rightarrow$ To find w ≡

Y: $N\times 1$, A: $N\times d$, w: $d\times 1$

$w = A^{-1} Y$ → 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

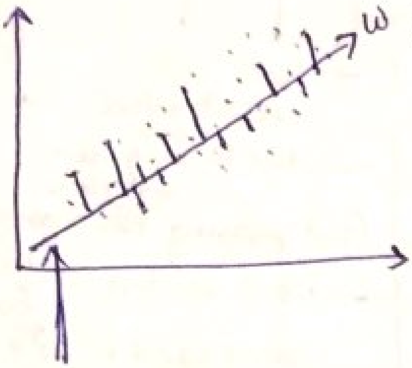
→ Doesnt minimize perpendicular (orthogonal) dist.

* $(A^T A)^{-1}$ ≡ 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, $\frac{\partial E}{\partial p} = 0$

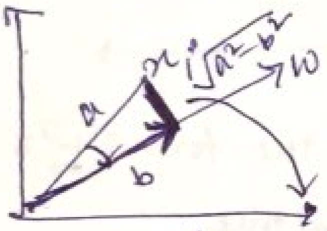
$$\frac{\partial}{\partial p} \sum_{i=1}^{N} \left(\underbrace{x^T x}_{0} + \underbrace{p^T p}_{2p} - \underbrace{2p^T x_i}_{2x_i} \right)$$

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

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

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

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



Normalized.

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

Norm of x's are fixed.

$$\sum_{i=1}^{N} \left(\underbrace{\|x_i\|^2}_{\text{independent of } w.} - (w^T x_i)^2 \right).$$

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

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

$(xw)^T xw$

$w^T x^T x w$

$$\boxed{\begin{array}{l} (w^T x)^T (w^T x) \\ x^T w w^T x \quad ? \end{array}}$$ wrong.

Max: $w^T x^T x w$.

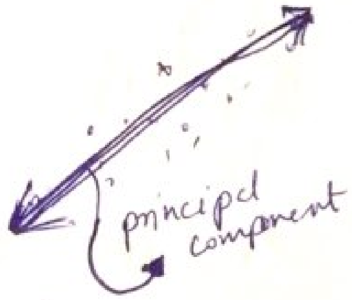
← look at regression.

Maximize ≡ $w^T X^T X w$ st $w^T w = 1$

↓ w is eigen vector of $(X^T X)$ ≡ PCA

(largest) eigen value ← covariance matrix

⇓ why?



$Ax = \lambda x$

$$\text{Max} (~~*~~ w^T \underbrace{X^T X w}_{\lambda w})$$

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

↓ 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 ↓ eigen value.

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

↓

Mainly used in PCA.

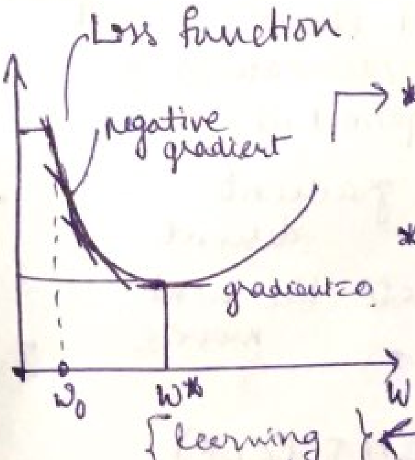
Gradient Descent ⇒

(x_i, y_i), i=1 . . . 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 termin^n.

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

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

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

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

R $\{0,1\}$

C

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

$$\begin{bmatrix}x^1\\x^2\\x^3\\x^4\end{bmatrix} = w_1x^1 + w_2x^2 + w_3x^3 + w_4x^4$$

$w^Tx \equiv$ Not restrictive

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

→ Find the w suiting those dimensions/features.

5d → In five dimension, it is linear.

Gradient Descent:

> Make a random guess.
↓
you should either increase w/decrease w to reach optimal w.

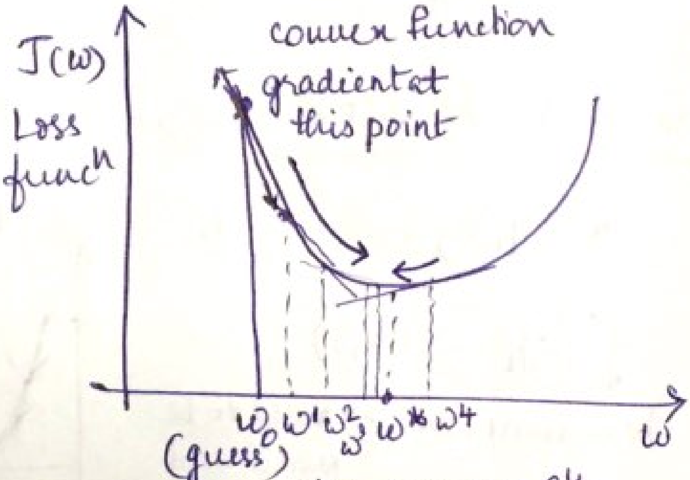
> check the gradient descent and calculate the next move ≡

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

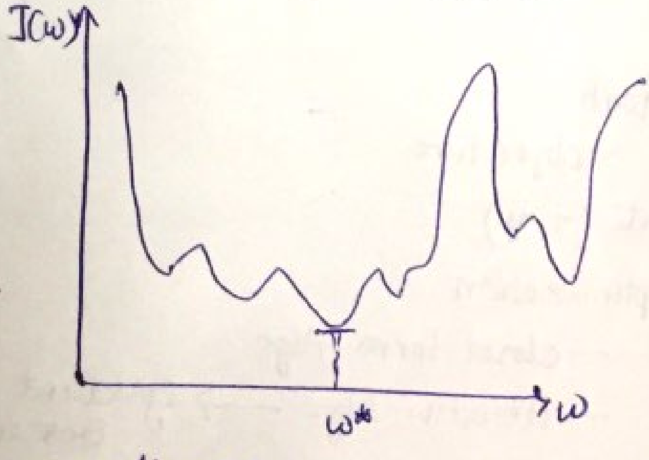
↓ amount of movement.

At the minimum;

$\Delta J(w) = 0$ and w, doesnt change → Algorithm converges



Non-convex f^n



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

$\nabla J(w_0) < \underline{0.001}$ → changing it doesnt give much af an advantage

In nonconvex function; the (local) $\underline{\text{minimum}}$ that we get $\underline{\text{maynot}}$ be the $\underline{\text{best}}$ one by using the same equation

What is $\underline{\eta\text{(eta)}}$? — Adaptive over time / iterations based on the behavior of algo.

↓

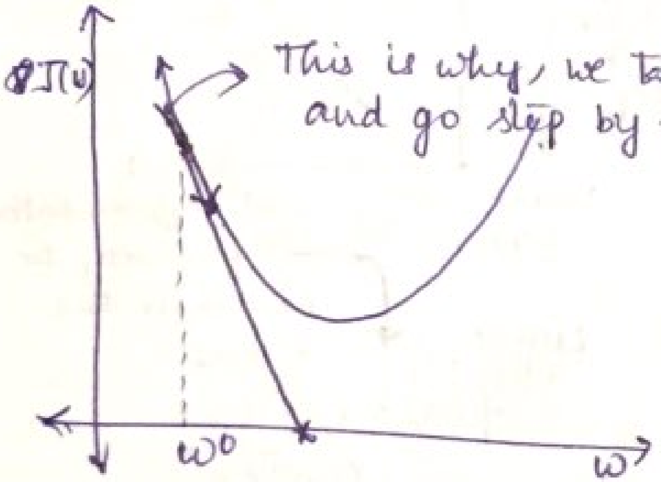
Learning Rate

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

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

when we make a move into the negative gradient

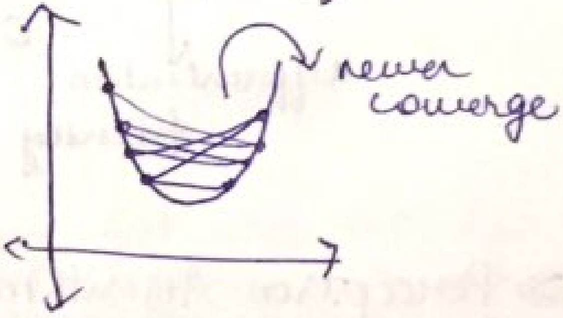
— Assume the function to be a line at the initial stage. and compute the slope.



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 prev step and present step are same.

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

$\underline{\text{for Ex}}$:

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

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

Taylor series expansion ≡

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

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

$$f(w^{n+1}) = f(w^n) + [w^{n+1} - w^n]\,\cancel{8}f'(w^n)$$

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).

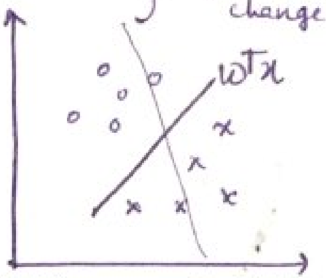
≫ Logistic Regression ≡

≫ ~~Perceptron Algorithm~~ ≡

But instead;

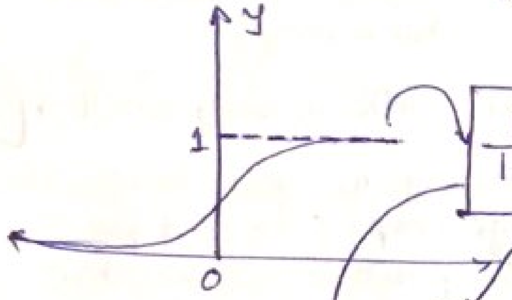
(instead of step) → search for Smooth step

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



minimum error. ≡ not differentiation friendly to eliminate the errors

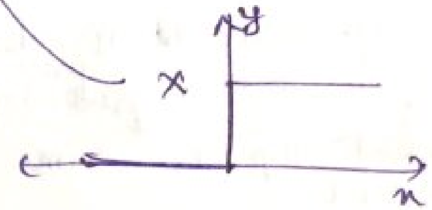
Linear classifier.
$f(x) = 1, w^T x > 0$
$0, w^T x \leq 0.$



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

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

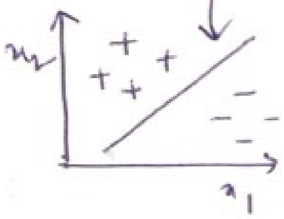
Perceptron Algorithm ≡ creating a new objective function.

↓ every sample thats gets misclassified is added to it

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

If the samples are "linearly seperable", (∃ 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.

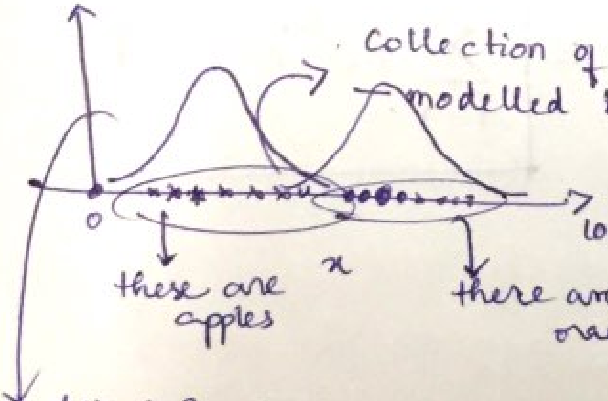
→ Discriminative classifier/model

② Generative Model (we donot model the boundary) ≡ reduced representation
↓ instead model the data [as in "a point"]

[Ex: model oranges and apples independently.]

↓ For Ex: 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.

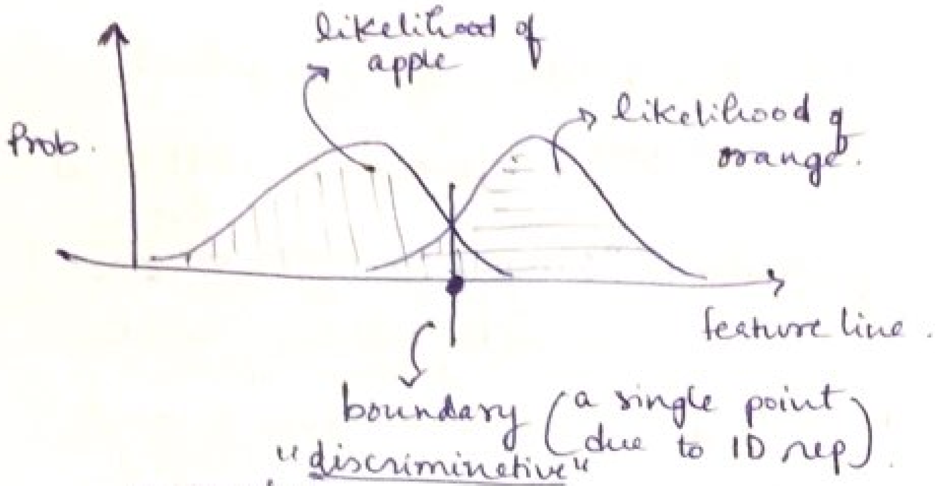
⇒ given density function } → we can generate new points [that is why ≡ generative model]

Discriminative Classifier } ≡ 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]



ω ≡ class

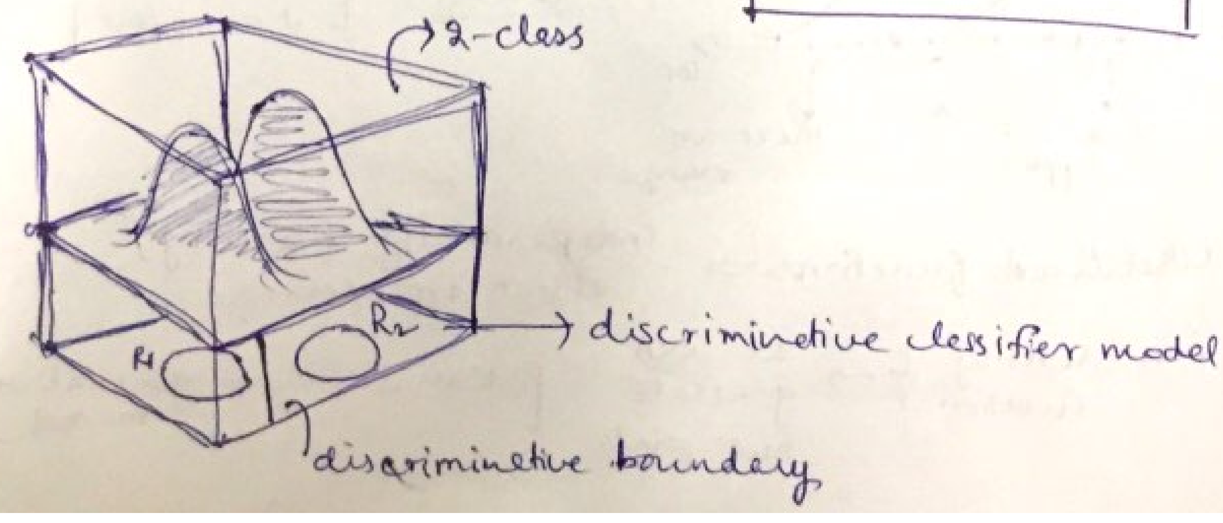
$P(\omega_1/x) \equiv$ "Probability" that given feature vector, x belongs to the ω_1 class. $[0<P<1]$

↓ feature vector

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

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

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



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

p [pdf]

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

(independent events)

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

$$\underset{\uparrow \text{ Joint Probability}}{P(\omega_1, x)} = P(\omega_1/x) \cdot p(x) \text{ (or) } p(x/\omega_1) \cdot P(\omega_1).$$

$$\boxed{\therefore \underset{\text{Posterior prob.}}{P(\omega_1/x)} = \frac{p(x/\omega_1) \cdot P(\omega_1)}{p(x)}}$$

→ Bayes Theorem.

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

→ Prior probability / <u>belief</u>

⇒ $p(x/w_i) \Longrightarrow$ likelihood.

⇒ $p(x) \longrightarrow$ 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)

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

Prob of having disease, given test came out +ve

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

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

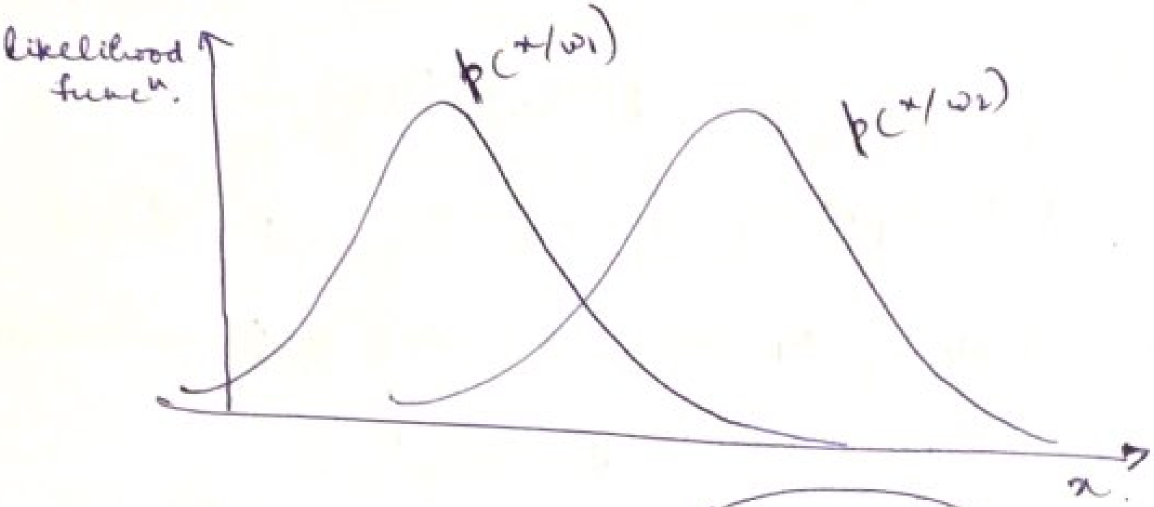
$P(+/D)$: person with D gets +ve result

$P(+/ND)$: person with ND

$$= 0.99 \times 0.001 + 0.01 \times 0.999$$
$$= 0.01 (0.099 + 0.999) = 1.098 \times 0.01$$

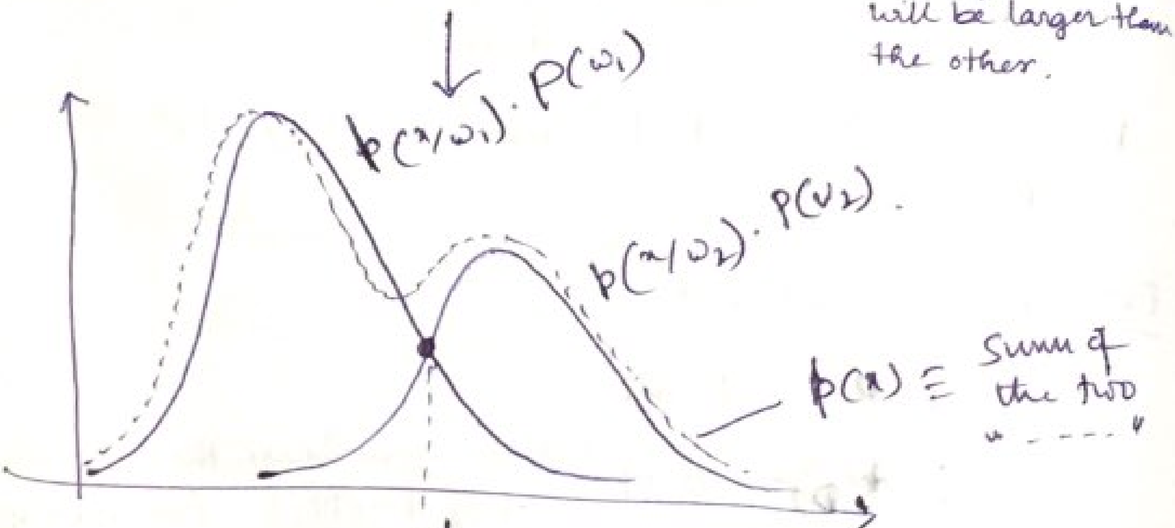
$$P(D/+) = \frac{0.99 \times 0.001}{0.01098} = 0.09 \ (9\%).$$

↑ chance of having the disease.



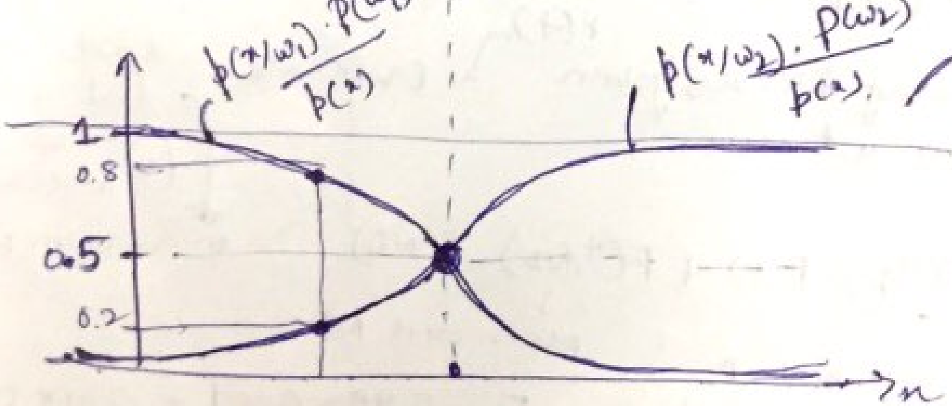
$$P(\omega_1/x) = \frac{p(x/\omega_1)\cdot P(\omega_1)}{p(x)}$$

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 ≡

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

after observing x, state q nature ≡ ω_j

probability thet state of nature, ω_j is true.

Pick ω_j thet has maximum prob.

Are all erross equelly costly? <u>No</u>

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

action, α_i

state of nature ≡ 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.

$E_1 \neq E_2$

→ there can be many actions

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

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

Prob that ω_j is true

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

c classes.

Strategy to pick the best action ≡ based on action plan

$\alpha(x)$ — if i see this observation, i'll do this.

given an x, what action needs to be taken.

Evaluate the action plan?

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

prob of observing x

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

"Action plans can be eval. based on risks"

— Integrate over all possible x,

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

To Minimise overall risk, by minimize $R(\alpha(x)/x)$ for 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^* = Bayes Risk.

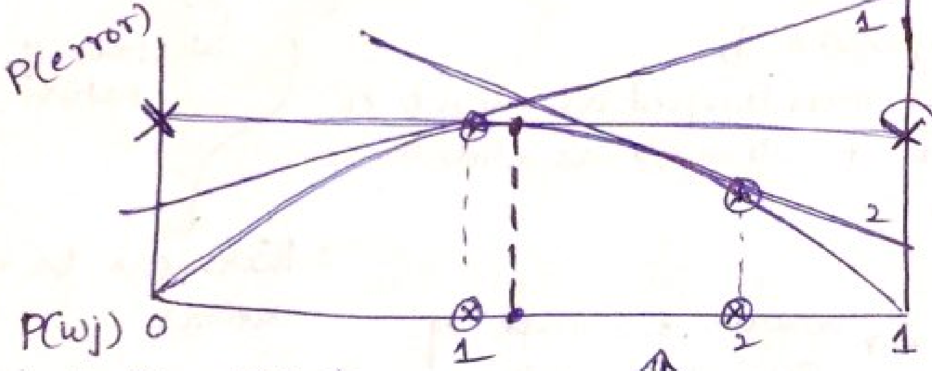
↖ Bayes's Action Plan.

↑

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

Risk ∝ P(error).

* $P(\omega_j)$ → can change (current assumptions)



Change in Risk = Linear vs ground truth

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

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

↑ "Neyman-Pearson criterion"!

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

$p(x)$ → 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 g either compute $g_i(x)$ or $g_i(x)$ (latter one) fquation of Normal density ⇒ $p(x) = \frac{1}{(2\pi)^{d/2}|\Sigma_i|^{1/2}} e^{-\frac{1}{2}(x-\mu_i)^t \Sigma_i^{-1}(x-\mu_i)} = P(x/\omega_i)$

(can be ignoved) constant

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

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

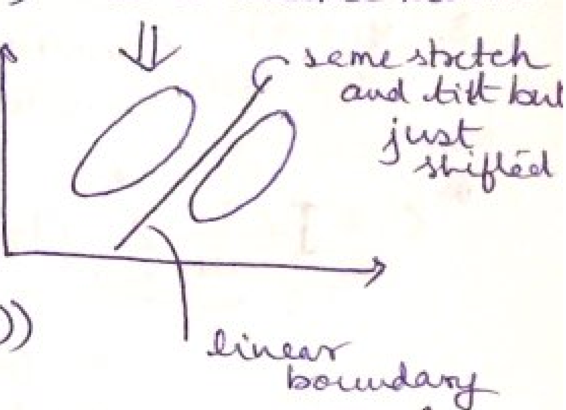
* $g_i(x) = -\frac{1}{2}\ln|\Sigma_i| - \frac{1}{2}(x-\mu_i)^t \Sigma_i^{-1}(x-\mu_i) + \ln P(w_i)$

Case 2: if $\Sigma_i = \Sigma$ (both the classes), if Kashmirand washington, variance may be same, same covariance matrin

⇓

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

$$= -\frac{1}{2}(\underbrace{x^t\Sigma^{-1}x}_{\text{independent of i, class}} - 2\Sigma^{-1}\mu_i^t x + \mu_i^t \Sigma^{-1}\mu_i) + \ln(P(w_i))$$



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

↓ tends to be "lineer"

Discriminant function.

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

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

In 3-D, it'll be a plane

* $\Sigma_i \neq \Sigma$

⇒ if covariance matrix is not the same, then the decision boundary will no more be a lineer function but a complex function. (it coulde be an elipse, hyperbole ...)

Case 3: $\Sigma_i = \Sigma_i$

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

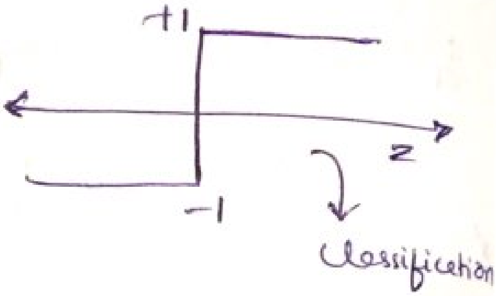
Loss :

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.

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

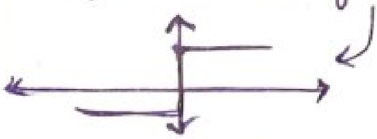


1. $y_i = +1$ $w^T x = 10$. ✓ correct $= 1 = y_i(g(w^T x))$

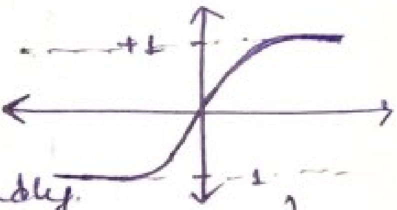
2. $y_i = -1$ $w^T x = -10$. ✓ correct $= 1 = y_i(g(w^T x))$

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

For this, instead of using



its better to use

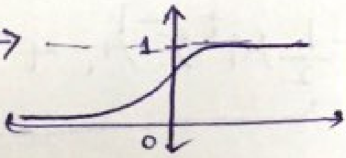


this is (Not Logistic Regression)

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

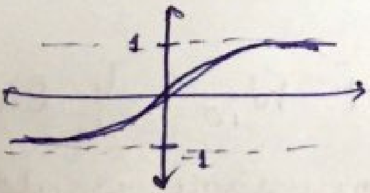
— Good
— Not optimisation friendly.

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



[Logistic Regression]

(or)



Gradient Descent:

1. start with w^0, $n = 0$
2. $w^{n+1} \leftarrow w^n - \underline{\underline{n}} \nabla J$ ↳ changes, when loss function changes
   → varies over time (iteration) [usually constant]
3. $n \leftarrow n + 1$
4. Repeat 2-4 until some <u>convergence criteria</u> is met.
   ?
   — Change in w is very very small.
   — change in loss is very small.

→ Loss function is not linear but the discriminant funcⁿ is linear.

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

approximated ≡ $f(y) = f(x) + (y-x)^T \nabla f(x) + \frac{1}{2}(y-x)^T H [y-x]$ ≡ $\boxed{H \text{ Hessian } d \times d \text{ matrix}}$ — operator matrix

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

$$\downarrow$$

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

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

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

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

only proves that

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

⇒ GD reduces the loss in each iteration if $\underline{\eta \text{ is +ve}}$ and $\underline{\text{small}}$

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)

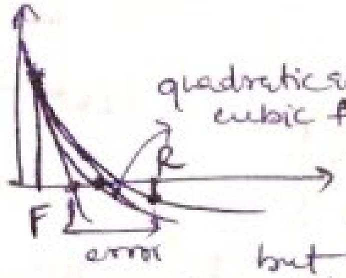
↳ Not used → maybe H might be singular. ✓
(inverse doesnt exist)

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 (bigger 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 ⇒ until no class is misclassified

How does this work? (Intuitive Approach)

$w^Tx \geq 0$, +1

$w^Tx < 0$, −1

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

initially $\underline{\underline{w < 0}}$ as $x > 0$ & $w^Tx < 0$

↓

with these values, w's going to increase ⇒ increase w

↳ w becomes +ve

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

$w^Tx < 0$ ⇕ $x < 0$ $\underline{\underline{w > 0}}$

$(t-o)x$ ⇒ −ve ⟶ w decreases.

↳ w becomes $\underline{\underline{-ve}}$

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

→ 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 − $\underset{w}{\text{argmin}} \sum_{x \in \epsilon} -y_i w^Tx$ ≡ OBJ Funcⁿ (misclassified samples)
- ∗ Logistic Regression

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

Algo depends on ∇J

Algo :-

∗ Initialize w^0, $k = 0$.

$w^{k+1} \leftarrow w^k + \eta y_k x_k$, if $x_k \in \Sigma$ → change is done per sample?

$k \leftarrow k+1$

Repeat until Σ is empty.

3 variations = → solⁿ changes with every sample (update for every sample)

- ∗ Online (imp: data streaming (continuously flowing samples))
- ∗ Batch ⟶ "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.

Mini Batch

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

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

| mini batch stochastic batch gradient |

* miniSGD.

↑ sample has to be replaced back

> online is a form of stochastic.

* 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 — ∃ '$\underline{w}$' such that data can be correctly classified.

↑ Cannot have a linear separating hyperplane for every problem

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

Termination criteria of perceptron. — ξ should be empty

↓ 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 $w^0 = 0$)

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

$$\vdots$$

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

→ $w^{*T} w^{k+1} > (k+1)\gamma$

↙ $w^{k+1} = w^k + y_i \, w_i^T x_i$

Squaring on both the sides. ≡

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

$\|x_i\|^2$ → Bounded

↓

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

→ (Perceptron algo guarantees that it converges in finite no. of steps)

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

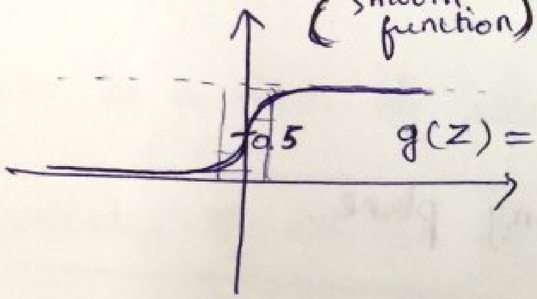
"(Converge in these many finite no of steps)"

* γ → large ; K → converge faster

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

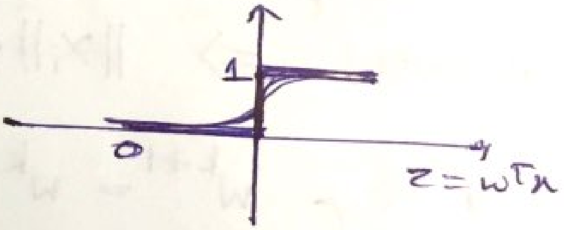
Logistic Regression ⟹

(smooth function)



←

$g(z) = \dfrac{1}{1+e^{-z}}$ ↘ can be $\dfrac{1}{1+e^{-kz}}$ (gets more closer to the axis)



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

↓

$P(y=1/x)$

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

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

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

↓

⇒ 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 ←

$J = \sum_{i=1}^{N} (y_i - g(w^T x_i))^2$ × Not a good function (due to local minimes)

either 1/0

smooth curve $= \frac{1}{1+e^{-z}}$

differentiable but it is not convex ?

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

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

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

→ ① * $J =$ objective function + $\boxed{\text{Regulerizer}}$?

*usually a function of w

additional constraints beyond objective function.

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

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

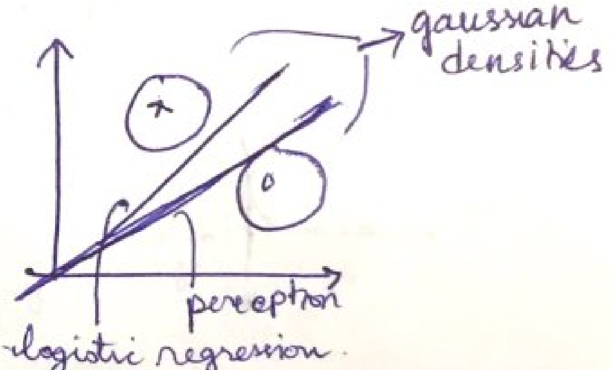
* [combining many binary to make a multi-class]

Perceptron :- (Recap) ⟶ finds the classifier line

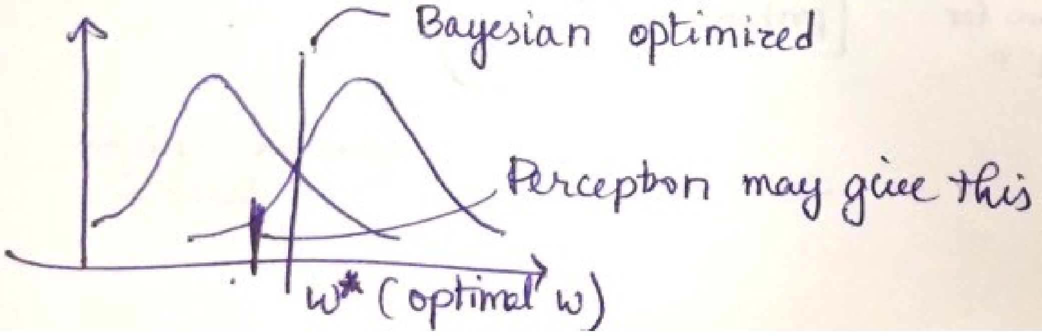
$$J = \sum_{x_i \in E} (\text{Loss})$$

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

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



may give one line which may be very close to the boundary (but not optimal in baysian sense)



Perceptron should work for future data also.
↓ (all the samples in the future)
"generalize"
(requirement).

* Computation was done on existing training data but the model has to be extended for future values also. — ?

$y_i w^T x_i > \underline{\gamma}$ [classified samples]

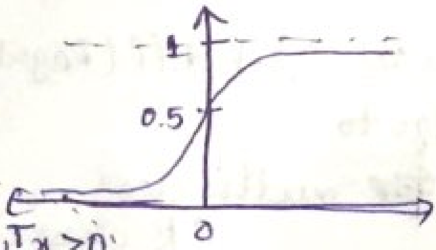
↓ at least γ away from the decision boundary.

Maximize "γ", (margin)

↓ why? make sure it is less prone to error in the case of future values

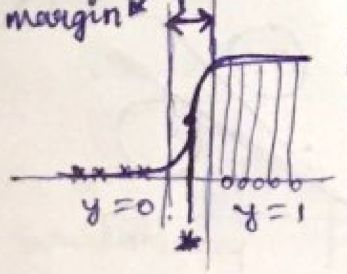
Logistic Regression ⇒

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

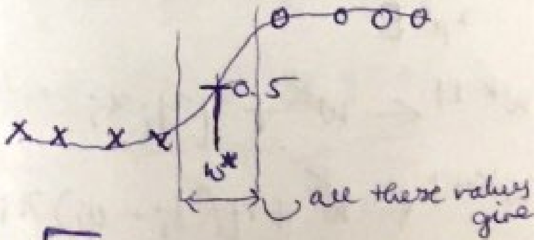


if $g(w^Tx) > 0.5$, $y = 1 \equiv w^Tx > 0$,
else $y = 0 \equiv w^Tx < 0$ → again the same classification.

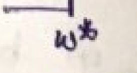
$P(y=1/x)$ — given x, probability that it belongs to class 1



$P(y=1/x)$



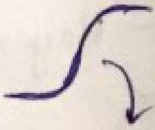
(Not the best) gives the first solution and stop



→ slide this (there exists a region)

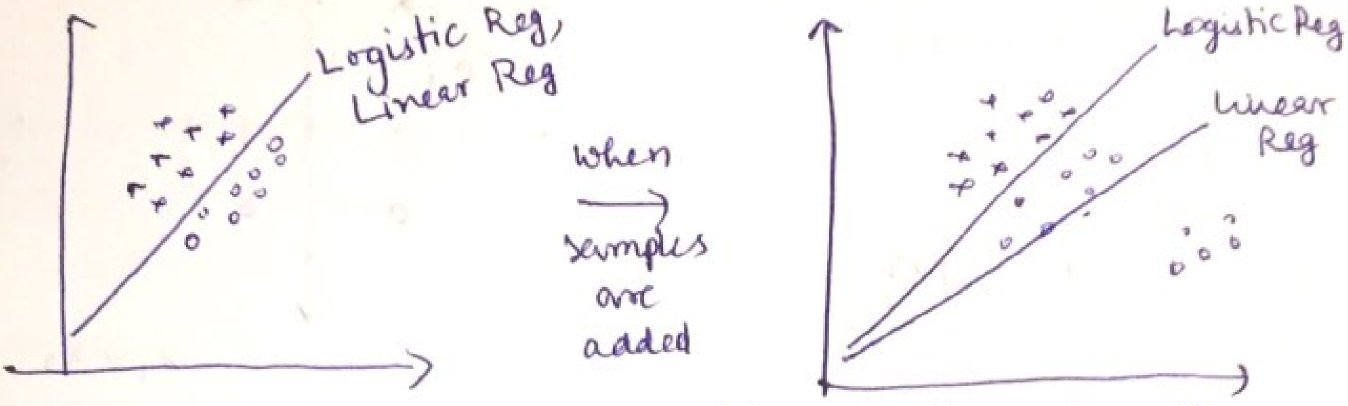
$\prod P(y=1/x_i)$ and also for $y=0$.

to maximize the product [probabilities]

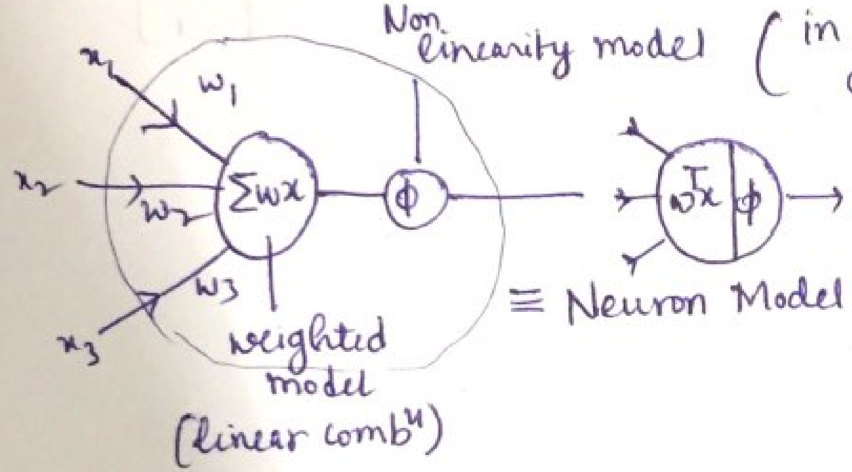


gives an unique solution

Asymmetric costs } – costs of misclassifying samples.



Linear Regression changes but logistic regression gives the optimal.



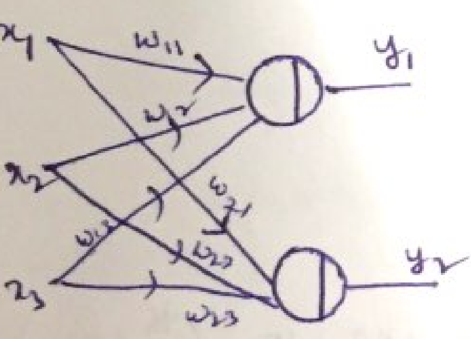
(in this case, a step)

$$\begin{cases} w^Tx > 0, & y = 1 \\ w^Tx < 0, & y = -1 \end{cases}$$

≡ Neuron Model (Perceptron works the same way)

⦶ Non linearity is presen

Perceptron



For Ex: Sigmoid, $\left(\frac{1}{1+e^{-w^Tx}}\right)$ step..

$$\Rightarrow \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \phi\left[\underset{2\times3}{\begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \end{bmatrix}} \underset{3\times1}{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}}\right]$$

$$\Downarrow$$

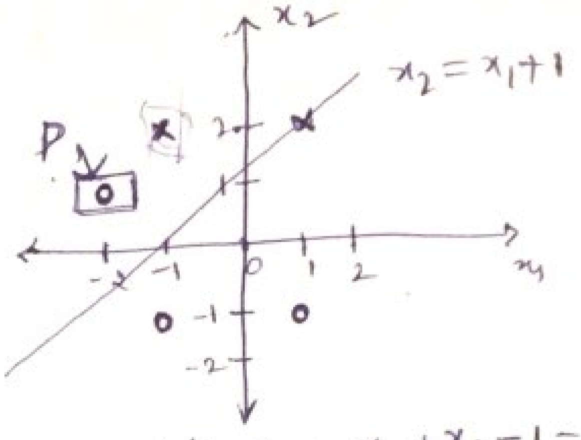
$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \phi \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$$

where let ϕ = sigmoid

$$y_1 = \frac{1}{1+e^{-z_1}}$$

$$y_2 = \frac{1}{1+e^{-z_2}}$$

Samples = { (1,2) +1 , (-1,2) +1 ,
(-2,1) -1 , (-1,-1) -1 ,
(1,-1) +1 }



Line ≡ $-x_1 + x_2 - 1 = 0$

$$\begin{pmatrix} w_2 \\ w_1 \\ w_0 \end{pmatrix} W_0 = \begin{bmatrix} +1 \\ -1 \\ -1 \end{bmatrix}$$

$$\boxed{\eta = 0.1}$$

$$P \equiv \begin{array}{c} \begin{array}{ccc} w_2 & w_1 & w_0 \end{array} \\ [1 \ -1 \ -1] \end{array} \begin{array}{cc} \begin{bmatrix} +1 \\ -2 \\ 1 \end{bmatrix} & \begin{array}{c} x_2 \\ x_1 \\ x_0 \end{array} \end{array}$$

(-2,1)

$o = w^T x \equiv 1 + 2 - 1 = 2 > 0 \Rightarrow \underline{+1}$

$t = -\underline{1}$

P (misclassified)

$\eta \ Y_i X_i$

$$w^1 \leftarrow w^0 + \underbrace{0.1 \times (-1)(x_p)}$$

$$w^1 = \begin{pmatrix} 1 \\ -1 \\ -1 \end{pmatrix} - 0.1 \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix} = \begin{bmatrix} 0.9 \\ -0.8 \\ -1.1 \end{bmatrix}$$

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

<u>Multiclass</u> :

* 2class — "w"
kclass — ?

$P(Y=c/x)$
$\uparrow$

$$> \frac{e^{w_c^T x}}{e^{w_1^T x} + e^{w_2^T x} + \dots e^{w_k^T x}}$$ (probability of being in c^{th} class)

$$> \frac{e^{w_c^T x}}{\sum_{i=1}^{k} e^{w_i^T x}}$$ (Soft Max)

In the case q 2 classes;

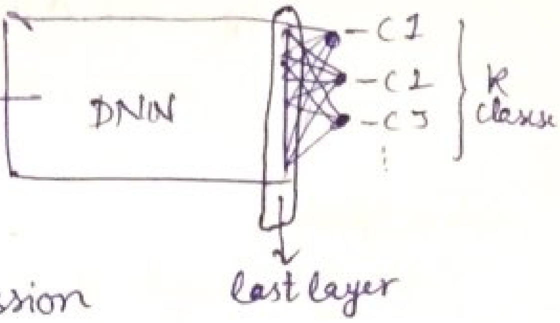
$$\frac{e^{w_1^T x}}{e^{w_1^T x} + e^{w_2^T x}} = \frac{e^{(w_1 - w_2)^T x}}{1 + e^{(w_1 - w_2)^T x}} \doteq \boxed{\frac{e^{w^T x}}{1+e^{w^T x}}}$$

Problems - multiclass

Solutions - binery.

uses soft max type of classifier

with logistic regression kind of loss.



K-classes

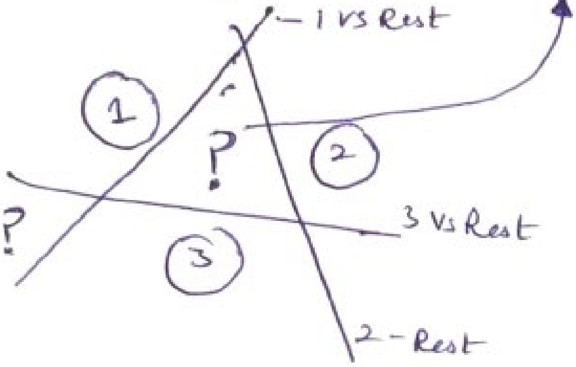
① K 1vs Rest classifier

(one class is positive whereas all the others are negative)

② $^KC_2$ 1vs 1 classifier

test for all K classifiers.

In, K 1vs Rest classifiers → what if all the classes say negative ("Rest")?



gives ambiguous results.

$^KC_2$ 1vs1 classifiers ⇒ majority voting

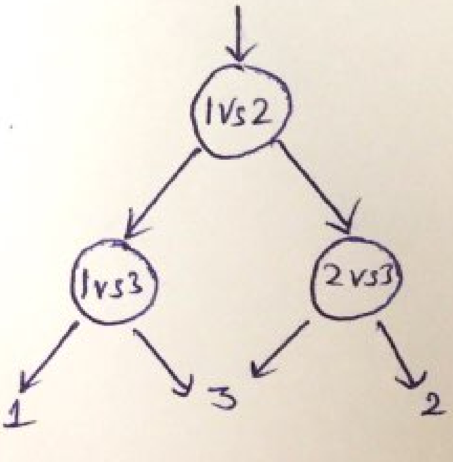
more computational complexity

(For Ex: 1000 class classification).

vs

DAG → [Evaluation complexity ≡ O(k)]

DAG — Directed Acyclic Graph.



Regularize Objective Function:-

define a new objective function "regularized"

$$J_R = J + \lambda \|w\|^2 \longrightarrow \text{small.}$$

[Ex: perception, logistic Regression, SVM]

$$J_R = J + \lambda \underset{\substack{\downarrow \\ \text{small.}}}{\|w\|^2} + 1000 \underset{\substack{\downarrow \\ 0}}{w_3^2} + 16 \underset{\substack{\downarrow \\ 0}}{w_4^2}$$

(to remove a lot of dimensions / features)

— computationally good at the test time.

W → sparse

(lot of elements = 0, supress ↑ features)

→ Find w as sparse as possible.

can use L_1 norm.

✓ L_2 norm (easier) ⇒ this to make it smell

($\|w\|^2$)