- ML is one of the many ways to achieve AI
- Deep learning ⊂ ML is yet another way to achieve AI
- GOAL OF ML
    - To create an algo to teach machine based on prior knowledge.

- Subset of AI
- Teach the machine on basis of prior knowledge
- Train a m/c to be able to make decisions
- Train a m/c to generalise from prior data (goal is to predict for future)
- Construct a model whose efficiency increases as we spend more time training it. ↳ incorrect term
  decision making accuracy

ML FORMAL DEF^N

Study of algos that allow computer programs to automatically improve through experience (Tom Mitchell) *

Role of statistics: inference from a sample
Role of CS: efficient algos to — 1) solve optimisat^n problem
2) representing & evaluating model for inference

* optimise a performance criterion using example data or past experience.

SEEMINGLY RELATED FIELDS TO ML

- Statiscal learning
- Pattern Recognition
- Data Mining
- Deep learning

] CONTRIBUTE TO AI IN SOME WAY

COMMON OBJECTIVE: analyze data with either of 2 goals
- prediction
- description

STATISTICAL LEARNING (SL): mostly about inferences & parameter estimations, while ML emphasises supervised, unsupervised, semi-superv:, online, reinforcement learning

Statistical learn: is based on smaller dataset (few dimensions) compared to ML (high dimension)

Extent of assumptions are more stringent in statistical learn, while ML is not that assumpt^n dependent

SL & ML differ in origin, history & emphasis.

PATTERN RECOGNITION (PR): typically an engg. problem to predict-based on patterns hidden in data

ML is a solution to solve problem of PR

Date :

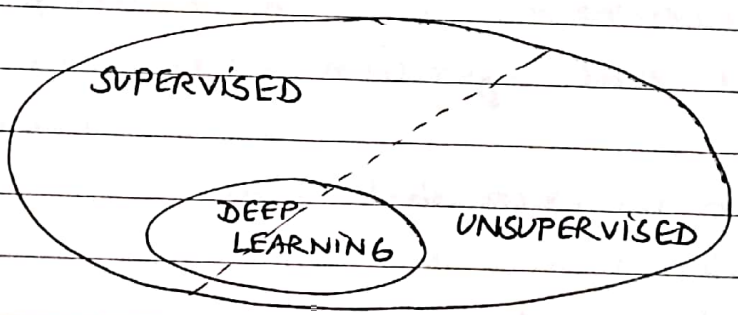
ML v/s DM

- DM is a process, while ML deals with science of creating algo that will make machines learn to predict

- DM is interactive & iterative process, while once a ML algo is designed, no human intervention is required.

- DM focused on scalability since its inception in 80's, ML has addressed scalability recently. (since deep learning came 5-6 yrs ago).

- Traditionally, ML didn't consider data cleaning etc. This routine of data cleaning was adopted recently.



MACHINE LEARNING

- Deep learning has effective architectures making AI a household thing now.

There is no need to learn in case of finding prime no.

learning is different from normal coding

learning is reqd. when human expertise doesn't exist (eg: unknown territory) or when humans are unable to explain their expertise (face & speech recognition) or when env. changes with time. (driving on highway v/s in city) or when solut^{n} needs to be adapted on case to case basis (diagnosing disease)

Date :

Deep Neural n/w machines make very accurate decisions but almost all of them fall short in explaining them.

Machines can't explain but can figure out the patterns humans can't

HYPOTHESIS: Build a model that is a good & useful approximation to the data.

Machine learning algo generalises an outcome based on specific instance values (eg: for a malignant cell cell wall thickness lies b/w 0·04 & 0·06).
It tries to establish rel^n b/w attributes

Machines only have power to model. From the data provided to machine, it must create a function $f(\underbrace{x_1, \ldots x_n}_{n \text{ attribute values}}) \rightarrow \{\text{dis}, \text{not dis.}\}$

Finding such a function results in abstraction
ML Algo attempt to approximate this function ABSTRACT
~~A better~~ Approximat^n & Accuracy

Gini, Info gain, Gain Ratio are 3 diff. hypotheses for the same task in decision tree

Data are cheap & abundant, labelling is often expensive

APPS OF ML

- Association
- supervised learning
  - classification
  - regression
- unsupervised learning
- Reinforcement learning (α-GO) — typically used for games

CLASSIFICATN : diff b/w low risk & high risk customers from their income & savings

hypothesis → discriminant : if income > θ_1 & savings > θ_2
THEN LOW-RISK
ELSE HIGH-RISK

REGRESSION : hypothesis → eqn of line

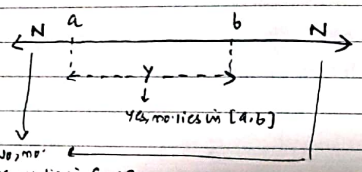
Date :

CHAPTER-2 SUPERVISED LEARNING

MACHINE LEARNING → Theoretical ML
→ Applied ML

Goal of ML is to abstract a funct^n but since it can't be predicted accurately, we approx. it

GUESS THE INTERVAL [a, b]



A hypothesis must be consistent with data
↓
if there are no conflicts

More the data, better the hypothesis

GIVEN DATA			
NO'	INSIDE	Hypothesis 1	Hypothesis 2
65	Yes	$(-\infty, \infty)$	$[0, 100]$
125	Yes	$(-\infty, \infty)$ ✓	✗ becomes inconsistent
		↳ we improve it as we have more data	
-80	NO	$[0, \infty)$ ✓	we make it $[0, \infty)$
		↳ we can improve it as $[-79, \infty)$	
-53	Yes	$[-79, \infty) \rightarrow [-60, 200]$	
		random values to tighten the range	
101	Yes	$[-60, 200]$	
110	NO	→ creates conflict with 125, Yes ∴ this is considered as noise as it creates inconsistensies in data	
153	No	$[-60, 152]$	

Date :

we evaluate a hypothesis using some metric
Error decreases as examples increase.
classes are called concepts.

concept learnt → inside the interval
complementary concept → outside the interval

we evaluate a hypothesis using some metric
Error decreases as examples increase.
classes are called concepts.

concept learnt → inside the interval
complementary concept → outside the interval

GOAL OF SUPERVISED LEARNING

- Accurately predict class labels for unseen instances
- Understand which features are important for class & how
- Assess quality of classifier

predictor: class in classificat^{n}, no. in regression
Pattern recognition came before ML

- Concept is subset of instance space X

For predicting whether a car C is family car (Y) or not (N)

PRICE	ENGINE POWER	FAMILY CAR
—	—	Y
—	—	N
—	—	Y
—	—	Y
—	—	N
—	—	N

→ Instance space, Training set
→ concept of family car
→ concept of n-family car

Date :

learning of concept : Find a description that's shared by all instances of class & NOT by any example of the other class.

For training set X

$$X = \{ \underbrace{x^t}_{\text{vector of features}}, \underbrace{r^t}_{\text{label for } t^{th} \text{ instance}} \}_{t=1}^{N}$$

as there are N instances in training set

x^t is a vector of features

r^t label for t^{th} instance

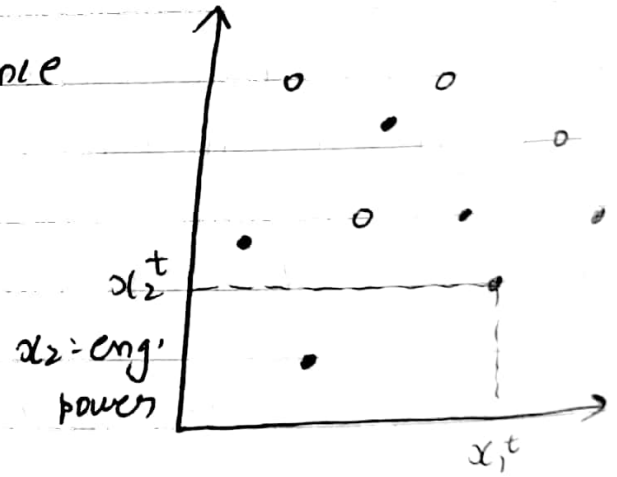
$$x = \begin{cases} x_1 \rightarrow \text{feature 1} \\ x_2 \rightarrow \text{feature 2} \end{cases}$$

in family car eg, price is a feature vector & so is engine power

x_1^t : x_1 feature of t^{th} instance

x_2^t : x_2 " " " "

$$r^t = \begin{cases} 1 & \text{if } x \text{ is +ve} \\ 0 & \text{if } x \text{ is -ve} \end{cases}$$



To learn a concept, of like 'yes' for family w car, we must find constraints s.t

$(p_1 \le \text{price} \le p_2)$ & $(e_1 \le \text{eng. power} \le e_2)$

There can be ∞ such constraint values, hypotheses

this forms a rectangle

All possible rectangles form hypotheses space

Date :

HYPOTHESES : educated guess
HYPOTHESES SPACE : set of possible hypotheses

GENERALIZATION : How well does hypotheses capture specific objects in training set to predict unseen objects (loss of precision & sometimes gain in understanding of description)

Machine can't be made intelligent by memorization

PAC : Probably Approximately ~~Let~~ Correct

EMPIRICAL ERROR : Model that machine constructs from data is just an approximation. ∴ there is some error which must be measurable, this measurement is empirical error

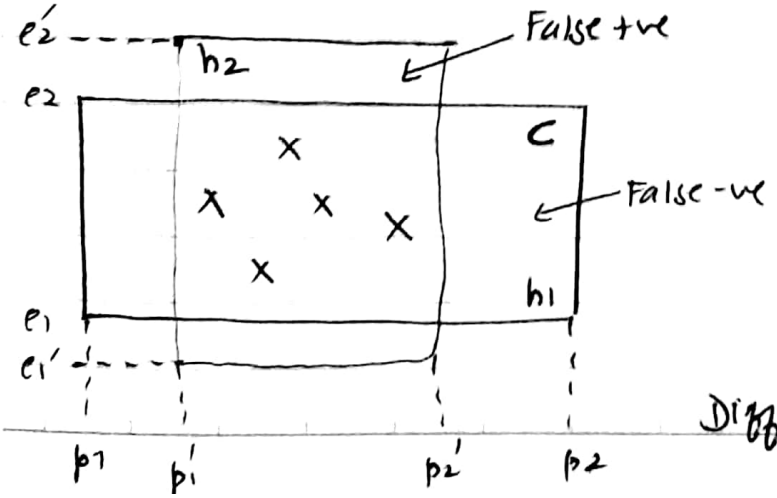
In predictive modelling, approximat^n is a theoretical concept

hypothesis : h
Hypothesis space : H s.t $h \in H$

→ Error is of a hypothesis h $= E(h|X) = \sum_{t=1}^{N} 1$ (if $\underbrace{h(x^t)}_{\text{predicted class}} \neq \underbrace{r^t}_{\text{actual class}}$)

↑ indicator funct^n

↓ denotes how many errors



Choice of algos & parameters defines the hypothesis space

Diff. algos have diff. hyp. space

Date :

task of algo is to navigate through H to obtain the best hypothesis
This makes ML as an optimizatⁿ problem

Date :

CHAPTER-3

BAYESIAN DECISION THEORY

Bayesian learning is probabilistic learning.

Estimat^n: p_0 (for heads) $= \frac{\#\text{ Heads}}{\#\text{ Tosses}} = \frac{\sum_t x^t}{N}$

$x^t = 1$ for heads
$\quad\quad 0$ for tails

Prediction of next toss : heads if $p_0 > 1/2$, tails otherwise

Bernoulli Thm HW

Bayesian learning combines prior knowledge with observed data & provides practical learning algos :

1) Naive Bayes learning
2) Bayesian Belief N/w learning

Bayesian learning provides flexible approach to learning compared to algos that eliminate inconsistent hypothesis. (don't reject directly)

Provides "gold standard" for evaluating other learning algos, even though sometimes they may be computationally intractable

Occam's razor HW

Bayesian learning provides the bound of error

Date :

How to decide class without seeing x?

A: Acc. to Bayesian theory, use prior probabilities of classes.
- choose C_1 if $P(C_1) > P(C_2)$, else C_2

If $\exists K$ classes, choose the class with max. prior probability.

seeing features of an instance will lead to better decision making

COURSE PREDICTION PROBLEM

No. of Eco students = 800 No. of engg. students = 1500
observe marks of Maths in 12TH (we are given marks of maths)
collect data
Plot histogram of Marks in Maths v/s count / Probability for eco & engg. separately & notice the density of distribution

Notice average marks in both (assuming avg in E = 78, B = 88)
Assume avg. marks in Eco in Maths = 70

$P(M=70|E)$ $P(M=70|B)$ → engg.

since avg. of Eco is closer to 70

$\therefore P(M=70|E) > P(M=70|B)$

Andvly $P(M=94|B) > P(M=94|E)$ This doesn't mean that high scorers in M always join B, they are ~~more~~ more likely to join B.

DECISION RULE

HYPOTHESIS : Let there be some marks threshold α

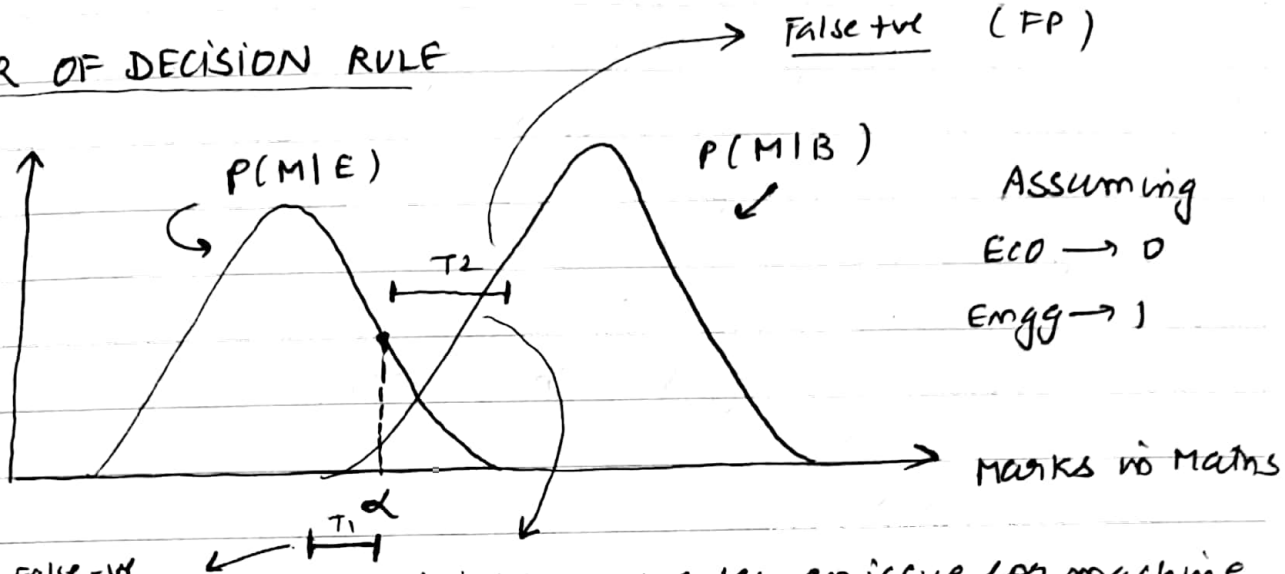
if Marks $\leq \alpha$ → Eco

Else → Engg.

What should be value of α ? → Based on popvlatn statistics

Date :

ERROR OF DECISION RULE

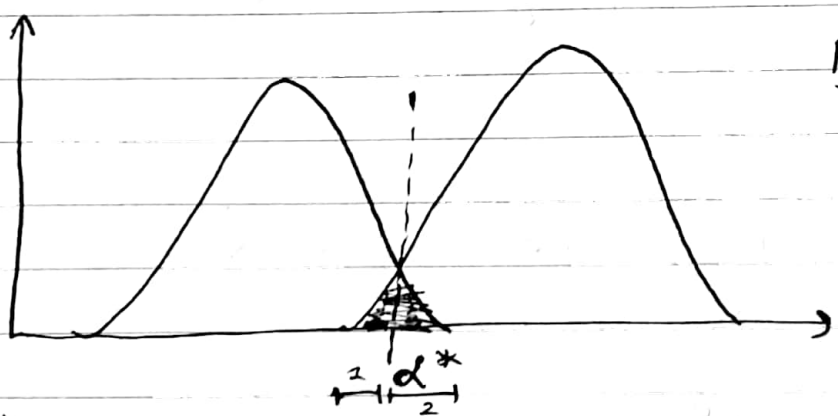


Assuming
ECO → 0
Engg → 1

↳ this range creates an issue for machine in decision making.

T1 : TYPE 1 : Actually BTech, classified ECO
T2 : TYPE 2 : " ECO, " BTech
] ERRORS = T1 + T2

More the area under the ~~error~~ curve, more is the occurrence
∴ FP occurs more than FN



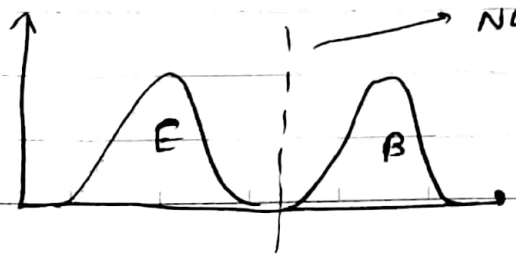
For Min-possible error

If T1 & T2 Errors have diff-costs : optimal boundary shifts

α^* : Optimal value of α : Min-possible error

$P(\alpha^* | E) = P(\alpha^* | MB)$

Mistakes = 1 + 2 = area under overlap curve



→ No mistakes by Bayes classifier
Clear decision Boundary
100 % Accuracy

Date :

The decision boundary in the plot = hypothesis of our model
No classifier gives error < Bayes Error → min. error probability
Bayes Error can't always be computed. It depends on the complexity of data available.

The $ distribution changes with Location / Time of observation
- job opportunities
- cost of courses
- family background

APRIORI PROBABILITY

- without measuring marks, what can we guess about the course student may opt ? → depends on locat^n / time
- Apriori Prob', P(E), P(B) : property of frequency of classes during expt. Not a property of marks of student

PREDICTION

$$\begin{cases} C=1 \text{ if } (P(C=1 \mid x_1, x_2) > 0.5) \\ C=0 \text{ otherwise} \end{cases}$$

OR

$$\begin{cases} C=1 \text{ if } P(C=1 \mid x_1, x_2) > P(C=0 \mid x_1, x_2) \\ C=0 \text{ otherwise} \end{cases}$$

Date :

BAYES' RULE

before we have observed the data

likelihood

prior probability

$$P(C|x) = \frac{P(x|C) \cdot P(C)}{P(x)}$$

$P(x)$ → normalising term

↳ evidence (object that we want to predict)

posterior prob. (after we observe the data)

- $P(C=0) + P(C=1) = 1$
- $P(x) = P(x|C=1) \cdot P(C=1) + P(x|C=0) \cdot P(C=0)$
- $P(C=0|x) + P(C=1|x) = 1$

] for 2 classes

For 'd' dimensions / attributes, we call it joint probability.

$P(x_1, x_2, x_3, \ldots, x_d)$ → joint probability

a random variable X of 'd' dimensions where each dimension can take a value from a range of values. This is an instance of possible combination of values that can be assigned to the dimensions of X.

Frequency = 0 ⇒ Prob. = 0

For plotting a histogram for 2 subjects, assuming X coord = Maths marks, Y coord = English marks, Frequency of student with that many marks in M & E = Height of histogram

For 3 subjects, random variable $X = [x_1\ x_2\ x_3]$
∴ we will have $(100)^3$ possible combinations of values
If we have only 1000 records, then most of the instances will have prob = 0 & most non-zero prob instances would be very less in frequency.

∴ Bayes classifier is computationally intractable.

FOR >2 CLASSES (BAYES' RULE)

we must find posterior prob for each class

$$P(C_i|x) = \frac{P(x|C_i) \cdot P(C_i)}{P(x)} = \frac{P(x|C_i) \cdot P(C_i)}{\sum_{m=1}^{K} P(x|C_m) \cdot P(C_m)}$$

- $P(C_i) \geq 0$ & $\sum_{i=1}^{K} P(C_i) = 1$
- DECISION RULE → choose C_i if $P(C_i|x) = \max_k P(C_k|x)$

BAYES CLASSIFIER

APRIORI : Without any measurement based on just locat^n / time - what can we guess about class membership (estimated from size of class populations)

CLASS COND^{N}AL (LIKELIHOOD) : Given the instance belongs to a particular class, what is the prob that its (combinat^n of) features exist (estimated from training set)

Eg: $P(<80, 65> | E)$ we know the class is E, we check for instances having class = E with instance values = <80, 65>

Date :

optimal Bayes classifier works on Joint prob distribut^n of data
Bayes classifier works on Bayes Theorem

MAXIMUM A POSTERIORI HYPOTHESIS (MAP HYPOTHESIS / HMAP)
choose hypothesis with max posterior probability

MAXIMUM LIKELIHOOD HYPOTHESIS (ML HYPOTHESIS) : If all classes are equally likely choose hypothesis which is most likely

eg: $P(C|x) = \frac{P(x|C) \cdot P(C)}{P(x)}$

$P(C)$ → this is same for all ∴ we classify on basis of $P(x|C)$

$P(x)$ → same throughout ∴ can be ignored

SALIENT FEATURES OF BAYESIAN LEARNING : This is a more flexible approach than many other algos

PRACTICAL DIFFICULTIES : 1) Typically require initial knowledge of many probabilities

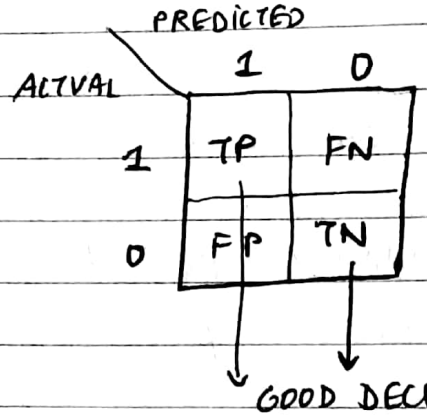
2) Significant computational cost reqd to determine Bayes optimal hypothesis

IMPLICAT^N OF MACHINE DECISIONS (eg: credit risk applicat^n)

→ GOOD DECISIONS
Accept low credit risk app
Reject high credit " "

↓ BAD DECISIONS
Reject low credit risk app
Accept high " " "

Date :



Rejecting low credit risk app = FN ∵ we wrongly reject a good app.

Accepting high credit risk app = FP

LOSSES & RISKS : loss incurred in case of wrong prediction / decision for x

Action α_i : choose class C_i

loss of α_i when correct class is C_k : λ_{ik}

Expected Risk for taking action α_i

$$R(\alpha_i | x) = \sum_i \lambda_{ik} P(C_k | x)$$

choose α_i if $R(\alpha_i | x) = \min_k R(\alpha_k | x)$

LOSS MATRIX λ_{11} : loss related to correct class 1 predicted as 1

possible to define loss funct^n for predict^n problem

Sometimes it's not possible to define all possible combinations

$$\lambda_{ik} = \begin{cases} 0 & \text{if } i = k \\ 1 & \text{if } i \neq k \end{cases}$$

$$R(\alpha_i | x) = \sum_i \lambda_{ik} P(C_k | x)$$

$$= \sum_{k \neq i} P(C_k | x)$$

$$= 1 - P(C_i | x)$$

There is only 1 case when $\lambda = 0$ when $i = k$

For min. risk, choose most probable class

Date :

REJECT OR DOUBT

$$\lambda_{ik} = \begin{cases} 0 : & \text{if } i = k \\ \lambda : & \text{if } i = k+1, \quad 0 < \lambda < 1 \\ 1 : & \text{otherwise} \end{cases}$$

$$R(\alpha_{k+1} | x) = \sum_{k=1}^{K} \lambda P(C_k | x) = \lambda$$

$$R(\alpha_i | x) = \sum_{k \neq i} P(C_k | x) = 1 - P(C_i | x)$$

choose C_i if $P(C_i|x) > P(C_k|x) \; \forall \; k \neq i$ & $P(C_i|x) > 1-\lambda$
reject otherwise

DISCRIMINANT FUNCTN AS CLASSIFIER

- works for both binary & multi-way classification
- for every class $i = 0, 1, \ldots, K$ define a function $g_i(x)$ mapping $X \to R$
- when decision on I/P X is to be made choose class with highest value of $g_i(x)$

DISCRIMINANT FUNCTIONS FOR BAYES CLASSIFIER

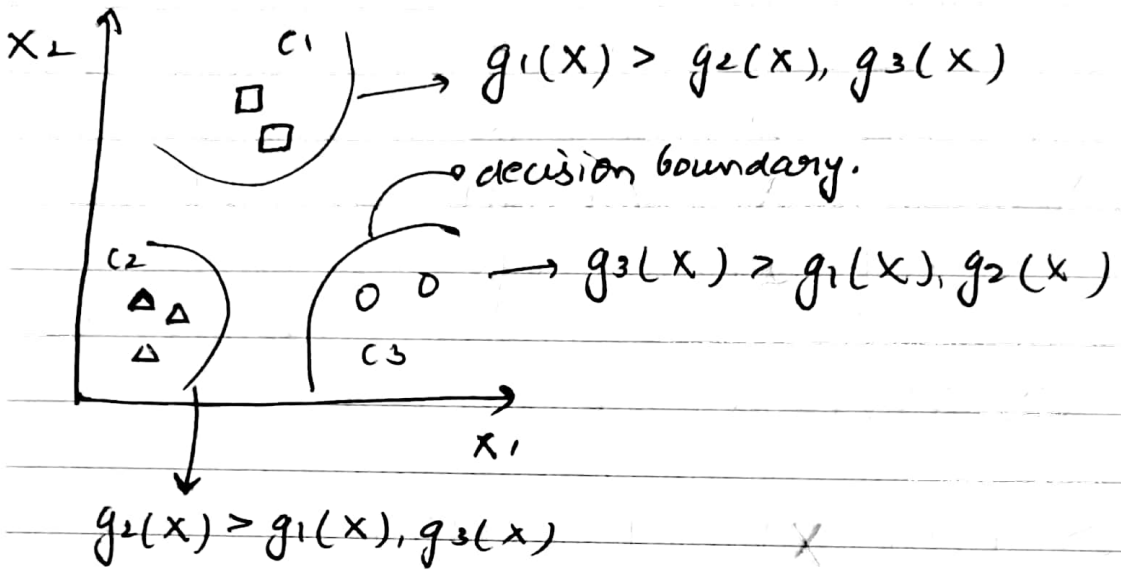
$$g_i(x) = \begin{cases} -R(\alpha_i | x) \longrightarrow \because g_i \text{ is to be maximised \& risk is to be minimized} \\ P(C_i | x) \\ P(x | C_i) \cdot P(C_i) \end{cases}$$

Generates K decision regions $R_1, \ldots, R_K$

$$R_i = \left[x \mid g_i(x) = \max_k g_k(x) \right]$$

$i = k$

Date :



K=2 CLASSES Dichotomizer (K=2) V/S Polychotomizer (K>2)

UTILITY : Utility of α_i if class is $k = U_{ik}$

Expected Utility = $EU(\alpha_i|x) = \sum_k U_{ik} P(S_k|x)$

Choose α_i with max utility

Value of info ∝ increase in expected utility

$EU(\alpha_1|x, z) > EU(\alpha_1|x)$

Date :

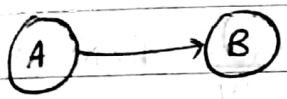
(GM) (UPTO 14.3)

CHAPTER-14 GRAPHICAL MODEL (PROBABILISTIC LEARNING)

→ representat^n method for data where Nodes = variables and edge b/w nodes = relationship

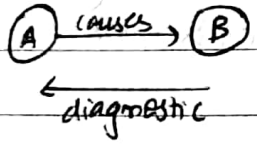
→ GOAL: extract a model which can be used to predict class label of an unseen object

Graphical Models are statistical models based on Bayesian learning



⇒ A causes B eg: smoking causes lung cancer

using this, we can find P(A|B). i.e. we are trying to diagnose



~~USE OF GM FOR ML~~ MOTIVATION FOR GM

1) Rel^n b/w features in training data are the quantities of interest for prediction

For Bayes classifier, you need joint prob. distribution to derive class conditional probabilities

For d-dimensional boolean data, we require $O(2^d)$ instances to estimate joint prob. distribution for each class. We simplify by assuming that attribute values are conditionally independent given class c_j i.e. $P(x = x_1, x_2, \ldots, x_d \mid c_j) = \prod_{i=1}^{d} P(x_i \mid c_j)$

HYPOTHESIS FOR NAIVE BAYES CLASSIFIER

$$h_{NB} = \text{argmax}_j \; P(c_j) \prod_m P(x_m \mid c_j)$$

Date :

Q- C = PlayTennis (Yes | No) |C| = 2

|Outlook| = 3
|Temp| = 3
|Humidity| = 2
|wind| = 2

] 3×3×2×2 combinatⁿs are possible = 36 combinations

If same 36 combo are there for Yes & for No then there is no distinction

$P(C|X) = [\pi P(X|C)] \cdot P(C)$

we need 8 prob. values given X

$P(Yes) = \frac{9}{14}$ $P(No) = \frac{5}{14}$

OUTLOOK : P(sunny | Yes) = 2/9
P(sunny | No) = 3/5
P(overcast | Yes) = 4/9
P(overcast | No) = 0
P(Rain | yes) = 3/9
P(Rain | No) = 2/5

Exact running time is H/W dependent

WIND : P(weak | yes) = 6/9
P(weak | No) = 2/5
P(strong | Yes) = 3/9
P(strong | No) = 3/5

TEMPERATURE : P(Hot | Yes) = 2/9
P(Hot | No) = 2/5
P(Mild | Yes) = 4/9
P(Mild | No) = 2/5
P(cool | Yes) = 3/9
P(cool | No) = 1/5

All these are calculated to create the model

we maintain (2 × (3+3+2+2)) = 20 counters for the model
(2 → Yes/No; 3+3+2+2 → attribute values)

HUMIDITY : P(High | yes) = 3/9
P(High | No) = 4/5
P(Normal | Yes) = 6/9
P(Normal | No) = 1/5

O(mw)
Time complexity ≈ O(m)
space complexity = O(C × (A1 + A2 + ... Ad))
(C → class value; A1 → no. of A1 values)

Date :

Predict the label for <sunny, ~~High~~ Hot, High, weak>

$$P(ND) \cdot \Pi\left(P(X_i \mid ND)\right)$$

$$= \frac{5}{14} \times \frac{3}{5} \times \frac{2}{5} \times \frac{4}{5} \times \frac{2}{5} = \frac{24}{35 \times 25} = \frac{24}{875} = 0.0274$$

$$P(Yes) \cdot \Pi\left(P(X_i \mid Yes)\right)$$

$$= \frac{9}{14} \times \frac{2}{9} \times \frac{2}{9} \times \frac{3}{9} \times \frac{6}{9} = \frac{4}{63 \times 9} = \frac{4}{567} = 0.00705$$

Date :

Not

Predict the label for <sunny, ~~High~~, High, weak>

$P(ND) \cdot \Pi (P(X | ND))$

$= \frac{5}{14} \times \frac{3}{5} \times \frac{2}{5} \times \frac{4}{5} \times \frac{2}{5} = \frac{24}{35 \times 25} = \frac{24}{875} = 0.0274$

$P(Yes) \cdot \Pi (P(X | Yes))$

$= \frac{9}{14} \times \frac{2}{9} \times \frac{2}{9} \times \frac{3}{9} \times \frac{6}{9} = \frac{4}{63 \times 9} = \frac{4}{567} = 0.00705$

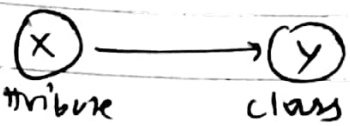
LAB WORK : Take a labelled dataset from UCI repository & find 10-fold cross validat^n accuracy.

- Naive Bayes is linear in time, popular because despite strong assumption, it often works surprisingly well.] FEATURE OF NAIVE BAYES

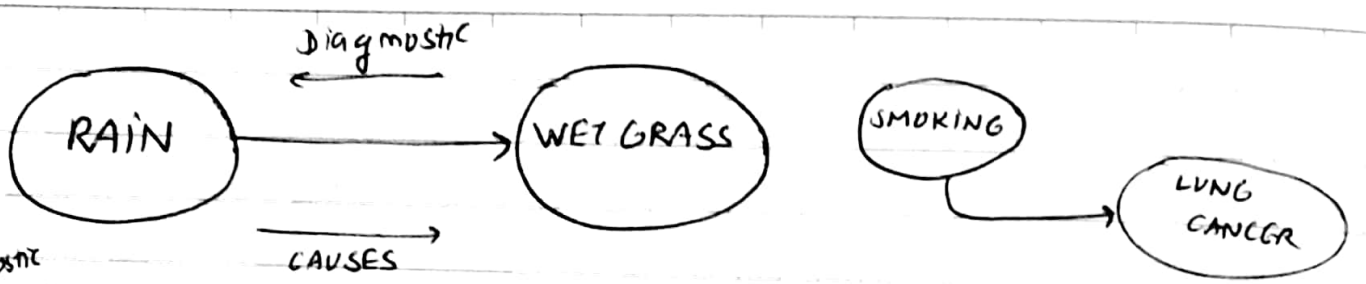
- Bayesian belief N/W provide an intermediate approach which allows strong conditional independence assumptions (based on domain knowledge) that apply to subsets of variables.

$x_1, \ldots x_6$ have $^6C_2$ relations
The structure is represented as a directed acyclic graph (DAG)

Parameters are conditional prob. in arcs



Date :



Diagnostic inference

$P(R) = 0.4$, $P(W|R) = 0.9$, $P(W|\sim R) = 0.2$

↳ Knowing that grass is wet, what's the prob. that rain is cause

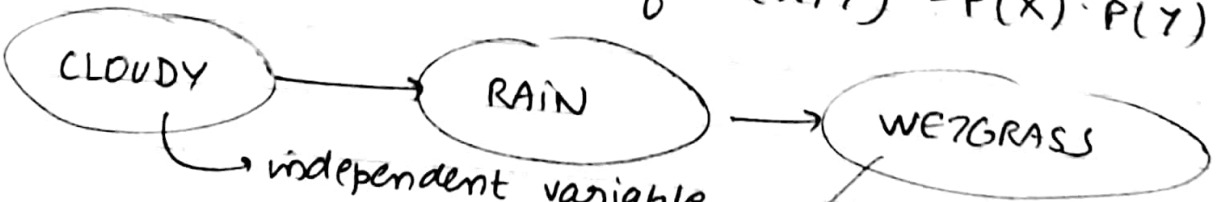
$$P(R|W) = \frac{P(W|R) \cdot P(R)}{P(W)} = \frac{P(W|R) \cdot P(R)}{P(W|R) \cdot P(R) + P(W|\sim R) \cdot P(\sim R)}$$

$$= \frac{0.9 \times 0.4}{0.9 \times 0.4 + 0.2 \times 0.6} = 0.75$$

If X is independent & $X \to Y$, then we must have $P(X)$

CONDITIONAL INDEPENDENCE

X & Y are independent if $P(X, Y) = P(X) \cdot P(Y)$



↳ independent variable

$P(C)$

$P(R|C)$
$P(R|\sim C)$

$P(W|R)$
$P(W|\sim R)$

for finding P(W) it doesn't matter if C occurs or not

$$P(W) = P(W|R) \cdot P(R) + P(W|\sim R) \cdot P(\sim R)$$

Date :

$$P(XYZ) = P(X) \cdot P(Y|X) \cdot P(Z|XY)$$

$$= \frac{P(X) \cdot P(YX) \cdot P(ZY)}{P(X) \quad P(XY)}$$

$$= \frac{P(XAY) \cdot P(YAZ)}{P(Y)}$$

$$P(Z|X,Y) = P(X,Y,Z) \,/\, P(X,Y) \qquad P(Y|X)$$

$$= P(X) \cdot P(Y|X) \cdot P(Z|Y) \,/\, P(X) \cdot P(XY)$$

$$= P(Z|Y)$$

[Event Y subsumes X]

[Event Y blocks event X & Z]

$P(Y)$, $P(Z)$ can be computed using Thm of total probability

Causal inference $P(Z|X)$

Diagnostic " $P(X|Z)$

people.du.ac.in/~vbhatnagar

↗ Causal inference

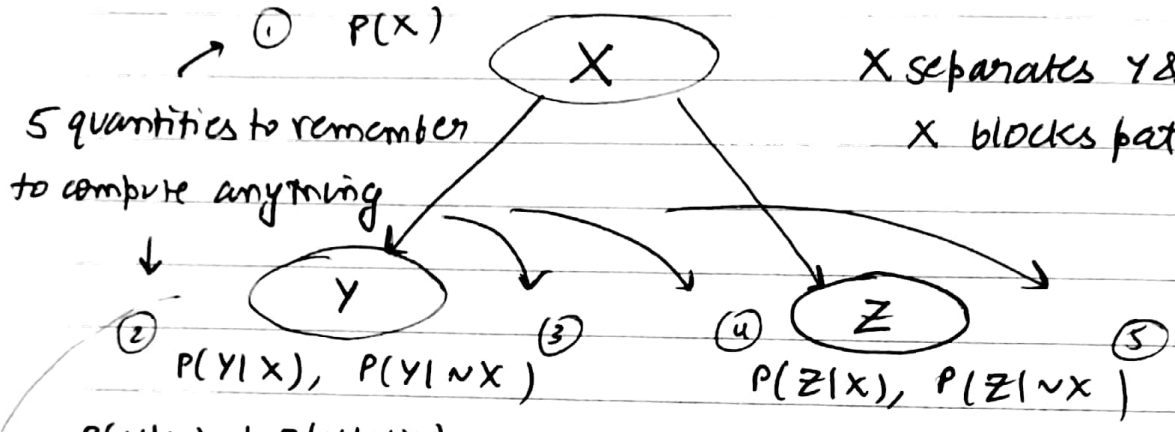
$$P(W|C) = P(W|R) \cdot P(R|C) + P(W|\sim R) \cdot P(\sim R|C)$$

$$P(C|W) = \frac{P(C|W) \cdot P(W)}{P(W)} \quad \frac{P(W|C) \cdot P(C)}{P(W)}$$

↳ Diagnostic inference

Date :

If a single variable influences multiple variables, i.e. for eg: X influences Y & Z



X separates Y & Z or X blocks path b/w X & Z

$P(Y|X) + P(Y|\sim X) \neq 1$ always because Y might also depend on some other factors. same as Y for Z.
↳ apart from X

- $P(X,Y,Z) = P(X) \cdot P(Y|X) \cdot P(Z|X)$ = joint probability of such a case

To check independent rel^n b/w Y & Z given X

$$P(YZ|X) = \frac{P(XYZ)}{P(X)} = \frac{P(X) \cdot P(Y|X) \cdot P(Z|X)}{P(X)} = P(Y|X) \cdot P(Z|X)$$

$P(Y) = P(Y|X) + P(Y|\sim X)$ · $P(Y|X) \cdot P(X) + P(Y|\sim X) \cdot P(\sim X)$

$P(Z) = P(Z|X) \cdot P(X) + P(Z|\sim X) \cdot P(\sim X)$

How to compute P(Y|Z) given we don't know they are indepdent

We know P(X|Y) & P(X|Z) are diagnostic

$$P(Y|Z) = \frac{P(Z|Y) \cdot P(Y)}{P(Z)}$$

Date :

we need this causal for calculating diagnostic

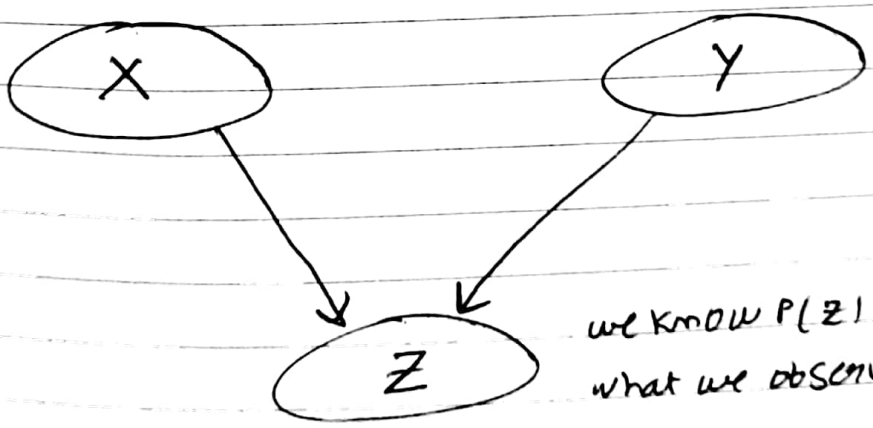
$$P(X|Y) = \frac{P(Y|X) \cdot P(X)}{P(Y)}$$

~~$P(Y|Z) = \frac{P(YZ)}{P(Z)} = \frac{P(Y) \cdot P(Z)}{P(Z)}$~~

$$P(Y|Z) = \frac{P(YZ)}{P(Z)} = \frac{P(Y) \cdot P(Z)}{P(Z)} = P(Y)$$

we don't use $P(X)$ as we are assuming X is not known

$$= P(Y|X) \cdot P(X|Z) + P(Y|\sim X) \cdot P(\sim X|Z)$$

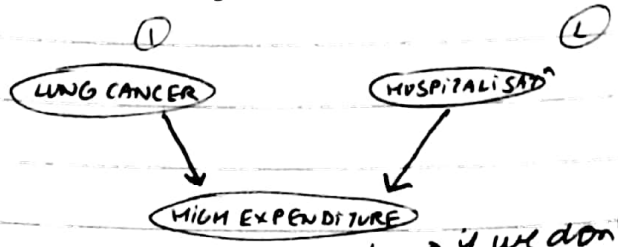


we know $P(Z|XY)$ ∴ this is what we observe

- $P(X,Y,Z) = P(X) \cdot P(Y) \cdot P(Z|X,Y)$

we can also obtain $P(Z|\sim XY)$, $P(Z|X\sim Y)$, $P(Z|\sim X\sim Y)$ apart from $P(Z|XY)$ from the provided data

eg.



→ if we don't observe this (high exp) then ① & ② are independent, else they are dependent

Date :

$P(W|R,S) + P(W|R,\sim S) + P(W|\sim R,S) + P(W|\sim R,\sim S)$

can be > 1

SPRINKLER → WET GRASS ← RAIN

P(W|S) = causal event

P(S|W) = diagnostic

Given $P(W|R,S) = 0.95$

$P(W|R\sim S) = 0.9$

$P(W|\sim RS) = 0.9$

$P(W|\sim R\sim S) = 0.1$

$P(S) = 0.2, \quad P(R) = 0.4$

$P(W) = P(W|RS) \cdot P(R) \cdot P(S) +$

using total prob thm.

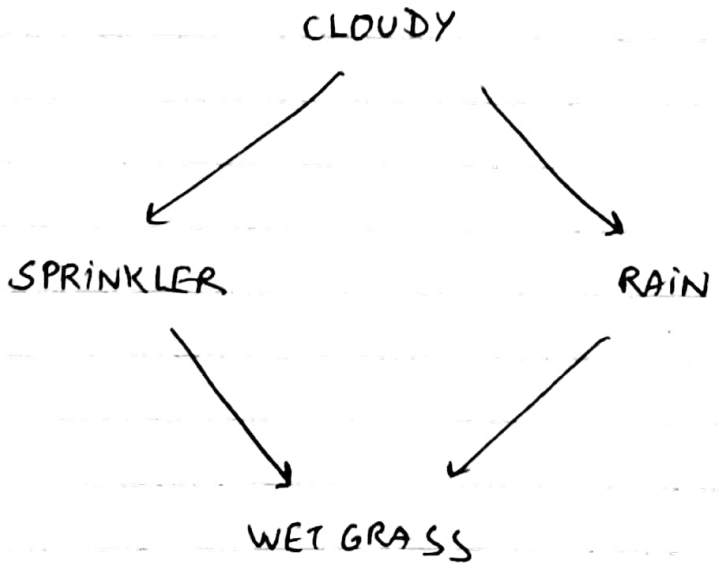
$P(W|(\sim RS)) \cdot P(\sim R) \cdot P(\sim S)$

$\cdot P(\sim R, \sim S)$

$P(W|S) = P(W|RS) \cdot P(R|S) + P(W|\sim RS) \cdot P(\sim R|S)$

$= P(W|RS) \cdot P(R) + P(W|\sim RS) \cdot P(\sim R)$

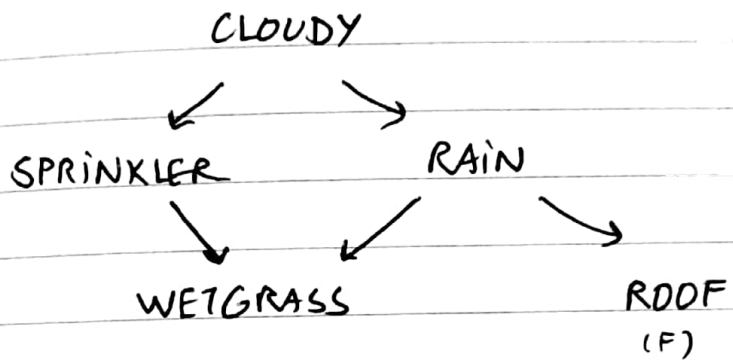
Knowing it has rained decreases the prob. that sprinkler is on



Date :

EXPLOITING LOCAL STRUCTURE

How to extend Bayesian N/W to a newly added variable



$$P(CSRWF) = P(C) \cdot P(S|C) \cdot P(R|C) \cdot P(W|SR) \cdot P(F|R)$$

$$= \prod_{i=1}^{d} P(X_i \mid \text{parents}(X_i))$$

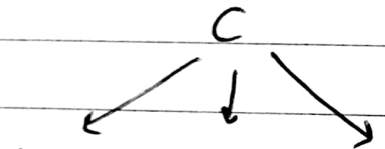
RAIN → WET

Since R is ind. ∴ we know P(R) & also P(W|R)
In bayes rule, we choose C & then calc. Prob.
In class., we calc. prob of a class given data
Bayes' rule inverts the arc

We have class conditional prob. from the data

Date :

NAÏVE BAYES CLASSIFIER



Given C, x_j are independent

$$P(x|c) = P(x_1|c) \cdot P(x_2|c) \cdot P(x_3|c)$$

Date : 23/1/2020

DECISION TREES (CH-9)

→ works on Divide & conquer

- It is a model induced from Training set
- can be used for classificat^n & regression
- prob. is involved in a very limited way.

CASE 1

CLASSIFICAT^N TREES

→ 2 attributes + there would be 1 class value ∴ =3 columns in training set

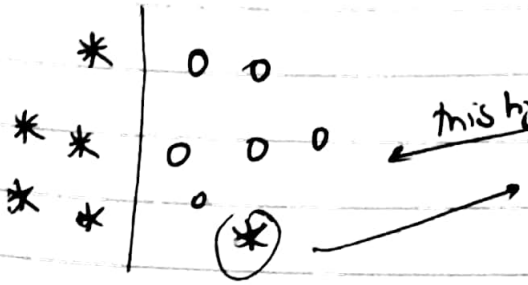
DISCRIMINANT FUNCTION : decision boundary b/w 2 classes divides data space into regions

Each region contains data points of a particular class.

Can we say Discrim. funct^n is classifier ?

↳ YES, we can. → choose o if $x_1 > V_{ii}$] HYPOTHESIS
(*) 1 otherwise

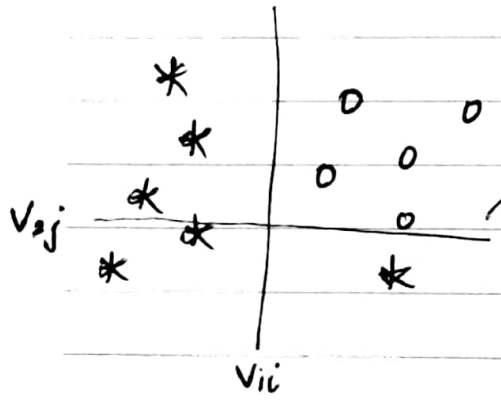
In reality, we don't get such perfect decision boundaries



this hypothesis is not consistent for this dataset
∴ we must revise our hypothesis

CASE 2

Date :

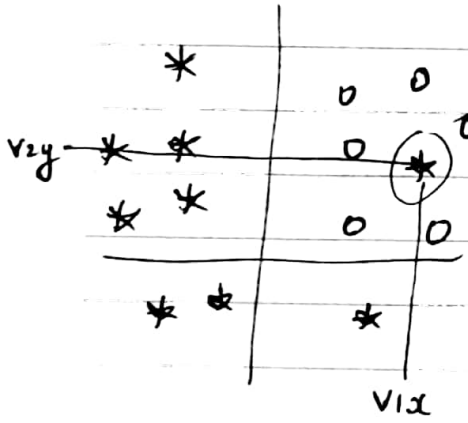


revised decision boundary

choose o if ($x_1 > V_{1i}$ & $x_2 > V_{2j}$)
1 otherwise
(*)

CASE 3

now this hypothesis is not consistent with this training set



how to develop our model in this case

choose o if ($x_1 > V_{1i}$ & $x_2 > V_{2j}$) & ($x_1 \neq V_{1x}$ & $x_2 \neq V_{2y}$)
1 otherwise
(*)

this hypothesis is particular for this data point, ∴ we tend to overfit our model over the provided data. We are making our hypothesis complex in order to be consistent

→ region s.t no 'o' pt lies in it (or on its boundary)

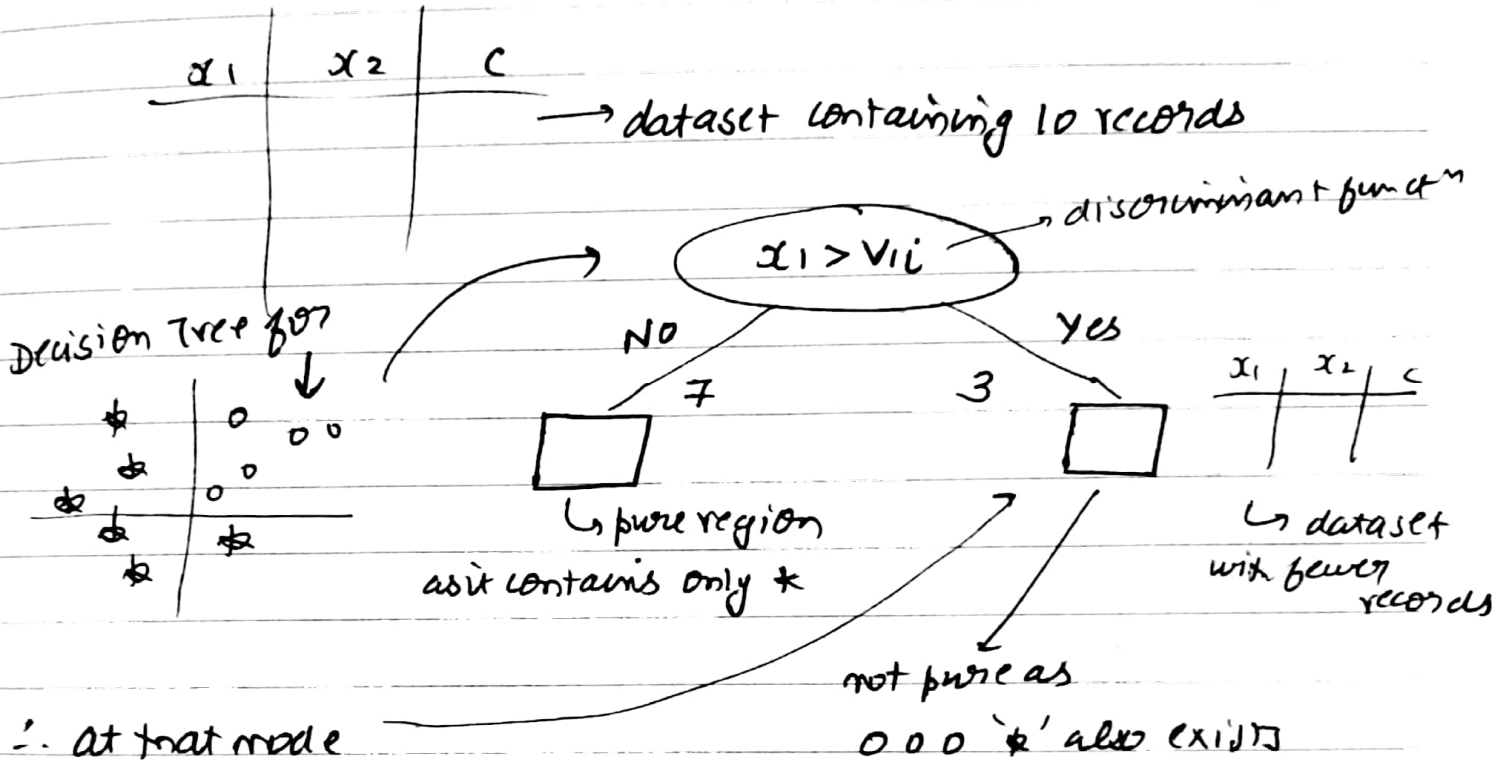
a better hypothesis would be to specify a non-overlapping region around (V_{1x}, V_{2y}) instead of simply specifying coordinates

Decision Tree provides a recursive method to divide the region until each pt is in a class.

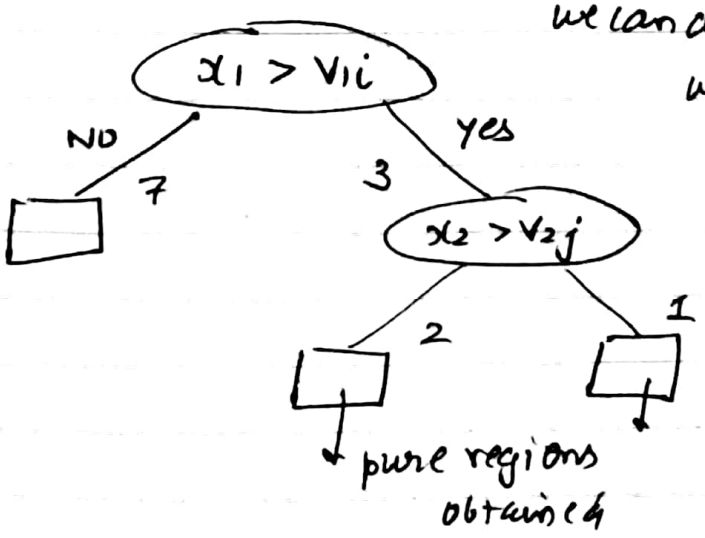
PURE REGION : all data pts of 1 class in a region → aim of our hypothesis

Date :

In a tree: root & internal nodes denoted by ⬭
leaf nodes denoted by ▭



we can do so recursively until we get pure partitions



Date :

Internal Decision Nodes : univariate → using single attr. x_i
multivariate → " all attr., x

Leaves : classificat^n → class labels or proportions
regression → Numeric ; " average or local fit

FORM OF DECISION NODES

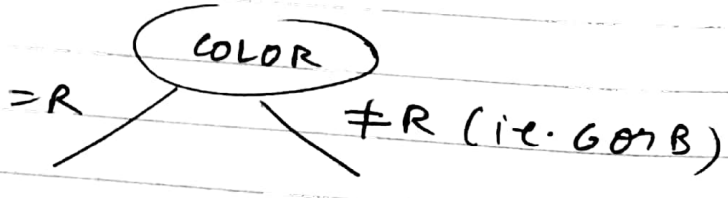
A (attribute)	O (operation)	V (value)
x_1	>	v_{1i}
x_2	>	v_{2j}

for numeric values ($>, <, \geq, \leq$)
→ also binary split

Discrete
↓
categorical
↙ ↘
Ordinal : { } Rank, { Good, Better, Best }
Nominal : { } City, { } Color

Decision on discrete values
→ n-way or binary split
($=, \neq$) usually used

OR



learning in decision tree is greedy ; find the best split recursively till some terminating criteria is met

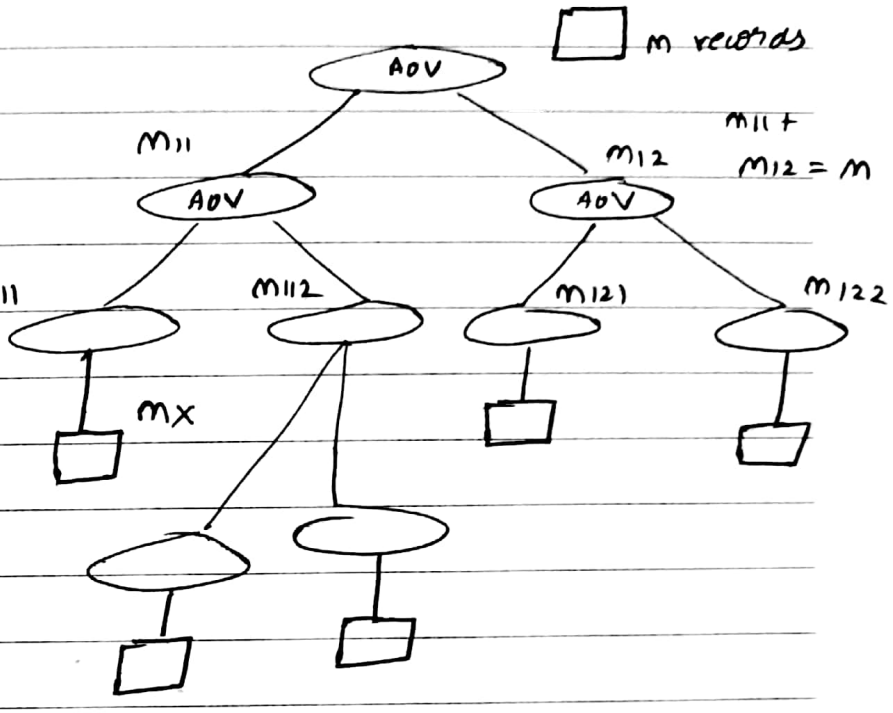
Decision Tree algo is a recursive partitioning algo.
Doesn't usually give accurate results but is popular ∵ it is transparent & we can explain our decision at each step

Date :

CLASSIFICAT^N TREES (ID3, CART, C4.5)

popular recursive partitioning algos

we define a region is pure using probability.



$= m_{11}$

$m_{111} + m_{112} + m_{121} + m_{122} = m_{12}$

$= m$

To check purity of m_x

$$P_{m_x}^{i} = \frac{\#\text{instances of } C_i}{|m_x|}$$

total no of records at m_x

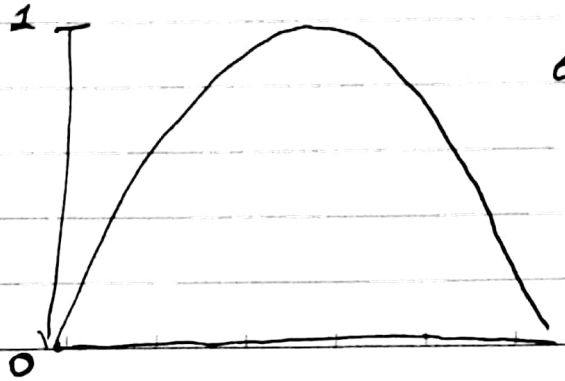
Node m_x is pure if $P_{m_x}^{i} = 0$ or 1

$|m_x| = N_{m_x}^{i} = \#$ instances of class i at node m

$\hat{}$ → denotes estimate

$$\hat{P}(C_i \mid x, m_x) = P_{m_x}^{i} = \frac{N_{m_x}^{i}}{N} \quad (N \rightarrow |m_x|)$$

ENTROPY : A more rigorous way to check purity



entropy is a measure of disorder in a system

entropy $\propto$ disorder

Date :

$$I_m = (-)\sum_{i=1}^{K} p_m^i \log_2 p_m^i$$

↳ information in bits

↳ $\because p_m^i$ is in $[0,1]$ $\therefore \log_2 p_m^i < 0$ $\therefore$ we have -ve sign outside

entropy measures the ~~ex~~ information in the data

If there are K classes each with K records, then entropy is max & prob. $= \frac{1}{K}$ for each class, $0 \log_2 0 = 0$

Entropy is min i.e. $= 0$ if ~~on~~ there is only 1 class

BEST SPLIT : If node m is pure, generate a leaf & stop, otherwise split & continue recursively

we should be able to predict with least questions possible

pick attribute with highest discriminating power

BSC SUBJECT | FLOOR OF CLASSES | 12TH MARKS | CLASS (CS/STATS/MATHS/OR)

↳ best att. in this case as each have classes on diff. floors

↳ OR students may have B.Sc CS, stats, maths students

↳ how to decide this? → using measures of goodness of split

Decision Tree is a non-parametric, hierarchical model

Importance of discriminant function decreases down the tree / power

∴ Root node has the discriminant funct" of highest power, importance

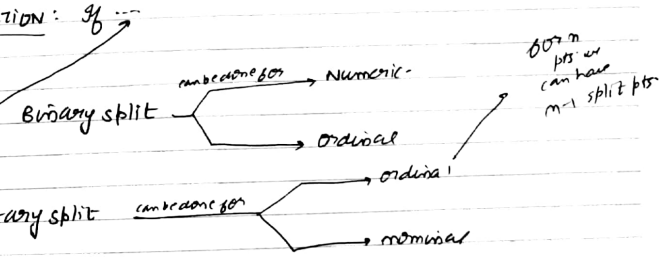
Date :

Each leaf defines a localised region in d-dimensional data space with high density of data pts of a class.

Decision Trees are one-way structures

$$\text{Entropy} \propto \frac{1}{\text{Purity}} \propto \text{Impurity}$$

INDUCTION : If ...



How to do split for nominal?

↳ by grouping

we find all possible subsets to get discriminant function

eg : weight is a continuous attribute, ∴ we can have ∞ split pts even for 100 records. we select best hypothesis (split pts) from hypothesis space

$$I_m' = -\sum_{j=1}^{n} \frac{N_{mj}}{N_m} \sum_{i=1}^{K} p_{mj}^i \log_2 p_{mj}^i$$

↳ no. of class labels

Date :

CHAPTER - 19

DESIGN & ANALYSIS OF MACHINE LEARNING EXPTS.

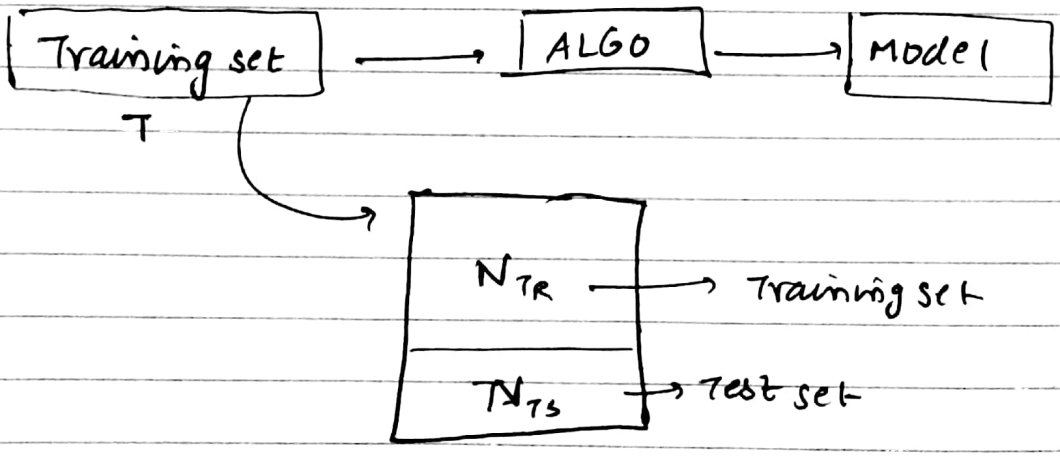
Each decision tree is a hypothesis
How to decide which DT will perform best when deployed?
How to quantify / assess quality of prediction?

We can create diff. DT each with its own impurity threshold
we choose the one with min. error / best accuracy.

DTi with threshold impurity θ_i →	θ_1	θ_2	θ_3
	hypothesis 1	hyp. 2	hypothesis 3

Training error, Test error

Model is a hypothesis
Classificat^n model constructs a hypothesis



$$|T| = N_{TR} + N_{TS}$$

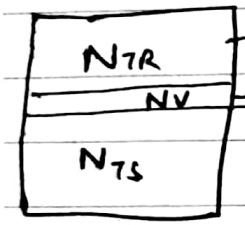
Date :

N_{TR} → Model → Training error

N_{TS} → Model → Test error

Training error < Test error ∴ our model is based on training error it might be that N_{TS} contains well behaved records which might degrade performance of our model when deployed in field.

To overcome this, we split our data into N_{TR} & N_{TS} multiple times randomly, so that test set can't be predicted



→ bigger than other two sets in size.

→ validation set (to tune parameters of classifiers)

DRAWBACK : less no of instances for training

MOTIVATION → Model selection v/s Model estimation

↳ Assessing quality of classifier / model / algo

↳ No free lunch thm (formalizes the statement that there is no universally best ML Algo)

OR

On an average, all algos perform equally badly

One algo may work accurately for one data but may perform worse for other data (even from same domain)

No algo works equaly well on all datasets

→ Diviaing dataset into N_{TR}, N_V, N_{TS} reduces size of training data & time to build the model

Algo preference on → Time complexity

→ Misclassificat^n error

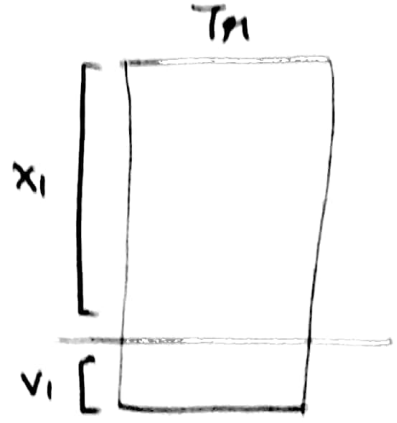
EVALUATING → K-cross Validation
→ Holdout
→ Bootstrap

RESAMPLING & K FOLD CROSS VALIDATION

K partitions → (K-1) for training
→ K^{TH} for testing

iterations = K

this makes sure all distributions are represented equally in Tr & Ts set



eg : for 100 instances, if K = 10 → 10 folds
→ 90 records for Tr & 10 records for Ts

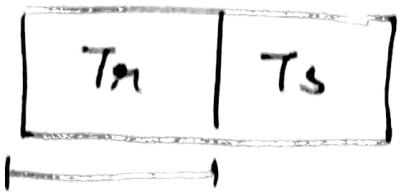
# of iterations = # of models = K

We take avg performance of classifiers over K iterations. Every instance is used (K-1) times for training & exactly once for testing

NOTATION: $\{X_i, V_i\}$: Tr & Ts Validation sets of i^{TH} fold

For smaller dataset, 10 fold CV is not option
→ do 5 times 2 fold CV

HOLDOUT



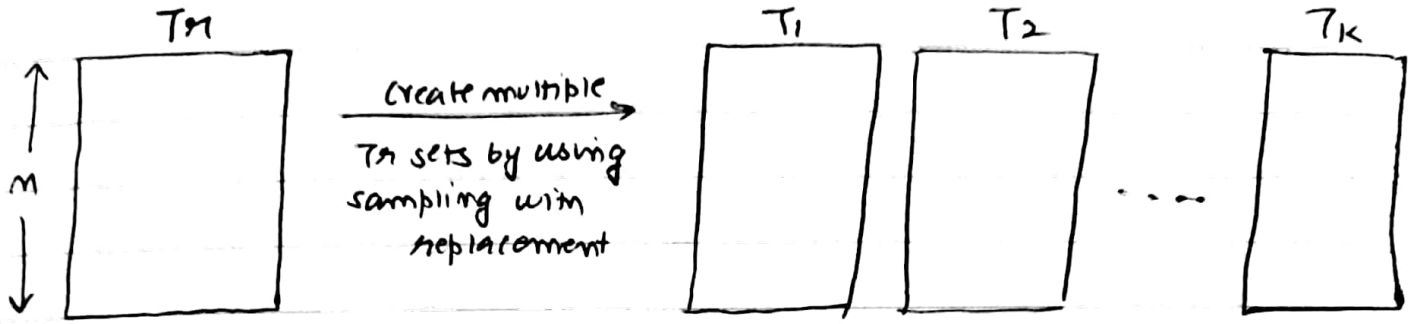
→ holding this throughout for training

BOOTSTRAPPING

- draws instances from data set with replacement

Date :

→ done for comparatively smaller datasets.



There will be only 36.8 % new instances

→ over multiple runs, all instances can be used in T_R as well as T_S

PERFORMANCE MEASURES Total = TP + FP + FN + TN

CONFUSION MATRIX

ACTUAL \ PREDICTED	1	0
1	TP	FN
0	FP	TN

$\frac{TP+TN}{Total}$ = Accuracy (ideally = 1 or 100 %), error rate = 1 − Accuracy

RECALL : Of all tve instances, how many can model predict as tve

$= \frac{TP}{TP + FN}$

$= \frac{TN}{TN+FP}$ (recall for −ve) = specificity

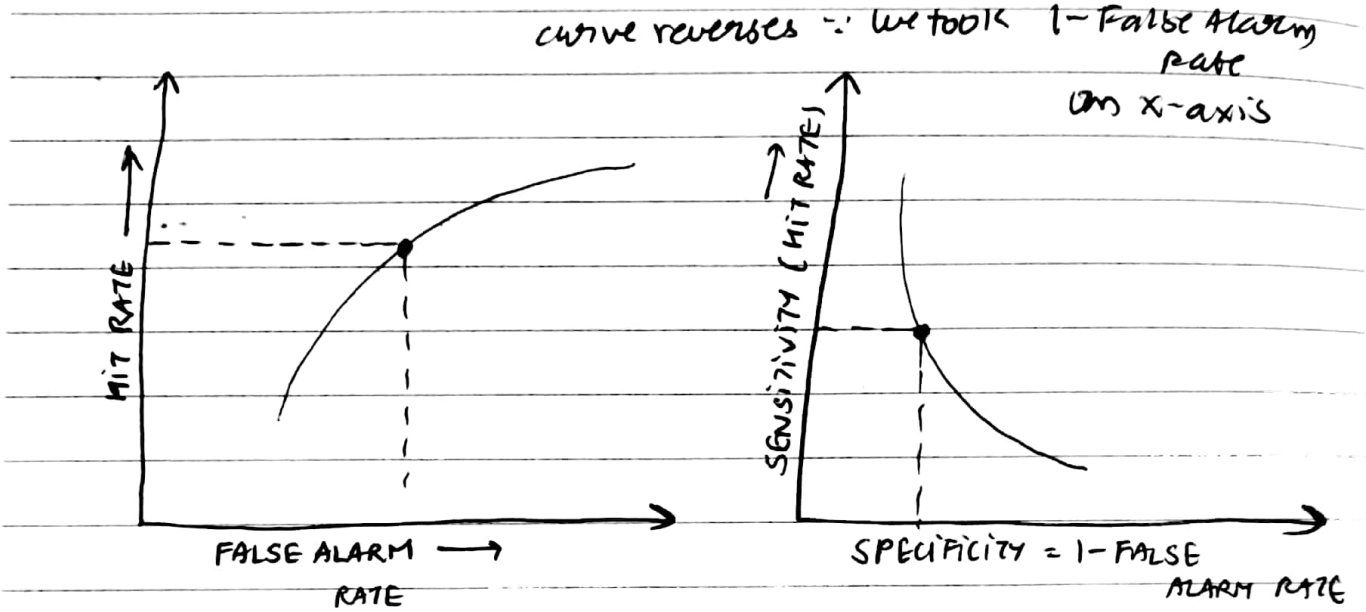
PRECISION : correct predictions of tve class out of total tve predictions

$= \frac{TP}{TP+FP}$

FALSE ALARM RATE : $\frac{FP}{FP+TN}$ (causes extra expenditure & expense)

Date :

ROC CURVE (Receiver Operating characteristics)



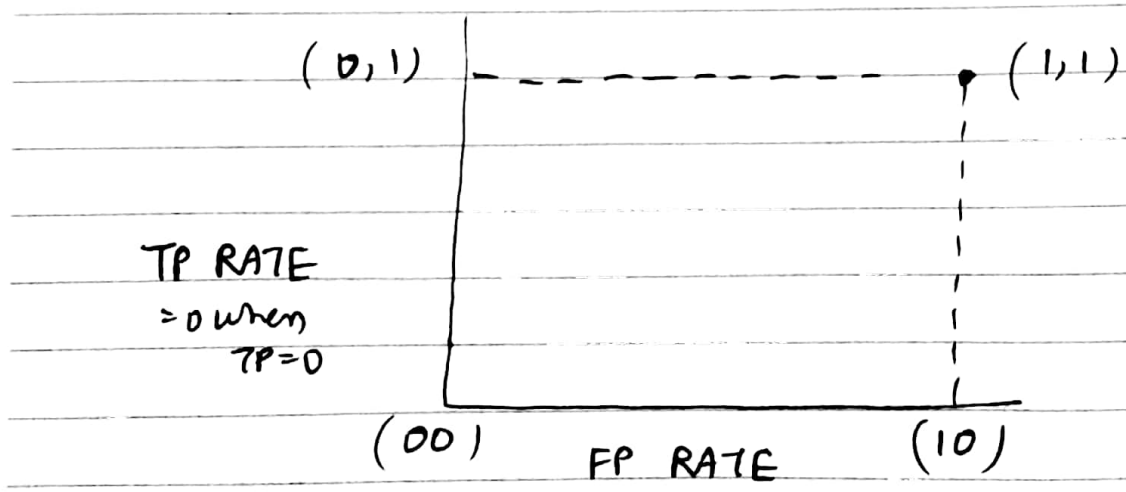
$$\text{Hit rate} = \frac{TP}{TP + FN} \qquad \text{False Alarm Rate} = \frac{FP}{FP + TN}$$

TP Rate = Recall of +ve class → sensitivity $= \frac{TP}{TP + FN}$

FP Rate = 1 − Recall of −ve class → specificity

$1 - \frac{TN}{FP + TN} = \frac{FP}{FP + TN}$

→ [0, 1]



When does (00) occur

∴ model always predicts −ve (either TN or FN)

Date :

(1D) : FP RATE = 1 $\Rightarrow \frac{FP}{FP+TN}$, TN = 0

TP RATE = 0 $\Rightarrow \frac{TP}{TP+FN}$, TP = 0

$\therefore$ no correct predictions

(1,1) : TP RATE = 1 $\Rightarrow$ FN = 0 & TP all tve sets are predicted tve

FP RATE = 1 $\Rightarrow$ TN = 0 $\therefore$ all are predicted tve

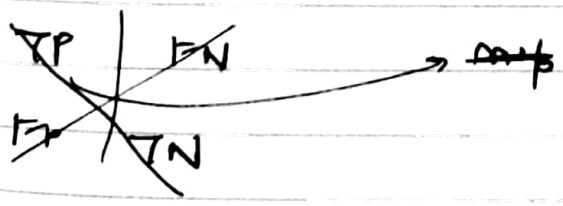
(0,1) : TP RATE = 1, FP RATE = 0] Ideal classifier

TP RATE = 1 $\Rightarrow$ FN = 0

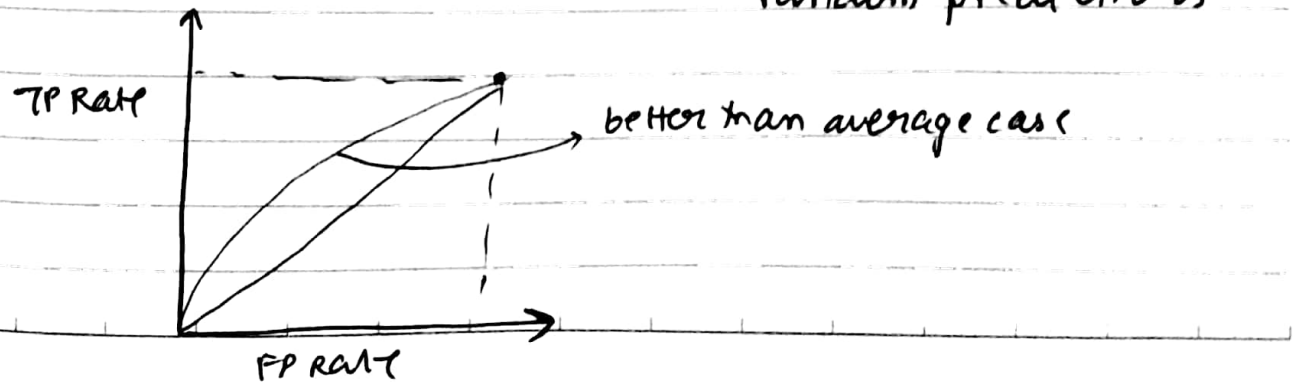
FP RATE = 0 $\Rightarrow$ FP = 0] Ideal case

If classifier is predicted tve class with 'p' prob. what is TPR?] *

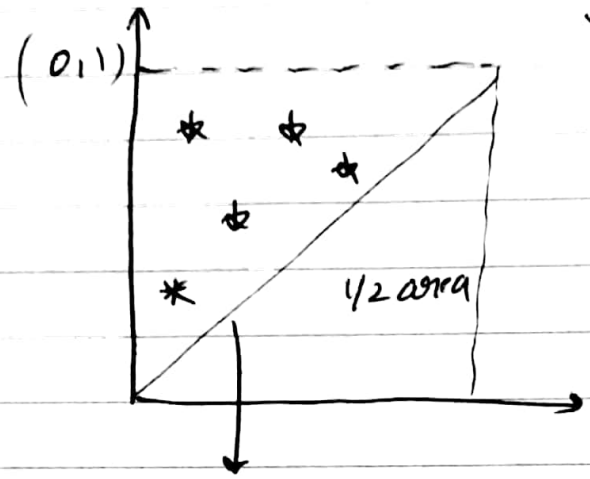
Total instances = $m_+ + m_- = m$



Roc curve for TP rate = FP rate when classifier is making random predictions



Date :

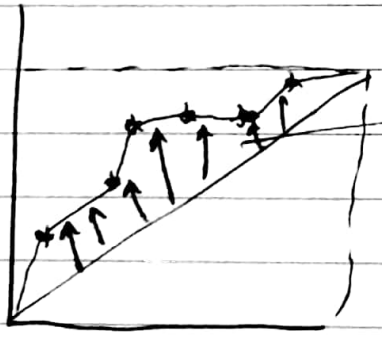


'*' closest to (0,1) (ideal case) must be chosen as the most appropriate classifier.

diagonal is the classifier that predicts with fixed probability (average performance case)

AUC : Area under curve

The curve which gives most ~~area under~~ AUC is best classifier (done during model selection)



Area above diag. = X

Total area = $\frac{1}{2}$ Total Area + X

MLE : Max likelihood Estimat^n (CH-4)

↓

Regression (CH-4)

↓

Reg. Trees

Date :

PARAMETRIC ESTIMATION

MAXIMUM LIKELIHOOD ESTIMATION (CH-4)

↳ estimating parameters of some density function

Given $n = 7$ records following Normal Distribution

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

75	
75	
85	11
92	235
83	92
62	63
77	62
	77
	549
	7
	78

$N_1(55, 5)$ $N_2(60, 7)$ $N_3(75, 10)$ $N_4(82, 15)$ ✓

higher variance explains deviation

values are closer around 82 as around 75 intuitively

Parametric Estimation is applied when we know the distribution

$X = \{ x^t \}_{t=1}^{N}$ → independently identically distributed variables

: training set with N instances

x is a random variable

$P(X, \theta)$: prob. density function

$$l(\theta | X) = p(x^1 | \theta) \cdot p(x^2 | \theta) \cdots p(x^m | \theta)$$

(x^2 → 2nd training element; x^m → m^{TH} training element)

likelihood of θ parameter

$$= \prod_{t=1}^{m} p(x^t | \theta) \quad — (1)$$

goal : identify θ which maximises (1)

Date :

$$\log\left(l(\theta|X)\right) = \log\left(\prod_{t=1}^{n} p(x^t|\theta)\right)$$

log likelihood (L)

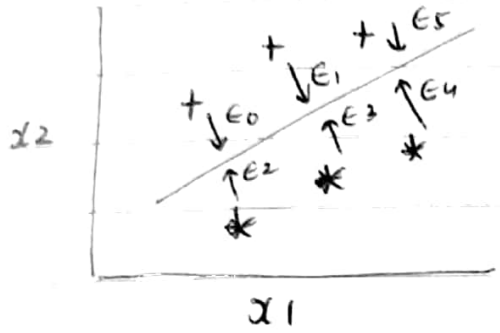
$$= \sum_{t=1}^{n} \log p(x^t|\theta)$$

$$\log\left(\frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}\right)$$

2 terms obtained where value is affected by μ & σ

~~constant~~ ∴ ~~ignored~~

REGRESSION



$$\sum \epsilon_i = 0 \quad \text{——— (1)}$$

$$r^t = f(x^t) + \epsilon \quad \text{——— (2)} \qquad \epsilon \sim N(0, \sigma^2)$$

using (1), $\mu = 0$, $\sigma = \frac{1}{2}\sum \epsilon^2 - \bar{\epsilon}^2 = 0$ ∴ σ^2 is some non zero value

Let $g(\cdot)$ denote the approximate function

Assume we know the functional form of $g(\cdot)$, but still the parameters are to be estimated.

↳ like we know the form of funct^n is normal distribution but we still need to know μ & σ^2 parameters

$g(x|\theta)$ approximates $f(x^t)$

↳ we don't know the parameters θ which we need to know from x

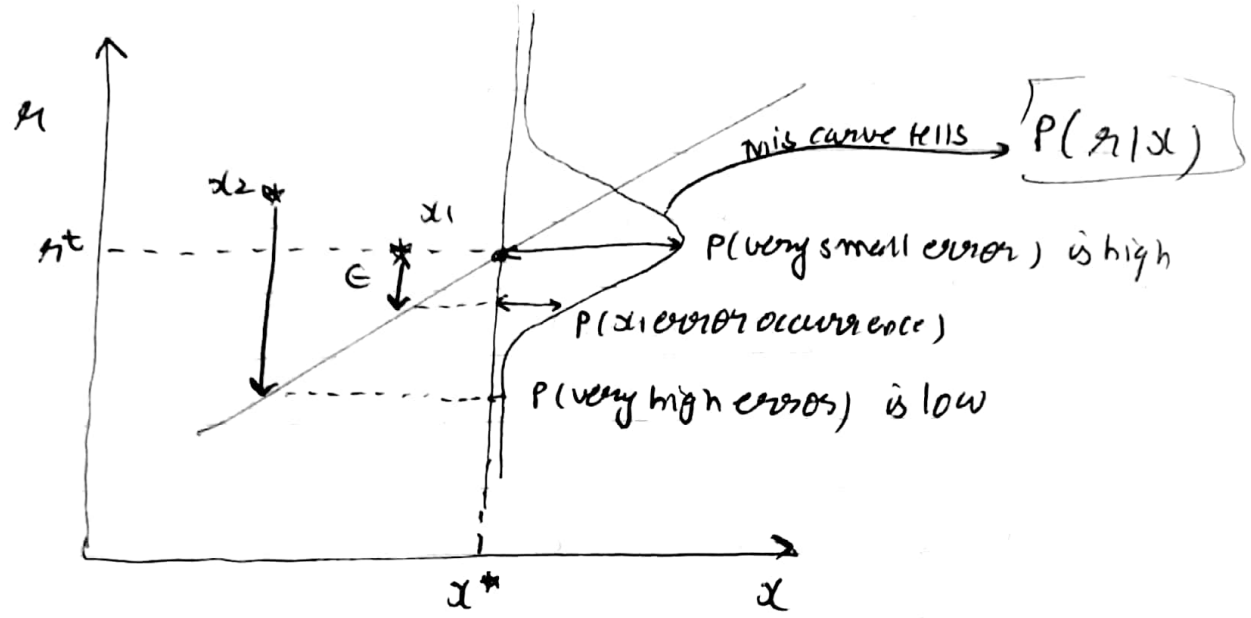
we are assuming $g(\cdot)$ mimics $f(\cdot)$

∴ in (2) $\hat{r}^t = g(x|\theta) + \epsilon$

In case of a straight line : $\hat{r}^t = w_0 + w_1 x^t$, $\theta = (w_0, w_1)$

for any new x^*, we substitute it in ↗ to get r^t, where (x^*, r^t) is a pt on line

Date :



$P(r|x)$ follows $N(g(x|\theta), \sigma^2)$

↳ we need to find this using max. value estimation

$$L(\theta|X) = \log \left[\prod_{t=1}^{N} \underbrace{P(x^t, r^t)}_{\text{same as } P(r^t|x^t) \cdot P(x^t)} \right]$$

$$= \sum_{t=1}^{N} \left[\log\left(P(r^t|x^t)\right) + \log\left(P(x^t)\right) \right]$$

$$= \sum_{t=1}^{N} \log\left(\underbrace{P(r^t|x^t)}_{I}\right) + \sum_{t=1}^{N} \log\left(\underbrace{P(x^t)}_{II}\right)$$

since x^t is independent variable $\therefore x^t$ is independent of θ

$\therefore$ II is omitted

$$L(\theta|X) = \sum_{t=1}^{N} \log\left(P(r|x^t)\right)$$

$$= \sum_{t} \log\left[\frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\left(\frac{r - g(x|\theta)}{\sigma}\right)^2} \right]$$

Date :

$$= \sum_t \left[\log \frac{1}{\sqrt{2\pi}\sigma} - \frac{1}{2\sigma^2} (r - g(x|\theta))^2 \right]$$

III IV

$$= \sum_t \log \frac{1}{\sqrt{2\pi}\sigma} - \frac{1}{2\sigma^2} \sum_t (r - g(x|\theta))^2$$

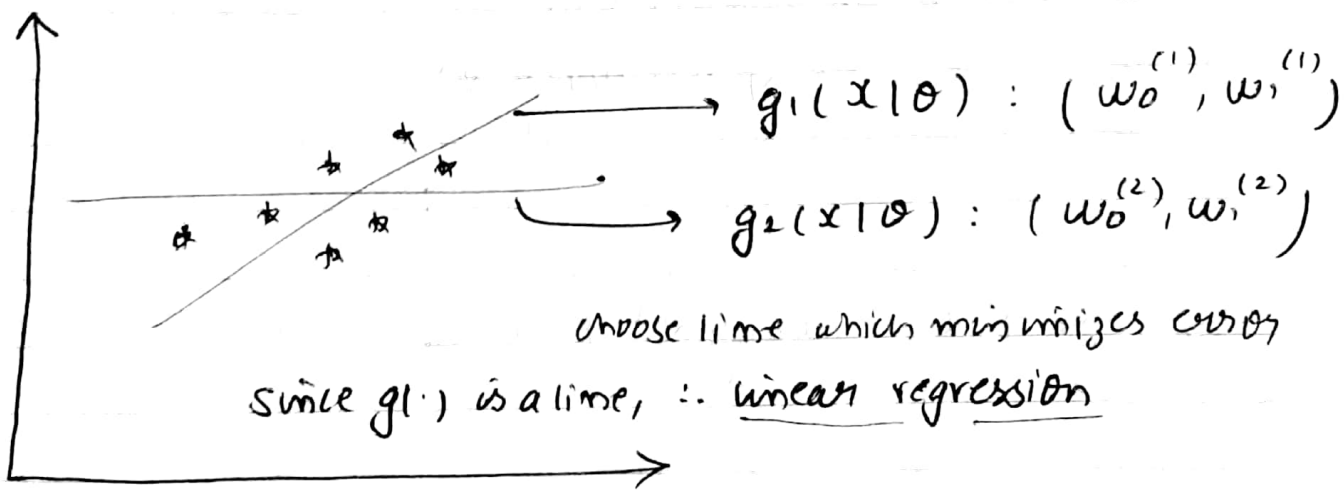
From the point of view of θ, σ is constant $\therefore$ III is neglected

$$\therefore L(\theta|X) = \underbrace{\frac{-1}{2\sigma^2}}_{\text{this is independent } \therefore \text{ neglecting}} \sum_t (r - g(x|\theta))^2$$

$$L(\theta|X) = \sum_t \overbrace{(r - g(x|\theta))^2}^{\text{squared error}}$$

we need to minimize this term

likelihood $\propto \frac{1}{\text{error}}$, we choose the line which minimizes error



choose line which minimizes error

since $g(\cdot)$ is a line, $\therefore$ linear regression

~~$\Sigma ab = \Sigma a \Sigma b$~~

Date :

In linear regression, $g(\cdot) = w_0 + w_1 x^t$

x	r
⋮	⋮

$$L(\theta|X) = \sum_t [r^t - g(x^t|\theta)]^2$$

$$= \sum_t (r^t - (w_0 + w_1 x^t))^2$$

We need to calculate $\frac{\partial L(\theta|X)}{\partial w_0}$, $\frac{\partial L(\theta|X)}{\partial w_1}$

$$\frac{\partial L(\theta|X)}{\partial w_0} \rightarrow \underline{2(r^t - (w_0 + w_1 x^t))(-1)} \quad \text{—①}$$

$$\frac{\partial L(\theta|X)}{\partial w_1} \rightarrow \underline{2(r^t - (w_0 + w_1 x^t))(-x^t)} \quad \text{—②}$$

Eq ① $= 0$

$$-2\left(\sum_t (r^t - (w_0 + w_1 x^t))\right) = 0$$

$$\Rightarrow \sum_t r^t = \sum_t w_0 + w_1 x^t$$

$$= N w_0 + w_1 \sum_t x^t \quad \text{—③}$$

Eq ② $= 0$

$$-2 x^t \left(\sum_t (r^t - (w_0 + w_1 x^t))\right) = 0$$

~~$\sum_t r^t x^t - w_0 \sum_t x^t + w_1 \sum_t (x^t)^2$~~

$$\sum_t r^t x^t = w_0 \sum_t x^t + w_1 \sum_t (x^t)^2 \quad \text{—④}$$

Date :

$$\underset{A}{\begin{bmatrix} N & \sum_t x^t \\ \sum_t x^t & \sum_t (x^t)^2 \end{bmatrix}} \underset{X}{\begin{bmatrix} w_0 \\ w_1 \end{bmatrix}} = \underset{B}{\begin{bmatrix} \sum_t r^t \\ \sum_t r^t x^t \end{bmatrix}}$$

we can calculate all these values from ..

x	r
:	:

we find X using $X = A^{-1}B$

FOR POLYNOMIAL REGRESSION

$$g(x|\theta) = w_0 + w_1 x^t + w_2 (x^t)^2 + \dots + w_k (x^t)^k$$

we need to estimate k+1 parameters

$$L(\theta|X) = \sum \left(r - (w_0 + w_1 x^t + \dots + w_k (x^t)^k) \right)^2$$

we do $\frac{\partial L(\theta|x)}{\partial w_0}, \frac{\partial L(\theta|x)}{\partial w_1}, \dots \frac{\partial (L(\theta|x))}{\partial (w_k)}$

$$\underset{A}{\begin{bmatrix} (k+1) \times (k+1) \\ \text{matrix} \end{bmatrix}} \underset{X}{\begin{bmatrix} x_0 \\ \vdots \\ x_k \end{bmatrix}_{(k+1)\times 1}} = \underset{B}{\begin{bmatrix} \sum r^t \\ \sum r^t x^t \\ \sum r^t (x^t)^2 \\ \vdots \end{bmatrix}_{(k+1)\times 1}}$$

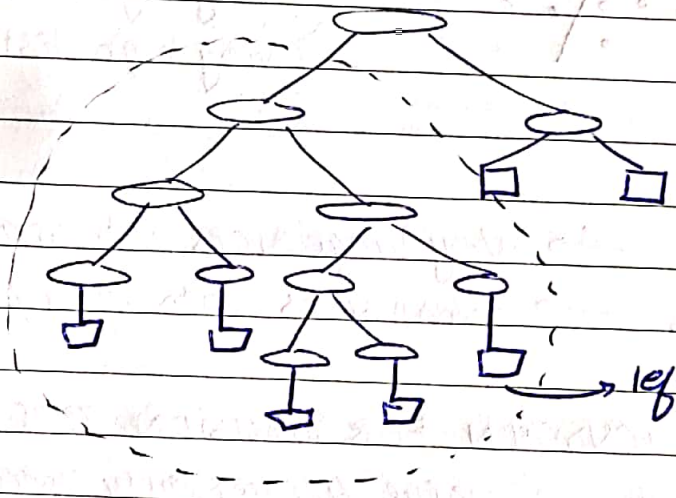
we can't find A^{-1} when $|A| = 0$ when

$$N \sum (x^t)^2 = \left(\sum x^t \right)^2$$

Date :

Regression trees are strictly binary in nature.
If threshold is small, the depth & 'bushyness' of tree ↑ses.

The structure of tree can be highly imbalanced which depends over the nature of the data. we have no control over it apart from calculating impurity measure for selecting the splitting attribute.



OVERFITTING v/s UNDERFITTING

$h_1(x)$ → $ax^2+bx+c>0$ (3 parameters) — we will choose $h_1(x)$ over $h_2(x)$ (eg: makes 12 errors) but still is easier to describe
HYPOTHESIS

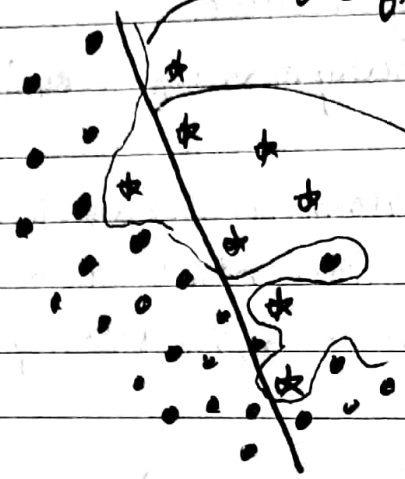
$h_2(x)$ → a piecewise curve (very difficult to describe) (numerous parameters)
OVERFITTED HYPOTHESIS
↳ complex of 2 reasons
— Too many ~~par~~ functions each with its parameters

simpler one is better hypothesis

$h_3(x)$ → $mx+c$ → simplest hypothesis but with way too many errors, so not always preferable to be selected. may underfit / too simple for data

Date :

very
(small training error, high test error)
→ overfitting hypothesis (memorizes the data)
hyperturned parameters

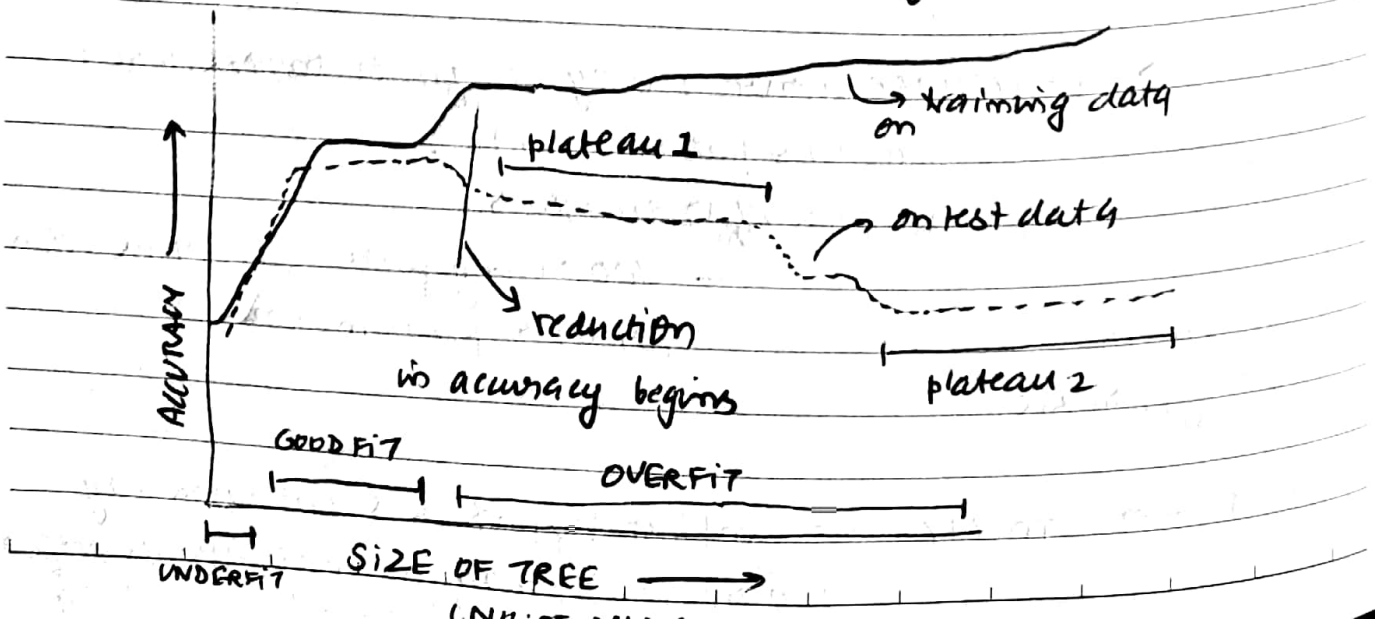


→ underfitting hypothesis
(very simple to describe but makes many mistakes)
(very high training error)
(very high test error)

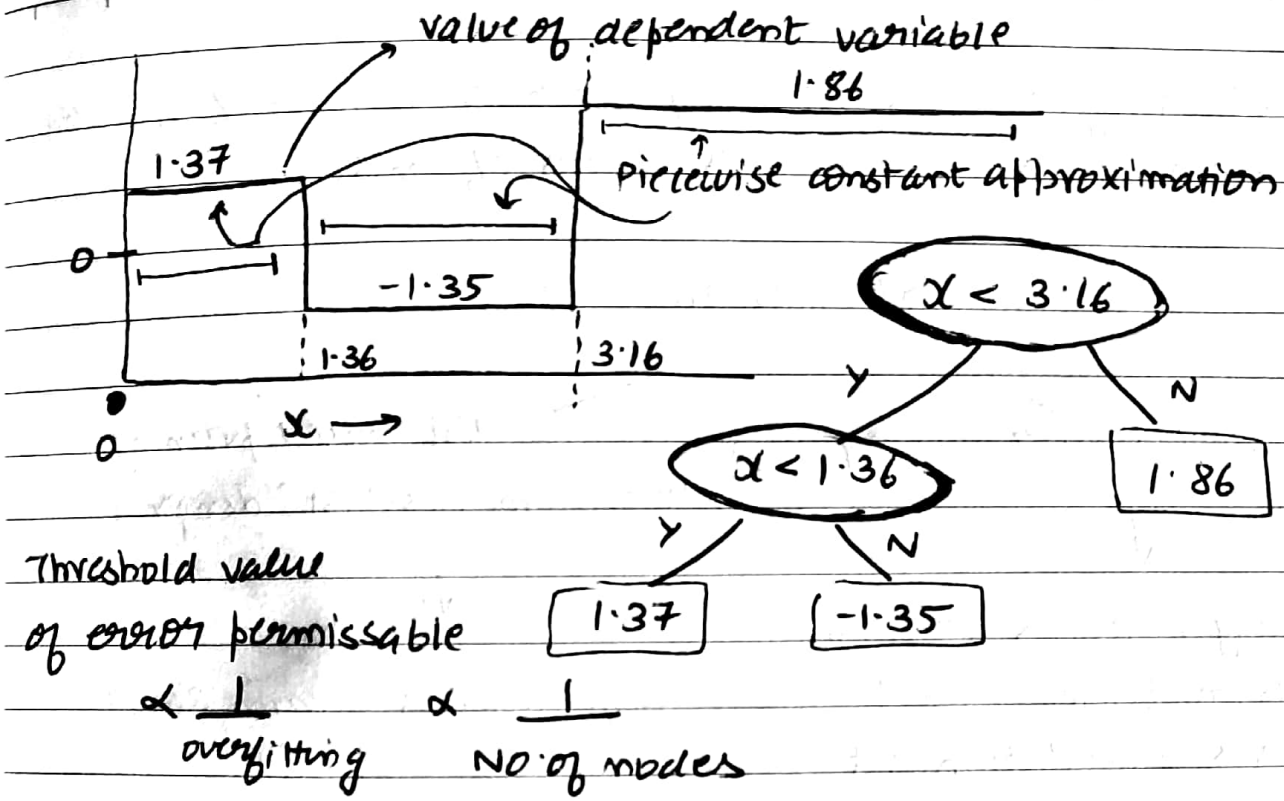
OVERFITTING ⟶ Very large trees with many leaves
UNDERFITTING ⟶ small trees with very few leaves.

TERMINATING CONDITION FOR DECISION TREES: Threshold value for impurity measure chosen

NO. of nodes = 0 & accuracy = 65%. ⇒ simply guessing the class based on prior probability



Date :



Threshold value of error permissable $\propto \frac{1}{\text{overfitting}}$ $\propto \frac{1}{\text{No. of nodes}}$

CAUSES OF OVERFITTING ⟶ Presence of noise (in data / class labels)

↳ lack of representative samples

↳ excessive parameter tuning

If mean squared error (MSE) > 0 ⟶ split else no split

Leaves with single instances ⇒ Pure leaves

PRUNING DECISION TREES ⟶ Remove subtrees for better generalisation

↳ prepruning, postpruning

⟶ Prepruning is faster, post pruning is more accurate (requires separate pruning set)

Date :

Post pruning involves replacing a subtree with a leaf provided the generalization error reduces because of this.

POST PRUNING

Reduced error pruning

↓

usually produces smallest version of tree., but it requires a validation set we are training on lesser data

MDL based pruning (Minimum description length)

↓

formalises Occam's Razor

→ tries to encode the model. Each of 'k' hypotheses are encoded. each with some length. we choose hypothesis with least length

HOW TO ENCODE DEC. TREE?

We need to encode every discriminant function. For 'd' attributes, we need $\log_2(d)$ bits for encoding

we need to encode hypothesis & errors on test set

length of model

if entropy = 0, no. of bits reqd. to encode = 0

A O V

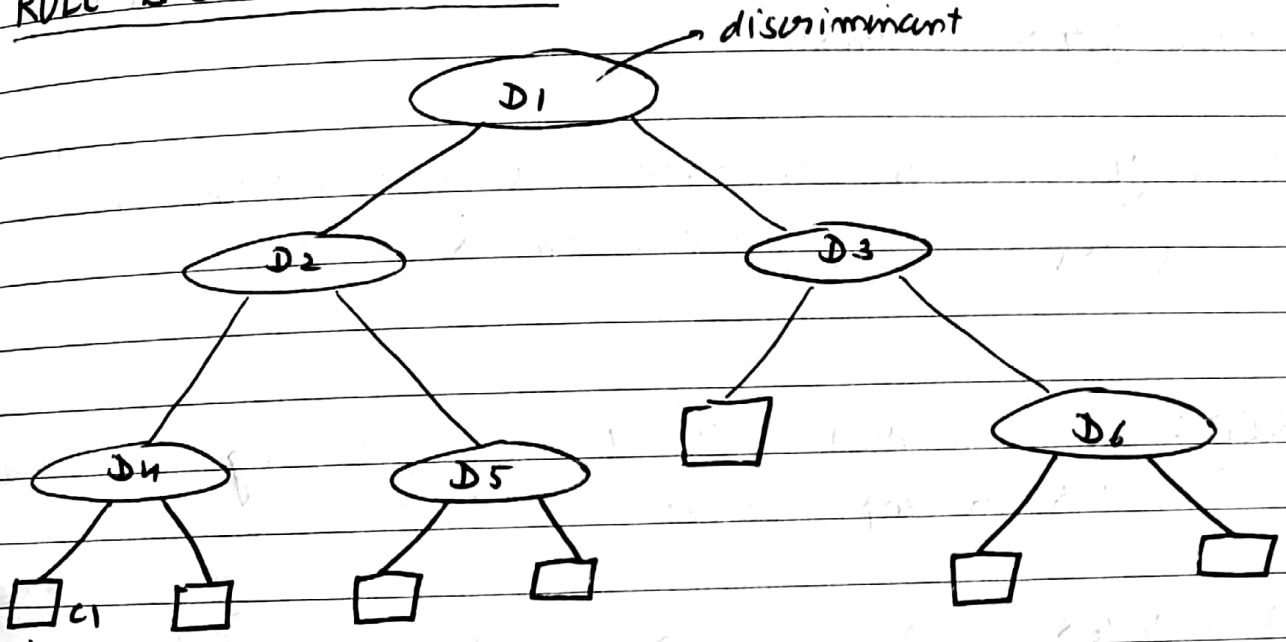
$\log_2 d$ 3 X

$\log_2(d) + 3 + X$ bits.

Reinforcement , VC, PAC, Regression , Look for slides
CH-1 CH2

Date :

RULE-BASED CLASSIFIERS



$(D1 \wedge D2 \wedge D4) \rightarrow C1$ — a way to represent hypothesis

If we only take all branches, then we get a RULESET
This is known as making rule based classifier from the tree

We can also make a rule based classifier from training set

Ruleset consists of rules which are conjunctions of terms.

Rule Induction ~ Tree Induction
(depth-first) (breadth first)
one rule at a time → adding conjunct to a rule until a quality criteria is matched

Ruleset (hypothesis)

$(A_1 O_1 V_1) \wedge (A_2 O_2 V_2) \wedge (A_3 O_3 V_3) \rightarrow$ class

Antecedent of Rule (body of rule) (description of concept)

consequent (head of rule) (concept)

Each rule is a sub-hypothesis

Date : _____

Rule covers an example if all terms of rule evaluate to true for the example.

SEQUENTIAL COVERING: Generate rules one at a time until all positive examples are covered.
(family of algos)

IREP : Incremental Reduced Error Pruning
Ripper is an improvement over IREP

2 APPROACHES → 1) learn the body & assign class
2) select the class & learn all the descriptions of that concept

There can be multiple descriptions for the same concept

Quality of classificat^n Rule : 1) COVERAGE
Fraction of records that satisfy the antecedent
2) ACCURACY
Fraction of records correctly classified

	COVERAGE	ACCURACY
R1	3/20	3/3 → of 3 records covered by this rule 3 are predicted correct by rule
R2	2/20	2/2

3 out of 20 records are covered by this rule

as class predicted by rule is same as class of that record in training set

Date :

	COVERAGE	~~ACCURACY~~
R1	4/20	
R2	3/20	
~~R3~~		

High coverage ⇒ large no. of rules satisfy those combo of attribute values

We add more conditions to a rule with less accuracy (& maybe high coverage), but it might lead to overfitting

We start with rules for the smallest class first

$C_1, C_2, \ldots C_k$ (sorted in increasing order of no. of instances / decreasing priority)

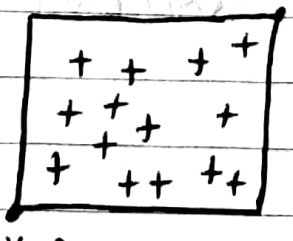
C_k → most instances ∴ rules at last

first rules for this

∵ there are few records with such combo of attribute values

∴ fewer rules would be reqd. for such cases

rules for such would be very specific hypothesis



Rule 1 (R1) describes this region

To develop a hypothesis for this, we need two coordinates to describe this region

Region covered by this hypothesis = $(x_h - x_l) \times (y_h - y_l)$

$(x_l \le x \le x_h) \wedge (y_l \le y \le y_h) \longrightarrow +$

Date : _____

An empty rule $\{\} \rightarrow C$ covers all the instances in data

$C_1 : 20$
$C_2 : 40$
$C_3 : 60$

$\{\} \rightarrow C_2$ coverage = 100 %.
accuracy = $\frac{40}{120}$ = 33.33 %.

$\{\} \rightarrow C_1$ coverage = 100 %.
accuracy = $\frac{20}{120}$

Adding more conditions to rule $\propto$ complexity $\propto$ constraints

start with empty rule &
keep on adding conjuncts

$\propto \frac{1}{\text{instances covered}}$

When to stop building a rule? : based on quality
values of coverage & accuracy

Adding each new test reduces rule's coverage
stop adding if accuracy > threshold & no significant ↑ in accuracy.

ripper algo

classifier ensemble

Date :

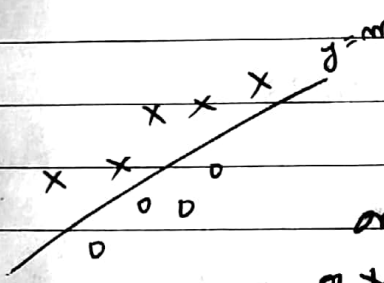
CHAPTER - 17

COMBINING MULTIPLE LEARNERS → combo of multiple classifiers of same or different types

- hybrid model
- one model for one region R1, other for other region R2, ...

- cross validat^n accuracy is more reliable than test accuracy.
- every classifier has its own assumptions

- Assumptions of a model → (leads to) Inductive Bias of model
  - it is the set of assumption that learner uses to predict O/P given I/P it hasn't yet encountered.



for any pt: $y = m$ gives the error $\forall x$ which depends on as to how far a value 'y' is from m of the line

$$E(\text{error}) = E(y - h(x))$$

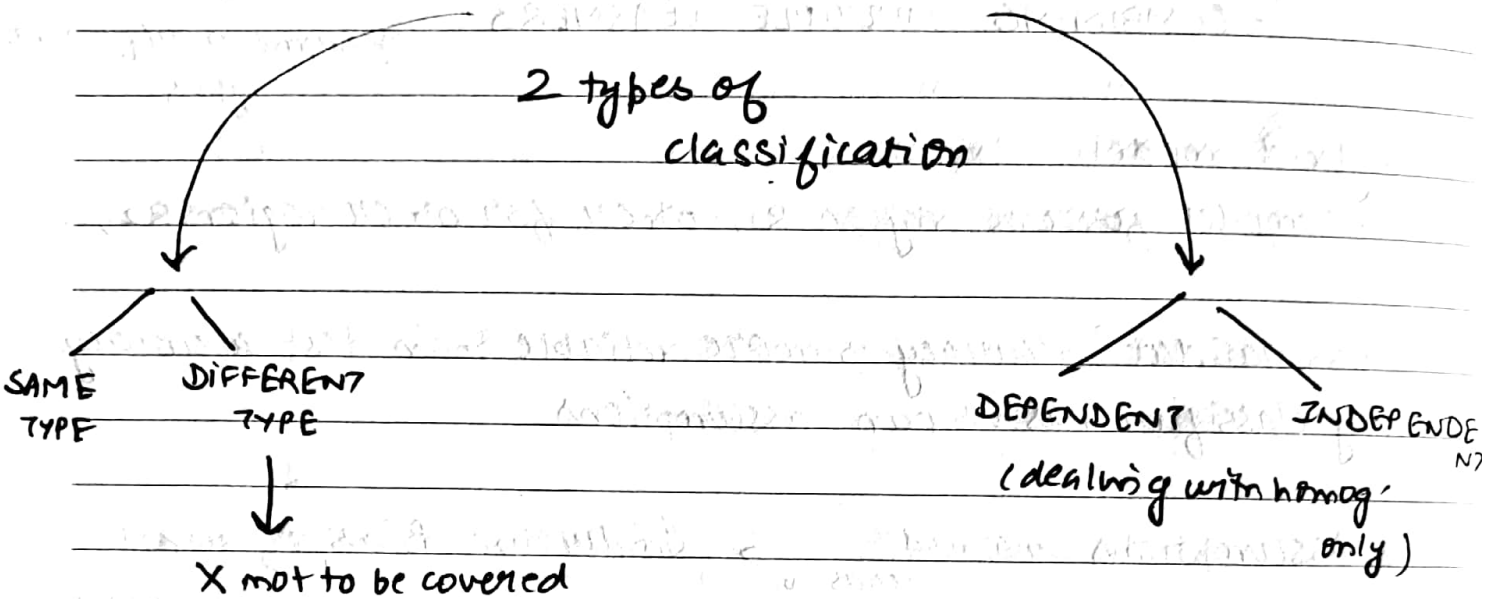
we combine multiple learners to reduce the overall errors
error may significantly reduce if we choose some other classifiers

we must —— 1) generate multiple learners ← SAME TYPE / DIFF. TYPE
2) combine those learners

generating multiple learners, all of same type are called Homogeneous Ensembles. We can also have heterogeneous ensembles if all learners aren't of same type.

Date :

GENERATING MULTIPLE LEARNERS



MOTIVATION FOR COMBINING MULTIPLE LEARNERS

1) Weak learner v/s Strong learner

↓

accuracy slightly > 50 %
which means its slightly better than random guess
(for binary classifiers)

↳ high prob. of being correct
like accuracy > 70%

- Combining multiple weak learners can result into a strong learner
- Combining multiple strong learners is computationally expensive & accuracy gains are not significant & sometimes even degrade. It's generation of such strong learners is also expensive

→ they predict correct at certain regions. So if we combine smartly we can use accuracy

Date :

2) Unstable classifier vs Stable classifier

↓ ↓

performance varies signif. if training set is manipulated slightly eg: Decision Tree Variance of model is very high on slight change in training set

performance doesn't affected significantly on slight change in training data

HOW TO GENERATE MULTIPLE CLASSIFIERS (ISSUE 1)

1) we can generate different Algorithms (only for heterogeneous)

2) By changing the hyperparameters (for homogeneous) Re-training same NN (neural n/w) will result in a diff. NN if initializations are different.

3) Different Modalities / Representation

↳ one classifier for images & one classifier for audio] classify a video

combine their results

Video 1	Images 1	Audios 1
Video 2	Images 2	Audios 2
⋮	⋮	⋮
Video m	Images m	Audios m

eg of creating a system of multiple classifiers

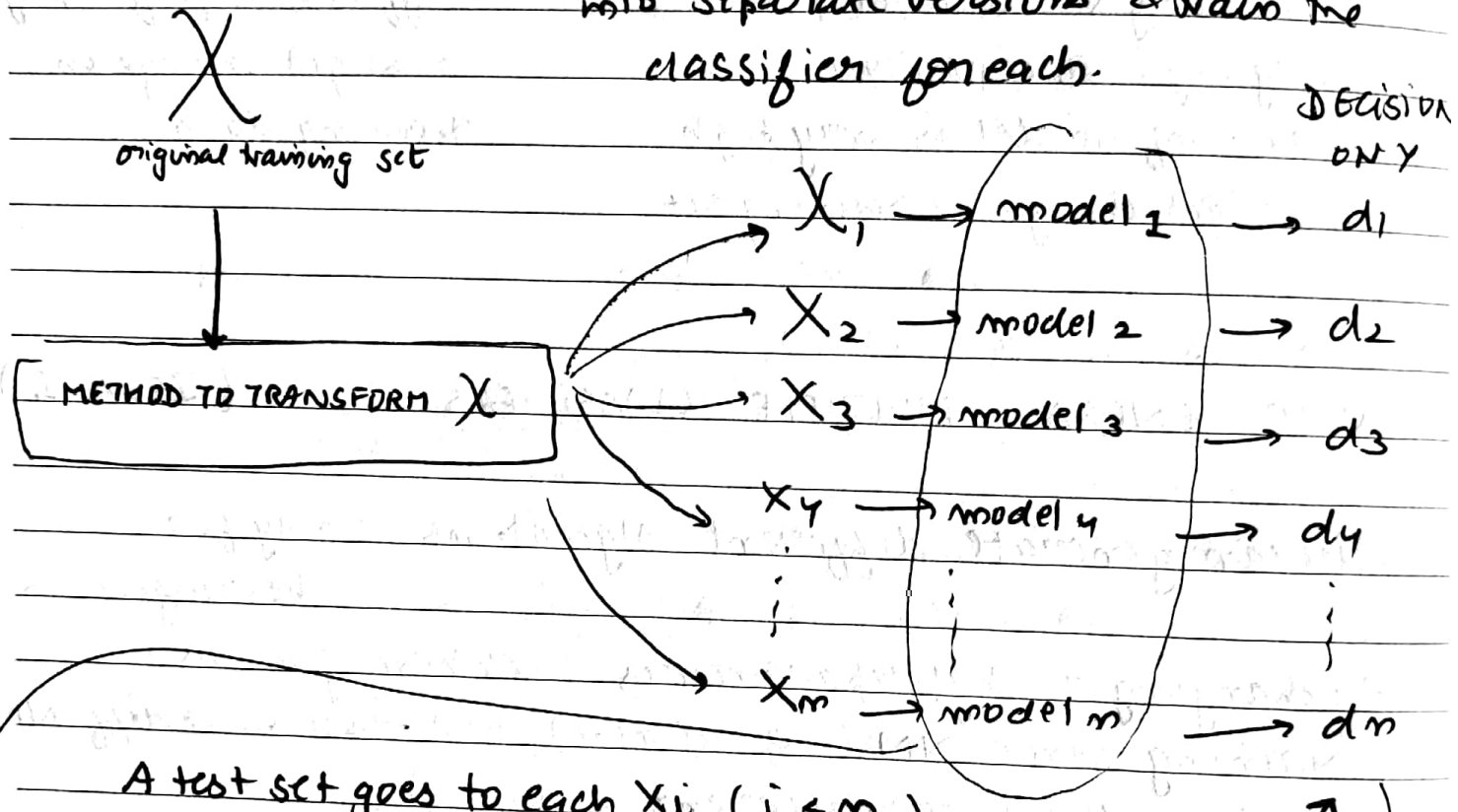
MODEL → M_{images} M_{audios}

→ combine to get O/P

Date :

4) Vary the training set (use different training set)

↳ how? : transform the same training set into separate versions & train the classifier for each.



A test set goes to each X_i ($i \leq m$) (Y)

COMBINER: $\hat{y} \leftarrow f(d_1, d_2, \ldots d_m)$

→ performance of such an ensemble will be better than expected if we have diverse classifiers i.e. each classifier makes mistakes in separate data regions. ∴ the o/p will be uncorrelated.

Date :

HOW TO COMBINE MULTIPLE LEARNERS

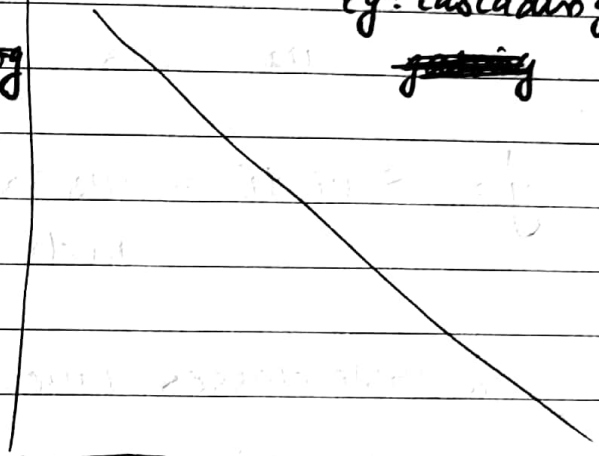
Global Methods

(combine O/P of all classifiers)

eg: voting, stacking, ~~gating~~ gating

Local Methods

eg: cascading, ~~gating~~



models →

$M_1 \quad M_2 \quad \cdots \quad M_L$

predictions →

$d_1 \quad d_2 \quad \cdots \quad d_L$

voting gives weight to each classifier →

$w_1 \quad w_2 \quad \cdots \quad w_L \quad s.t \left(0 \le w_i \le 1 \;\&\; \sum w_i = 1 \right)$

combine them →

$\hat{y} = \sum w_i d_i$ (weighted sum of O/P)

In case of classification : $\hat{y} = \sum w_i$ (for binary classification)

if $\sum w_i > 0.5$ then class 1

else class 0

Case 1: odd no. of classifiers

M_1	M_2	M_3	M_4	M_5
d_1 (0)	d_2 (1)	d_3 (0)	d_4 (0)	d_5 (1)
w_1	w_2	w_3	w_4	w_5

$\hat{y} = \sum w_i d_i = w_2 d_2 + w_5 d_5$

Date :

CASE 2 : even no of classifiers

M_1	M_2	M_3	M_4	
d_1	d_2	d_3	d_4	
(0)	(0)	(1)	(1)	→ 2 say '1' & 2 say '0'
w_1	w_2	w_3	w_4	in this case weights come into play

$$\hat{y} = \sum w_i d_i = w_3 d_3 + w_4 d_4$$
$$= w_1 d_1 + w_2 d_2 \text{ (if } w_1 = w_2 = w_3 = w_4)$$

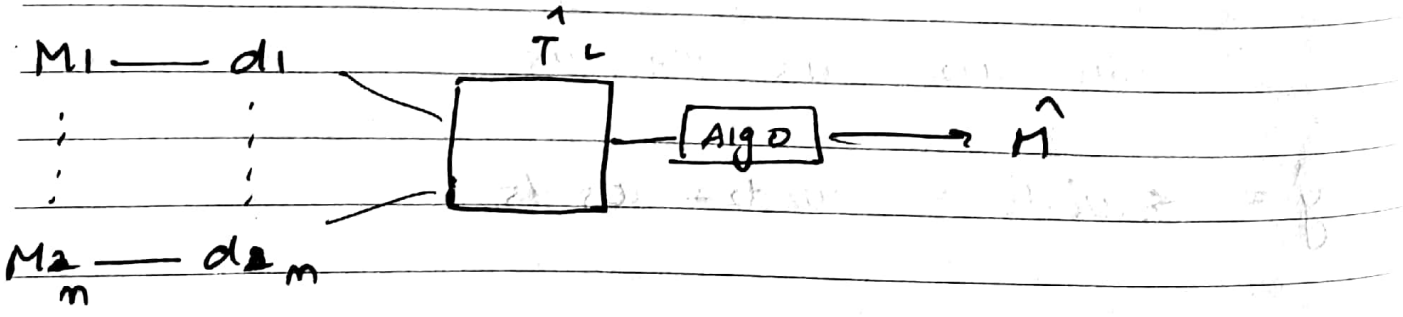
CASE 3 : multiple classes (we must use indicator functions)

d_1	d_2	d_3	d_4	d_5	d_6	d_7
(1)	(2)	(2)	(3)	(1)	(3)	(1)

$$y_i = \sum_j I(d_j = C_i)$$ → no of models predicting class i

Final class $\hat{y}$ = $\underset{i}{\text{argmax}}\ y_i$ → assuming all weights are same

METALEARNING (learning the learnings)

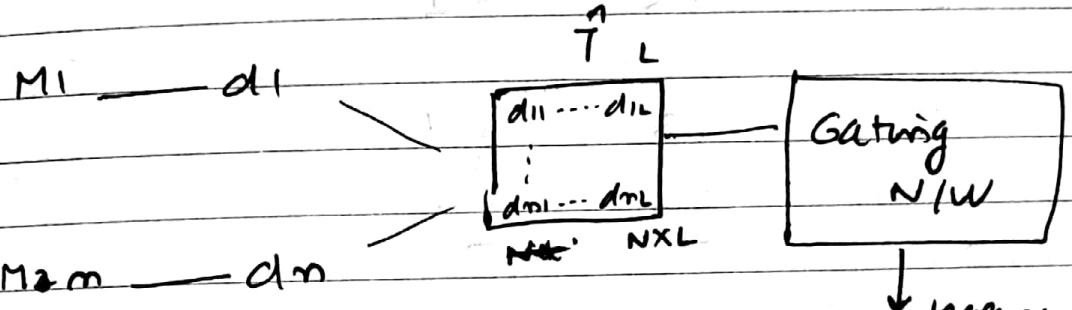


→ can maintain data privacy.

Date :

Stacking & Gating are also Global Methods

↓ a metalearning method



↓ learns gating funct^n's

Voting & Gating functions are diff: as weights in Gating functions are learnt from the ~~network~~ training set

$\sum_j W_j d_j$ ← each assigns probabilities to models (just like weights)

j ↓ diff: from weights in voting's

CASCADING : every model is assigned some confidence level (threshold) & when instance is to be predicted representation of instance is checked with each model If confidence of a model for that instance is low, we proceed to next model. It's a local prediction as only one model is going to predict the class.

Combining Methods for Global Methods : Average, Min, Median, Product (depending on applicat^n)

↓ in Regression

If algo is predicting Prob: of class, then average can be used

Date :

	C_1	C_2	C_3
d_1	0·2	0·5	0·3
d_2	0·7	0·2	0·1
d_3	0·3	0·5	0·2
d_4	0·6	0·3	0·1
SUM	—	—	—
or AVERAGE	sum & average are correlated methods as sum ∝ average.		

Diff algo have diff inductive bias ∴ of which they might focus on diff data regions. ∴ combining such algos can result in better decision boundaries.

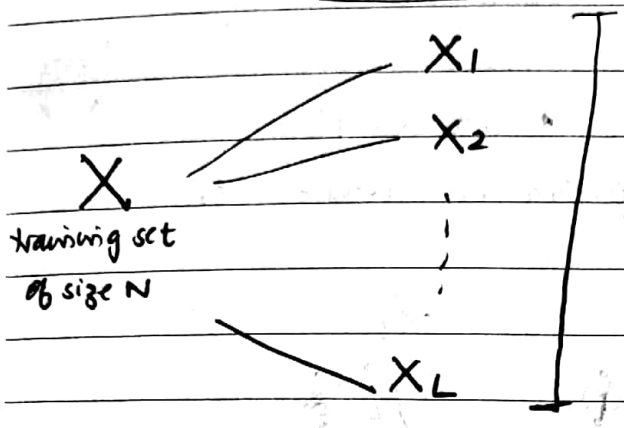
Gain ~~Gain~~ ∝ Diversity of classifiers

NFL (NO FREE LUNCH) THEOREM : There is no algo that is always the most accurate

Date :

BAGGING (ENSEMBLE METHOD) → Bootstrap Aggregation

random sampling with replacement



random samples with replacement
each of size N
without replacement makes $X_1 = X_2 = \dots X_L$
∴ no diversity remains

in case of multiclasses,
we use indicator functions

$X_1 \rightarrow M_1$
$X_2 \rightarrow M_2$
$X_L \rightarrow M_L$

$$\sum_L w_j d_j \left(\frac{1}{L} \forall j\right)$$

we take average as $w_j = \frac{1}{L} \forall j$

How much repetition can we expect?

$\left(1 - \frac{1}{N}\right)^N$: prob. of an instance not coming in X

$\approx \frac{1}{e} \approx 0.632$ ~~0.37~~

There is 37% chance that an instance not coming in X
There are 63% unique elements in each training set

Date :

χ = $< x, y >$, $x \in \mathbb{R}^d$, $y \in \mathbb{R}$ | $y \in C$

- d dimensional data
- $y \in \mathbb{R}$ → numeric classes
- $y \in C$ → in case of discrete class values
- C → class set

training set

ASSUMPTIONS

- $\chi \sim P_{xy}$ — training set comes from the distribution defined by x & y
- $x_0 \sim P_{xy}$

1) Create L training sets $\chi_1, \chi_2, \ldots, \chi_L$

2) Create models → $h_{\chi_1}(\cdot), h_{\chi_2}(\cdot), \ldots h_{\chi_L}(\cdot)$

assumptions : 1) weak learners

2) unstable learners → high variance

$h_1(x_0), h_2(x_0), \ldots, h_L(x_0)$

for x_0 → if we use hypothesis h_1 → $< x_0, h_1(x_0) >$

" " " " h_2 → $< x_0, h_2(x_0) >$

$< x_0, h_L(x_0) >$

In Bagging, we take average of $h_i(x_0)$, i.e. expected value of h over L classifiers → $E(h(x_0)) = h_B(x_0)$

we want to find out how far (on average) is each prediction from $h_B(x_0)$ or $E(h(x_0))$ → bagged estimate

$= E\left((h(x_0) - E(h(x_0)))^2 \right)$

= variance for x_0

Date :

eg. of unstable classifiers : decision trees, KNN, ANN
A linear classifier is a stable classifier

Stable classifiers have high bias -
→ the error in prediction, due to the assumptions taken.

$x, y \in \mathbb{R}$

$\epsilon \sim N(0, \sigma^2)$ → all errors cancel out ∴ $\mu = 0$

$$y = f(x) + \epsilon$$

y → dependent variable / predicted

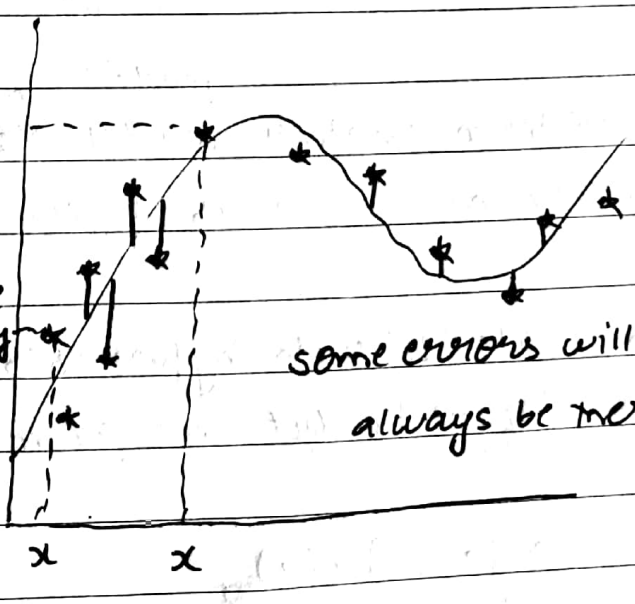
$f(x)$ → independent variable / predictor

ϵ → error / noise associated with each record

For simulated data, we know $f(x)$, but it isn't so in real life

x	y
-	-
-	-
-	-
-	-
-	-
-	-
-	-
-	-
-	-
-	-

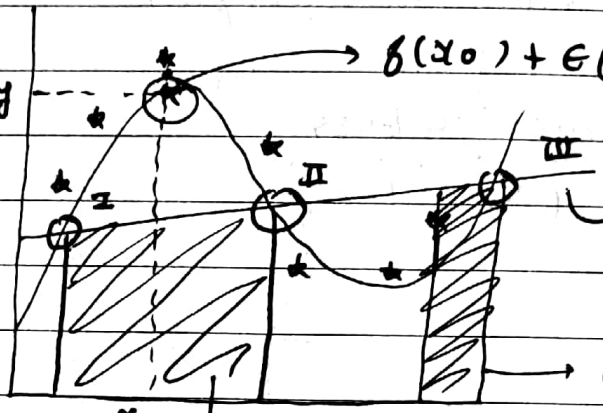
assuming f to be some sinusoidal wave



some errors will always be there.

Date :

Suppose we learn our model with these 2 parameters α, γ



∴ our model is overestimating in one data region while underestimating at other

only I, II, III are predicted without error as $h(x_0) = f(x_0)$

Bias for x_0 (of the model) = $h(x_0) - f(x_0)$

predicted — true

Suppose we train another model h_2, it could be better or worse than h_1. ∴ Bias $h_1(x_0)$ will be diff. from $h_2(x_0)$

We can create 'L' such hypotheses each with their own error regions. If we predict x_0 from all of them:

$h_1(\cdot), h_2(\cdot), \ldots\ldots, h_L(\cdot)$

↓ ↓

$h_1(x_0), h_2(x_0) \ldots\ldots, h_L(x_0)$

$E_L(h(x_0)) = \frac{1}{L} \sum h_i(x_0)$

Date :

The variance would lie somewhere in mid of all predicted values for x_0.

$$E\left[(h(x_0) - E(h(x_0)))^2\right]$$

Bias in Decision trees comes because of assumptions (attribute selection measures, depth of tree).

Bias, Variance, Noise
(B) (V) (N)

$$\text{Error} = B^2 + V + N$$

$$\text{Squared error} = (y_0 - h(x_0))^2$$

Mean squared error for x_0

$$= E_L\left[(y_0 - h(x_0))^2\right]$$

$$= E\left(y_0^2 - 2h(x_0)y_0 + h^2(x_0)\right)$$

$$= E(y_0^2) - 2E(h(x_0))E(y_0) + E(h^2(x_0))$$

(A) (B) (C)

— (2)

We know

$$E((x - \bar{x})^2) = E(x^2) - [E(x)]^2$$

$$\Rightarrow E(x^2) = E((x - \bar{x})^2) + [E(x)]^2$$

— (1)

Date :

using ① in ② to compute $E(y_0^2)$

A → $E(y_0^2) = E((y_0 - E(y_0))^2) + [E(y_0)]^2$

C → $E(h^2(x_0)) = E((h(x_0) - E(h(x_0)))^2) + (E(h(x_0)))^2$

Putting these A & C in ②

$= E((y_0 - E(y_0))^2) + [E(y_0)]^2 - 2E(h(x_0))E(y_0)$

$+ E((h(x_0) - E(h(x_0)))^2) + [E(h(x_0))]^2$

($h(x_0)$ — predicted; $E(h(x_0))$ — average prediction of weak learners → varying)

$= E((y_0 - f(x_0))^2)$ — error

$+ E((h(x_0) - E(h(x_0)))^2)$ — variance

$+ E(f(x_0))^2 - 2E(h(x_0)$

$E(f(x_0)$

$+ E(f(x_0))^2$

$f(x_0)^2 - 2h(x_0)f(x_0) + f(x_0)^2$

$= E((f(x_0) - h(x_0))^2)$

Bias (square of bias)

Date :

1) Find out of bag accuracy
2) select unique out of bag instances, & for each find bias & variance

1000 pts. take randomly, learn a linear regression model from this dataset.
Create L sets : $x_1 \dots x_L$ L Models.
$M_1 \dots M_L$

Bagging only reduces the variance only thereby reducing overall error $E = B^2 + \underline{V} + N$

Variance of hypothesis for x_0 = $E\left((h(x_0) - E_L(h(x_0)))^2\right)$
Overall variance = $E_x\left[E\left((h(x_0) - E_L(h(x_0)))^2\right)\right]$

Date:

for classificatⁿ

Adaptive Boosting Algorithm (staged classifier)

works stage wise incremental approach to making ensemble

ADABOOST (proposed in 1990)

↳ all its claims are provable mathematically

↳ much better ensemble method than bagging

↳ boosts performance of weak classifiers

Boosting: general philosophy based on idea that if you create a model M1 & predict the instances in training set, then M1 will predict some correctly while some will be predicted incorrectly.

(I) X_1 (all have weight = 1/N) ⟶ M_1 ⟶ correctly predicted — (1)

(contains the initial training set) ⟶ Incorrectly predicted — (2)

Now, we will create another M_2 which focuses on incorrectly predicted instances of M_1

(II) X_2 ⟶ M_2 : we assign more weights to instances in (2)

↓ derived from X_1 s.t incorrectly predicted in M_1 are given more importance. M_2 must focus on (2)

X_1 = training set X (of N instances) from which sampling is done ~~without replacement~~ with P(an instance is selected) = $\frac{1}{N}$

$\frac{1}{N}$ = Weight of i^{TH} instance in 0^{TH} iteration = w_i^0, $\sum w_i = 1$

Incorrectly classified in (I) will have weight $> \frac{1}{N}$ &

correctly ~~pre~~ classified in (I) will " " $< \frac{1}{N}$

Date :

$X_2 \longrightarrow M_2$ → correctly predicted — (3)
→ incorrectly predicted — (4)

$W_i^{(1)} = \frac{1}{N}$, $W_i^{(2)}$ → $> \frac{1}{N}$ if x_i is wrongly pred.
→ $< \frac{1}{N}$ if x_i is correctly pred.

$W_i^{(3)}$ → $> w_i^{(2)}$ if wrongly
→ $< w_i^{(2)}$ if correctly.

We want to stop when:

1) 'L' classifiers (iterations) have been completed
2) If accuracy < 50 %. i.e. random guess then stop (in case of binary classifier)

We carry on the correctly predicted instances for in next iterations so that size of set remains appreciable

$$w_i^{l} = f(w_i^{l-1}, \text{performance measure})$$

After (2) all incorrectly predicted will have same weight

- Bagging had its classifiers working parallelly. Adaboost works sequentially, As X_i can't be made without knowing X_{i-1}.
- All classifiers are independent in bagging. & Each training set is dependent on performance of model created by prev. training set in case of adaboost.

Date :

HOW TO COMBINE PREDICTIONS ? → weighted sum of predictions

$$h_{ADABOOST}(x) = \sum_{i} \alpha_i \, h_i(x) \quad \text{—— (5)}$$

i ← no. of classifiers

α_i → weight of i^{th} classifiers

$h_i(x)$ → prediction of i^{th} classifier

Adaboost was initially used for binary classificat^n. ✓ ✗

with prediction classes = {1, -1}

-1 taken instead of 0

to denote wrong prediction for mathematical convenience

Bagging was initially for regression purposes.

(5) => $h_A(x) = \{\sum_i \alpha_i h_i(x)\}$ $\text{sign}(\sum_i \alpha_i h_i(x))$

HOW TO ASSIGN WEIGHTS α_i ?

$$\alpha_i = g(\text{performance})$$

DECISION CRITERIAS

→ 1) Adjust weights (generating Training set / Models)

2) Determine weights of classifiers

Date :

LOSS FUNCTION

$$\text{ERROR}_{(0/1 \text{ Loss})} = \sum_{x \in X} I(y \neq h(x)) \quad \text{---} \quad (6)$$

Loss Function followed by Adaboost is Exponential loss

$$\text{loss}_{exp}(x) = e^{-yh(x)} \quad \text{where} \quad y = +1, -1$$
$$h(x) = +1, -1$$

For correct prediction:

y	$h(x)$	$yh(x)$
1	1	= 1
-1	-1	

$\Rightarrow$ loss $= e^{-1} = \frac{1}{e}$

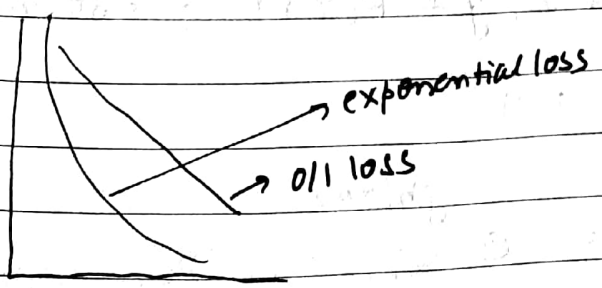
For wrong prediction

y	$h(x)$	$yh(x)$
1	-1	-1
-1	1	-1

$\Rightarrow$ loss $= e$

$$\text{loss for classifier} = \text{loss}_{exp}(h(x)) = \sum_{x} e^{-yh(x)}$$

Exponential loss function will reduce faster than (6)



Date :

→ uses exponential loss

ADABOOST → staged, incremental, dependent classifiers, combines using weighted additive function

$$y_i = \text{Sign}\left(\sum_{L} \alpha_i h(x_i)\right)$$

Sign → ∵ class can be +1 or −1

L → over all models

$y = h(x)$ ⇒ correct prediction Loss = 0

$y \neq h(x)$ ⇒ incorrect prediction Loss = 1

~~$\sum_{x} \sum_{a} I(y \neq h(x))$~~

$\sum_{x \in X} I(y = h(x))$: correct prediction

$\sum_{x \in X} I(y \neq h(x))$: training error

$\sum_{x \in T} I(y \neq h(x))$: test error

$\frac{|X| - \sum I(y \neq h(x))}{|X|}$: training accuracy

$\frac{|T| - \sum_{x \in T} I(y \neq h(x))}{|T|}$: test accuracy

- $\text{Loss}_{exp}(x_0) = e^{-y h(x_0)}$

Date :

- $Loss_{exp}(X) = \sum_{x \in X} e^{-yh(x)}$

Risk is expected loss $= E(Loss_{exp}) = E(e^{-yh(x)})$

if $y = h(x)$, $y \cdot h(x) = 1$, $loss = e^{-1}$ — (1)

if error, i.e. $y \neq h(x)$, $y \cdot h(x) = -1$, $loss = e$ — (2)

∴ using (1) & (2),

~~$E(e^{-yh(x)})$~~

$$\sum_{x \in X} e^{-yh(x)} = \underbrace{\sum e^{-yh(x)}}_{\text{correct}} + \underbrace{\sum e^{-yh(x)}}_{\text{wrong}}$$

(I) (II)

We must reduce (II) as it contributes more

Ideally, (II) = 0

least value = ne^{-1}

Adaboost was originally designed for improving classification trees

Bagging was " " " regression trees

$M1 \longrightarrow M2 \longrightarrow M3 \longrightarrow \dots \longrightarrow ML$

$M1$: $y = h_1(x_0)$ → one specific record

$M2$: $y = \alpha_1 h_1(x_0) + \alpha_2 h_2(x)$

$M3$: $y = \sum \alpha_i h_i(x_0)$

ML: model from L^{th} training set

Date :

HOW TO ASSIGN WEIGHTS TO HYPOTHESIS?

→ upto some (L-1), we have the hypothesis $H(x) = \sum_{i=1}^{L-1} \alpha_i h_i(x_0)$

→ suppose $h(x_0)$ is hypothesis at stage L &
α is the weight of $h(x_0)$

we have to find α.

Now, $y_0 = \underbrace{H(x_0) + \alpha h(x_0)}_{\text{hypothesis of } x_0 \text{ at } L^{th} \text{ stage}}$ → prediction for y_0 at stage L

loss (x_0) at L^{th} stage $= e^{-y(H(x_0) + \alpha h(x_0))}$

loss $(X) = E_{xy}\left(e^{-y(H(x) + \alpha h(x))}\right)$

↳ at L^{th} stage

$$= E_{xy}\left(\underbrace{e^{-yH(x)}}_{\text{loss at L-1 stages we already have it so let it be constant}} \cdot e^{-y\alpha h(x)}\right)$$

this is expected loss
$e^{-\alpha y h(x)}$ will be pointwise loss

$$= e^{-yH(x)} \cdot E_{xy}\left(e^{-\alpha y h(x)}\right)$$

loss (X) will be minimum if for every instance loss is minimum, in this case if $e^{-\alpha y h(x)}$ is min.

Date :

IMPORTANT NOTATIONS

$E_{xy}\left[(y - h(x))^2\right]$: Mean squared training error

$e^{-yh(x)}$: exponential loss , pointwise loss

$E_{xy}\left(e^{-yh(x)}\right)$: expected loss over entire training set.

$$h(x) = \sum_{i=1}^{L} \alpha_i h_i(x)$$

$\underbrace{e^{-yH(x)}}_{\text{This is constant \& we are assuming to have minimised it at } (L-1)^{th} \text{ stage}} \cdot E_{xy}\left(e^{-\alpha y h(x_0)}\right)$

$\Rightarrow$ minimize $E_{xy}\left(e^{-\alpha y h(x_0)}\right)$ — (1)

we already have y from training set, $h(x_0)$ is random

we know that $E(x) = \sum x_i * P(x_i)$

(1) is proceeded to have 2 cases, when $y = h(x)$ & $y \neq h(x)$

$\therefore$ (1) $= \left[e^{-\alpha} * P(y = h(x_0)) + e^{\alpha} * P(y \neq h(x_0)) \right]$

$e^{-\alpha}$ ← correct prediction → : $y \cdot h(x) = 1 * 1 = 1$

e^{α} ← incorrect prediction → : as $y \cdot h(x) = -1$

∴ we took classes as 1, -1 & not 1, 0

y	$h(x)$	loss
0	0	e^0
0	1	e^0
1	0	e^0
1	1	e^1

→ incase of correct prediction = incorrect prediction which shouldn't happen

we take $\{-1, 1\}$ to reduce loss quickly

(as what is adv. of predicting correctly ? → ANOMALY

Date :

let incorrect prediction be denoted by ϵ

$\therefore$ correct prediction be denoted by $1-\epsilon$

$\therefore$ (1) $= e^{-\alpha}(1-\epsilon) + e^{\alpha}\epsilon$ —— (2)

in order to minimize it, we must partial differentiate it w.r.t α, i.e. we have to find the weight that minimizes loss

$$\frac{\partial E(\text{Loss}_{\text{exponential}})}{\partial \alpha} = -e^{-\alpha}(1-\epsilon) + e^{\alpha}\epsilon \quad \text{—— (3)}$$

put (3) $= 0$ $\rightarrow -e^{-\alpha}(1-\epsilon) + e^{\alpha}\epsilon = 0$

$\Rightarrow e^{-\alpha}(\epsilon - 1) + e^{\alpha}\epsilon = 0$

$\Rightarrow \frac{\epsilon}{e^{\alpha}} - \frac{1}{e^{\alpha}} + e^{\alpha}\epsilon = 0$

$\Rightarrow \epsilon - 1 + e^{2\alpha}\epsilon = 0$

$\Rightarrow \epsilon(e^{2\alpha} + 1) - 1 = 0$

$\Rightarrow \epsilon = \frac{1}{e^{2\alpha}+1}$

$\Rightarrow e^{2\alpha} + 1 = \frac{1}{\epsilon}$

$\Rightarrow e^{2\alpha} = \frac{1}{\epsilon} - 1$

$= \frac{1-\epsilon}{\epsilon}$

$\Rightarrow 2\alpha = \log\left(\frac{1-\epsilon}{\epsilon}\right)$

$\Rightarrow \alpha = \frac{1}{2}\log\left(\frac{1-\epsilon}{\epsilon}\right)$

Date :

Reduction is very fast

$e^{-yh(x)}$

	$\{0, 1\}$			$\{-1, 1\}$		
	y	$h(x)$	Loss	y	$h(x)$	Loss
Correct Pred	0	0	e^0 (1)	-1	-1	e^{-1} (.37)
	1	1	e^{-1} (.37)	1	1	e^{-1} (.37)
Wrong Pred	0	1	e^0 (1)	-1	1	e (2.7)
	1	0	e^0 (1)	1	-1	e (2.7)

penalty smaller with $\{-1, 1\}$

This has bigger penalty

Decrease in error = $v - (1 - 0.37)$

Decrease in error = $v - (2.7 - 0.37)$

Base-learners

↓

These are the classifiers used

Decision Trees don't deal with probabilities

weights of sample records = prob. distribution of sample

For a s record, let p_1 be its prob. of selection in first iteration

Let p_2 be " " " " " second "

Then $p_2 < p_1$ for correct prediction

$p_2 > p_1$ for wrong prediction

ITERATION 1	ITERATION 2	Normalising
$w_1^{(1)}$	$w_2^{(1)} / \sum w_2^{(i)}$	If $\epsilon = 0.5$ $\alpha = \frac{1}{2}\log\left(\frac{1-\epsilon}{\epsilon}\right) = 0$ ∴ we ignore such classifiers
$w_1^{(N)}$	$w_2^{(N)} / \sum w_2^{(i)}$	If $\epsilon < 0.5$ α = +ve as $\log\left(\frac{1-\epsilon}{\epsilon}\right) > 1$ this classifier is better than random guess
prob. distribution in first round as $\sum w_i = 1$ as each $w_i = \frac{1}{N}$	we divided by $\sum w_2^{(i)}$ to get prob distribution otherwise $\sum w_2 > 1$	If $\epsilon > 0.5$ this classifier is worse than random guess & we stop reject this classifier

$\beta = \frac{\epsilon}{1-\epsilon}$ we can $\frac{1-\epsilon}{\epsilon} = \beta$ but calculations are to be varied

1ST iteration	2ND iteration	
1/4	1/6	1/6 × 3/4
1/4	1/6	1/6 × 3/4
1/4	1/2	1/2 × 3/4
1/4	1/2	1/2 × 3/4
$\sum = 1$	$\sum = \frac{4}{3} \neq 1$	$\sum = 1$

∴ now its a prob. dist.

Date :

DISCRIMINANT FUNCTION

Bayes classifier — Rule-based classifier — Decision Tree

Probabilistic — Non-Probabilistic — Non-Probab.

Non-parametric

here, we assume the form of probability distribution func. ∴ parametric

doesn't use probability for classificat^n its used only in impurity measures

learns the hierarchy, not always one at a time for class

learns description of each class one at a time

as conjunction of conditions

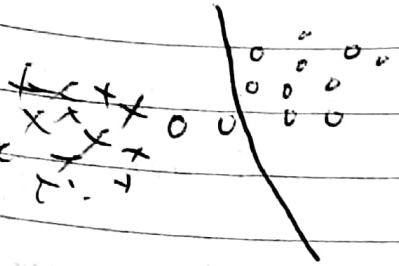
we calculate prior probabilities from the data, also the likelihood & class conditional probability.

Decision trees looks at data as a whole while Rule based looks one class at a time.

→ importance of conditions at global level

All classifiers learn decision regions & decision boundaries

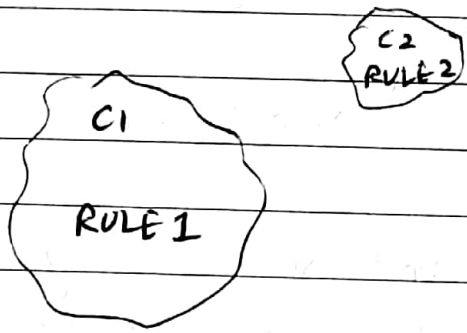
Bayes classifier learns a decision region using class conditional prob. prob. dist. function



Date :

- Decision tree learns decision boundaries based on importance of conditions in hierarchical manner.

Rule based classifier learns decision boundary (arbitrary) using class description. One class can be described by a no. of regions.



Non linear classifiers :: boundaries are non linear
Bayes classifier is typically non linear but it can be shown linear under certain constraints

DISCRIMINANT FUNCTN: A function which discriminates b/w classes

For each class, we have a discr. function $g_i(x)$ for given x

$g_1(x)$
$\vdots$
$g_k(x)$

→ assign class with max value.
(used by Naive Bayes classifier)

Let $g_0(x)$ & $g_1(x)$ be 2 ~~discr~~ function values

$g(x) = [g_1(x) - g_0(x)] > 0 \Rightarrow g_1(x) \therefore C_1$
else C_0

if $g = 0$ then either

Date :

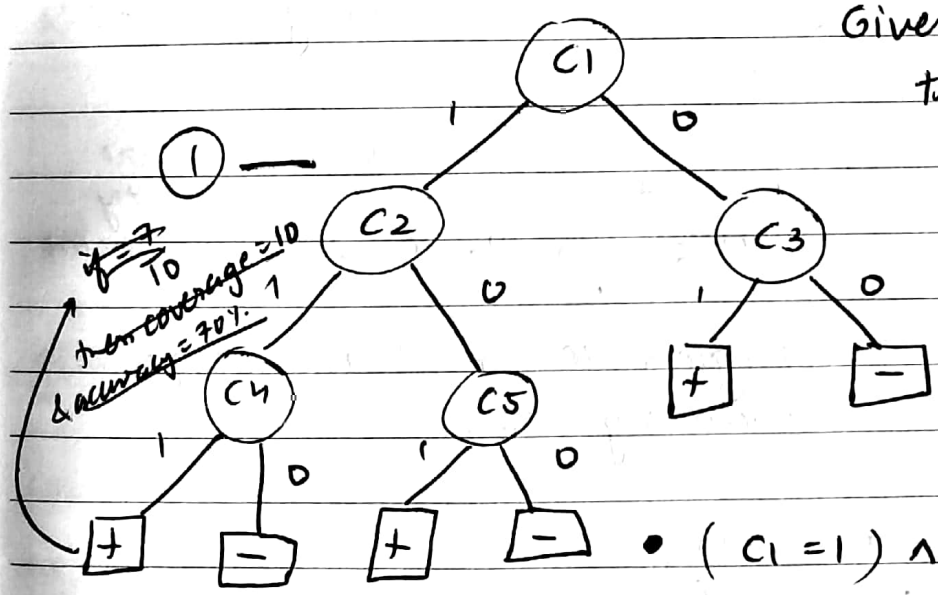
Linear Discriminant Function (in syllabus)

↳ class boundary is linear → divides data into 2 regions

1-D data : a point is a boundary

2-D data : a line (i.e. $y = mx + c$ is a boundary)

3-D data : a plane



Given this decision tree, create the corresponding ruleset.

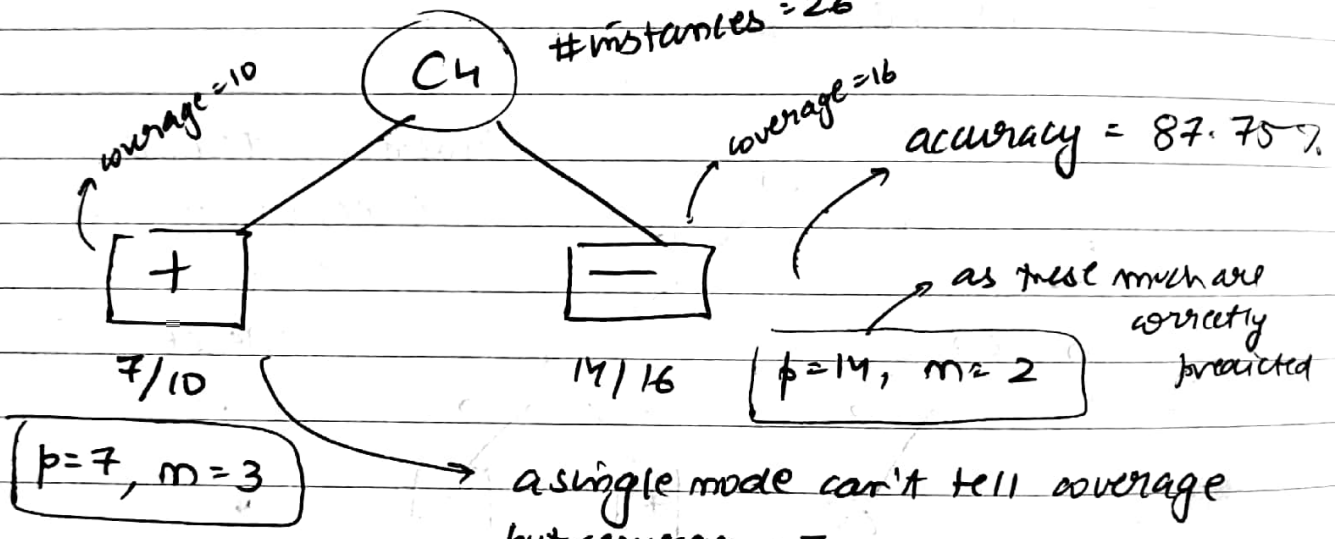
- $(C_1 = 1) \wedge (C_2 = 1) \wedge (C_4 = 1) \longrightarrow$ '+'
- $(C_1 = 1) \wedge (C_2 = 1) \wedge (C_4 = 0) \longrightarrow$ '−'
- $(C_1 = 1) \wedge (C_2 = 0) \wedge (C_5 = 1) \longrightarrow$ '+'
- $(C_1 = 1) \wedge (C_2 = 0) \wedge (C_5 = 0) \longrightarrow$ '−'
- $(C_1 = 0) \wedge (C_3 = 1) \longrightarrow$ '+'
- $(C_1 = 0) \wedge (C_3 = 0) \longrightarrow$ '−'.

'+' rules :
- $(C_1 = 1) \wedge (C_2 = 1) \wedge (C_4 = 1)$
- $(C_1 = 1) \wedge (C_2 = 0) \wedge (C_5 = 1)$
- $(C_1 = 0) \wedge (C_3 = 1)$

Arrange acc. to no. of Rules :
- $(C_1 = 0) \wedge (C_3 = 1)$
- $(C_1 = 1) \wedge (C_2 = 1) \wedge (C_4 = 1)$
- $(C_1 = 1) \wedge (C_2 = 0) \wedge (C_5 = 1)$

Date :

① since is a binary tree, ∴ the no. of rules for each class were same. However, this might not always be true in case of multiclass rules. & to create a decision tree from ruleset, no. of rules from each class must be same.

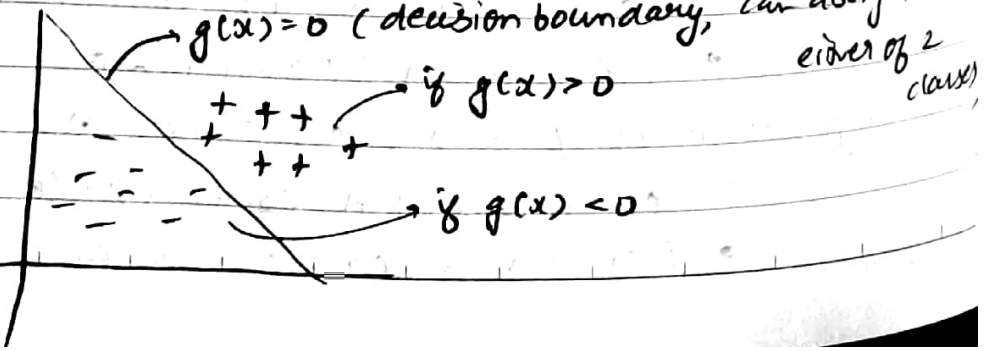


$$\frac{7+14}{26} = \frac{21}{26}$$

DISCRIMINANT FUNCTION (CONTD.)

LINEAR DISCRIMINANT FUNCTION FOR 2D DATA

$g(x) = mx + c$, $g(x) = 0$ defines the linear boundary



Date :

IN 3D DATA

linear Discr. function takes form of a plane with one class above the plane & one class below the plane

& each instance $X = (x_1, x_2, x_3)$

if $g(x) = 0$, $\Rightarrow$ pt lies on decision boundary plane of the 2 classes

IN d-D DATA

↳ called hyperplane.

$$X = \{x_1, x_2, \ldots, x_d\} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix}_{d\times 1} \qquad w = \begin{bmatrix} w_{i1} \\ \vdots \\ w_{id} \end{bmatrix}_{d\times 1}$$

~~$g(x) = w_0 + w_1^T$~~

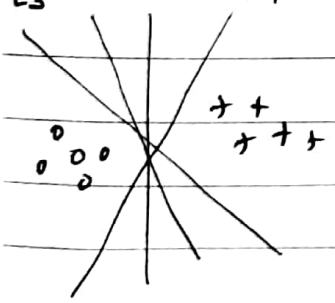
$$g_i(x) = \underbrace{w_{i0}}_{\text{scalar}} + \overbrace{w_i^T x}^{\text{scalar}} \rightarrow \text{scalar}$$

scalar + scalar gives real value of $g_i(x)$

↳ tells locat^n of plane in d-D space wrt origin

↳ locat^n of hyperplane

'm' slope defines orientat^n of line

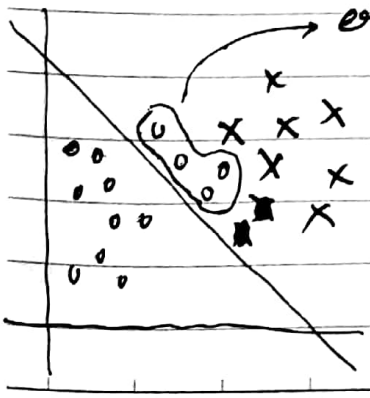


↳ choose the line which maximizes the margin

this can be done by SVM (support vector machines)

• I)

How to predict a line? : start with a random value of m & c



error, we need to shift the boundary → how much to shift? (η)

if we shift above a lot, $g(x)$ will drastically change & then might a lot of instances incorrectly.

Date :

η should not be very big (as before)

η should not be very small as convergence to soln will be very late

Each time, we should change the plane by different amount

→ depends on the amt of error, Shift ∝ error

If data is not linearly separable (completely)

then loss function will never be 0.

Date :

η should not be very big (as before)
η should not be very small as convergence to soln will be very late
Each time, we should change the plane by different amount

↳ depends on the amt of error, Shift $\propto$ error

If data is not linearly separable (completely)
then loss function will never be 0.

For the Bayes classifier

Takes 'd' dimensional data ⟶ o/p is prob. [0, 1]

$B(X)$: $\mathbb{R}^d \longrightarrow \mathbb{R}$

$B(X) > 0.5 \longrightarrow$ class 1
else $\longrightarrow$ class 0

] only if 2 class problem

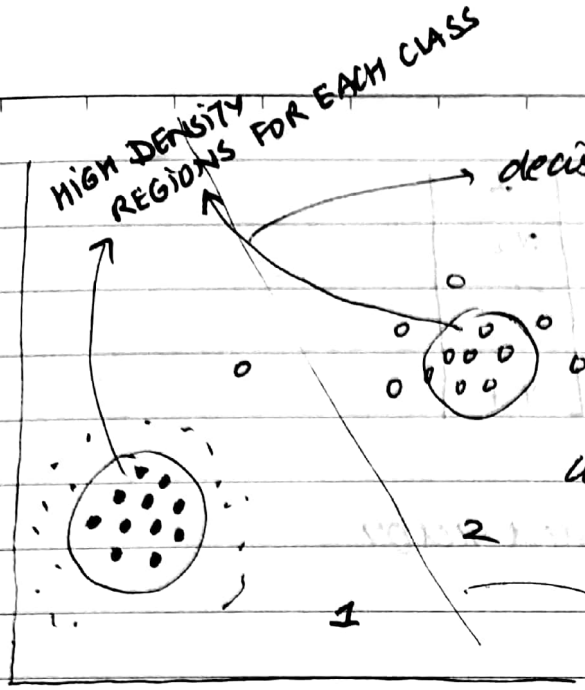
$g_i(x|\phi_i)$: discriminant function for i^{th} class

In discrim. function we assume the model & not density unlike in Bayes classifier where d prob-density funct^n is assumed

Discriminant functions are being assumed linear

Discriminant based classificat^n assumes a model for $g_i(x|\phi_i)$, no density estimation

Date :



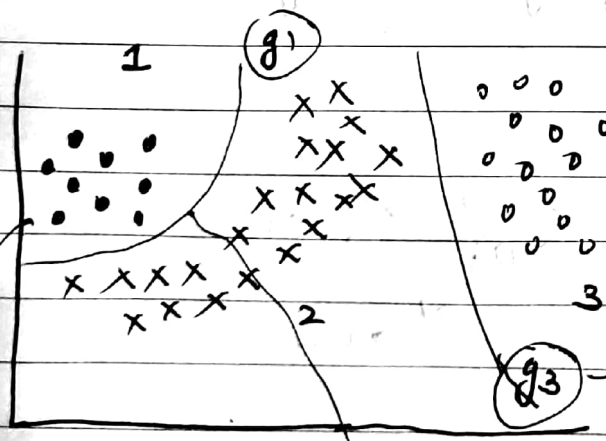
decision boundary, discriminant function learns this instead of the density regions of each class which is computationally expensive.

How to learn the boundary & s.t loss funct^n is minimized.?

what model of $g_1(x)$ & $g_2(x)$ minimize loss function.

$$g_i(x|w_i, w_{i0}) = w_i^T x + w_{i0} = \sum_{j=1}^{d} w_{ij} x_j + w_{i0} \quad (1)$$

'd-1' dimensional hyperplane



g_3 decides if 'x' belongs to g_3 ~~or not~~ class 3 or not

in this case it can have class 1 or class 2

if $g_1 > \gamma$ then class 1

non linear decision boundary

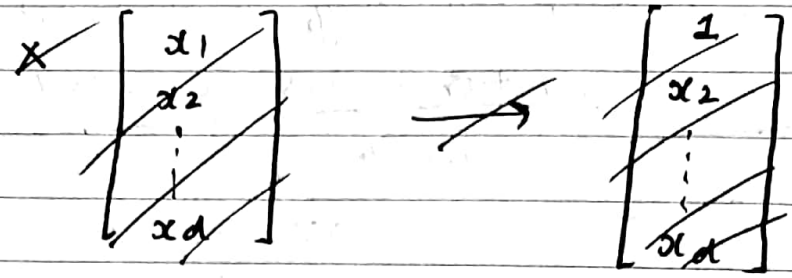
$g_3 > 0$ then class 3

'm'/slope : determines orientation of the line.

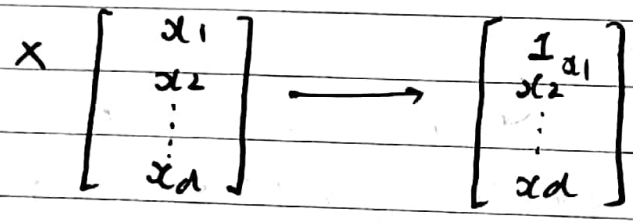
in (1), +ve weights contribute to class

-ve weights inhibit that class eg # hours jogging $\propto \frac{1}{\text{diabetes}}$

Date :



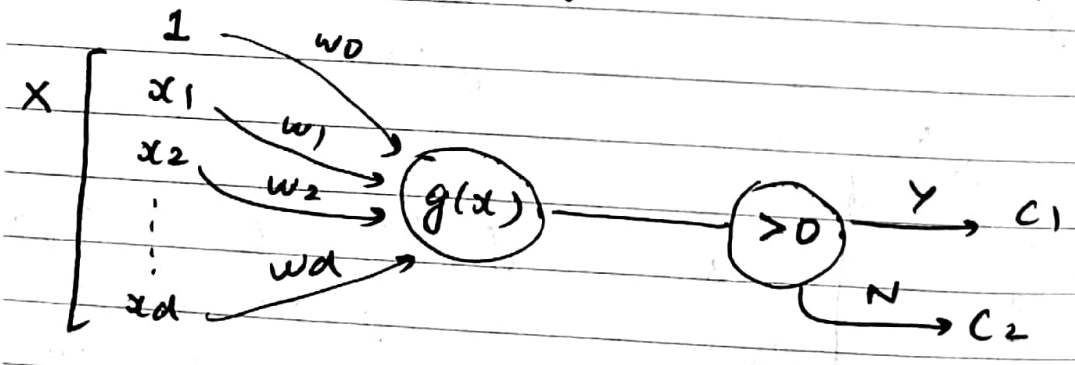
$g(x) = w^T x$ → augmented vector



before

$$w_0 + \sum_{j=1}^{d} w_j x_j$$

$$= \sum_{j=0}^{d} x_j \cdot w_j \quad \text{with } x_j = 1$$



$y = mx + c$

$\Rightarrow mx - y + c = 0$

substituting variables, x, y

$\Rightarrow w_0 + w_1 x + w_2 y = 0$

$w_0 = c$
$w_1 = m$
$w_2 = -1$

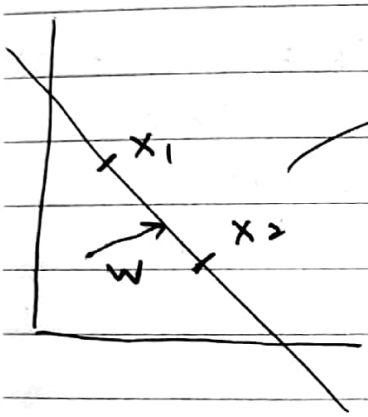
$g(x) = w_0 + w_1 x + w_2 y$

Date :

$g(x) = \underbrace{w^T x}$ → orientation of the plane

$= [\underbrace{w_0}\ w_1 \ \dots\ w_d] \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix}$

w_0 → tells location of plane



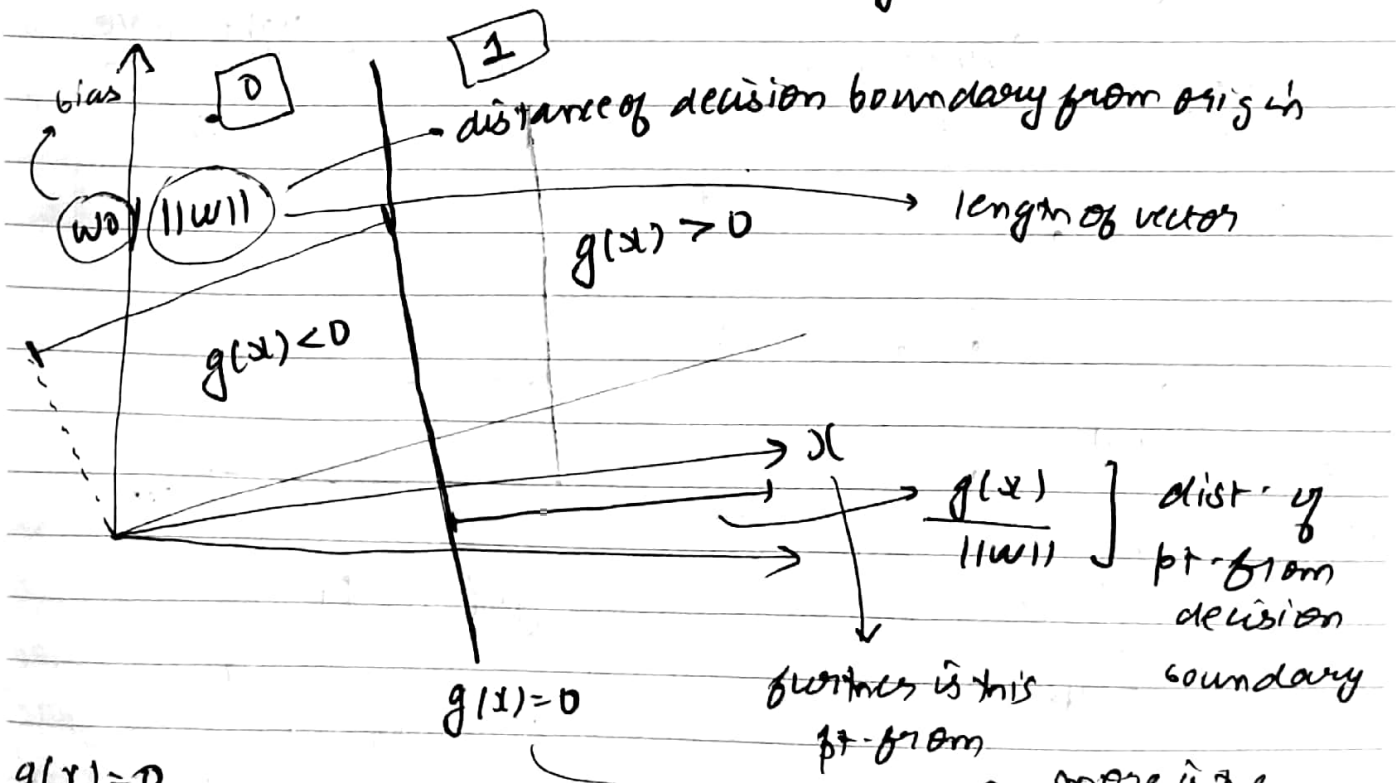
$\Rightarrow g(x_1) = g(x_2)$

$\Rightarrow w^T x_1 = w^T x_2$

$\Rightarrow w^T x_1 - w^T x_2 = 0$

$\Rightarrow \underbrace{w^T}(x_1 - x_2) = 0$ → vector

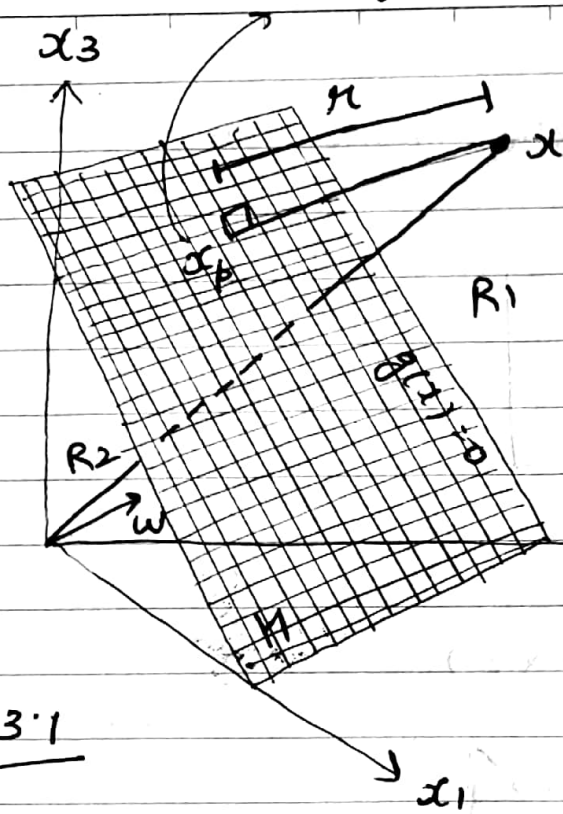
→ this vector is normal to any pt. that lies on plane



$g(x) = 0$ is a hypothesis

Date :

⊥ of x on plane, r is the distance



$g(x) = w^T x$

$$[w_0\ w_1\ w_2\ w_3] \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

10.3.1

Show that

$$x = x_p + \frac{r \cdot w}{\|w\|}$$

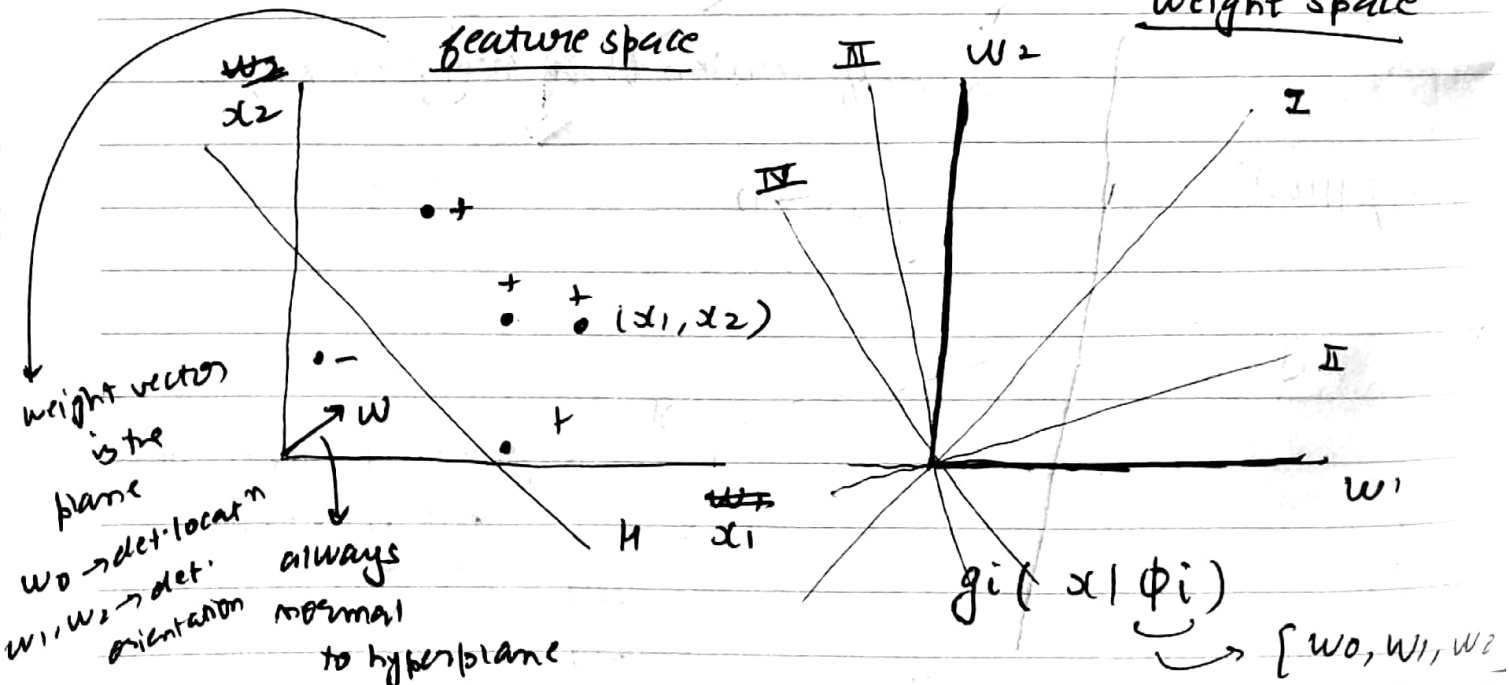
We know $r = \frac{g(x)}{\|w\|}$

$$w^T x = \begin{cases} > 0 \rightarrow R_1 \\ < 0 \rightarrow R_2 \\ = 0 \rightarrow \text{Hypothesis } (H) \end{cases}$$

(parametric space)

weight space

feature space



weight vector is the plane

w_0 → dist. location

w_1, w_2 → dist. orientation

always normal to hyperplane

$g_i(x | \phi_i)$ → $[w_0, w_1, w_2$

assuming separating hyperplane passes through origin

$\therefore w_0 = 0$

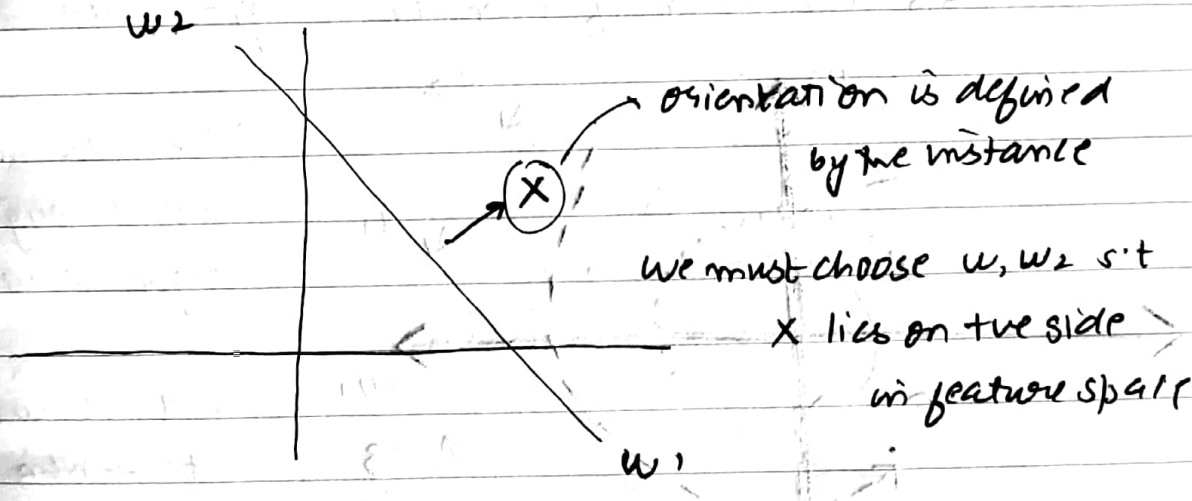
Date :

I, II, III, IV are the possible separating hyperplanes that predict '+' in R1 & '−' in R2

we must choose the hyperplane that minimizes loss function

Hyperplane in feature space must be complementary of some hyperplane in weight space.

Multiple separating hyperplanes exist in 360° ~~space~~ weight space



Weight space is comp. to feature space

$W^T X = X^T W$

$W^T X$ → orientation in weight feature space

W vector is normal to plane

$X^T W$ → feature orientation in ~~weight~~ space

X are normal to plane a

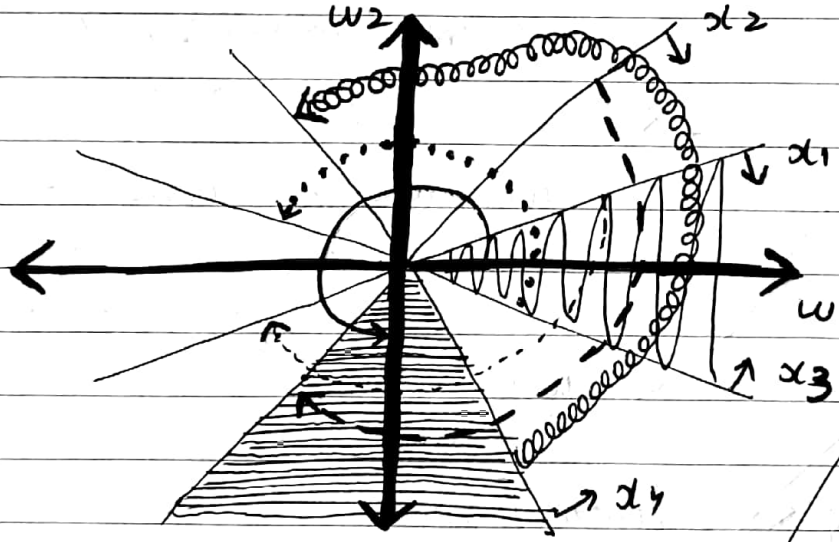
$[x_1, x_2, \ldots, x_d]$

Date :

	X_1	X_2	C	
$x_1 \rightarrow$	x_{11}	x_{12}	1	+
$x_2 \rightarrow$	x_{21}	x_{22}	1	+
$x_3 \rightarrow$	x_{31}	x_{32}	0	−
$x_4 \rightarrow$	x_{41}	x_{42}	0	−

$x_1, x_2, .. x_4$ are the 4 records

data pts: det: orien: of plane in weight space



All planes pass through origin $\therefore w_0 = 0$

Find w_1, w_2 s.t. ~~when~~ x_1, x_2 evaluate to + & x_3, x_4 evaluate to '−' in feature space.

for x_1, if we choose w_1, w_2 from

(I) region covered by '---→'

then x_1 evaluates to true

↓ correspond

w_1, w_2 to a plane in feature space s.t

(II) for x_2, if we choose w_1, w_2 from region covered by '— — →', then x_2 evaluates to true

for x_1, x_2 both to evaluate true, we must take intersection of regions in (I) & (II) to select w_1, w_2

this region is a semi-∞ region

Date :

(III)

for x_3, if we choose w, w_2 from region covered by '.....>', x_3 evaluates to +ve

but x_3 belongs to -ve class

for x_1, x_2, x_3 to evaluate to +ve we take intersection for regions (I), (II), (III)

(IV) same for x_4

(I) $\cap$ (II) $\cap$ (III) $\cap$ (IV) leads to +ve prediction for all instances

But goal is $\longrightarrow$ det. +ve for x_1, x_2

det. -ve for x_3, x_4

$\therefore$ region denoted by MMMM results to 2 false +ves

$\therefore$ solution space is region denoted by  , as it predicts x_1, x_2 as +ve & x_3, x_4 as -ve

Date :

GOAL : to determine weight vector that min. loss function
- multivariate regression
- trial error method by adjusting hyperplane

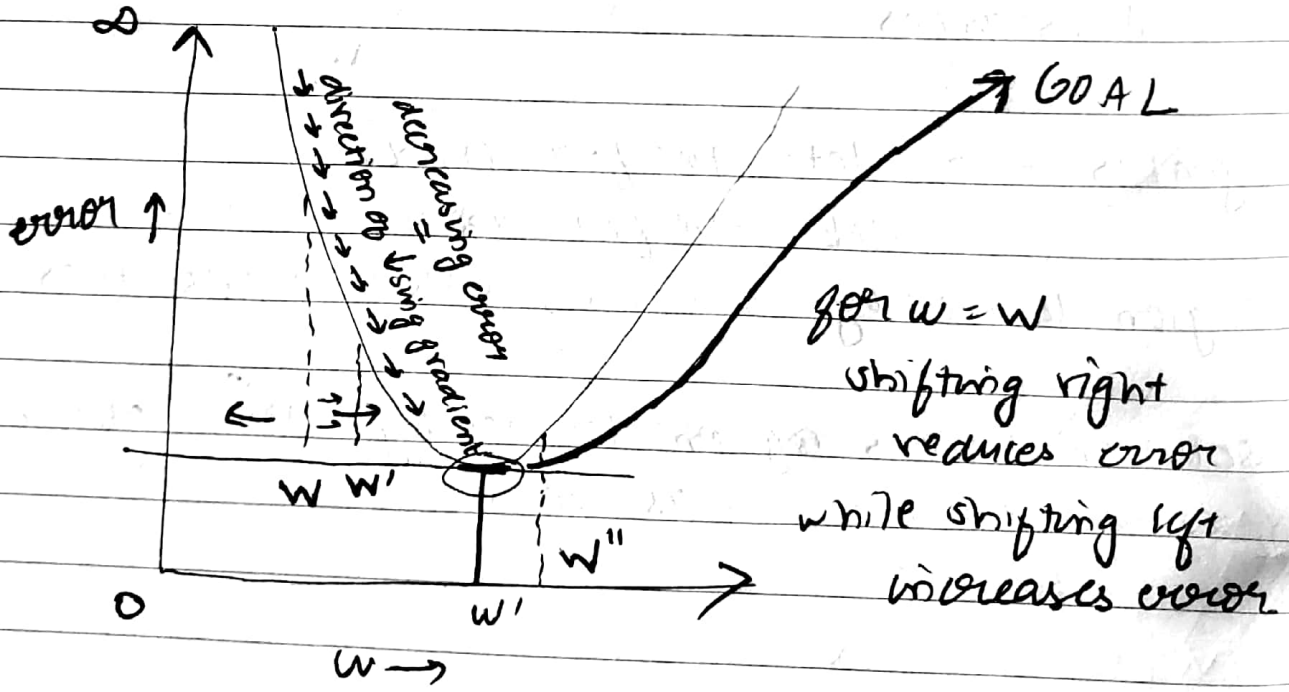
Squared error is the typical loss function

$$L(x) = [y - h(x)]^2$$

Training error =

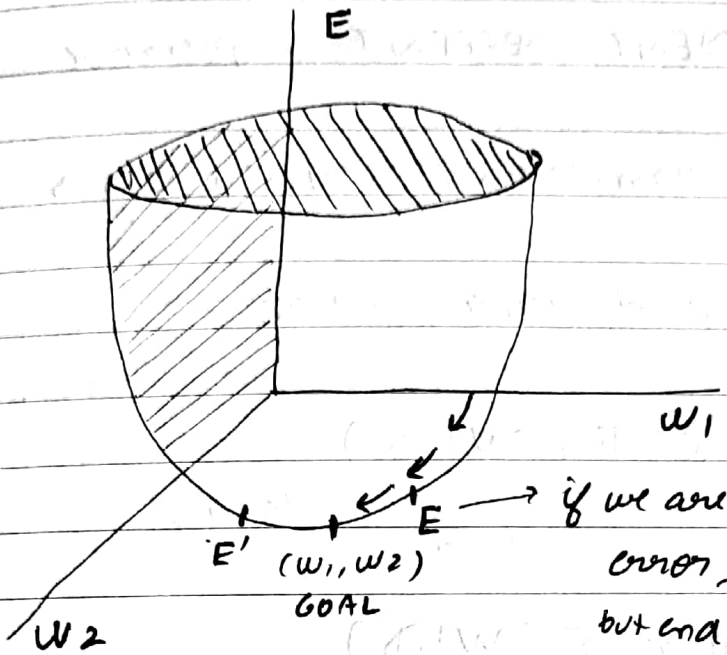
$$\sum L(xi)$$

convex function



suppose we take a large step & end up overshooting to right of goal i.e. w'', error increases

Date :



→ if we are here & to reduce error, we shift it to left but end up overshooting to E'

& if E' = E, then we keep on oscillating until we don't change the step size

∴ as error becomes smaller we reduce step size

Wide data : columns >> rows or some rows or some columns are dependent ∴ of which X^TX becomes non invertible.

Inverse time complexity - $O(n^3)$

Analytical method is good for small data else gradient descent method

Date :

(GD) GRADIENT DESCENT → Iterative

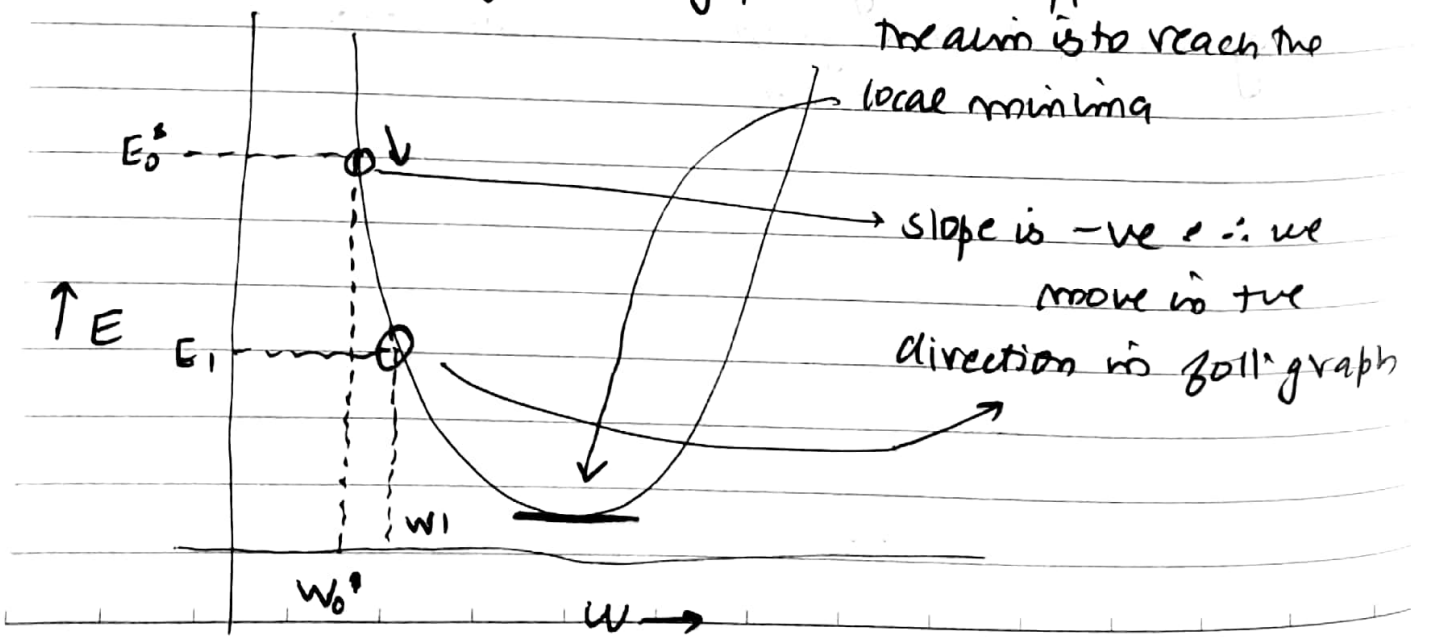
GOAL: Optimize parameter vector W s.t loss is minimized

LOSS FUNCTION: squared error function

We need to minimize $E(W|D)$ → error, data; $W = [w_0, \dots w_d]$ → dimensions of data

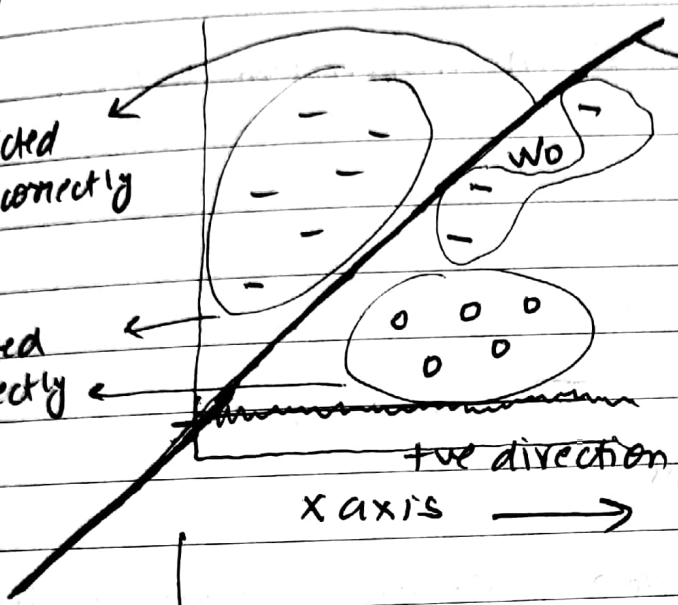
$$W^* = \arg\min_W E(W|D)$$ → D or X

- Gradient descent is good for smooth convex objective funct^n
- It reduces error at each step
- Need to determine step size (learning rate)
- GD is good for large 'n' w.r.t analytical method
- we start randomly at any pt. for w, suppose at w_0^0 the aim is to reach the local minima



Date :

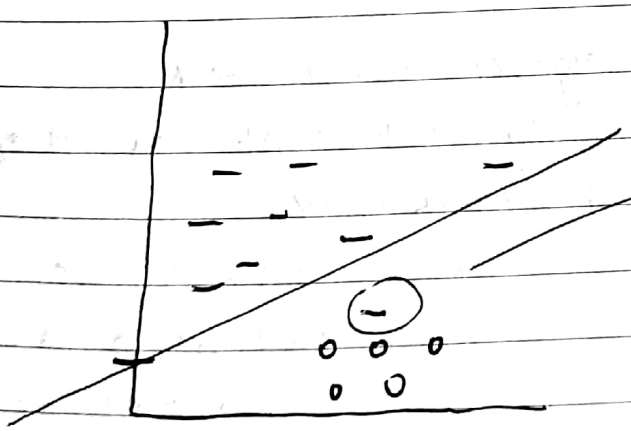
The graph for w_0 in data space is:



we expect the hyperplane to move to → direction

UPDATED TO

Learning Rate is very critical parameter in arriving at a good solution



Only 1 erroneous prediction remains

GRADIENT VECTOR $\nabla_w E = \left[\frac{\partial E}{\partial w_1}, \frac{\partial E}{\partial w_2}, \ldots \frac{\partial E}{\partial w_d} \right]^T$

$\Delta w_i = -\eta \frac{\partial E}{\partial w_i}$ → learning Rate, → opposite direction

↳ slope -ve ⇒ move in the +ve to ↓ error

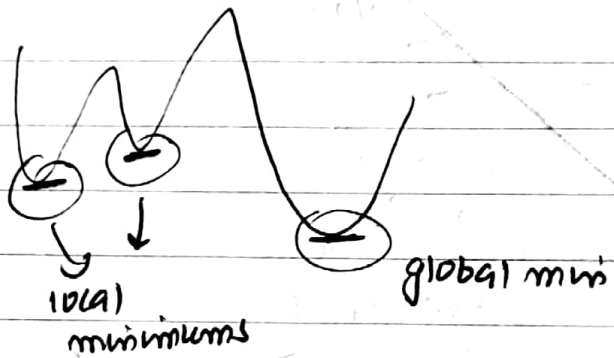
Updated weight ← $w_{i+1} = w_i + \Delta w_i$

W_0 remains same ∴ we assumed initial line passing thru w_0

Date :

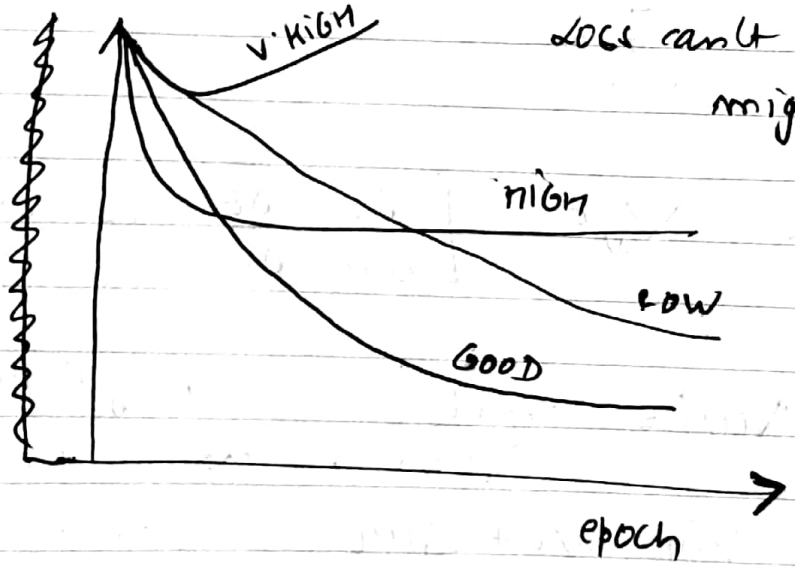
Random initialization akin to random separating hyperplane

No guarantee of finding global minimum for non-convex function



How to decide learning rate (LR)? : Critical since determines how fast or slow we will move towards optimal weights

very high ~~learn~~ LR: divergence is indicated
low LR: converging v-slowly, not feasible as lot of computational power is used
High LR: might become constant & doesn't reduce loss can't be 0 as data might not be linearly separable



Date :

Popular Heuristics for computing LR

1) Exact line search - bisection method

I) LB UB

find mid pt

II) LB MP UB

III) depending on how loss changes choose

LB MP

OR

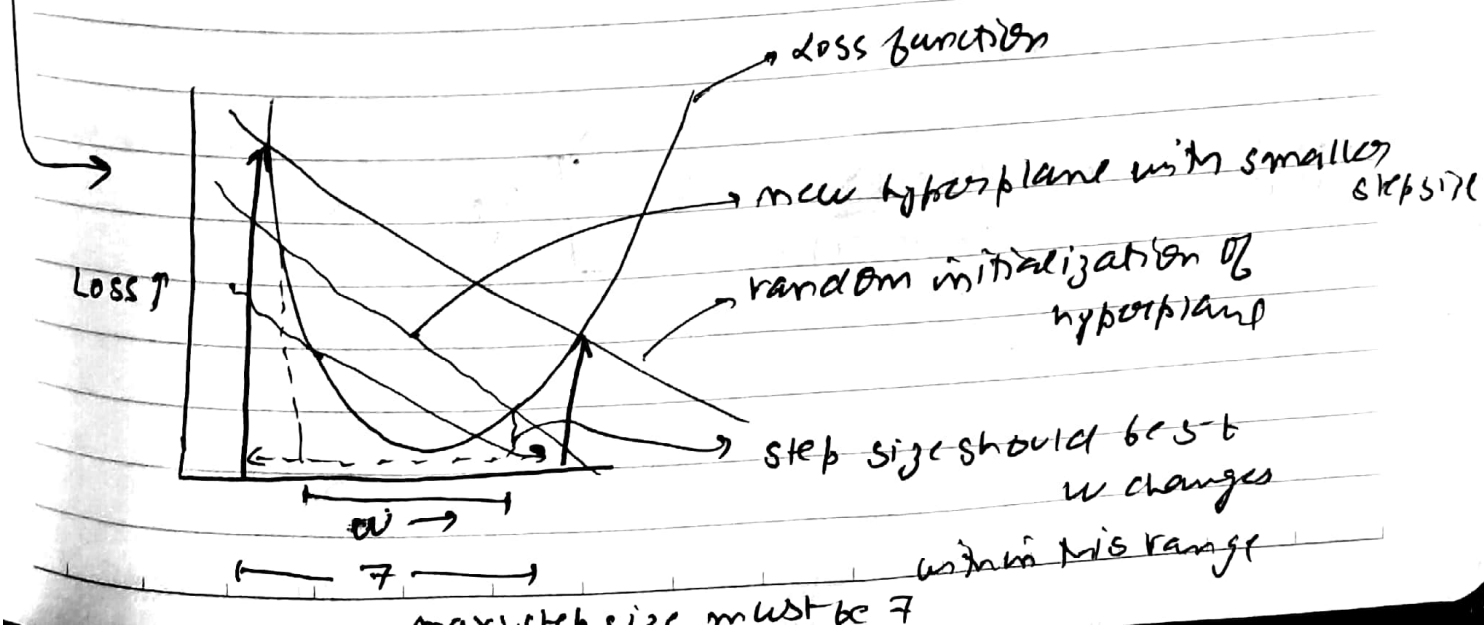
MP UB

2) Inexact line search

3) Decaying learning rate can be $\frac{1}{K}$, $\frac{1}{\sqrt{K}}$, etc. depends on data

function should be inverse

4) Fixed learning rate



Date :

assumptions: 1) class conditional densities are Gaussian
2) and share covariance matrix

Probabilistic classification as Discriminant Based classificat

$$y = h(\mathbf{x}) = \begin{cases} 1 & \text{if } P(C_1|X) > P(C_2|X) \text{ or} \\ & \bullet\ P(C_1|X) > .5 \text{ or} \\ & \bullet\ \frac{P(C_1|X)}{P(C_2|X)} > 1 \text{ or} \\ & \bullet\ \log\left(\frac{P(C_1|X)}{P(C_2|X)}\right) > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$\log\left[\frac{P(C_1|X)}{P(C_2|X)}\right] = \log\left[\frac{P(C_1|X)}{1-P(C_1|X)}\right]$$

logit transformation
↳ log odds of posterior prob of C_1

$$y = \frac{P(C_1)}{P(C_2)} \quad \text{odds}$$

$$= \frac{P(C_1|X)}{1-P(C_1|X)}$$

$$\log y = \log\left(\frac{P(C_1|X)}{1-P(C_1|X)}\right) \quad \text{log odds}$$

Date :

$$\log\left[\frac{P(C_1|X)}{P(C_2|X)}\right] = \log\left[\frac{P(X|C_1)\cdot P(C_1)}{P(X|C_2)\cdot P(C_2)}\right]$$

$$= \log\left[\frac{P(X|C_1)}{P(X|C_2)}\right] + \log\left[\frac{P(C_1)}{P(C_2)}\right]$$

to derive this $\Rightarrow$ $w^T x + w_0$ i.e. can we interpret bayesian classifier as discriminant func^n

$\rightarrow$ not augmented vectors

~~For bayesian classifier~~:

~~$w_i = \Sigma^{-1} \mu_i$~~

~~$w_{i0} = -\frac{1}{2} \mu_i \Sigma$~~

$C_1 \sim N(\mu_1, \Sigma)$

$C_2 \sim N(\mu_2, \Sigma)$

$$\left[\log\left(\frac{P(C_1|X)}{P(C_2|X)}\right)\right]$$

we need to find $P(X|C_1)$ & $P(X|C_2)$

$\rightarrow$ for d dimensional X

$$= (2\pi)^{-d/2} |\Sigma|^{-1/2} \exp\left[\frac{-1}{2}(X-\mu_1)^T \Sigma^{-1} (X-\mu_1)\right]$$

Date :

solving $\log\left(\frac{y}{1-y}\right) = \log\left(\frac{P(x|c_1)}{P(x|c_2)}\right) + \log\frac{P(c_1)}{P(c_2)}$

$= w^T x + w_0$ ← solving this

For 1 dimensional

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

$$\frac{P(x|c_1)}{P(x|c_2)} = \frac{(2\pi)^{-d/2} |\Sigma|^{-1/2} \exp\left(-\frac{1}{2}(x-\mu_1)^T \Sigma^{-1} (x-\mu_1)\right)}{(2\pi)^{-d/2} |\Sigma|^{-1/2} \exp\left(-\frac{1}{2}(x-\mu_2)^T \Sigma^{-1} (x-\mu_2)\right)}$$

$$\log\left[\frac{P(x|c_1)}{P(x|c_2)}\right] = \log P(x|c_1) - \log(x|c_2)$$

$\mu_1 = \mu_{i2}$ Mean of i^{th} feature of class 1

↳ a vector to store mean of all attributes

$$\left(\frac{x-\mu}{\sigma}\right)^2 = (x-\mu_1)^T \Sigma^{-1} (x-\mu_1)$$

$$\downarrow$$

$$\langle (x_1-\mu_{11}), (x_2-\mu_{21}), \ldots (x_d-\mu_{d1}) \rangle$$

$$\left[(x_1-\mu_{11}), (x_2-\mu_{21}), \ldots (x_d-\mu_{d1})\right]_{1\times d} \begin{bmatrix} \sigma_1^2 & 0 & 0 & \cdots & 0 \\ \vdots & \ddots & & & \vdots \\ 0 & \cdots & & & \sigma_d^2 \end{bmatrix}_{d\times d} \begin{bmatrix} x_1-\mu_{11} \\ \vdots \\ x_d-\mu_{d1} \end{bmatrix}$$

Date :

$$= \log\left(\frac{\exp\left(-1/2 (x-\mu_1)^T \Sigma^{-1} (x-\mu_1)\right)}{\exp\left(-1/2 (x-\mu_2)^T \Sigma^{-1} (x-\mu_2)\right)}\right)$$

$$= \log\left(\exp\left(-\frac{1}{2}(x-\mu_1)^T \Sigma^{-1} (x-\mu_1)\right)\right) - \log\left(\exp\left(-\frac{1}{2}(x-\mu_2)^T \Sigma^{-1} (x-\mu_2)\right)\right)$$

$$= \underbrace{-\frac{1}{2}(x-\mu_1)^T \Sigma^{-1} (x-\mu_1)}_{c_1} + \underbrace{\frac{1}{2}(x-\mu_2)^T \Sigma^{-1} (x-\mu_2)}_{c_2}$$

$$-\frac{1}{2}\left(\frac{(x_1-\mu_{11})^2}{\sigma_1^2} + \frac{(x_2-\mu_{21})^2}{\sigma_2^2} + \ldots \frac{(x_d-\mu_{d1})^2}{\sigma_d^2}\right)$$

$$+\frac{1}{2}\left[\left(\frac{x_1-\mu_{12}}{\sigma_1}\right)^2 + \ldots + \left(\frac{x_d-\mu_{d2}}{\sigma_d^2}\right)^2\right]$$

→ open all brackets to get

$$-\frac{1}{2}\left[\frac{(\mu_{11}^2-\mu_{12}^2)}{\sigma_1^2} + \ldots + \frac{(\mu_{d1}^2-\mu_{d2}^2)}{\sigma_d^2}\right]$$

$$+\left[\frac{(\mu_{11}-\mu_{12})x_1}{\sigma_1^2} + \ldots + \frac{(\mu_{d1}-\mu_{d2})x_d}{\sigma_d^2}\right]$$