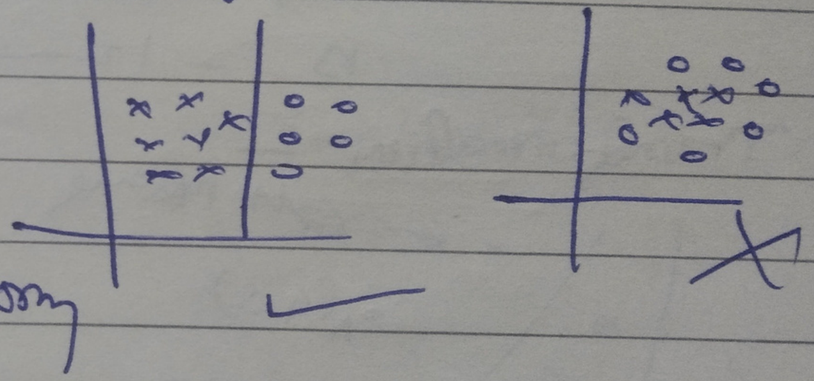
Logistic Regression

Logistic Regression :- data should be linearly separable

(or) almost linearly separable through line

first approach

(a) Perceptron trick :-



here we take

line $Ax+By+C=0$

in log.reg not $y=mx+b$ form

$Ax_1+Bx_2+\dots C=0$

We have to find A, B, C ?

So How we can find ~~with~~ with perceptron trick.

first Assume any value of A, B, C

say $A=1, B=1, C=0$

Step Assume any value of A, B, C

Say A = 1, B = 1, C = 0

for ← loop say 1000 times

data के अंदर किसी (randomly select) point को select करेंगे और check करेंगे क्या वो value (point) सभी जगह पर है यदि नहीं है line के उसके acc change करेंगे फिर next time loop →

कुछ तक loop चलेंगे

1. 1000, 10000 ...
2. while convergence (जब तक Misclassified point है)

How to identify +ve region (-ve region ?

→ point put करके देख लो और +ve हो तो +ve region
-ve हो तो -ve region में

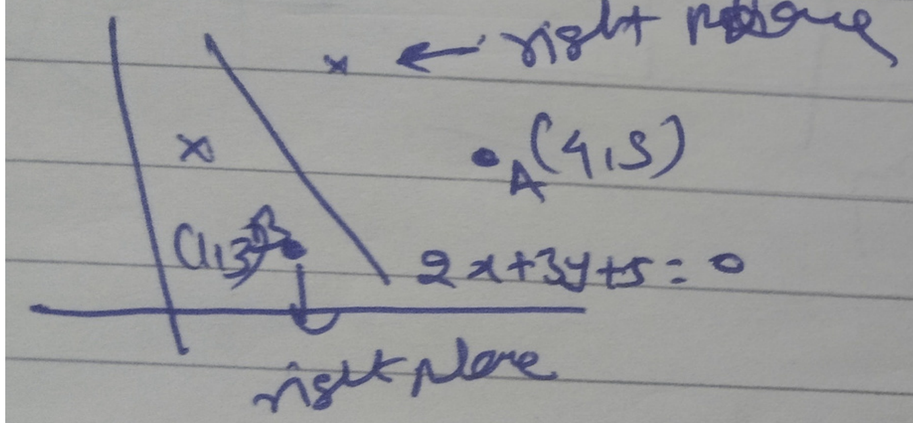
→ How to move line ? By change A, B, C.

C को ↑↓ करने से Parallel changes.

A से ~~about~~ axis से about

B — 〃 —

Transformation: —



acc to A (4, 5, 1) ← Add 1

line	2	3	5
Point	4	5	1
	−2	−2	4

$-2x-2y+4=0$

new line

↑ subtract

acc to B (1, 3, −1) ↑ add

each D(1, 3, -1) ← add

line	2	3	5
Port	1	3	1
	3	6	6

← add

$3x + 6y + 6 = 0$

def step(z)
return 1 if z > 0
else 0

इसमें कुछ बदला transform हो रहा

take a small → learning rate

$\alpha = 0.01$ say

2	3	5
0.01	0.03	0.01

→ so gradual change

↓ Now change

Algo. for perceptron trick

let data

	CGPA	IQ	placed
1	7.5	81	1
1	8.9	109	1
1	7.0	81	0
x_0	x_1	x_2	

add

$$w_0 + w_1 x_1 + w_2 x_2 = 0$$

$$w_0 x_0 + w_1 x_1 + w_2 x_2 = 0$$

$$\sum_{i=1}^{2} w_i x_i = 0$$

$+ \longrightarrow 1$

$- \longrightarrow 0$

epoch → 1000 $\eta = 0.01$

for i range(epoch)

randomly select a student.

if $y_i \in N$ & $\sum w_i x_i \geq 0$

placement नहीं हुआ

← गलत जगह लगाया है

update $w_{new} = w_{old} - \eta(x_0 \; x_1 \; x_2 \ldots$

main

if $y_i \in P$ & $\sum w_i x_i < 0$

$w_{new} = w_{old} + \eta(x_0 \ldots)$

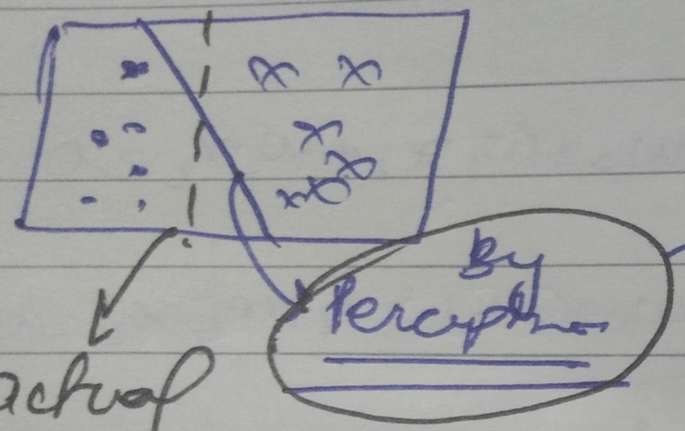
```
X = np.insert(X, 0, 1, axis=1)
weights = np.ones(X.shape[1])
lr = 0.1
for i in range(1000):
    j = np.random.randint(1, 100)
    y_hat = step(np.dot(X[j], weights))
    weights = weights + lr*(y[j] - y_hat)*X[j]
```

direct write

$$\boxed{w_n = w_{old} + \eta(y_i - \hat{y}_i)X_i}$$

very

Prob. in perceptron : —



→ इन को separate करने के लिए कोई रूल नहीं

But logistic wale symmetrically cut करते हैं (better model)

actual (from sklearn lib) logistic reg.

So we use logistic reg model instead of perceptron

→ test error भी बढ़ता है तो रखता है।

How How to change ?

By perceptron

Possible solⁿ $\boxed{w_n = w_0 + (y_i - \hat{y}_i)\, x_i}$

Possible solⁿ $\boxed{w_n = w_0 + (y_i - \hat{y}_i)\, x_i}$

Now misclassify point का भी classify नहीं होगा
classify का भी नहीं होगा

misclassify point ← line के ही तरफ पड़ेगा
classify ⟶ line से दूर से पड़ेगा

So centre में जाकर line खड़ा करेगा

यदि point, line के पास है ज्यादा तेज
push/pull करेगा।

By perceptron $w_{new} = w_{old} + \eta(y_i - \hat{y}_i)\, x_i$

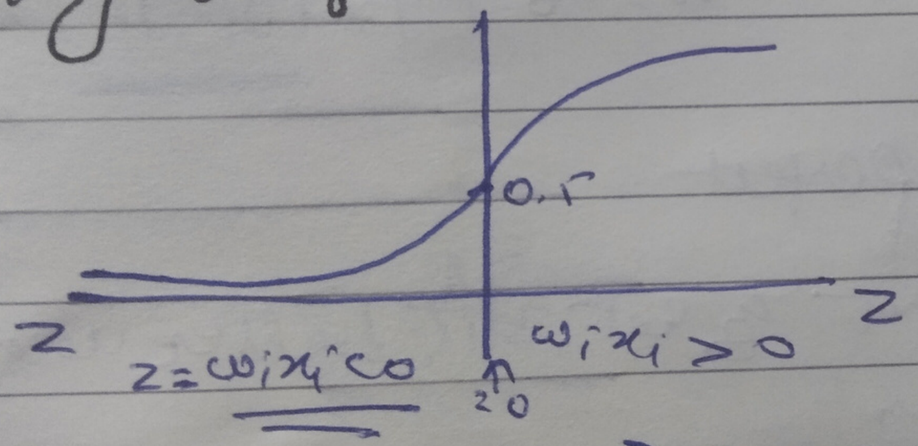
1 या 0 और 2 का या

इसलिए यहाँ पर already classify point
मुझे नहीं चाहिए रहे थे।

So we want ही वो diff value है।

so we use $\boxed{\text{Sigmoid fn}}$ *

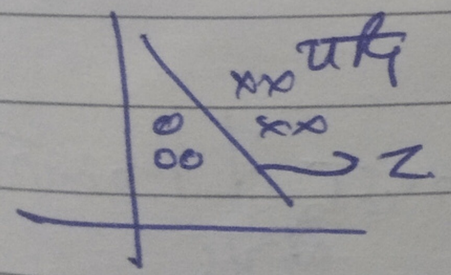
Sigmoid fn



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

$$-\infty < z < \infty$$

$$\boxed{0 < y < 1}$$



यदि point line के ऊपर → $\sum w_i x_i = 0, \sigma = 0.5$

$\sum w_i x_i > 0 \rightarrow, \sigma > 0.5$

$< 0 \rightarrow, \sigma < 0.5$

say prob

यदि line पर point हो

say prob

मतलब line पर point की
placement होने की Probability 0.5

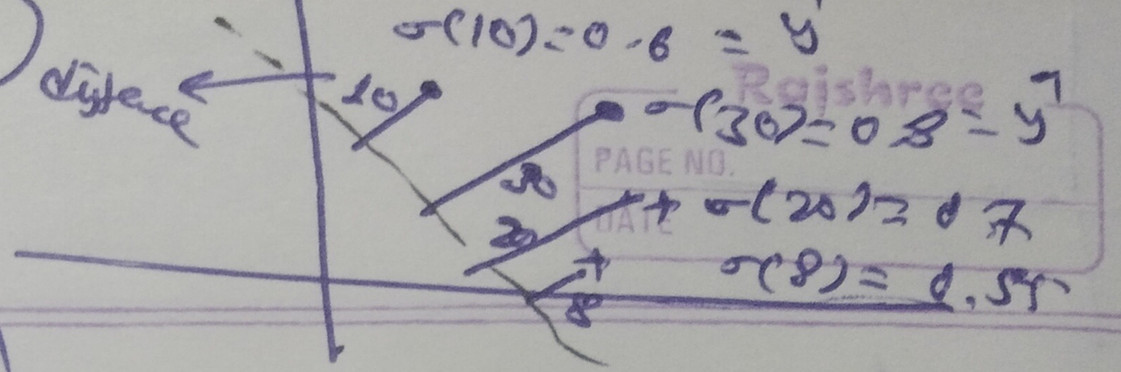
~~(+ve region) से जितना दूर उतना~~
~~उतनी Prob~~

जितना ज्यादा line से दूर +ve region में हो उतना ज्यादा
chance (more prob)

जितना ज्यादा line से दूर -ve region में उतना कम chance

Sigmoid ⟶ लेने का मतलब उसका Prob. में change
कर दिया
(chance कि कितना chance है)

eg



y_i	$\hat{y}_i$ let	$y_i - \hat{y}_i$
1	0.6	0.4
1	0.8	0.2
0	0.65	−0.65
1	0.30	0.70

→ पहला वाला point

→ तो ज्यादा push करेगा

$$w = w_0 + \eta \times \underline{(y_i - \hat{y}_i)} \cdot x_i$$

But still NOT perfect

log Reg stream out करेगा और better उसका रहेगा।

व्यापारी करते हैं so find using loss fn

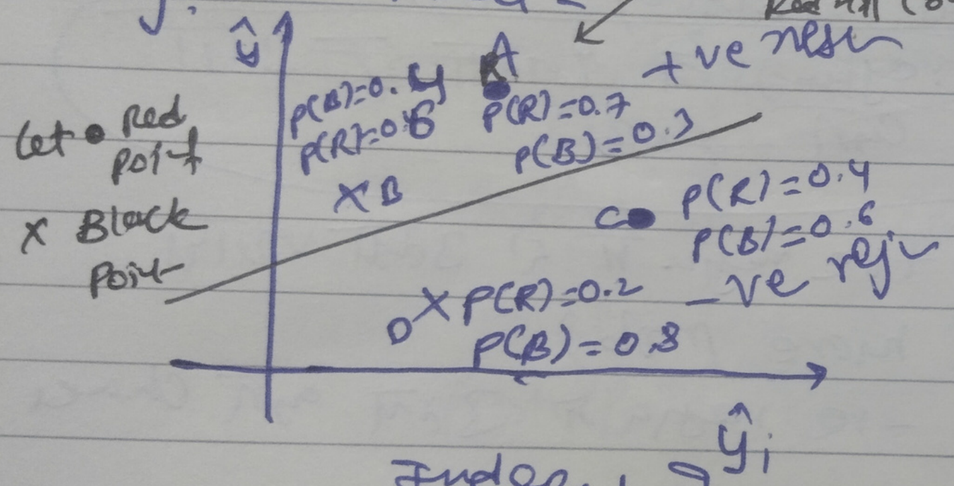
Loss f^n (Maximum Likelihood)

→ data के लिए Model Prediction आसानी से Probability के आधार

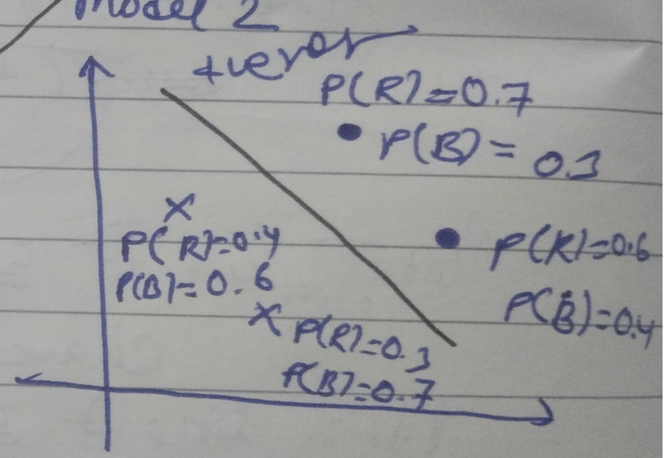
to derive the concept of loss f^n we will use the concept of maximum likelihood

eg.

Model 1



Model 2



let • Red point
× Black point

2 Point Red के 3 सही Prob = 0.7
Red गलत (Blue है) = 0.3

Independent है तो direct Mult

model 1 = ~~0.6 × 0.7 ×~~ .

= (0.4) (0.7) (0.4) (0.8)

Maximum likelihood →

= ~~0.176~~ 0.089

3 सही color की Prob मिल color की है

model 2 = 0.7 × 0.6 × 0.6 × 0.7

= ~~0.089~~ 0.176

Maximum Likelihood Estimation is a method that determines values for the parameters of a model. The parameter values are found such that they maximise the likelihood that the process described by the model produced the data that were actually observed.

So model 2 is better

This Method is best but Multiply करने पर Number छोटा or गायेगा तो Computer करने में problem होगा। (यदि बहुत points हो तो say Lossss

So Convert into log

$$\log \max x = \log(0.7) + \log(0.4) + \ln(0.4) + \ln(0.8)$$

$\log \max = \log(0.7) + \log(0.4) + \ln(0.4) + \ln(0.8)$

but all give -ve

so change in negative

$\log(\max) = -\log 0.7 - \log 0.4 - \log 0.4 - \log 0.8$

$\boxed{\text{cross entropy}}$

The summation of log of maximum likelihood is cross entropy

इसलिए Minimus करना है मतलब (cross entropy को कम करना है)

Model best ← Matlab cross entropy is Minimum

So. जिसका कि कुछ Point के अलग-अलग Prob लिए अलग color wise

तो

↓ कुछ का $\log \hat{y}_i$, कुछ का $\log(1-y_i)$

So we use for very point

$$-y_i \log \hat{y}_i - (1-y_i)\log(1-\hat{y}_i)$$

for for Red point (A Point)

$$-1 \log 0.7 - (1-0) \log(1-0.7)$$

$$= -\log 0.7 - 0$$

So we use

automatic zero

$$L = \sum^{n}$$

So we use

$$L = \sum_{i=1}^{n} -y_i \log(\hat{y}_i) - (1-y_i)\log(1-\hat{y}_i)$$

$$L = \frac{-1}{n} \sum_{i=1}^{n} \left[(y_i \log \hat{y}_i) + (1-y_i)\log(1-\hat{y}_i) \right]$$

for average error

logloss error

(or) Binary cross entropy

for Minimum

use gradient descent

$\sigma(x) = \frac{1}{1+e^{-x}}$ $\sigma'(x) = \sigma(x) \cdot [1 - \sigma(x)]$

Rajshree
PAGE NO.
DATE

Using gradient Descent :-

$$\hat{y}_i = \sigma(w_1 x_{i1} + w_2 x_{i2} \ldots w_n x_{in} + w_0)$$

$$\hat{y} = \begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_m \end{bmatrix} = \sigma\left(\begin{bmatrix} 1 & x_{11} & x_{12} \ldots & x_{1n} \\ & & & \\ 1 & x_{m1} & x_{m2} \ldots & x_{mn} \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_n \end{bmatrix} \right)$$

$$\boxed{\hat{y} = \sigma(xw)}$$

$$L = -\frac{1}{m} \left[\sum_{i=1}^{m} y_i \log \hat{y}_i + \sum_{i=1}^{m} (1-y_i) \log (1-\hat{y}_i) \right]$$

$$\sum_{i=1}^{n} y_i \log(\hat{y}_i) = y_1 \log \hat{y}_1 + y_2 \log \hat{y}_2 \cdots\cdots + y_m \log \hat{y}_m$$

$$= \begin{bmatrix} y_1 & y_2 & \cdots & y_m \end{bmatrix} \begin{bmatrix} \log \hat{y}_1 \\ \log \hat{y}_2 \\ \vdots \\ \log \hat{y}_m \end{bmatrix}$$

$$= \begin{bmatrix} y_1 & \cdots & y_m \end{bmatrix} \log\left(\begin{bmatrix} \hat{y}_1 \\ \vdots \\ \hat{y}_m \end{bmatrix} \right)$$

$$= y \log \hat{y}$$

$$\sum y_i \log \hat{y}_i = y \log(\sigma(xw))$$

Bernoulli $x_i = 1$ ✓ yes. $x_i = 0$ ✗ Not

$$\log L = \sum x_i \log p + \sum (1 - x_i) \log(1-p)$$

Compare

$$L = -\frac{1}{m}\left[y \log \hat{y} + (1-y) \log (1-\hat{y}) \right]$$

where $\hat{y} = \sigma(xw)$

Now Use grad. descent → समस्या solve करने की कोई other method नहीं है इसलिए हम solve using gradient descent

L_{min} ?

for i in epochs:

$$w = w - \eta \frac{\Delta L}{\Delta w}$$

$$\frac{\Delta L}{\Delta w} = \left[\frac{\partial L}{\partial w_0}, \frac{\partial L}{\partial w_1}, \dots \frac{\partial L}{\partial w_n} \right]$$

$n+1$

$$L = -\frac{1}{m}\left[y \log \hat{y} + (1-y) \log (1-\hat{y}) \right]$$

$$\frac{dL}{dw} = -\frac{1}{m}\left[y \frac{d}{dw} \log \hat{y} + (1-y) \frac{d}{dw} \log(1-\hat{y}) \right]$$

$$= \cancel{-\frac{1}{m}\left[y \right] (1-\hat{y}}$$

$$= -\frac{1}{m}\left[y(1-\hat{y})x - \hat{y}(1-y)x \right]$$

$$\frac{\Delta L}{\Delta w} = -\frac{1}{m}\left[y - \hat{y} \right] x$$

$$\boxed{\frac{\Delta L}{\Delta w} = -\frac{1}{m}(y-\hat{y})x}$$

$\hat{y} = \sigma(wx)$

$\frac{d\,\sigma(wx)}{dw}$

$= \sigma(wx)(1-\sigma(wx)) \frac{d}{dw}(wx)$

So By grad. descent

$$\boxed{w_{new} = w_{old} + \eta \frac{1}{m}(y-\hat{y})x}$$

$$w = \begin{bmatrix} w_0 \\ \vdots \\ w_n \end{bmatrix}$$

Assumptions, Advantage- Disadvantages

Log. Reg.

- Basic Assump. Linear Regression b/w indep features & the log odds
- Advantages. very easy to understand, good accuracy for many simple data sets and it performs well when the dataset is linearly separable
  - It makes no assumptions about distributions about distributions of classes in feature space
  - Log. reg is less inclined to overfitting but it can overfit in high dimensional datasets. One may consider L1 & L2 technique to avoid.
  - easier to implement, interpret, and very efficient to train

- Disadv. Sometimes lot of feat. engin, If the indep feat are correlated
  - quite prone to noise & overfitting, If the no of observation is lesser than no of features, Logistic Regression should not be used, otherwise, it may lead to overfitting.
  - Non-linear problem cannot be solved because it has a linear decision surface. Linearly separable data is rarely found in real-world scenarios.
  - It is tough to obtain complex relationship using LR. More powerful & compact alg such as Neural Networks can easily outperform this algo. • In LinearReg indep & dep ~~factors~~ variable are related linearly. But logistic reg. needs that indep variable are linearly related to log odds $\log P/1-P$
- Missing values Sensitive to missing values
- Impact of outliers: Like lin. reg, estimates of the logistic reg are sensitive to the unusual observation. outliers, high leverage, & influential observations. Numerical examples & analysis are presented to demonstrate the most recent outlier diagnostic methods using

Classification Metrics

Classification ~~Matrix~~ metrics :- करने के लिये ML Model कितना अच्छा perform कर रहा है।

↳ Same as regression metrics

(I) Accuracy → वो metric जो बतायेगा कितना अच्छा Model perform कर रहा है।

How much accuracy is good?

→ Not fix any.

It depends on prob. that we are solving.

(data के ऊपर depend करेगा)

eg (1) Let 100 data का heart disease check करने वाला एक hospital ने 99 का सही बता दिया 1 का गलत होगा 99% accuracy

→ But

फिर भी Bad, we will not accept ।
क्योंकि हम नहीं चाहते है कि 100 में से 1 ~~की जान (Probably) चली जाये~~ से 1 बार गलती कर रहे है। तो 1 की जान ~~चली जा सकती है~~, ~~So But ऐसा नहीं चाहेंगे~~ जा सकती है so not accept 99% acc.

eg (2) Let swiggy के लिये Model Build किया कि किसी particular customer का monthly behavior देख कर पता कर रहे कि ये Person particular restaurant से खाना order करेगा या नहीं करेगा।

Let here 80% is accuracy then Here we can accept the model

The Prob. with Accuracy?

Accuracy score 80% बताता है लेकिन ये नहीं बताता है कि galti का type क्या है।

Let take example of placement.

actual	predict
Y	N
N	Y

> Both mistake But Both diff

अतः accuracy score पर नहीं बताता कि कौन सी type की mistake है।

So we use Confusion Matrix

Type & Nature

So we use Confusion Metrics

Confusion Metric: — → [illegible] का Type & Nature [illegible]

		Prediction 1	Prediction 0
Actual	1	True Positive (TP)	False Neg (FN) Type-II error
	0	False Positive (FP) Type-I error	True Neg (TN)

Let

1 ← Heart disease [illegible]

0 ← [illegible]

1 ← Heart dis predict [illegible]

0 ← [illegible] [illegible]

Predict

T/F → actual

P/N → Prediction

eg

	1	0
1	26	6
0	0	29

26 → लोगों को Model ने बोला कि Heart disease है और actual में उनके पास।

So Conf Mat → accuracy score निकाल सकते हैं but सही नहीं।

$$\text{acc score} = \frac{TP + TN}{TP + TN + FP + FN}$$

Type-I error FP Type-II error FN

Confusion Matrix for Multi-classification Problem

Prediction

actual	0	1	2
0			
1			
2			

$$\frac{\text{diag element}}{\text{total}} = \text{accura}$$

Q When accuracy is misleading. ⇒

→ In Imbalanced dataset

eg.

एक model बना रहे airport security के लिये 1 000000 data है उसमे से बताना है कि कितने terrorist है test train में 99999 लोग terrorist नहीं है 1 है।

तो accuracy 99.999% → 100%

⊙ तो Misleading

यदा data balance नहीं था कुछ बहुत ही ज्यादा और कुछ बहुत ही कम।

(III)

Precision & Metrics :-

Type-I mistake

↑ Solve this type of prob by Precision & Recall Metric

Example

(A)

M1

	Sent to spam	Not sent to spam
Spam	100	170 FN
Not spam	30 FP	700

(B)

M2

	Sent to spam	Not sent to spam
Spam	100	190
Not spam	10	70

Which model is better ?

acc same

दोनों बराबर है लेकिन
M2 में पर कोई
व्यक्ति स्पाम
गलती नहीं करेगा

Precision:- (Positive Predictive) ⇒ The proportion of positive cases that were correctly identified.

Recall:- (Sensitivity) The proportion of actual +ve cases which are correctly identified.

	1st Model	2nd Model
In first	FN = 170	FN = 190
	FP = 30	FP = 10

Let <u>example</u> किसी company ने mail किया कि इस mail का reply करो कि you want placement और Not.
और suppose mail spam हो गया तब क्या ? ↘ case

↓

बहुत बड़ी गलती है (FP)

So for checking which model is best check FP

FP should be less

So Model B is selected

that is Precision: → with Precision then best

Precision: What proportion of predicted +ve is truly +ve.

$$\boxed{\text{Precision} = \frac{TP}{TP+FP}} = \frac{\text{Positive actual all}}{\text{Positive student Positive}}$$

Model A $\frac{100}{130}$

Model B $\Rightarrow \frac{100}{110}$.

Pre $P_A < P_B$ so B is selected

(Ir) Recall

	(A) Detected Cancer	(A) Not Detected	(B) Detected Cancer	(B) Not Detected
Has Cancer	1000	2000 (FN)	1000	500
No Cancer	800 (FP)	8000	500	8000

accuracy

FP → Detected पर्सन But असली में actually

FN → असली पर्सन But नहीं

FN is More ~~dang~~ dangerous here

so (A) is select. / replay

that's Recall

Recall :- What proportion of actual +ve is correctly classified ?

Type-II मतलब

$$\text{Recall} = \frac{TP}{TP+FN} \Rightarrow \frac{\text{कितना Detect किया}}{\text{पूरा कितने को cover}}$$

ज्यादा Recall
than better

$$R_A = \frac{1000}{1200} \qquad R_B = \frac{1000}{1500}$$

$R_A > R_B$

so A is selected

F1 score: - Some times model detect नहीं कर पाता

क्या Type I / Type II Mistake है

में से कौनसी More Probabalmetic (संभावना)

है so we use F1 score

eg.: photo देखकर बताना है कि ये Cat / Dog है।

तो यह पर हमें ज्यादा नहीं कह सकते कि

कौनसा High precision / High Recall model है।

so दोनों को देखना पड़ेगा.

F1 score ⇒ Combination of Precision & Recall.

(average of both) → H.M.

$$\boxed{F1 = \frac{2 \cdot P R}{P + R}}$$

HM हमेशा छोटे के साथ रहता

Why NOT AM?

eg. $P = R = 80$ — A Model

$P = 60$ $R = 100$ — B Model

Central में रहता है

$AM = \frac{80+80}{2} = 80$ $\quad$ $\frac{60+100}{2} = 80$

Same

$HM = \frac{2 \times 80 \times 80}{160} = 80$ $\quad$ $\frac{2 \times 60 \times 100}{160} = 75$

F1 Penalize किए क्योंकि हमारा Precision कम है

(F1 को छोटा)

P कम तो F1 भी कम

eg $P = 0$ $R = 100$

Mean = 50 $\quad$ HM = 0

Penalize to P (small)

How to find (Multi-Class Precision & Recall)

actually में Precision / Recall सभी class के लिये निकाला जाता है (Binary में समय 1 वाले के लिये ही निकाला था)

eg 3 class

		Predicted			
		Dog	Cat	Rabbit	Total
Actual	Dog	25	5	10	40
	Cat	0	30	4	34
	Rabbit	4	10	20	34
	Total	29	45	34	

P_D = Precision of Dog

$P_D = \frac{25}{29} = 0.86$, $P_C = \frac{30}{45} = 0.66$, $P_R = \frac{20}{34} = 0.58$

$P_D = \frac{25}{29} = 0.86$, $P_C = \frac{30}{45} = 0.66$, $P_R = \frac{20}{34} = 0.58$

Combine Precision

Macro-Precision : → take average

$$\frac{0.86 + 0.66 + 0.58}{3} = 0.70$$

Weighted Precision → $\frac{40}{108} \times 0.86 + \frac{34}{108} \times 0.66 + \frac{34}{108} \times 0.58$

$= 0.71$

Macro is used when balanced (nearly equal class size)

weighted → unbalanced

In above example (all nearly balanced) so Macro

Recall $R_D = \frac{25}{40} = 0.62$ $R_C = \frac{30}{34} = 0.88$ $R_P = \frac{20}{34} = 0.58$

→ Macro Recall $\frac{0.62 + 0.88 + 0.58}{3}$

→ Weighted ✓

Same as F1 score

add this for multi-class.

use precision_score(y_test, y_pred, average = 'macro')

(or = 'None')

(or 'weighted')

'None' → सभी का अलग-अलग आ जायेगा

'macro' → Single value

'weighted' → —"—

(or) use direct

सभी का औसत-औसत
का उपयोग

'macro' → single value

'weighted' → —"—

(or) 'weighted'

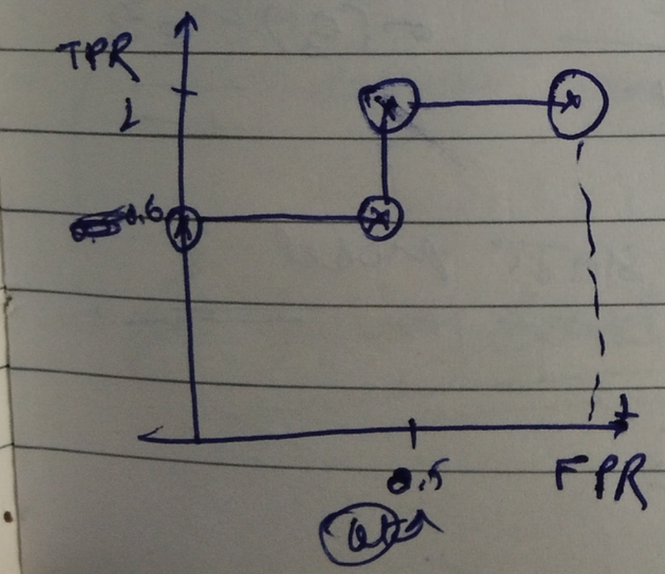
(en) use direct

from sklearn.metrics import classification_report

print(classification_report(y_test, y_pred))

finish

ROC & AUC



eg

y	$\hat{y}_{pred}$	$\hat{y}(0)$	$\hat{y}(0.2)$
1	0.8	1	1
0	0.91	1	1
1	0.4	1	1
1	0.3	1	1
0	0.2	1	0
1	0.7	1	1

→ threshold value

$TPR = TP/(TP+FN)$

$FPR = FP/(FP+TN)$

$TPR = \frac{4}{4+0} = 1$, $FPR = \frac{2}{2+0} = 1$ (1,1)

$\frac{4}{4+0} = 1$, $\frac{1}{1+1} = 0.5$

(1, 0.5)