* Performance prediction of models : Accuracy → metric
↓
classification, regression
(KNN)

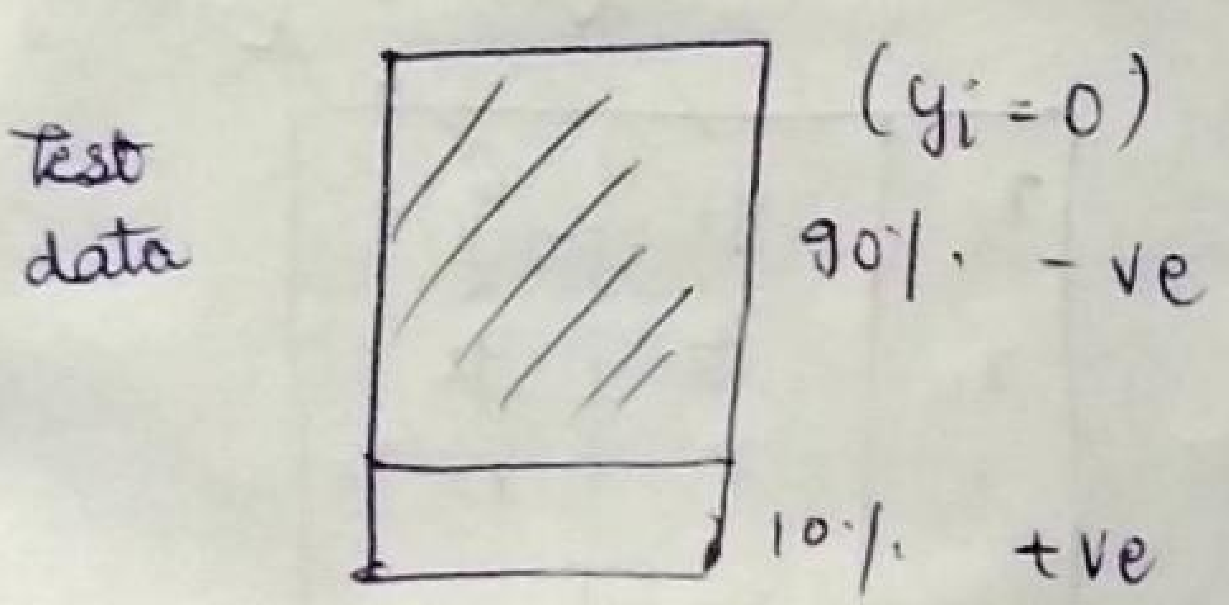
Accuracy = $\frac{\text{no of correctly classified pts}}{\text{total no. of points in } D_{test}}$

easy to understand measure

Performance is always measured on test data.

2|01

① Imbalanced data :→



dumb model $x_q \rightarrow$ -ve
{return -ve}

↳ model → accuracy = 90% = 0.9
high accuracy

So, accuracy is not at all a good measure to predict if a model is performing good or not when dataset is imbalanced.

②

	x	y	M_1	M_2	$\hat{y}_1$	$\hat{y}_2$
+ve	x_1	1	0.9	0.6	1	1
	x_2	1	0.8	0.65	1	1
-ve	x_3	0	0.1	0.45	0	0
	x_4	0	0.15	0.48	0	0

{Test set}

M_1 & M_2 return a prob score
KNN

{ $x_q \rightarrow$ prob $(y_q = 1)$
$(0 \leq p \leq 1)$

$\hat{y}$ = Predicted value

⇒ predicted class labels are exactly the same in M_1 & M_2.

⇒ M_1 is better than M_2 by looking at probability scores.

accuracy ⟶ cannot use prob-score
↓
$\hat{y}_1$ & $\hat{y}_2$ → by looking at $\hat{y}_1$ & $\hat{y}_2$, M_1 & M_2 are having same accuracies.

21.2
* Confusion-matrix :- doesnot / cannot process prob-scores

binary classification task (0, 1) ← 2 classes

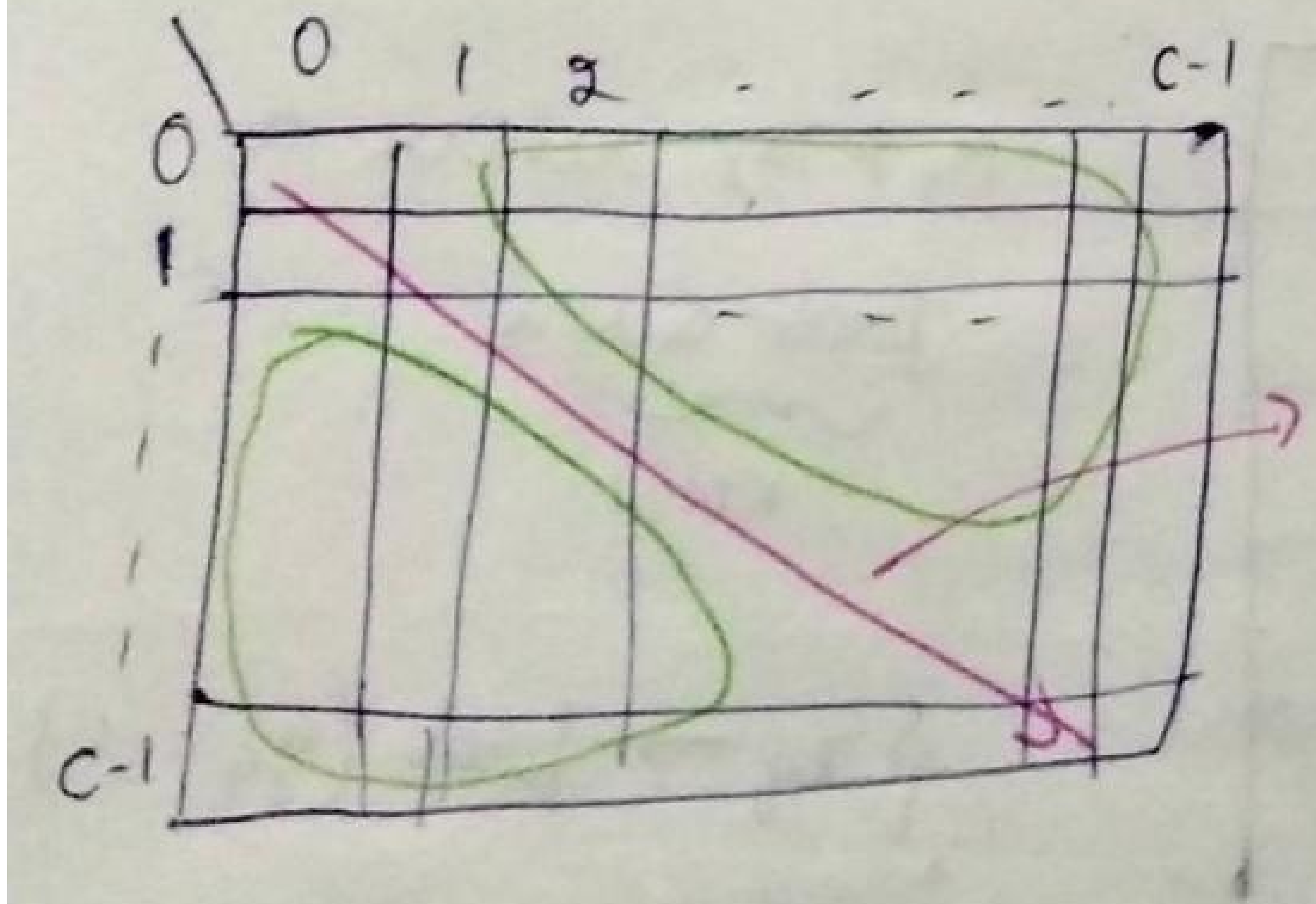
$$\begin{array}{c|cc} & \multicolumn{2}{c}{\text{actual } (y_i)} \\ \text{predicted } (\hat{y_i}) & 0 & 1 \\ \hline 0 & a & b \\ 1 & c & d \end{array}$$

	binary	binary
x_1	y_1	$\hat{y}_1$
x_2	y_2	$\hat{y}_2$
⋮	⋮	⋮
x_n	y_n	$\hat{y}_n$
data point	actual class label	predicted class label.

a = no. of points such that
$y_i = 0$ & $\hat{y_i} = 0$

⇒ a & d should be high &
c & d should be low

multiclass-classification :- (c-Classes)



principal diagonal values should be high

and off-diagonal values should be small for a sensible model.

predicted ↓ \ actual →	0	1
0	True negative	false negative
1	False positive	True positive
	N	P

True positive (TP)

↓ are yours predicted label matching with actual? (T/F)

→ what is the predicted label?

N = sum of false positive and true negative

P = sum of false negative and true positive

n = total no. of points (N+P)

$$\text{True positive Rate (TPR)} = \frac{\text{True positives}}{P}$$

$$\text{True negative Rate (TNR)} = \frac{\text{True negatives}}{N}$$

False ~~negative~~ positive Rate (FPR) = $\frac{\text{false positives}}{N}$

$$\text{False negative Rate (FNR)} = \frac{\text{false negatives}}{P}$$

For ex :→

predicted ↓ \ actual →	0	1
0	850 TN	6 FN
1	50 FP	94 TP
	N = 900	P = 100

Test :→ 900 -ve
100 +ve } imbalanced

↑ TPR = 94%

↑ TNR = $\frac{850}{900}$

FPR = $\frac{50}{900}$ ↓

FNR = 6% ↓

For a sensible model, TPR & TNR should be high & FPR & FNR should be low.

dumb model :→ all -ve

→ actual

predicted ↓

	0	1
0	900 TN	100 FN
1	0 FP	0 TP

900 = N P = 100

900 -ve

100 +ve

xq :→ -ve : '0'

terribly low ← × TPR = 0%   FPR = 0% ✓

✓ TNR = 100%   FNR = 100% ×

For ex :→ To diagnose a cancer patient if he/she has cancer or not.

Domain specific

→ act

pred. ↓

	0	1
0	TN	FN
1	FP	TP

our objective is not to miss any cancerous patient

V.low FNR should be there as, FNR means to say that patient does not have cancer but it actually has cancer.

If FPR is high → that's okay also, as if a patient doesn't have cancer and it predicted to be cancer patient then he/she will go through more powerful tests to prove him non- Cancerous.

interpretable in english sentences.

21.3

★ Precision, Recall & F_1-Score :→

act →

pred ↓	0	1
0	TN	FN
1	FP	TP
	N	P

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

↓

of all the points the model declared / predicted to be +ve, what %age of them are actually +ve

Recall :→ same as TPR $= \frac{TP}{P}$

↓

of all the actual positive points, what %age of them, are predicted positive by the model.

we want precision to be high & Recall should be high.

Precision ↑ (0-1)

Recall ↑ (0-1)

F_1-score is combination of Precision & Recall.

Imp for kaggle competitions ↑

$$F_1\text{-score} = \left(2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}}\right)$$

↓

harmonic mean of precision & recall

$\boxed{F_1\text{-score} \uparrow \text{ if Precision} \uparrow \text{ \& recall} \uparrow}$

31/4

→ Reciever Operating characteristic curve & AUC

(ROC) curve ↳ designed by electronics & radio engineers during second world war.

$\hat{y}_{\tau_2}=0.92$	$\hat{y}_{\tau_1}=0.95$	x	y	$\hat{y}$	
1	1	x_1	1	0.95	$\to \tau_1$
1	0	x_2	1	0.92	$\to \tau_2$
0	0	x_3	0	0.80	
0	0	x_4	1	0.76	
0	0	x_5	1	0.71	

model → 1, 0

↳ gives score (like ~~prob score~~)

① Sort your data in decreasing order of $\hat{y}$

② Thresholding → Take any value

(τ)

↓

tow

ⓐ $\tau_1 = 0.95$

if $\hat{y} \geq \tau_1$.

①

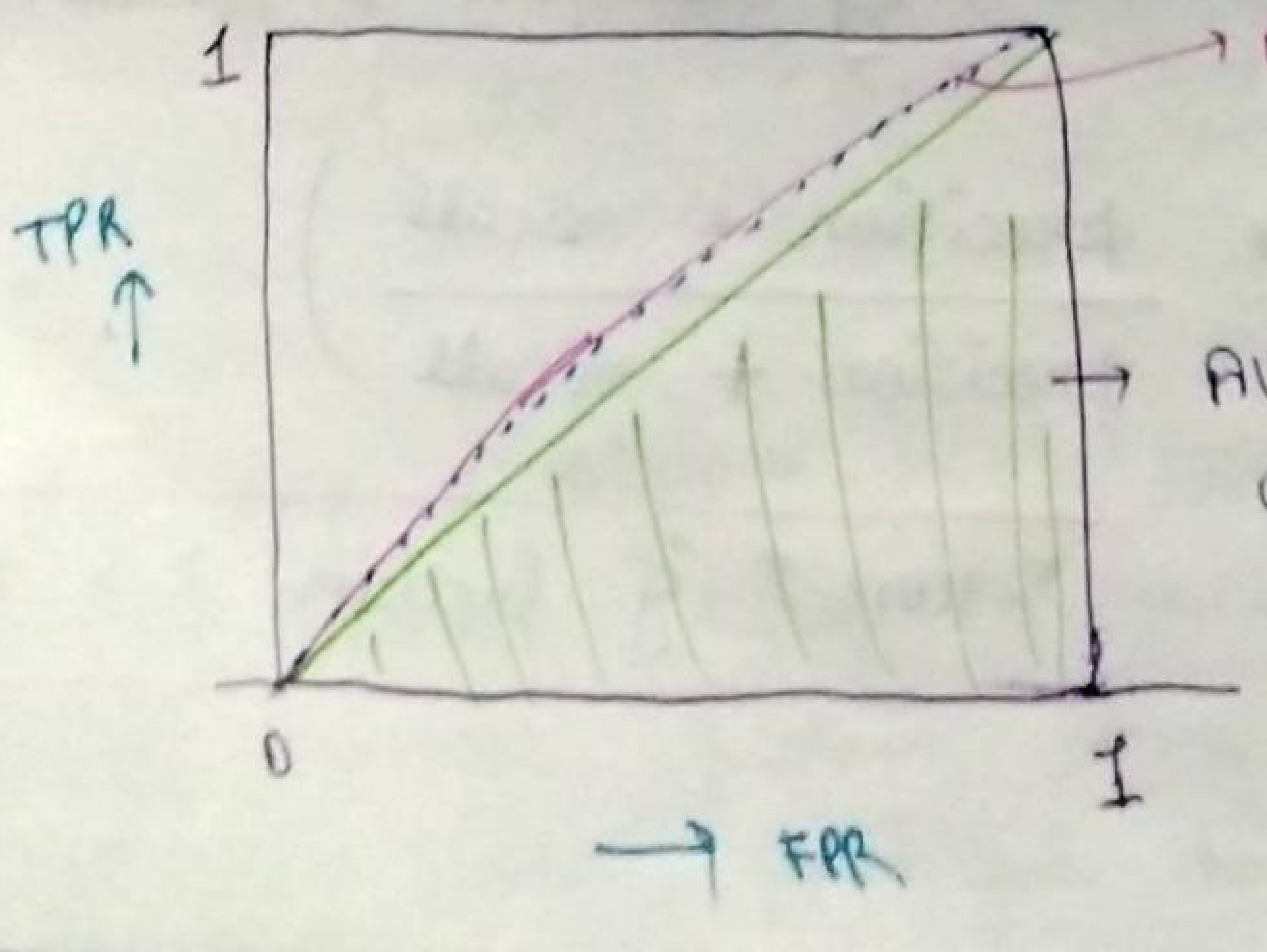
else ⓪

For each $\tau_1, \tau_2, \tau_3, \ldots \tau_n$

↓ we can easily get

FPR$_1$, TPR$_1$

↓ FPR$_2$, TPR$_2$ and so on



→ AUC → area under curve

can lie b/w ⓪ to ①

↓ terrible

↓ v.good.

AUC :→

1. Imbalanced data → AOC can be high with a dumb model too / Simple model.

2. AUC is not dependent on the $\hat{y}$ scores but it depends on the decreasing order

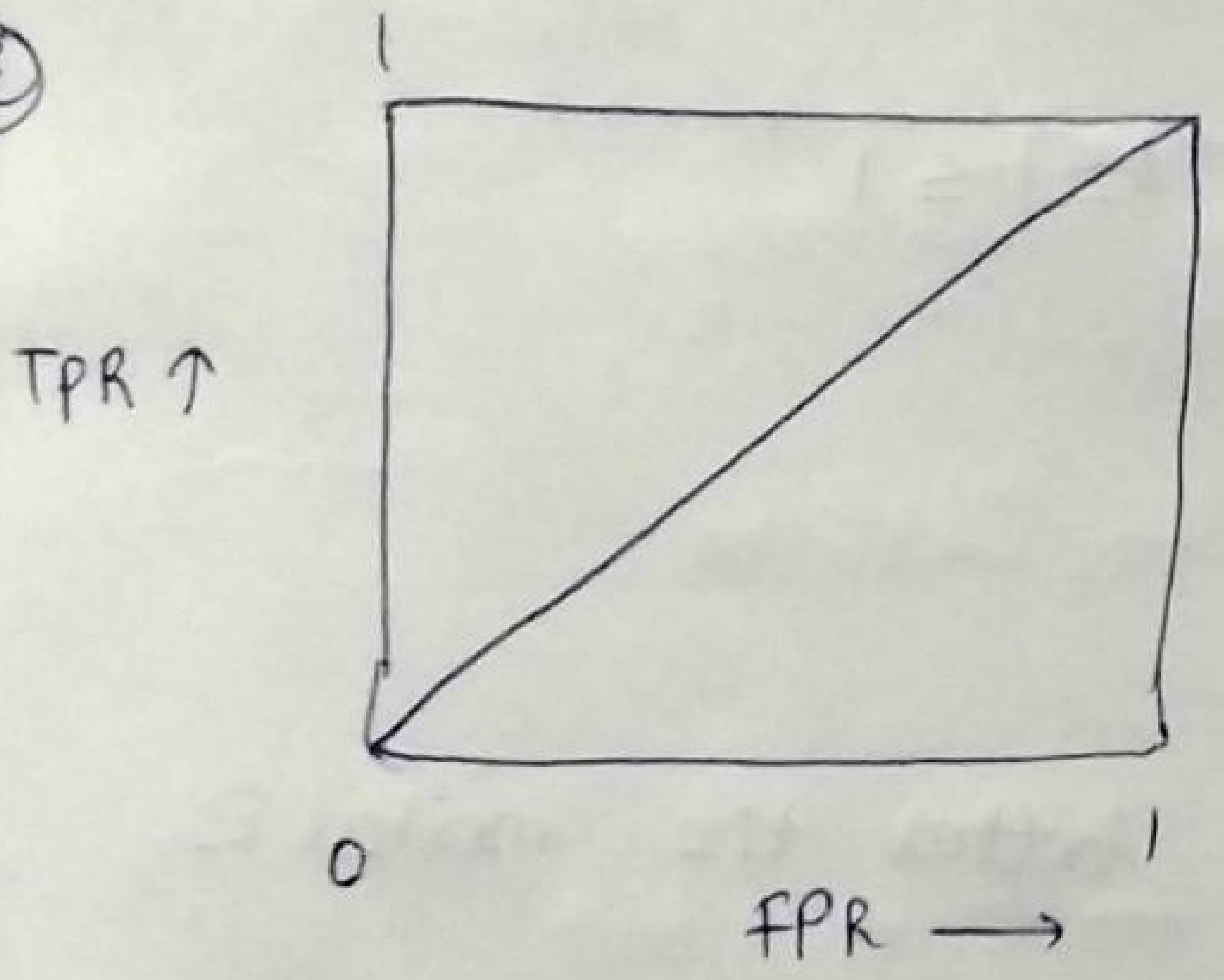
		M_1	M_2
x_1	1	0.95	0.2
x_2	1	0.92	0.1
x_3	0	0.80	0.08
x_4	1	0.76	0.07
x_5	1	0.71	0.06

$AUC(M_1) = AUC(M_2)$

as AUC only depends on order.

and not at all depend on actual value

3.

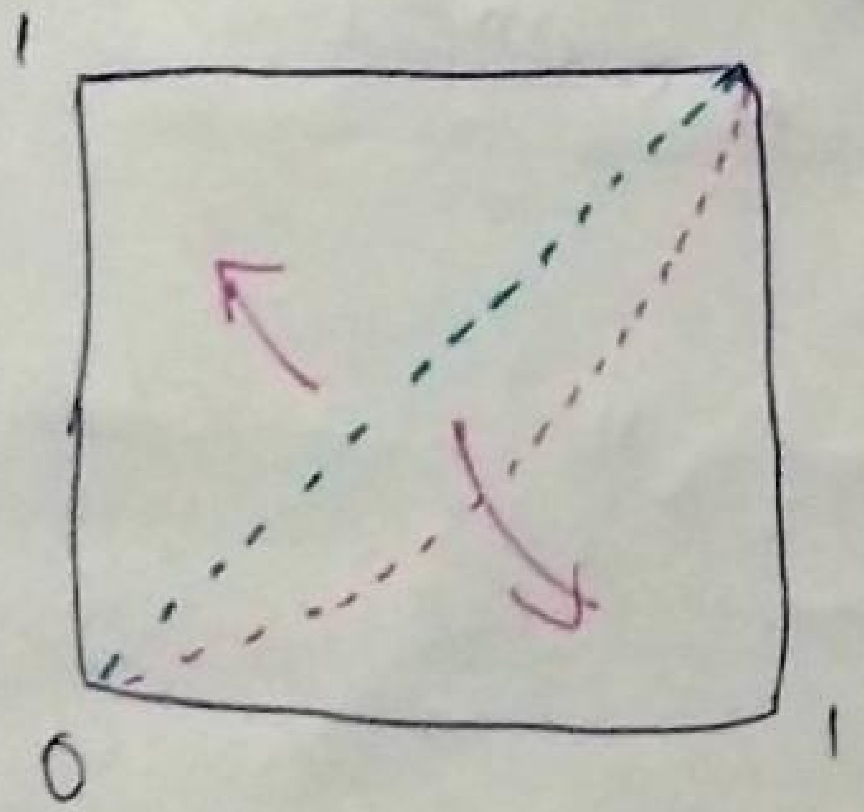


random model

↳ x_q → 1 or 0

{ AUC (random - model) = 0.5 }

4. Model M :→



Let AUC (M) = 0.2 → means you did something wrong in modelling

↳ worse than random model.

AUC →

0.5 to 1 → ✓

0.5 → random model

0.0 to 0.5 → worst

↳ swap class labels, AUC = 1 - 0.2

$\hat{y}_i = 0 \longrightarrow 1$ } swa

$\hat{y}_i = 1 \longrightarrow 0$ } - pping

Scanned by TapScanner

21.5

→ (0 to ∞)

log-loss :→ uses prob-scores.

↙ as small as possible

Binary classification :→

x	y	$\hat{y} = P$
x_1	1	0.9
x_2	1	0.6
x_3	0	0.1
x_4	0	0.4

x_1: → $-\log(0.9) = 0.0457$

x_2: $-\log(0.6) = 0.22$ ←

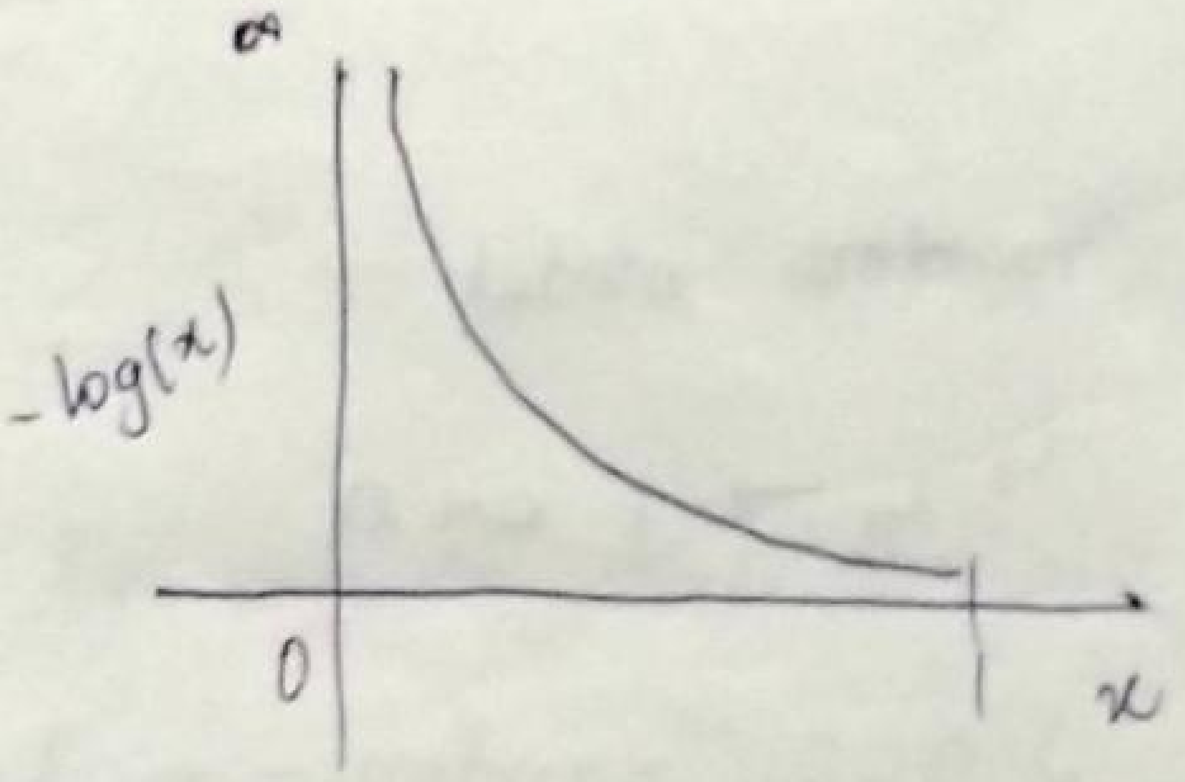
x_3: → $-\log(0.9)$ ↳ 0.0457

x_4: $-\log(0.6) = 0.22$ ←

Test set of n- points

$$\text{log-loss} = -\frac{1}{n}\sum_{i=1}^{n}\left\{\left(\log(P_i) * y_i\right) + \left((1-y_i) * \log(1-P_i)\right)\right\}$$

log-loss :→ average of negative log prob of correct class label.



$0 \le p \le 1$

Smaller the log loss, better the model is.

Multiclass log loss :→

$$-\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{c} y_{ij}\log(P_{ij}) \rightarrow \text{probability that } x_i \in \text{class } j$$

y_{ij} ↓ $=1$ if $x_i \in$ class j

otherwise 0

Best case of log loss is 0, but other values are not easy to make sense.

21.6

R^2 (or) coefficient of determination:→

↓ For regression

Test:→ $y_i \in \mathbb{R}$

$i = 1$ to n x_i , y_i , $\hat{y}_i$, $\boxed{e_i \Rightarrow y_i - \hat{y}_i}$ ↓error

↑ actual output ↑ model predicted output

$$\boxed{SS_{Total} = \sum_{i=1}^{n} (y_i - \bar{y})^2}$$

Sum of squares

y_i ↓ actual $\bar{y}$ ↳ mean of actual class labels

, here $\bar{y} = \frac{1}{n} \sum_{i=1}^{n} y_i$

$\bar{y}$ → avg. value of y_i's in test data.

For regression,

the simplest model you can build is called the average model → simple mean model

$$x_q \longrightarrow \text{mean}(y_i) = \bar{y} = y_q$$

SS_{total} → sum of squared errors using simple-mean model.

residues

$$\boxed{SS_{res} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} e_i^2}$$

residue $= e_i = y_i - \hat{y}_i$

y_i ↓ actual $\hat{y}_i$ ↳ predicted

$R^2 = \left(1 - \frac{SS_{res}}{SS_{tot}} \right)$ Case1 :→ $SS_{res} = 0 \leftarrow \boxed{e_i = 0} \rightarrow R^2 = 1$

phenomenal model

↓ Best value

Case 2 :→ if $SS_{res} < SS_{tot}$; $R^2 = 0$ to 1

Case 3 :→ if $SS_{res} = SS_{tot}$; $R^2 = 1 - 1 = 0 \rightarrow$ model is same as simple mean model.

Case 4 :→ if $SS_{res} > SS_{tot}$; $R^2 = 1 - (\text{greates} > 1) = -ve$

model is worse than simple mean model.

21.7

Median absolute deviation of errors :→

$SS^2_{res} = \sum_{i=1}^{n} e_i^2$ if one e_3 is very large

R^2 is not very robust to outliers.

$x_i \longrightarrow y_i, \hat{y}_i, e_i$, $|e_i|s \rightarrow 0 \rightarrow$ great

$|e_i|s \longrightarrow$ large → not so good.

if $e_i \rightarrow$ random variable,

mean ← $\text{median}(e_i)$ = central value of errors → small

acting like these

~~std.~~ dev. ← $\text{MAD}(e_i) = \text{median}\Big(\underbrace{|e_i - \text{median}(e_i)|}_{\text{deviation}} \Big) \rightarrow$ small.

abs

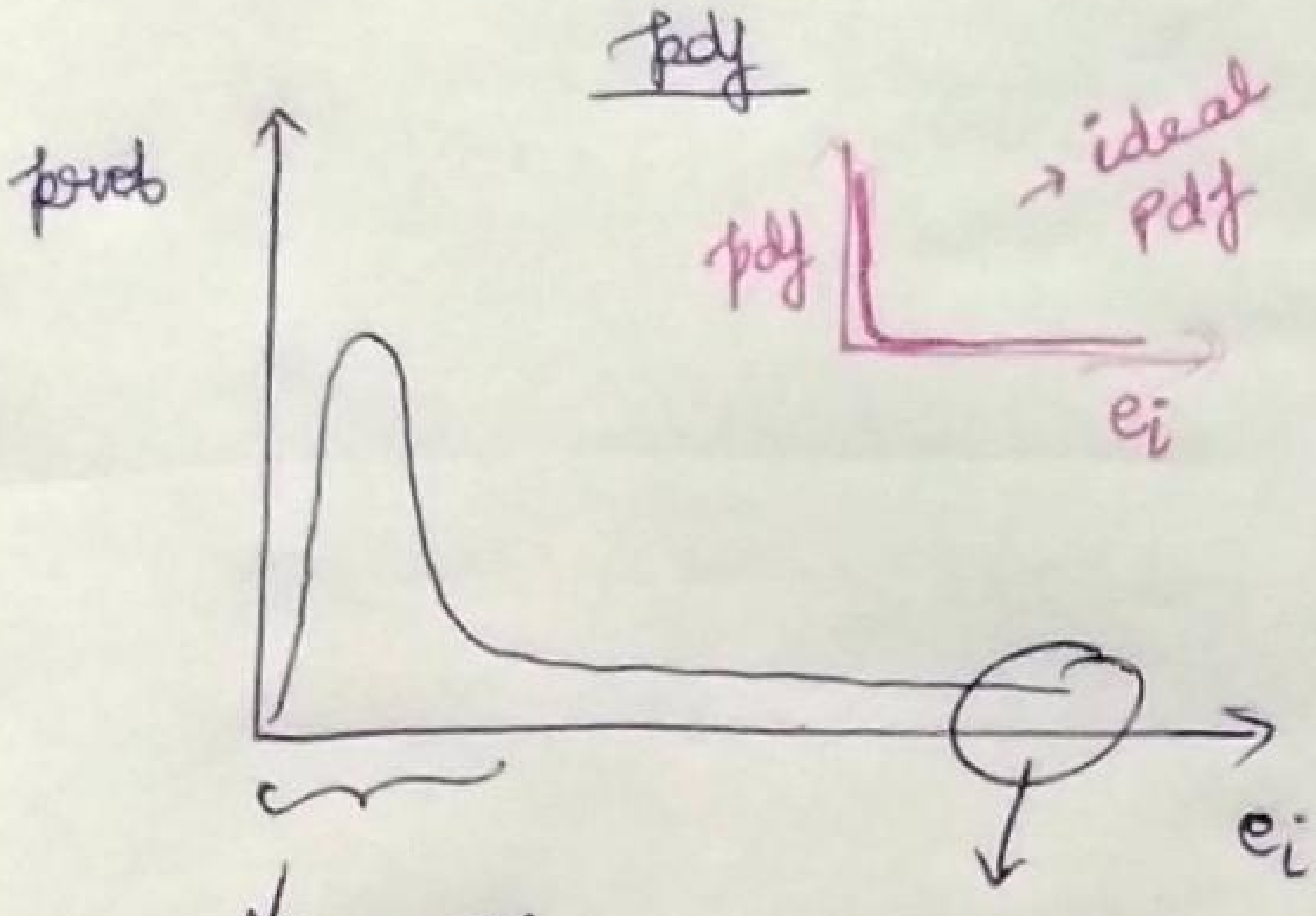
mean or $\boxed{\text{median}}$ of eis → used to understand if the errors are small or large.

std-dev or MAD

↓

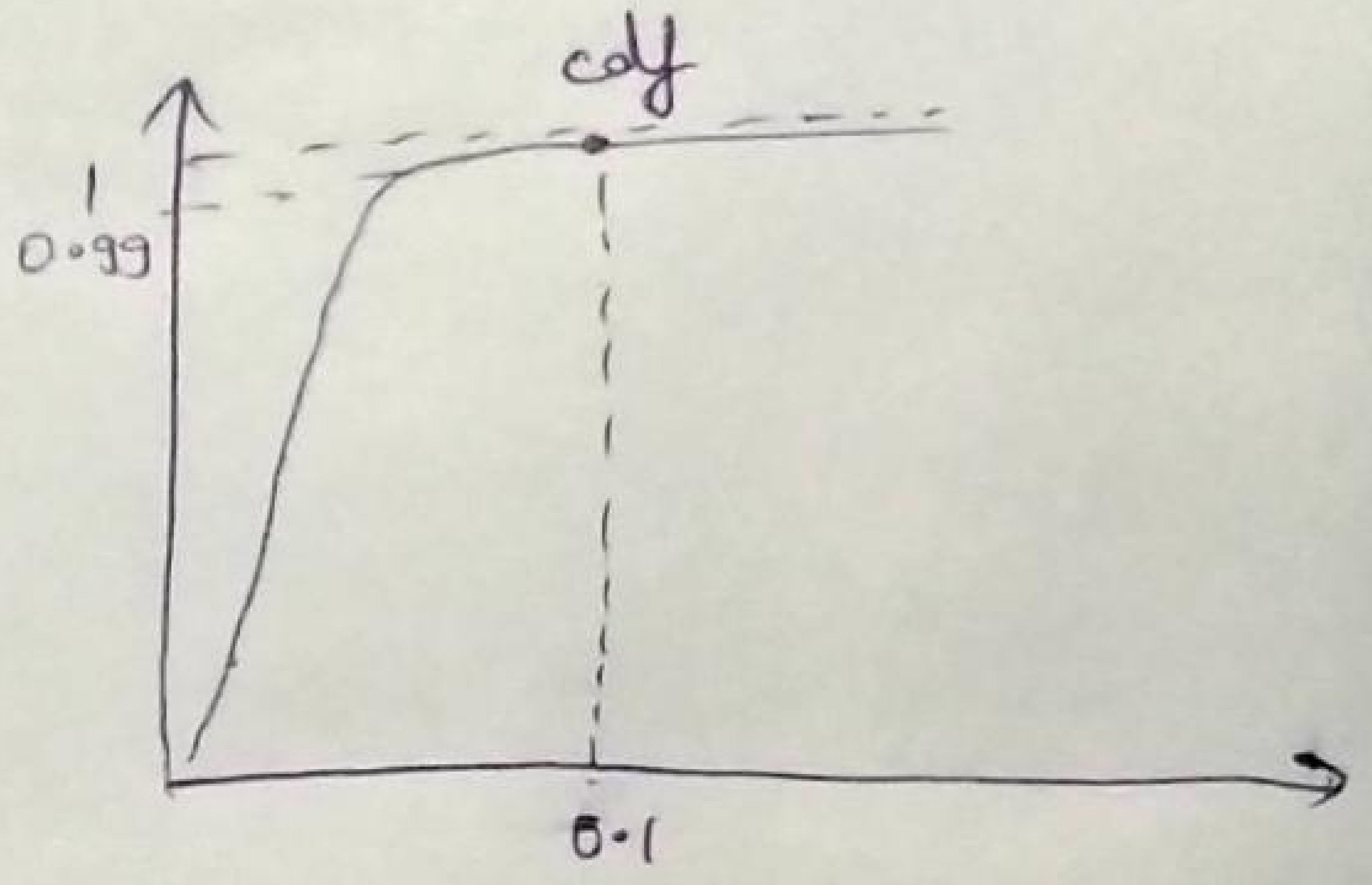
robust to outliers

21.8 **Distribution of errors :→**



most errors are small

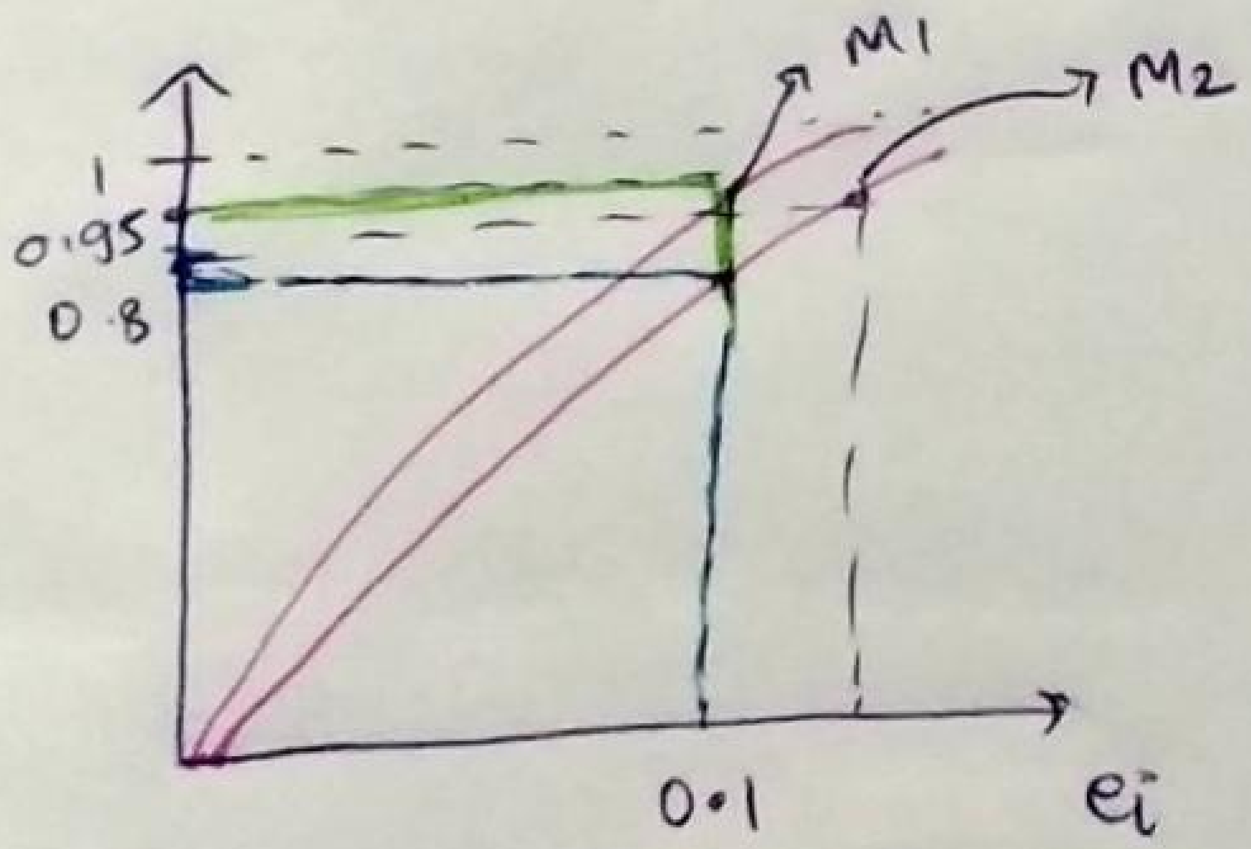
very few errors are large



99% of errors are < 0.1

1% errors are ≥ 0.1

Models M_1 & M_2



M_2 cdf is below M_1

M_1 :→ 95% errors are below 0.1

M_2 :→ 80% errors are below 0.1.

So, M_1 is a better model for regression than M_2