→ "Performance Measurements of a Model" :-

Here we will understand, as how to measure the performance of a model,
Here we will be focusing on "classification & Regression" models"

Note :- K-NN can be used for both 'classification as well as regression'.

One measure that we have already seen is called 'Accuracy'

let's define Accuracy, & then we will see where it is useful & where it is not useful. :-

$$\text{Accuracy} = \frac{\text{\# of correctly classified points}}{\text{Total \# points in } D_{Test}}$$

It lies b/w 0 & 1
'bad' 'better'

One of the biggest advantages of accuracy is,
it is very very easy to understand the performance of a model
because. if you tell me that, our D_{Test} has 100 pts

100 pts ⟶ 60 +ive $\xrightarrow{M}$ 53 +ive . 7 -ive
↳ 40 -ive ⟶ 35 -ive , 5 +ive

Now of these 60 +ive pts, Model 'M' predicted 53 of them as +ive & 7 as -ive
& Of the 40 -ive pts, model 'M' predicted 35 of them as -ive & 5 points as +ive.

So in total we made (our model) made 12 errors..

errors :- 12 (incorrectly classified)
Correctly classified :- 88 pts

∴ Accuracy is 88%

Note :- Performance of any model is measured only on the "test-data"

P.T.O

There are some problems associated with ~~imbalanced~~ "Accuracy"

Case① Imbalanced data :-

let's say in our test-set, 90% of points belong to ["-ive class"] & only 10% of points belong to ["+ive class"]

Test data

$y_i=0$ 90% "-ive"
$y_i=1$ 10% "+ive"

When; we have something like this Imagine, if we have a model & also our model is a ["dumb model"]

"Model is dumb" let's assume this model says that, given any query point "x_q" label it as -ive

$$\{ x_q \longrightarrow -ive \}$$

Now becoz our data is 90% -ive & 10% +ive, so if we run our model "M" on our "test.data", Remember this is a "dumb model" then the accuracy will be - 90% or 0.9

dumb model $\longrightarrow$ acc: 90% or 0.9 becoz in the underlying test data, we have 90% of "-ive points".

So, becoz of the imbalanced data., even a dumb model gets high accuracy.

Note:- Accuracy is not a useful measure, when we have imbalanced data.

Case②. Just an example,

Imagine, we have our "test data" comprising of 4 points (x_1, x_2, x_3, x_4). let's assume for each of those points, belongs to some classes as shown.

	"x"	"y"	M_1	M_2
+ive class	x_1	1		
	x_2	1		
-ive class	x_3	0		
	x_4	0		

$1 \rightarrow$ +ive class

$0 \rightarrow$ -ive class

Let's assume we have two Models "M_1" & "M_2".

P.T.O

When we run 'M_1' & 'M_2' on our test set, let's assume that these models return a "probability score".

Let's say these models instead of returning '1' or '0', they return a probability score.

which mean, given a datapoint 'x_q' they are returning the probability score as.

$$\underbrace{x_q \longrightarrow Prob(y_q = 1)}$$

{ Given a datapoint 'x_q', what is the probability that $y_q = 1$ }

Probability score → lie b/w "0 & 1" $\{0 \le P \le 1\}$

	X	Y	M_1	M_2
+ve	x_1	1	0.9	0.6
	x_2	1	0.8	0.65
-ve	x_3	0	0.1	0.45
	x_4	0	0.15	0.48

doing good

Now if we compare ['M_1'] & ['M_2'] & intuitively speaking we can say that "M_1" is better

$\hat{y}$: 'Predicted value'

[$\hat{y}_1$ → "Predicted value of M_1"
$\hat{y}_2$ → "Predicted value of M_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

→ Here we can see that both the Models are predicting the same values {same class labels}

But by looking at the probability values, we know that 'M_1' is better than "M_2"

"Accuracy Measure" cannot use "probability scores", It can only use predicted class labels. so using Accuracy we can say that both "M_1" & "M_2" are having same Accuracy but we know that, $\boxed{M_1}$ is working better than $\boxed{M_2}$.

→ "Confusion Matrix, TPR, FPR, FNR, TNR"

Let's understand what a confusion matrix is, & what problems does it solve. that accuracy has.

Let's start with a simple ["binary classification task"].

In binary classification task, we have two classes {0, 1}

As the name suggests "Confusion matrix" is a matrix. So we create a grid of 2×2

Actual → (Actual values) (y_i)

Predicted ↓ (Predicted values) ($\hat{y_i}$)

$$\begin{array}{c|cc} & 0 & 1 \\ \hline 0 & a & b \\ 1 & c & d \end{array}$$

"Test-Data"

		"M"
x_1	y_1	$\hat{y_1}$
x_2	y_2	$\hat{y_2}$
x_3	y_3	⋮
⋮	⋮	⋮
x_n	y_n	$\hat{y_n}$

"datapoints" (Actual "Class labels") Predicted class labels

Note :→ Confusion Matrix doesnot take "probability score".

Just like y_i's, $\hat{y_i}$'s are also binary values.

Now given this data, we can say,
how many points do we have, which are actually "0" & which are predicted "0".

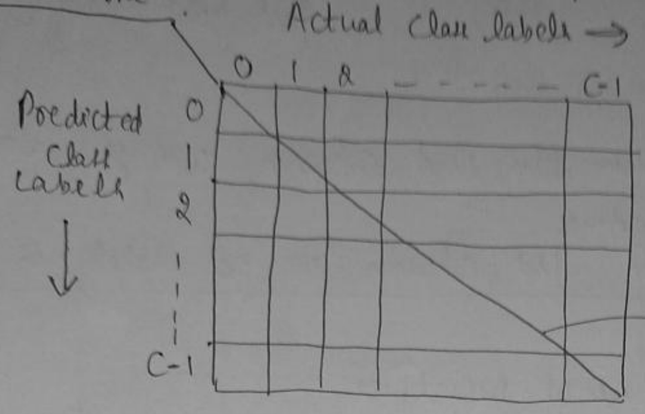
Let's say in the first cell, we get a value "0"

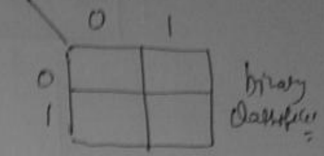
a: # points such that $y_i = 0$ & $\hat{y_i} = $ "0".
b: # points such that $y_i = 1$ & $\hat{y_i} = $ "0"
c: # points such that $y_i = 0$ & $\hat{y_i} = $ "1"
d: # point such that $y_i = 1$ & $\hat{y_i} = $ "1"

This matrix is called a Confusion matrix because it tells us of all the four possibilities that are possible.

Now if instead of a binary classification, we have a "multiclass classification" setting (5)

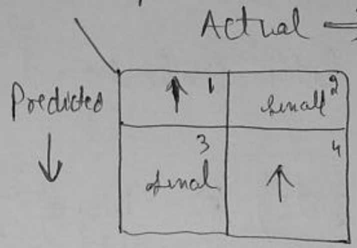
Suppose we have "C-classes", then we basically draw a "C×C matrix"





Where "C" is the number of Classes.

Now one thing we notice is, if the model is sensible & not dumb, then our model should predict most of the data points which are actually correct



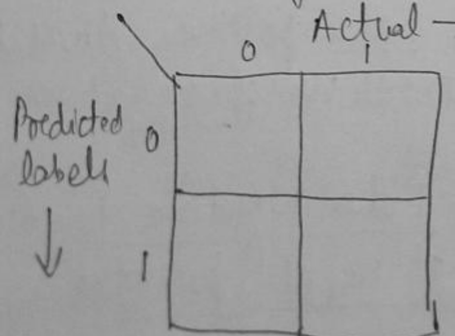
∴ Values in Cell no. 1 & 4 should be high

Similarly in a multiclass setting the cells belonging to the principal diagonal of this "C×C" matrix should have a "high-value".
& all the off diagonal elements should have a relatively small values

So, in Confusion matrix, the Principal diagonal elements should have a high value.

Now let's define a bunch of things :-

If this is a confusion matrix for a "binary classification setting"



There are special names for each of these cells in the confusion matrix

Predicted ↓ \ Actual →	0	1
0	'TN'	'FN'
1	'FP'	TP

To remember: T P

2nd part means ("What you predicted")

first part means ("Are you correct")

If we sum the 2nd column, we get total no. of "Positives"

if we sum the 1st column, we get total no of "Negatives"

N:- Total no of Negatives

P:- Total no. of Positives. & "n" = N+P

Total no. of points.

Now given these things let's understand some more points :-

1) "TPR" (True positive rate) :- No. of True positives divided by Total no. of positives.

$$\left[TPR = \frac{TP}{P} \right].$$

2) "TNR" (True Negative Rate) :- No. of True negatives divided by Total no. of Negatives.

$$\left[TNR = \frac{TN}{N} \right]$$

3) "FPR" (False Positive Rate) :- No. of False positives divided by Total no. of negatives.

4) "FNR" (False Negative Rate) :- No. of False Negatives divided by Total no. of positives.

These four rates are very useful.

Let's take a simple example to understand it even better.

Let's consider a test dataset comprising of 1000 pts in total,

Out of that { 900 are −ve } Imbalanced data set.
{ 100 are +ve }

Let's draw the confusion matrix

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

Let's say our model predicted the values as shown in ("confusion matrix")

Now we have ~~these~~ four values.

Now let's look at our four rates.

$$TPR = \frac{TP}{P} = \frac{94}{100} = 94\%$$

$$TNR = \frac{TN}{N} = \frac{850}{900} \quad \text{large / good}$$

$$FPR = \frac{FP}{N} = \frac{50}{900}$$

$$FNR = \frac{6}{100} = 6\%$$

So, our model is said to be good, if TPR is high, TNR ↑ & FPR & FNR are low.

$$\begin{bmatrix} TPR \uparrow & FPR \downarrow \\ TNR \uparrow & FNR \downarrow \end{bmatrix} \quad \text{Model is good}$$

Even if our dataset is imbalanced, just by looking at four rates we can say, ~~that this~~ model is sensible

Now imagine if we have a dumb model

We have a dumb-model with same data.

Predicted ↓ \ Actual →	0	1
0	900 TN	100 FN
1	0 FP	0 TP

$N = 900$ $P = 100$

900 = -ive pts
100 = +ive pts

Let's assume our dumb model is predicting all all test points to be -ive

Now lets find the four numbers

$$TPR = \frac{TP}{P} = \frac{0}{100} = 0\%$$

$$TNR = \frac{TN}{N} = \frac{900}{900} = 100\%$$

$$FNR = \frac{FN}{P} = \frac{100}{100} = 100\%$$

$$FPR = \frac{FP}{N} = \frac{0}{100} = 0\%$$

By looking at these four numbers, we can conclude that our model is doing 'something stupid' becoz we want TPR & TNR to be high but here it is not

∴ Even in an imbalanced dataset, confusion matrix & these four rates can help us understand how our model is performing.

Remember. Accuracy did not do a good job with imbalanced datasets.'

Which of these four rates (TPR, TNR, FPR & FNR) is more important, That is very very domain specific