→ "Precision, Recall & F1-score" :>

There are some related measures called "Precision & Recall" & they are often used in 'information retrieval problems'

They are often used when we have a large corpus of text. They are used in search engines.

Predicted ↓ / Actual →	0	1
0	TN	FN
1	FP	TP

"N" "P"

The formula for precision is

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

Let's first understand what does precision means Intuitively :>

Precision (Pr) :> Of all the points, the model declared / predicted to be +ive, what % age of them are actually 'tive'.

↓

In precision, we are not worried too much about the negative class, same is the case with recall also.

Note :> 'Precision & recall' tells are about the ["positive-class"] & not about the ['negative-class"]

Recall :> It is nothing but the TPR (True positive rate)

$$TPR = \frac{TP}{P}$$

In both the cases, Recall & Precision basically revolves around "TP" only.

So, 'Precision & Recall" are the metrics, which are very useful when we care about mostly the ["positive class"].

Let's understand recall intuitively, Recall basically means of all the points which actually belong to ["positive class"], how many, the model detected to be of 'positive class'

"Precision" :→ of all the points that the model predicted to be positive, how many are actually positive.

"Recall" :→ of all the actually positive points, how of them are predicted to be positive by the model.

Now given these two measures, Is there any way for us to combine ["Precision"] & ["Recall"] into one measure (single measure).

Note :→ we always want "Precision" & "Recall" to be high.

{ Pr ↑ (0 to 1) Re ↑ (0 to 1) }

There is a measure called "F1-score" which combines both these metrics ("Precision" & "Recall").

$$\text{"F1-Score"} = \left(2 * \frac{Pr * re}{Pr + re} \right)$$

Interpreting "F1-score" is much more harder than interpreting "Precision" & "Recall"

Given Precision, & Recall, it is much easier to understand, but there will be some instances where everything has to be converted into just a single metric, that's where we use "F1-score".

The formula is based on "Harmonic Mean"

$$F1 = \frac{2}{\left(\frac{1}{recall} + \frac{1}{Precn.} \right)} \rightarrow \frac{1}{2}\left(\frac{1}{re} + \frac{1}{Pr} \right)$$

$$= \left[Avg\left(inv\text{-}recall \; ; \; inv.\text{-}pr. \right) \right]^{-1}$$

F1-score" is often used in lots of "Kaggle competitions" & F1-score is high, if "Precision" & "Recall" both are high.

Precision & Recall in F1-score are represented as.

$$F1\text{-score} = \left(2 \times \frac{(P) * (R)}{P + R} \right)$$

These numbers lie b/w "0 & 1"

F1 score also lies b/w "0 & 1"

["1" being very good & "0" being very bad.]

→ Receiver operating characteristic curve & AUC :-

(ROC)

"Area under the curve"

Other very interesting metric is called "ROC" curve & "AUC" (Area under the curve)

This curve has very nice interesting history, This curve has very nice interesting history. Actually. It was designed by "Electronics & Radio Engineers" during "Second World War" to predict how well their missiles are working.

Let's understand, what "ROC" is :-

Imagine if we have a dataset like (comprising of five datapoints) for each datapoint, we have corresponding class labels

"x"	"y"	G
x_1	1	
x_2	1	
x_3	0	
x_4	1	
x_5	1	

The class labels (Actual class labels) associated with these data points are (1,1,0,1,1)

Now let's assume that our model, which does "binary classification" not only gives us the class labels associated with the datapoints, It also gives us a score like a probability score. It could be any score here, a score that represents as: More the score, more is the chance that it belongs to Class 1,

or vice versa.

Let's assume that our model gives the probability scores associated with each point as:

	x	y	$\hat{y}$
x_1		1	0.95
x_2		1	0.92
x_3		0	0.80
x_4		1	0.76
x_5		1	0.71

Now the probability score that 'x_i' belongs to class '1' is let's say 0.95

similarly for x_2 let's say it is 0.92

x_3 " " " 0.80

x_4 " " 0.76

x_5 " " 0.71

⇒ decreasing order of $\hat{y}$"

first thing that you would notice here is, we have sorted our data in decreasing order of $\hat{y}$.

1) So, first step in 'ROC' plotting is,
Take all your data & sort it in decreasing order of "$\hat{y}$".

2) Once we have sorted our data in decreasing order of "$\hat{y}$", we will do "thresholding" (T)

So, "thresholding" works as follows:⇒

We can take any value of $\hat{y}$ for thresholding.

Let's take the first largest value as threshold.

as $T_1 = 0.95$,

After taking $T_1 = 0.95$, we will say that, if $\hat{y} \geq T_1$ then declare it to be "class 1"

else declare it to be "class 0"

$T_1 = 0.95$.
if $\hat{y} \geq T_1$
else
1
0

for every "datapoint" in our dataset, we will get "class labels" & we can compute "TPR & FPR"

So, when we have $T_1 = 0.95$, All the points for which $\hat{y} \geq 0.95$, will have a final class label of 1 else it is "0"

x	Actual y	$\hat{y}$	Pred. $\tilde{y}$ $\tau_1=0.95$
x_1	1	0.95	1
x_2	1	0.92	0
x_3	0	0.80	0
x_4	1	0.76	0
x_5	1	0.71	0

Here we are finding Associated class labels when $\tau_1=0.95$ (threshold=0.95) let's represent it with $\tilde{y}$

&, Corresponding $\tilde{y}$ given "$\tau_1=0.95$" we can compute "TPR" & "FPR"

Next we will take $\tau_2=0.92$ & we compute $\tilde{y}$ given $\tau_2=0.92$

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

Corresponding to $\tau_2=0.92$, we can even compute "TPR" & "FPR".

Similarly, we can keep changing the threshold values, so if we have "n" points in our dataset we could have "n" thresholds.

& if we have "n" thresholds $(\tau_1, \tau_2, \tau_3, \dots \tau_n)$

Now for each "τ" we can get corresponding "FPR & TPR"

$\tau_1, \tau_2, \tau_3, \tau_4, \dots \tau_n$

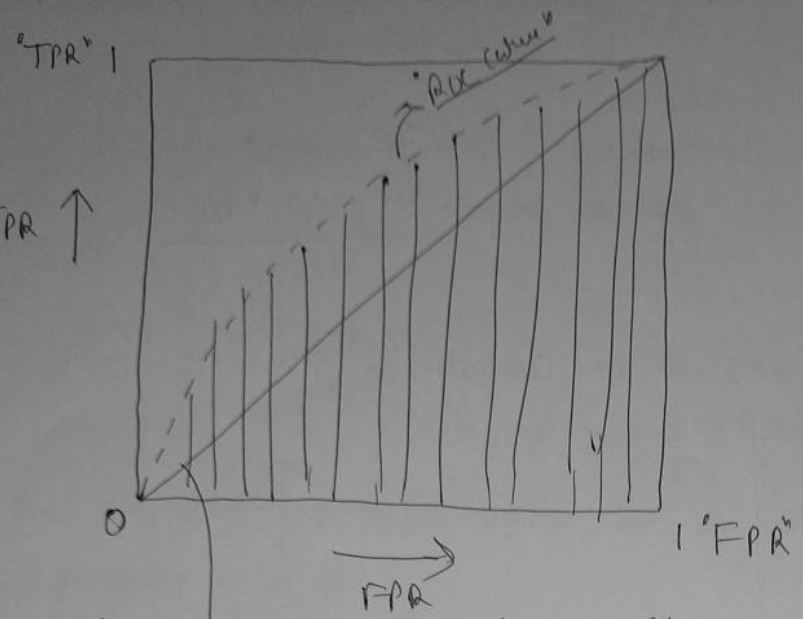
FPR_1 FPR_2 FPR_3 FPR_4 ..... FPR_n

TPR_1 TPR_2 TPR_3 TPR_4 TPR_n

Now we will draw a plot, where x-axis is "FPR" & y-axis is "TPR"

P.T.O

Value of "FPR & TPR" can lie b/w "0 & 1"



Now we will plot the (TPR, FPR) pairs

Typically, if our model is sensible, we could get a line/curve as shown above. This line/curve is called "Receiver operating characteristic" curve.

Let's draw another line as shown above, This straight line divides the entire region into two parts.

So area under this straight line is = "0.5"

Now "AUC" = It basically the total area under this whole "ROC curve"

This total area under the curve (AUC) can lie between 0 & 1

Higher the value of "AUC" .It is better.

[1 mean very good
0 mean terrible]

AUC is only useful for "binary classification" tasks.

There are extensions to "ROC" for multiclass classification also, but in general, we will typically use it for binary classification only

Some properties of AUC: (some downfalls of AUC)

① If we have 'imbalanced data' → AUC can be high even for a dumb model for simple reason

Note:- AUC can be impacted by "imbalanced data".

② If we look at AUC we will note that it does not care about the actual 'ŷ'. It is only about the sorting of ŷ.

So, AUC is not dependent on the ŷ score, it depends only on the ordering of ŷ

Let's say we have five datapoints in our dataset, & we have two models operating on this dataset. The corresponding ŷ's generated by these models 'M1' & 'M2' is as:-

"x"	"y"	M_1 ŷ	M_2 ŷ
x_1	1	0.95	0.2
x_2	1	0.98	0.1
x_3	0	0.80	0.08
x_4	1	0.76	0.07
x_5	1	0.71	0.06

For both these models 'M1 & M2' the AUC curve will be same because sorted order of 'ŷ' for both 'M1' & 'M2' is going to be same.

"AUC (M_1) = AUC (M_2)"

because it doesn't care about actual scores;

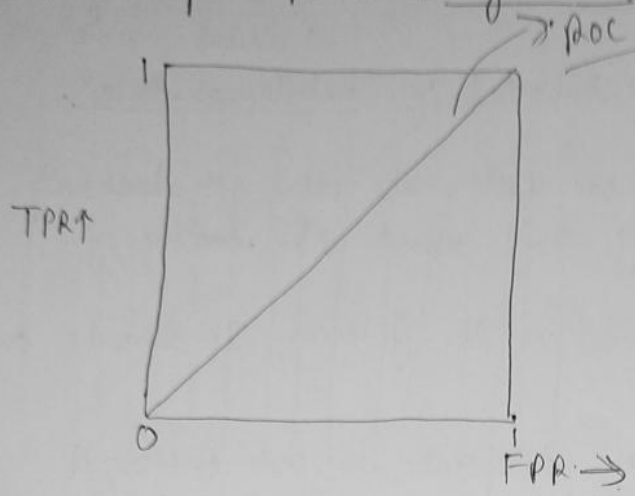
Note:- "AUC" is often used in ML a lot especially for "binary classification".

③ Suppose we have a model which randomly decides about the class, given a query point 'xq', the model randomly assigns a class label, such a model is called a random model.

Random model

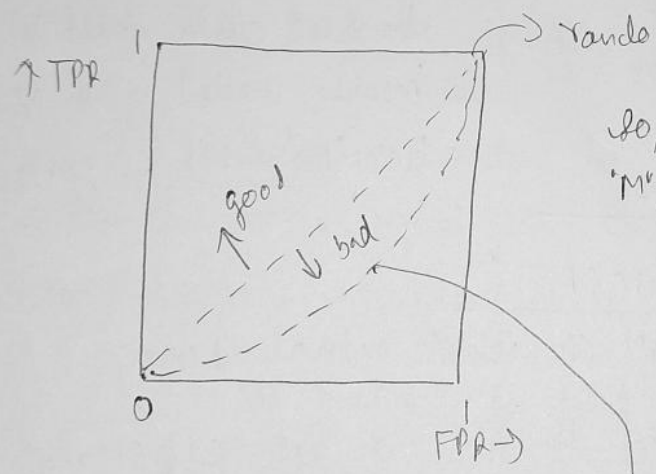
↳ x_q → "1 or 0"

So, if a model is random, not dumb then ROC for that random model will looks like a straight line



∴ 'AUC' of a random model will be exactly "0.5".

(4) Suppose, we train a model 'M' & we plotted the 'ROC'



So, we computed the AUC of model 'M', & let's say it is 0.2

'AUC(M) = 0.2'

Which is worst than random

0.2 basically mean that ROC curve looks like as shown above. Everything above the straight line, means it is a good model & everything above that straight line means it is a good model

∴ If we see 'AUC' value b/w 0.5 to 1, it means a good model.

& If we see 'AUC' value of '0.5', we say that model is not doing anything feasible, it is just a random model.

if AUC lies b/w "0.0 to 0.5" then this is what we do.

if $\hat{y_i} = 0$ (class label) simply change the class labels

$\hat{y_i} = 1$

[So, if model o/p is '0' change it to '1'
& if it is '1' change it to '0']

↓ simple swapping of class labels

Now if we do so,
then the modified model will have an AUC of

"1 − 0.2 = 0.8"

which is good

So, if you ever see a model with an AUC less than 0.5 simply switch the "class labels". & when we swap the class labels, we get an AUC = 1 − original AUC

In this case "1 − 0.2 = 0.8".

———— • ————