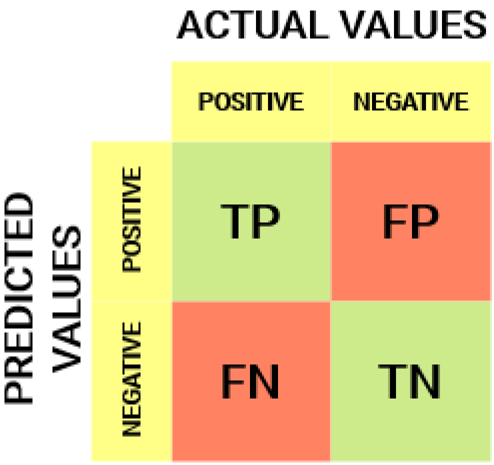
Module 3

What Is a Confusion Matrix?

A confusion matrix is a performance evaluation tool in machine learning, representing the accuracy of a classification model. It displays the number of true positives, true negatives, false positives, and false negatives. This matrix aids in analyzing model performance, identifying mis-classifications, and improving predictive accuracy.

A Confusion matrix is an N x N matrix used for evaluating the performance of a classification model, where N is the total number of target classes. The matrix compares the actual target values with those predicted by the machine learning model. This gives us a holistic view of how well our classification model is performing and what kinds of errors it is making.

For a binary classification problem, we would have a 2 x 2 matrix, as shown below, with 4 values:



		ACTUAL VALUES	
		POSITIVE	NEGATIVE
PREDICTED VALUES	POSITIVE	TP	FP
	NEGATIVE	FN	TN

Let's decipher the matrix:

- The target variable has two values: **Positive** or **Negative**

- The **columns** represent the **actual values** of the target variable
- The **rows** represent the **predicted values** of the target variable

But wait – what's TP, FP, FN, and TN here? That's the crucial part of a confusion matrix. Let's understand each term below.

Important Terms in a Confusion Matrix

True Positive (TP)

- The predicted value matches the actual value, or the predicted class matches the actual class.
- The actual value was positive, and the model predicted a positive value.

True Negative (TN)

- The predicted value matches the actual value, or the predicted class matches the actual class.
- The actual value was negative, and the model predicted a negative value.

False Positive (FP) – Type I Error

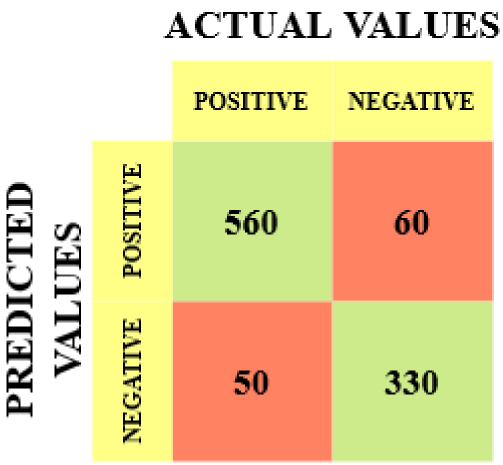
- The predicted value was falsely predicted.
- The actual value was negative, but the model predicted a positive value.
- Also known as the type I error.

False Negative (FN) – Type II Error

- The predicted value was falsely predicted.

- The actual value was positive, but the model predicted a negative value.
- Also known as the type II error.

Let me give you an example to better understand this. Suppose we had a classification dataset with 1000 data points. We fit a classifier (say logistic regression or decision tree) on it and get the below confusion matrix:



		ACTUAL VALUES	
		POSITIVE	NEGATIVE
PREDICTED VALUES	POSITIVE	560	60
	NEGATIVE	50	330

The different values of the Confusion matrix would be as follows:

- True Positive (TP) = 560, meaning the model correctly classified 560 positive class data points.
- True Negative (TN) = 330, meaning the model correctly classified 330 negative class data points.
- False Positive (FP) = 60, meaning the model incorrectly classified 60 negative class data points as belonging to the positive class.
- False Negative (FN) = 50, meaning the model incorrectly classified 50 positive class data points as belonging to the negative class.

This turned out to be a pretty decent classifier for our dataset, considering the relatively larger number of true positive and true negative values.

Remember the Type I and Type II errors. Interviewers love to ask the difference between these two! You can prepare for all this better from our Machine Learning Course Online.

Why Do We Need a Confusion Matrix?

Before we answer this question, let's think about a hypothetical classification problem.

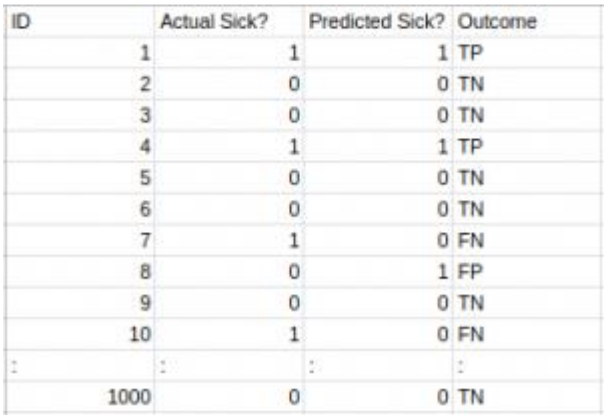
Let's say you want to predict how many people are infected with a contagious virus in times before they show the symptoms and isolate them from the healthy population (ringing any bells, yet?). The two values for our target variable would be Sick and Not Sick.

Now, you must be wondering why we need a confusion matrix when we have our all-weather friend – Accuracy. Well, let's see where classification accuracy falters.

Our dataset is an example of an imbalanced dataset. There are 947 data points for the negative class and 3 data points for the positive class. This is how we'll calculate the accuracy:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

Let's see how our model performed:



ID	Actual Sick?	Predicted Sick?	Outcome
1	1	1	TP
2	0	0	TN
3	0	0	TN
4	1	1	TP
5	0	0	TN
6	0	0	TN
7	1	0	FN
8	0	1	FP
9	0	0	TN
10	1	0	FN
:	:	:	:
1000	0	0	TN

The total outcome values are:

TP = 30, TN = 930, FP = 30, FN = 10

So, the accuracy of our model turns out to be:

$$Accuracy = \frac{30 + 930}{30 + 30 + 930 + 10} = 0.96$$

96%! Not bad!

But it gives the wrong idea about the result. Think about it.

Our model is saying, "I can predict sick people 96% of the time". However, it is doing the opposite. It predicts the people who will not get sick with 96% accuracy while the sick are spreading the virus!

Do you think this is a correct metric for our model, given the seriousness of the issue? Shouldn't we be measuring how many positive cases we can predict correctly to arrest the spread of the contagious virus? Or maybe, out of the correct predictions, how many are positive cases to check the reliability of our model?

This is where we come across the dual concept of Precision and Recall.

Precision vs. Recall

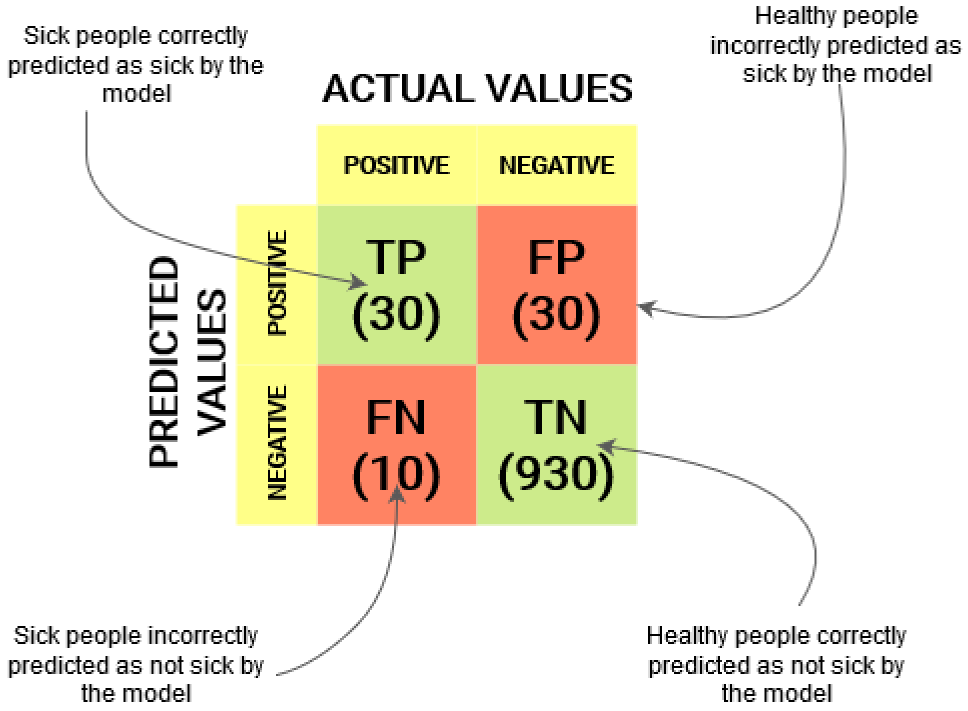
Precision tells us how many of the correctly predicted cases actually turned out to be positive.

Here's how to calculate Precision: $Precision = \frac{TP}{TP + FP}$

This would determine whether our model is reliable or not.

Recall tells us how many of the actual positive cases we were able to predict correctly with our model.

And here's how we can calculate Recall:


Sick people correctly predicted as sick by the model
Healthy people incorrectly predicted as sick by the model
ACTUAL VALUES
PREDICTED VALUES
Sick people incorrectly predicted as not sick by the model
Healthy people correctly predicted as not sick by the model

	POSITIVE	NEGATIVE
POSITIVE	TP (30)	FP (30)
NEGATIVE	FN (10)	TN (930)

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

We can easily calculate Precision and Recall for our model by plugging in the values into the above questions:

$$Precision = \frac{30}{30+30} = 0.5$$

$$Recall = \frac{30}{30+10} = 0.75$$

50% percent of the correctly predicted cases turned out to be positive cases. Whereas 75% of the positives were successfully predicted by our model. Awesome!

Precision is a useful metric in cases where False Positive is a higher concern than False Negatives.

Precision is important in music or video recommendation systems, e-commerce websites, etc. Wrong results could lead to customer churn and be harmful to the business.

Recall is a useful metric in cases where False Negative trumps False Positive.

Recall is important in medical cases where it doesn't matter whether we raise a false alarm, but the actual positive cases should not go undetected!

In our example, Recall would be a better metric because we don't want to accidentally discharge an infected person and let them mix with the healthy population, thereby spreading the contagious virus. Now you can understand why accuracy was a bad metric for our model.

But there will be cases where there is no clear distinction between whether Precision is more important or Recall. What should we do in those cases? We combine them!

What Is F1-Score?

In practice, when we try to increase the precision of our model, the recall goes down, and vice-versa. The F1-score captures both the trends in a single value:

$$F1 - score = \frac{2}{\frac{1}{Recall} + \frac{1}{Precision}}$$

F1-score is a harmonic mean of Precision and Recall, and so it gives a combined idea about these two metrics. It is maximum when Precision is equal to Recall.

But there is a catch here. The interpretability of the F1-score is poor. This means that we don't know what our classifier is maximizing – precision or recall. So, we use it in combination with other evaluation metrics, giving us a complete picture of the result.