- **Accuracy**

$$Accuracy = \frac{\#\ Correctly\ Classified\ Points}{\#Total\ Points}$$

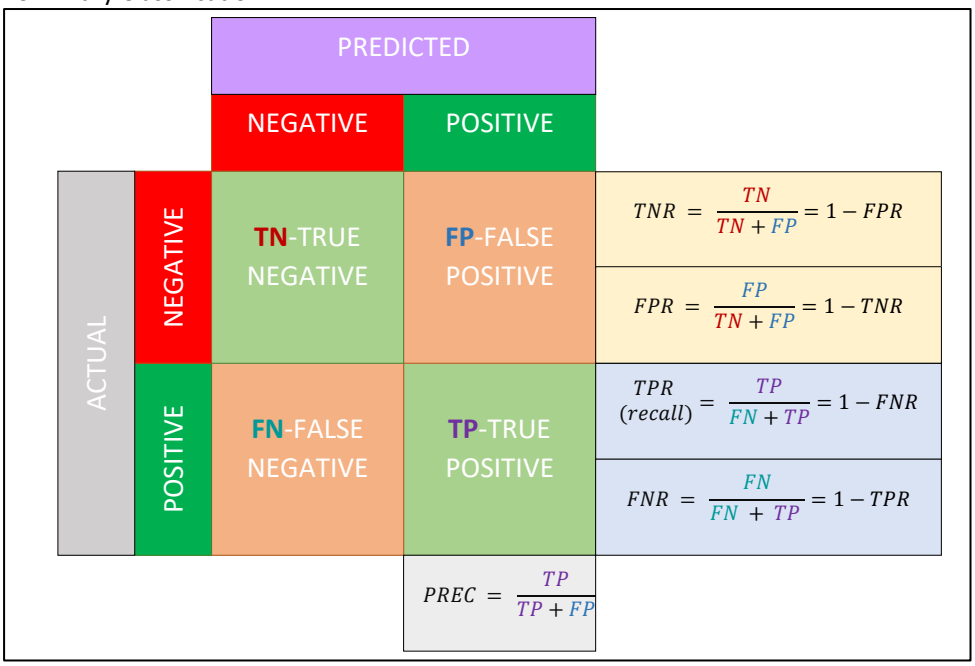
  - Accuracy lies between 0 and 1 (0 is bad and 1 is best)
  - In case of imbalanced dataset, accuracy is not a good metric. For example, if we have a dataset containing 90% positive points and 10% negative points whenever we run the model on this dataset, it will always give accuracy ≥ 90% as there are very few numbers of negative points.

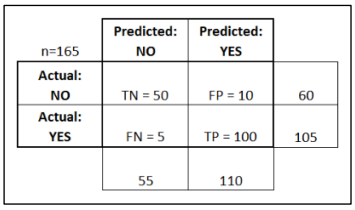
Class	Distribution	Prediction	Accuracy
Positive	90%	100%	90%
Negative	10%	0%	0%

  - If a model returns probability score. Let's say, there are two models M1 and M2 for classification. If a point p1 belongs to positive with the probability score 0.95 in M1 and it belongs to positive in M2 with probability score 0.65. Here we know that model M1 is better than M2.
  - Accuracy can say that an inferior model is working similar to a powerful model. thus, accuracy cannot give an idea whether a model is inferior or powerful

Data Points		Model M1		Model M2	
x_i	y_i	[Prob (y_q = 1)]	Prediction	[Prob (y_q = 1)]	Prediction
x_1	1	0.9	1	0.6	1
x_2	1	0.8	1	0.65	1
x_3	0	0.1	0	0.45	0
x_4	0	0.15	0	0.48	0

- **Confusion matrix, TPR, FPR, FNR, TNR**
  - For Binary Classification





Example-

⇨ $TPR\ (recall) = \frac{100}{105} = 0.95$ When it's actually yes, how often does it predict yes?

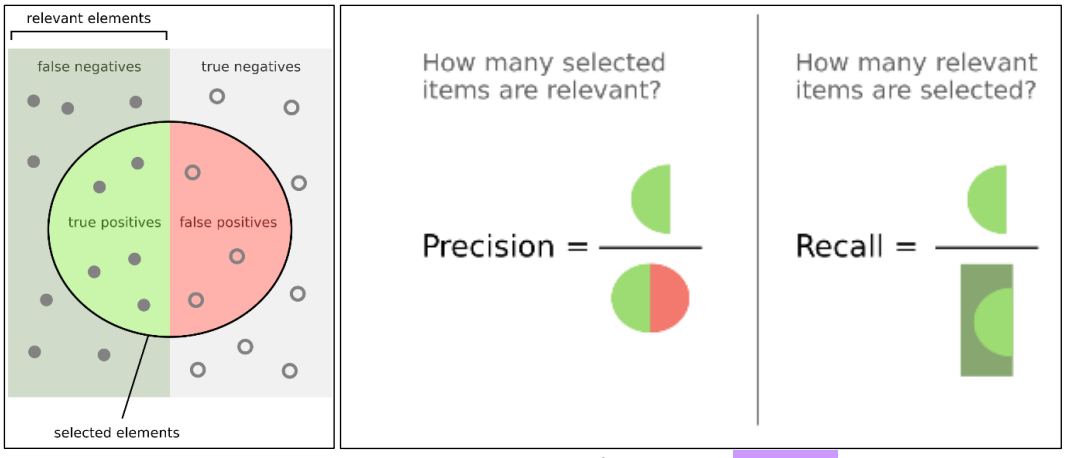
⇨ $FNR = \frac{5}{105} = 0.05$ When it's actually yes, how often does it predict no?

⇨ $FPR = \frac{10}{60} = 0.166$ When it's actually no, how often does it predict yes?

⇨ $TNR = \frac{50}{60} = 0.833$ When it's actually no, how often does it predict no?

⇨ $PREC = \frac{100}{110} = 0.91$ When it predicts yes, how often is it correct?

⇨ $F1\ Score = 2 * \frac{PREC*REC}{PREC+REC} = \frac{TP}{TP+\frac{1}{2}(FP+FN)} = 2 * \frac{0.91*0.95}{0.91+0.95} = 0.92957$

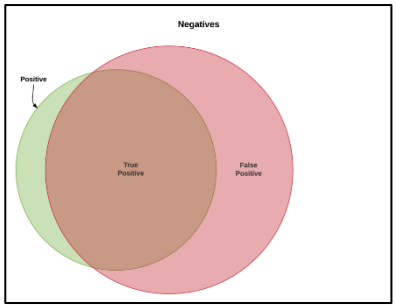


- Precision tells how precise is the model such that out of all positive predicted points; how many are actually positive
- Recall tells how much it can recall from the model i.e. out of all actual positive points; how many are predicted positive?
- In medical domain, we want a high recall. i.e., We don't want any patient who actually has a disease the model say that he/she doesn't have it.
- But we can have low precision i.e., the patient actually doesn't have a disease but it's okay to say that he/she has. That is not life threatening.
- We use the probability threshold to maintain the precision-recall trade-off, say, by setting a lower threshold (earlier 0.5) to 0.4, now we are going to have more points predicted in the positive class. However, now we increase recall, decrease precision.

- **More about precision & Recall**
  - Suppose we are predicting whether we should quarantine a person based on his sickness. The positive class is having flu & negative class is not having flu
  - If you are quarantining patients that have the flu an algorithm with a high recall would be able to quarantine most of the people who have the flu.
  - High recall would quarantine every single sick person. It asks: "How close did we come to quarantining every person who had the flu?"
  - An algorithm with high precision would quarantine a group of people and most of the people it quarantined would have the flu.
  - Precision does not care how many sick people we missed. It only asks: "Did everyone we quarantined have the flu?"
  - CASE1- low recall, low precision

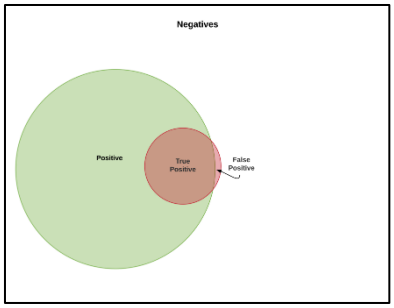
- The algorithm is probably underfitting the data or the wrong algorithm for this task.
- Algorithm isn't good at finding the people who have the flu, small green-red overlap (low recall),
- Among the stuff it chose, the red circle, it picked tons of people who didn't have the flu (low precision).

  - CASE2- high recall, low precision



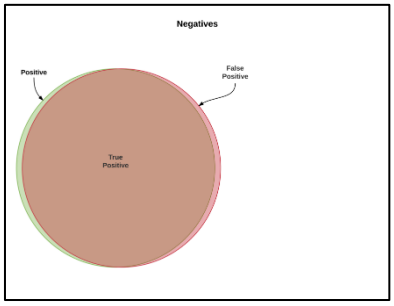
- The model is getting most of the things it is looking for, big green-red overlap, (high recall)
- But it's also grabbing a lot of stuff it's not supposed to, big red-white space overlap (low precision).

- CASE3- low recall, high precision



  - The model is missing most of the things it's looking for
  - but among the things it has selected most of them are correct. The green-red overlap is small (low recall).
  - red-white overlap is also small (high precision).

- CASE3- high recall, high precision

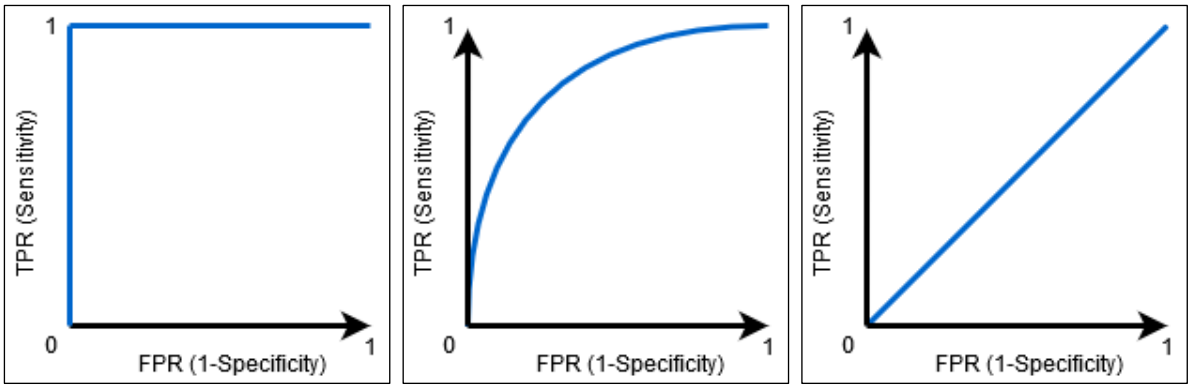


  - The predictions our algorithm made and the actual number of people who have the flu has a high overlap (high green-red overlap, high recall).
  - We also have very few people that are not sick (red-white space) that our algorithm said had the flu (high precision).
  - This is the holy grail right here, an algorithm that can find all the sick people without also grabbing too many healthy people.

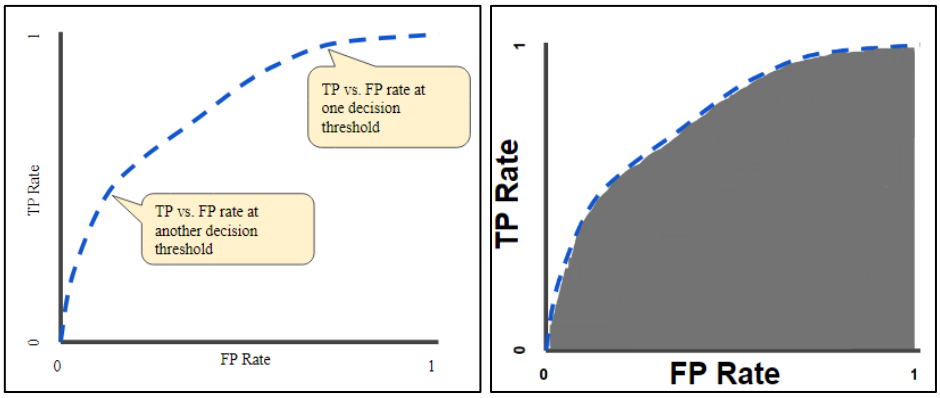
- Example
  - The justice system tries to have a high precision and thus ends up having low recall. Because of a lack of evidence many criminals will not be convicted (low recall) but at least the government can be sure that the people it is imprisoning are indeed guilty (high precision).
  - Inspections for airplane Engines have a high recall and low precision. You want to make sure you get every single defective engine, it's ok to grab many working engines as well but it is critical that every defective engine get caught, so an airplane engine does not fail mid-flight.
- Precision and Recall are not opposites. It is possible to have both but it can be hard to produce a model that does this. In practice you can usually turn knobs on your model to make it more or less relaxed in its predictions, which will create a trade-off between precision and recall.
- By making fewer positive predictions you can get higher precision
- By making more positive predictions you can get your higher recall.
- But using two values, we cannot determine if one algorithm is superior to another.
- F1 score is the harmonic mean of the precision and recall. The highest possible value of an F-score is 1.0, indicating perfect precision and recall, and the lowest possible value is 0, if either the precision or the recall is zero.

$$F1\ Score\ =\ 2 * \frac{PREC * REC}{PREC + REC} = \frac{TP}{TP + \frac{1}{2}(FP + FN)}$$

- **Receiver Operating Characteristic Curve (ROC) curve and AUC**
  - ROC is a probability curve and AUC represent the degree or measure of separability. It tells how much the model is capable of distinguishing between classes. Higher the AUC, the better the model is at accurate prediction.



- When AUC = 1, then the classifier is able to perfectly distinguish between all the Positive and the Negative class points correctly.
- If AUC = 0, then the classifier is predicting all Negatives as Positives, and all Positives as Negatives.
- When 0.5<AUC<1, there is a high chance that the classifier will be able to distinguish the positive class values from the negative class values. This is because the classifier is able to detect more numbers of True positives and True negatives than False negatives and False positives. AUC can be high even for a dumb model when the data set is imbalanced.
- When AUC=0.5, then the classifier is not able to distinguish between Positive and Negative class points. either the classifier is predicting random class or constant class for all the data points.
- An ROC curve plots TPR vs. FPR at different classification thresholds. Lowering the classification threshold classifies more items as positive, thus increasing both False Positives and True Positives. AUC is "Area under the ROC Curve." That is, AUC measures the entire two-dimensional area underneath the entire ROC curve from (0,0) to (1,1).



- AUC provides an aggregate measure of performance across all possible classification thresholds.
- One way of interpreting AUC is as the probability that the model ranks a random positive example more highly than a random negative example.
- AUC is not dependent on the predicted values, rather it considers the ordering; if two models give same order of predicted values then AUC will be same for both the models;

- **Log Loss**
  - Binary classification problem: Log Loss uses probability scores

$$Log\ Loss = \frac{-1}{n}\sum y_i * \log p_i + (1 - y_i) * \log(1 - p_i)$$

- **Coefficient of determination (R^2)**
  - In regression problems, we will be predicting $\hat{y}$ whereas the actual class will be y_i
  - mean of the observed data is $\bar{y} = \sum_{i=1}^{n} y_i$
  - Residual error, $e_i = y_i - \hat{y}$
  - Total Sum of Squares $TSS = \sum_i (y_i - \bar{y})^2$
  - Residual Sum of Squares $RSS = \sum_i (y_i - \hat{y})^2 = \sum_i (e_i)^2$
  - $R^2 = 1 - \frac{RSS}{TSS}$
  - If the predicted values exactly match the observed values, which results in RSS = 0 and $R^2 = 1$
  - A baseline model, which always predicts $\bar{y}$, will have R^2=0.
  - Models that have worse predictions than baseline will have a negative R^2

R^2	RSS	Case
1	0	Best Case (Perfect Predictions)
0 < 1	RSS < TSS	(R^2) of the data fit the regression model.
0	RSS = TSS	Base Model (Always Predicts $\bar{y}$)
< 0	RSS > TSS	worse than Base Model

  - If Median & MAD of errors is small, we know that our model is good
  - We can compare two models by plotting pdf & cdf of errors