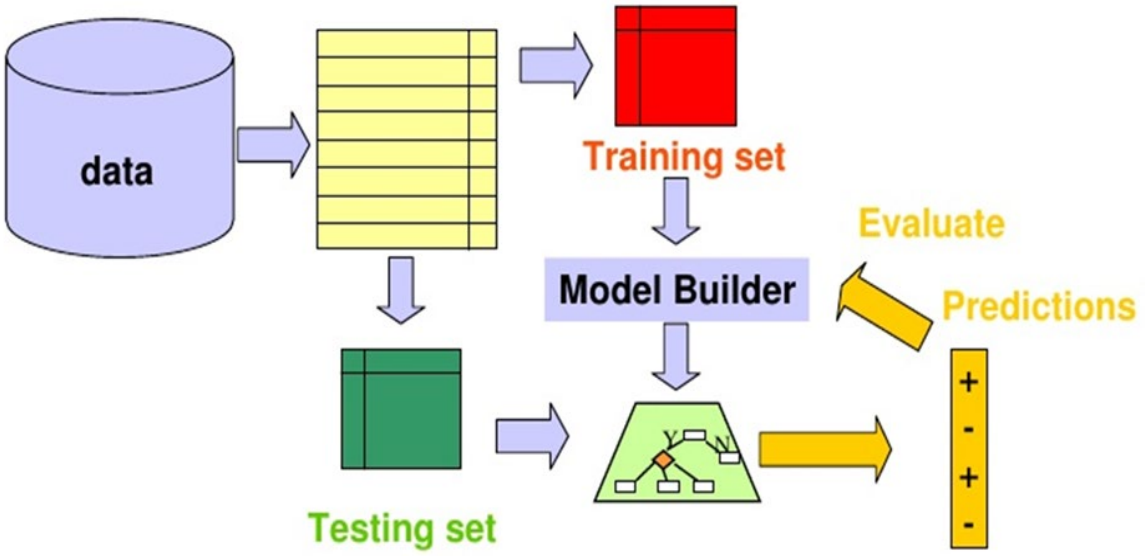
7. Performance Evaluation for Classifiers

In supervised learning, a classification model is trained using the labelled data. How can we understand the performance of the model? Next, we introduce a few model evaluation procedures and metrics.

7.1 Evaluation Procedure

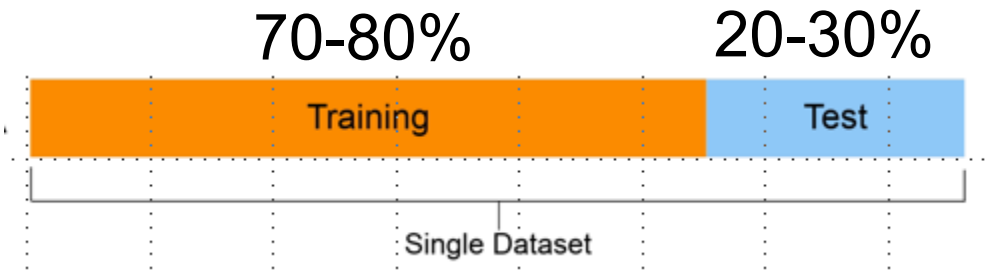
7.1.1 Holdout method

In this method a part/subset of the available data is held back (that is how the name holdout originates) for evaluation of the model. This subset of the data is used as the test data for evaluating the performance of a trained classifier.

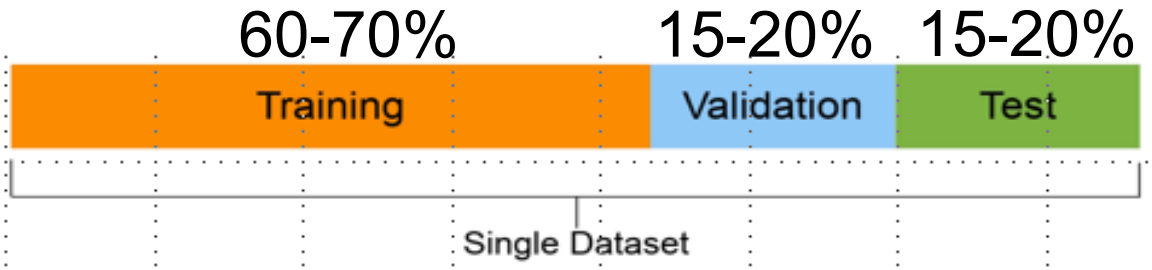


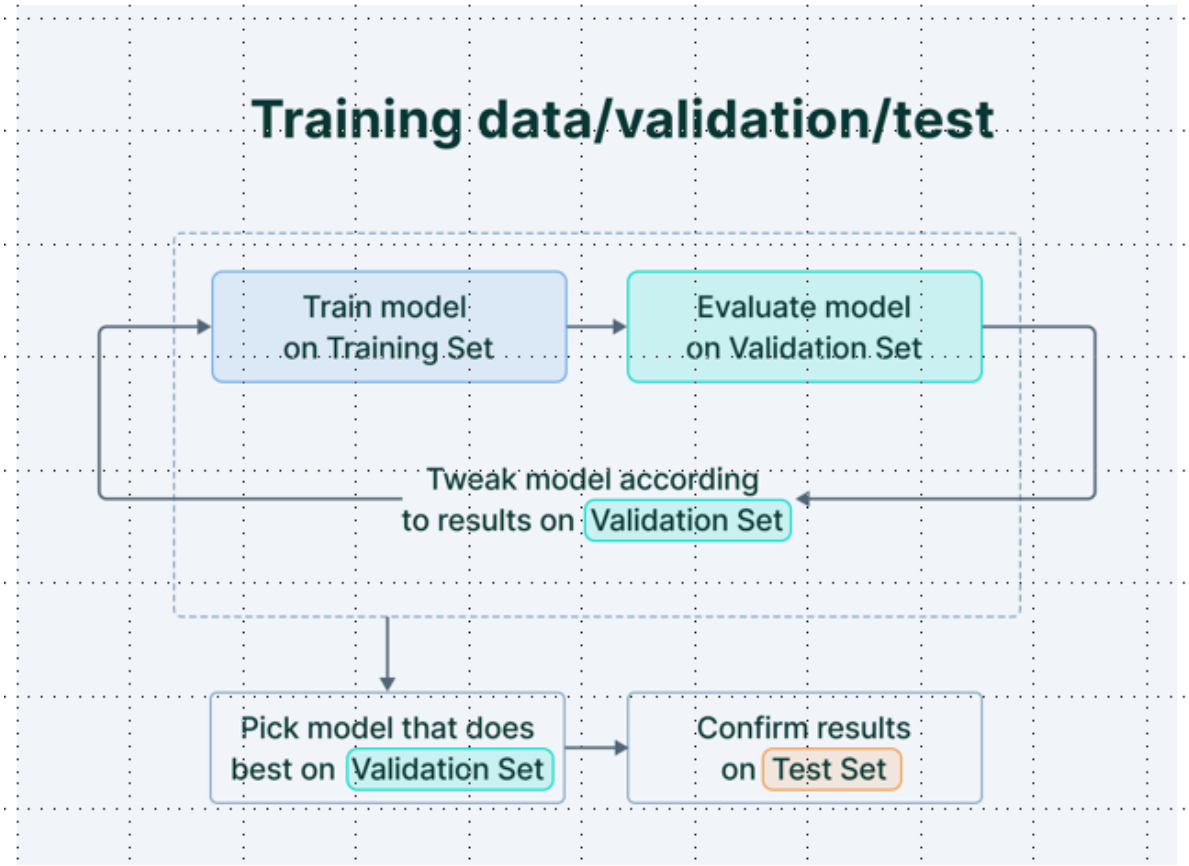
Once the model is trained using the training data, the labels of the test data are predicted using the trained model. Then the predicted value is compared with the actual value of the label. This is possible because the test data is a part of the available data with known labels. The performance of the model is measured by various metrics, which will be introduced later.

Usually, 70%–80% of the available data (randomly sampled) is used for classifier training, and the remaining 20%–30% is used as test data.



Quite often, the available data is partitioned into 3 portions – **a training and a test data, and a validation data**. The validation data is used for measuring the model performance during training stage. After the model is finalized by the training and validation data, the test data is used only for once, to measure and report the final performance of the model.


60-70%
15-20%
15-20%
Training
Validation
Test
Single Dataset


Training data/validation/test
Train model on Training Set
Evaluate model on Validation Set
Tweak model according to results on Validation Set
Pick model that does best on Validation Set
Confirm results on Test Set

An obvious problem in this method is that the division of data of different classes into the training and test data may not be proportional. This situation is worse if the overall percentage of data related to certain classes is much less than other classes (i.e. class imbalance problem). This may happen despite the fact that random sampling is employed for test data selection.

This problem can be addressed to some extent by applying **stratified random sampling** in place of random sampling. In stratified random sampling, random sampling is conducted on each individual class. For example, 70% and 30% of the samples in each class are randomly selected as training and test data. This ensures that the generated random partitions have equal proportions of each class.

7.1.2 Repeated holdout method

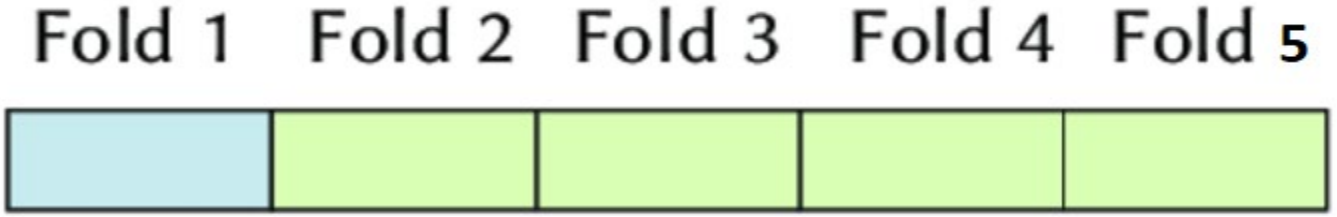
A special variant of the holdout method, called repeated holdout, is often employed to ensure the randomness of the composed training and test data sets.

In repeated holdout, several random holdouts are used to measure the model performance. In the end, the average of all performances is taken.

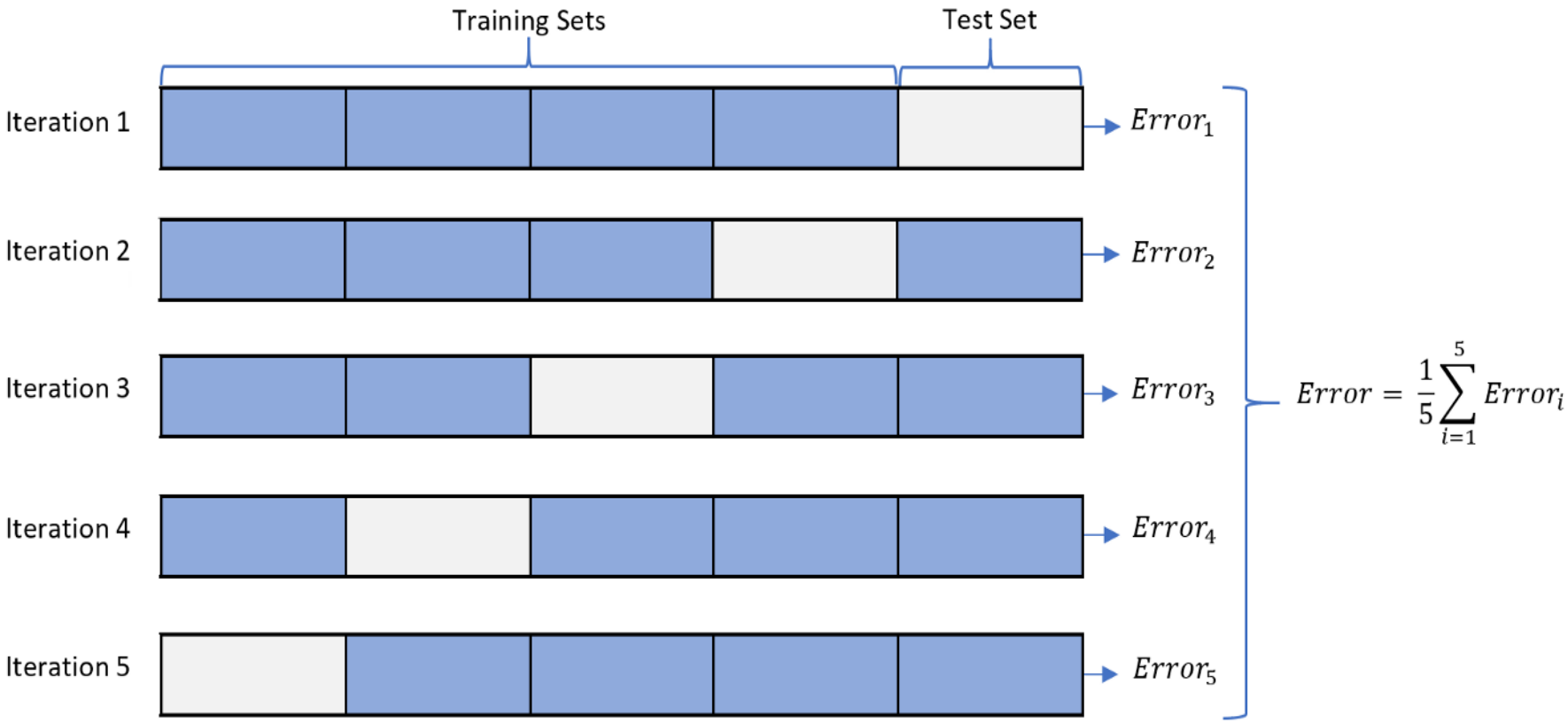
As multiple holdouts have been drawn, the training and test data (and also validation data, in case it is drawn) are more likely to contain representative data from all classes and resemble the original input data closely.

7.1.3 K-fold cross-validation method

The process of repeated holdout is the basis of *k*-fold cross-validation technique. In *k*-fold cross-validation, the data set is divided into *k*-completely distinct or non-overlapping random partitions called folds:



Each of the k folds (i.e. 5 in the above example) is given an opportunity to be used as a held back test set, whilst all other folds collectively are used as a training dataset. A total of k models are fit and evaluated on the k hold-out test sets and the mean performance is reported as illustrated below.



5-fold cross validation

7.1.4 Leave-one-out cross-validation (LOOCV)

An extreme case of k-fold cross-validation is to set $k = N$, which is the total number of samples. In this extreme case, in each fold, only one sample is used as test data, the remaining $N - 1$ samples are used as training data. This extreme case is called **Leave-one-out cross-validation.**

This is done to maximize the count of data used to train the model. It is obvious that the number of iterations (i.e. folds) for which it has to run is equal to the total number of samples in the data set. Hence, it is computationally very expensive and is used only when the number of samples is small.

7.1.5 Repeated k-fold cross-validation

The estimate of model performance via $k-$fold cross-validation can be noisy. This means that each time the procedure is run, a different split of the dataset into k folds is used, and in turn, a different mean estimate of model performance is obtained. The amount of difference in the estimated performance from one run of k-fold cross-validation to another is dependent on the difference of training-test data division.

On solution to the problem is to repeat the k-fold cross-validation process multiple times and report the mean performance across all folds and all repeats. This approach is generally referred to as **repeated $k-$fold cross-validation**.

Procedure of Repeated k-Fold Cross-Validation

(1) Set $repeat = 1$

(2) Perform k-fold cross validation and obtain the evaluation performance (averaged over k folds) of the current repeat.

(3) Let $repeat = repeat + 1$ and reshuffle the data

(4) Go to Step (2) until the total number of repeats reach to a pre-set value such as 50.

(5) Calculate the average performance over all the repeats

7.2 Performance Evaluation Metrics

For a two-class problem, we assume one class is positive, and another is negative. Usually, the class of interest is named as positive, for example, “cancer” is named positive, while “normal” is named as negative. The classification result of a sample may have the 4 scenarios:

(i) **True positive (TP):** a sample of positive class is correctly classified into positive class

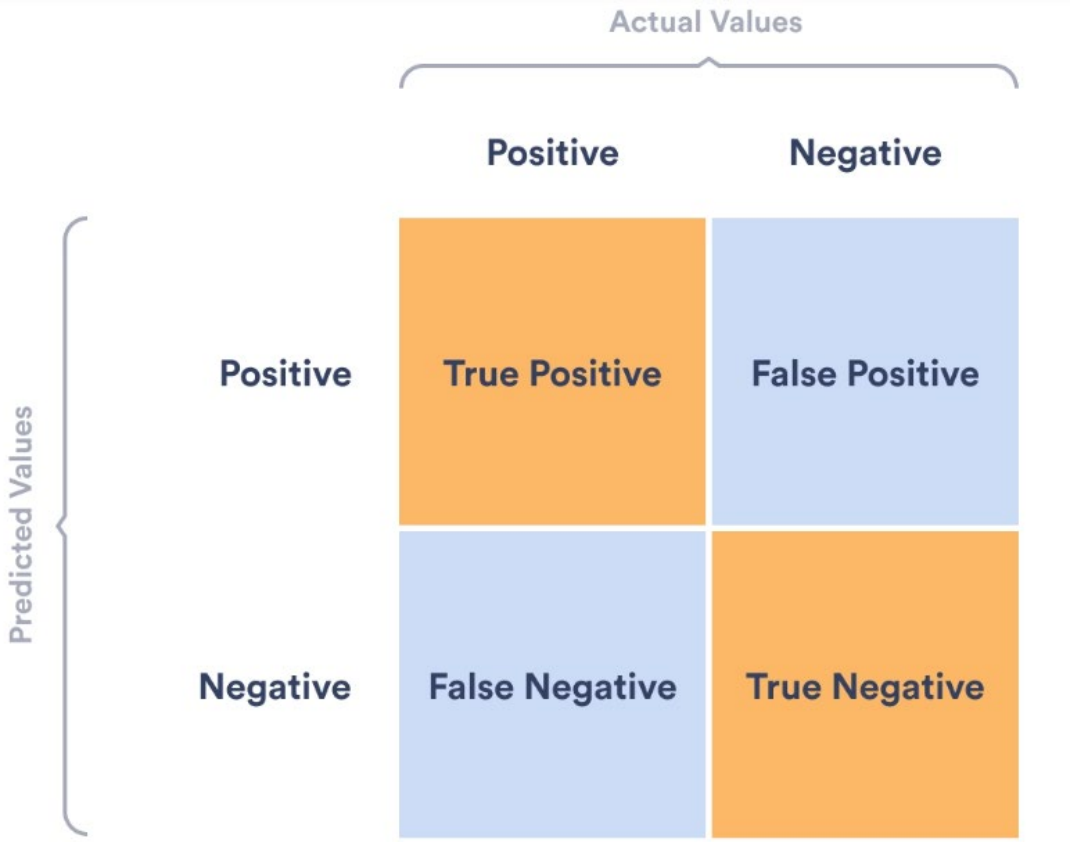
(ii) **False positive (FP):** a sample of negative class is wrongly classified into positive class

(iii) **True negative (TN):** a sample of negative class is correctly classified into negative class

(iv) **False negative (FN):** a sample of positive class is wrongly classified into negative class

7.2.1 Confusion matrix

A confusion matrix is a summary of classification results as shown below:



An example:

	Actual Positive	Actual Negative
Predicted Positive	85	4
Predicted Negative	2	9

7.2.2 Accuracy

Accuracy is the proportion of correctly classified samples:

$$Acc = \frac{TP + TN}{TP + FP + TN + FN} = \frac{85 + 9}{100} = 0.94 = 94\%$$

7.2.3 Error rate

Error rate is the proportion of wrongly classified samples:

$$ER = \frac{FP + FN}{TP + FP + TN + FN} = \frac{4 + 2}{100} = 0.06 = 6\%$$

7.2.4 Sensitivity and Specificity

Sensitivity of a classifier measures the proportion of positive samples that have been correctly classified. It is measured as:

$$Sensitivity = \frac{TP}{TP + FN} = \frac{85}{85 + 2} = 97.7\%$$

Specificity of a classifier measures the proportion of negative samples that have been correctly classified. It is measured as:

$$Specificity = \frac{TN}{TN + FP} = \frac{9}{9 + 4} = 69.2\%$$

7.2.5 Precision and Recall

Precision gives the proportion of positive predictions which are truly positive:

$$Precision = \frac{TP}{TP + FP} = \frac{85}{85 + 4} = 95.5\%$$

Recall measures the proportion of correct prediction of positives to the total number of positives:

$$Recall = \frac{TP}{TP + FN} = \frac{85}{85 + 2} = 97.7\%$$

7.2.6 F-score

F-score is measure of classifier performance that combines the precision and recall:

$$F = \frac{2 \times Precision \times Recall}{Precision + Recall} = \frac{2 \times 0.955 \times 0.977}{0.955 + 0.977} = 96.6\%$$

7.2.7 Receiver operating characteristic (ROC) curve

In the classification problems, we use discriminant function to decide which class the sample belong to. In the Bayes decision rule, this discriminant function is the posterior probability. For a two-class problem:

$$\text{Decide } \omega_1 \text{ if } P(\omega_1|x) > P(\omega_2|x);$$

$$\text{Decide } \omega_2 \text{ if } P(\omega_2|x) > P(\omega_1|x).$$

In this decision rule, we implicitly assume the threshold is 0.5. If the posterior probability belonging to a class is higher than 0.5, then the sample is assigned to the class. But in some applications, for example, the diagnosis of a patient into “cancer” or “normal”, a different threshold instead of 0.5 might be a more prudent choice.

Setting different thresholds for classifying positive class will inevitably change the 4 metrics in the confusion matrix: TP, FP, TN, FN, and hence the other evaluation metrics of the classification model. One of these thresholds will probably give a better result than the others, depending on whether we are aiming to lower the number of False Negatives or False Positives.

We can generate different confusion matrices and compare the various metrics. But that would not be a prudent thing to do. Instead, what we can do is generate a plot between some of these metrics so that we can easily visualize which threshold gives us a better result.

The ROC curve solves just this problem!

Receiver Operating Characteristic (ROC) visualizes the efficiency of a classification model in the detection of true positives while avoiding the occurrence of false positives.

We define the following measures:

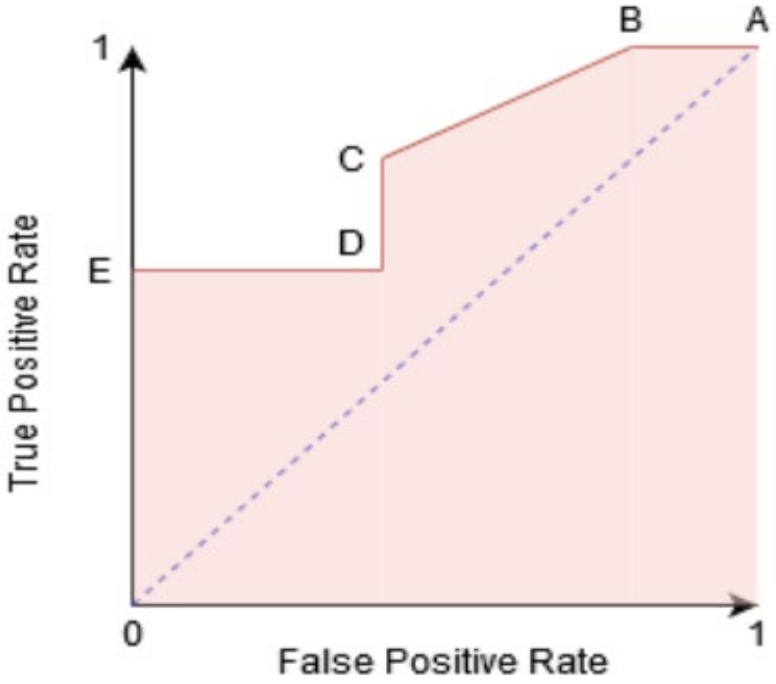
True Positive Rate (TPR)

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

False Positive Rate (FPR)

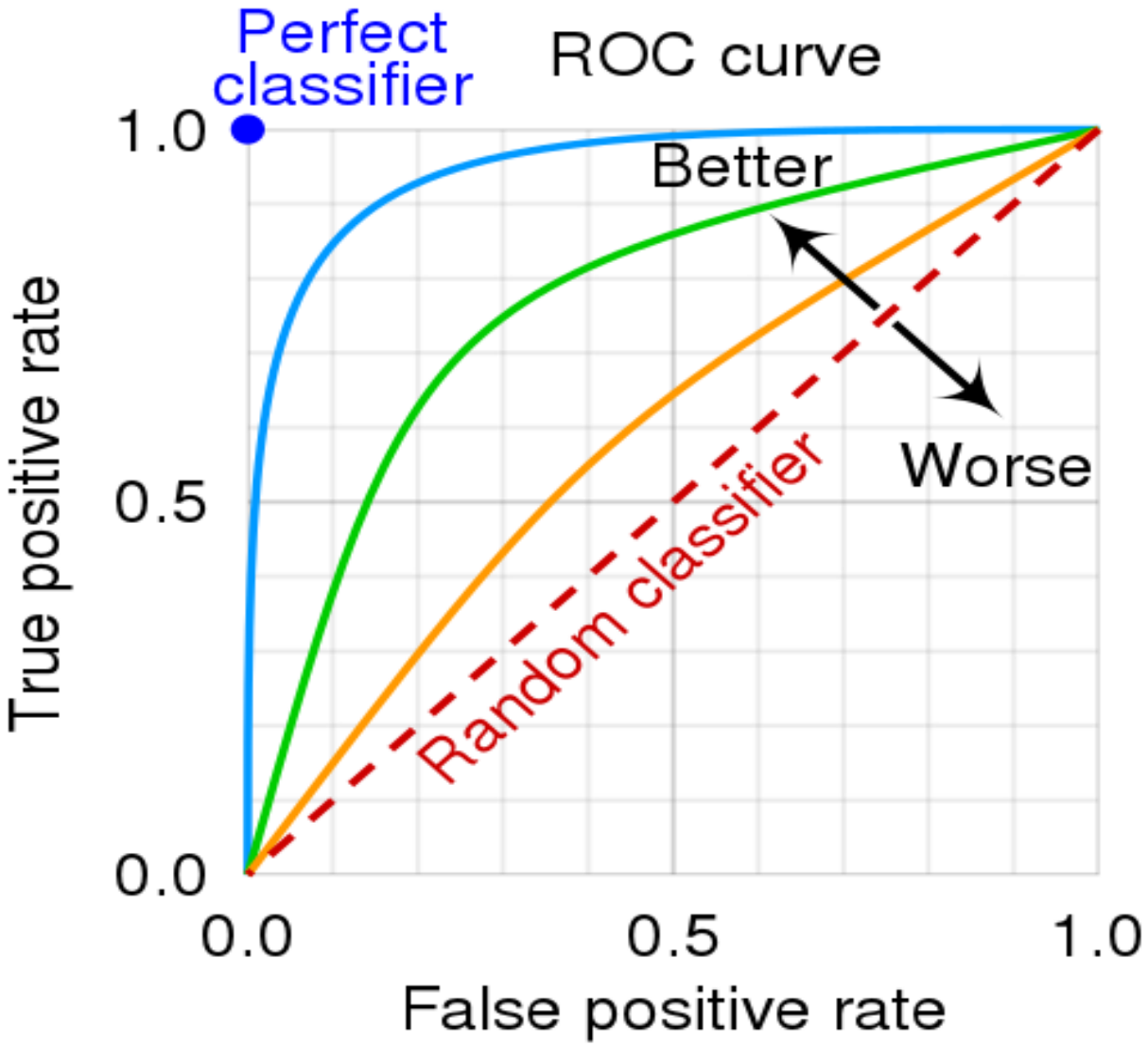
$$FPR = \frac{FP}{FP + TN} = 1 - specificity$$

In the ROC curve, the false positive rate is plotted (in the horizontal axis) against true positive rate (in the vertical axis) at different classification thresholds.



The **Area Under the Curve (AUC)**, i.e. the shaded part in the above figure, is the measure of the ability of a classifier to distinguish between classes.

1) When **AUC=1**, the classifier is able to perfectly distinguish between all the Positive and the Negative class points correctly;
2) When **AUC=0**, the classifier would be predicting all Negatives as Positives, and all Positives as Negatives.
3) When **AUC=0.5**, then the classifier is not able to distinguish between Positive and Negative class points, meaning the classifier is predicting random class.
4) When **0.5<AUC<1**, there is a high chance that the classifier will be able to distinguish the positive class from the negative class, meaning the classifier is able to detect more numbers of True positives and True negatives than False negatives and False positives.


Perfect classifier
ROC curve
1.0
Better
0.5
Worse
Random classifier
True positive rate
0.0
0.0
0.5
1.0
False positive rate