Lab-14: Evaluation Metrics to evaluate machine learning algorithm

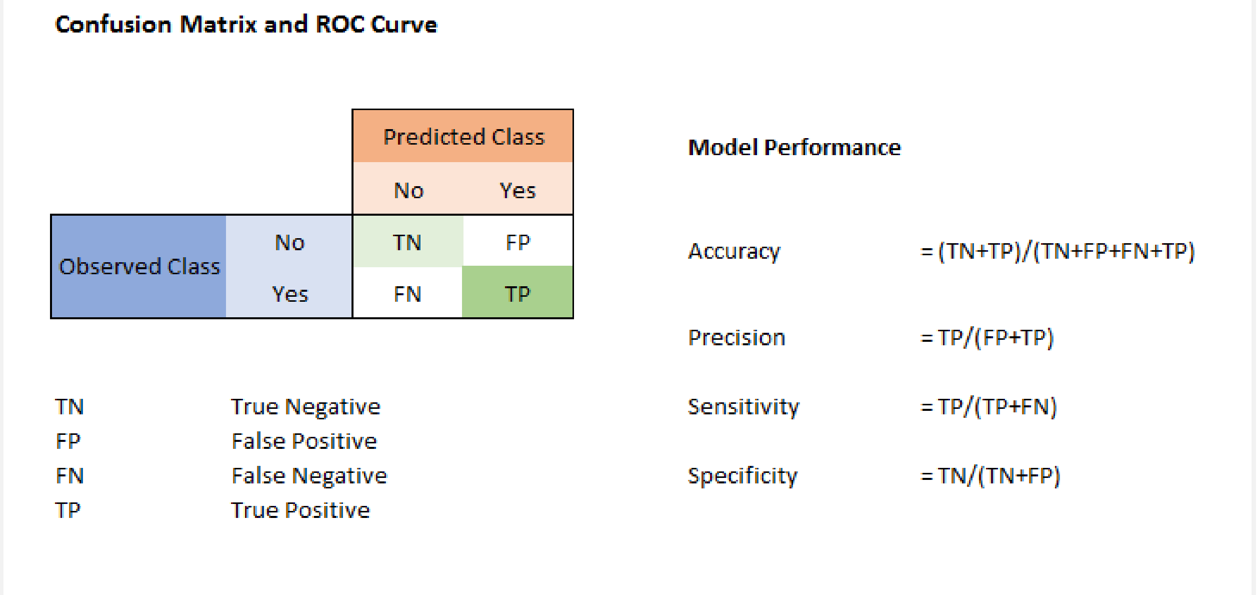
14.1 Objectives:

1. To learn evaluation metrics to evaluate machine learning algorithms

14.2 Confusion Matrix

Confusion Matrix as the name suggests gives us a matrix as output and describes the complete performance of the model.

Lets assume we have a binary classification problem. We have some samples belonging to two classes : YES or NO.



Confusion Matrix and ROC Curve

		Predicted Class	
		No	Yes
Observed Class	No	TN	FP
	Yes	FN	TP

TN	True Negative
FP	False Positive
FN	False Negative
TP	True Positive

Model Performance

Accuracy	= (TN+TP)/(TN+FP+FN+TP)
Precision	= TP/(FP+TP)
Sensitivity	= TP/(TP+FN)
Specificity	= TN/(TN+FP)

Confusion Matrix

There are 4 important terms :

- **True Positives** : The cases in which we predicted YES and the actual output was also YES.
- **True Negatives** : The cases in which we predicted NO and the actual output was NO.
- **False Positives** : The cases in which we predicted YES and the actual output was NO.
- **False Negatives** : The cases in which we predicted NO and the actual output was YES.

Accuracy for the matrix can be calculated by taking average of the values lying across the **“main diagonal”** i.e

$$Accuracy = \frac{TruePositive + TrueNegative}{TotalSample}$$

Confusion Matrix forms the basis for the other types of metrics.

1 Area Under Curve

Area Under Curve(AUC) is one of the most widely used metrics for evaluation. It is used for binary classification problem. *AUC* of a classifier is equal to the probability that the classifier will rank a randomly chosen positive example higher than a randomly chosen negative example. Before defining *AUC*, let us understand two basic terms :

- **True Positive Rate (Sensitivity)** : True Positive Rate is defined as *TP/ (FN+TP)*. True Positive Rate corresponds to the proportion of positive data points that are correctly considered as positive, with respect to all positive data points.

$$TruePositiveRate = \frac{TruePositive}{FalseNegative + TruePositive}$$

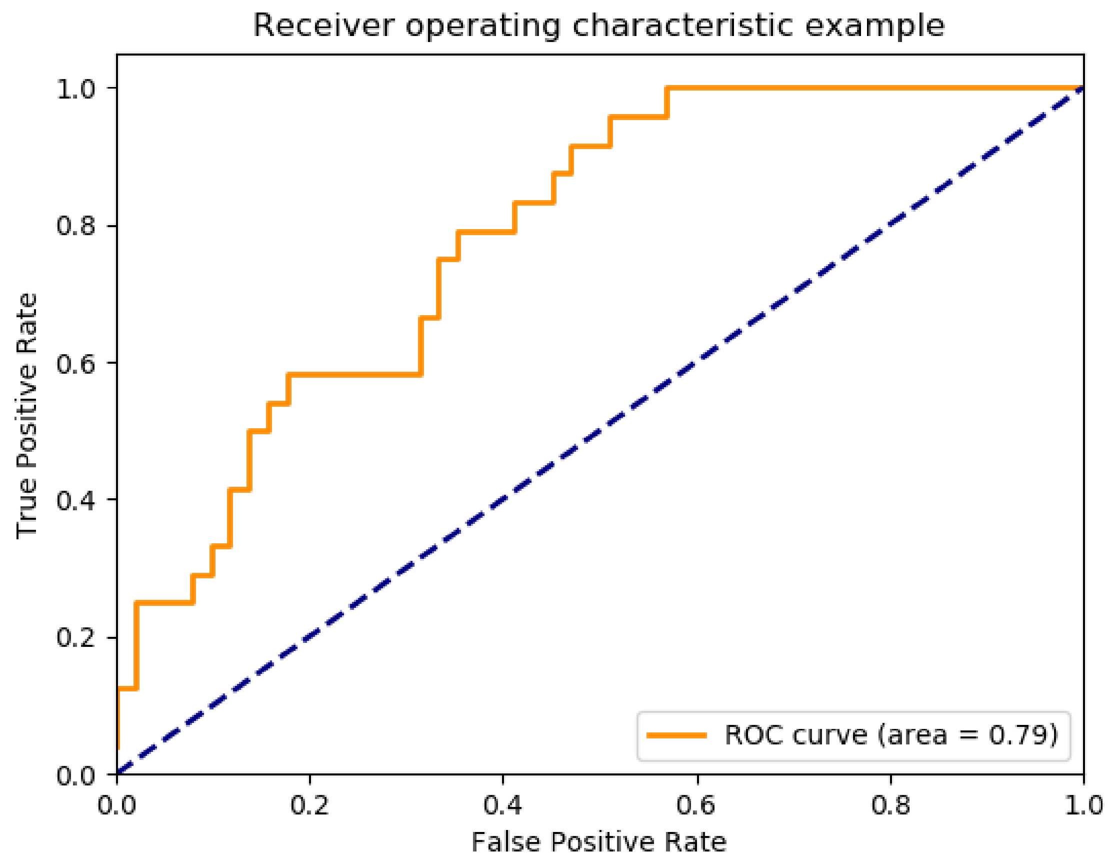
- **True Negative Rate (Specificity)** : True Negative Rate is defined as *TN / (FP+TN)*. False Positive Rate corresponds to the proportion of negative data points that are correctly considered as negative, with respect to all negative data points.

$$TrueNegativeRate = \frac{TrueNegative}{TrueNegative + FalsePositive}$$

- **False Positive Rate** : False Positive Rate is defined as *FP / (FP+TN)*. False Positive Rate corresponds to the proportion of negative data points that are mistakenly considered as positive, with respect to all negative data points.

$$FalsePositiveRate = \frac{FalsePositive}{TrueNegative + FalsePositive}$$

False Positive Rate and *True Positive Rate* both have values in the range **[0, 1]**. *FPR* and *TPR* both are computed at varying threshold values such as (0.00, 0.02, 0.04, …., 1.00) and a graph is drawn. *AUC* is the area under the curve of plot *False Positive Rate vs True Positive Rate* at different points in **[0, 1]**.



As evident, *AUC* has a range of [0, 1]. The greater the value, the better is the performance of our model.

2 F1 Score

F1 Score is used to measure a test's accuracy

F1 Score is the Harmonic Mean between precision and recall. The range for F1 Score is [0, 1]. It tells you how precise your classifier is (how many instances it classifies correctly), as well as how robust it is (it does not miss a significant number of instances).

High precision but lower recall, gives you an extremely accurate, but it then misses a large number of instances that are difficult to classify. The greater the F1 Score, the better is the performance of our model. Mathematically, it can be expressed as :

$$F1 = 2 * \frac{1}{\frac{1}{precision} + \frac{1}{recall}}$$

F1 Score tries to find the balance between precision and recall.

- **Precision :** It is the number of correct positive results divided by the number of positive results predicted by the classifier.

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives}$$

- **Recall :** It is the number of correct positive results divided by the number of ***all*** relevant samples (all samples that should have been identified as positive).

Recall
$$= \frac{TruePositives}{TruePositives + FalseNegatives}$$

3 Mean Absolute Error

Mean Absolute Error is the average of the difference between the Original Values and the Predicted Values. It gives us the measure of how far the predictions were from the actual output. However, they don't gives us any idea of the direction of the error i.e. whether we are under predicting the data or over predicting the data. Mathematically, it is represented as :

$$MeanAbsoluteError = \frac{1}{N}\sum_{j=1}^{N}|y_j - \hat{y}_j|$$

4 Mean Squared Error

Mean Squared Error(MSE) is quite similar to Mean Absolute Error, the only difference being that MSE takes the average of the **square** of the difference between the original values and the predicted values. The advantage of MSE being that it is easier to compute the gradient, whereas Mean Absolute Error requires complicated linear programming tools to compute the gradient. As, we take square of the error, the effect of larger errors become more pronounced then smaller error, hence the model can now focus more on the larger errors.

$$MeanSquaredError = \frac{1}{N}\sum_{j=1}^{N}(y_j - \hat{y}_j)^2$$

Mean Squared Error

Source: https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234

Python:

```
from sklearn.metrics import accuracy_score, confusion_matrix,
roc_auc_score,plot_roc_curve, classification_report
cm=confusion_matrix(y_test,result)
tn=cm[0,0]
fp=cm[0,1]
print(confusion_matrix(y_test,result))
print('auc: ',roc_auc_score(y_test,result))
plot_roc_curve(classifier,x_test,y_test)
plt.show()
print(classification_report(y_test,result))
print(accuracy_score(y_test,result))
```