

# Concepts in Machine Learning-CST 383 KTU-Minor Notes-Dr Binu V P

This course enables the learners to understand the fundamental concepts and algorithms in machine learning. Its a part of KTU minor course in Machine Learning-Dr Binu V P, 9847390760

## Classification Assessment



February 28, 2022

Evaluation of a machine learning model is crucial to measure its performance. Numerous metrics are used in the evaluation of a machine learning model. Selection of the most suitable metrics is important to fine-tune a model based on its performance. We should know the methods to assess the performance of classifiers, and to compare multiple classifiers.

### CLASSIFICATION PERFORMANCE MEASURES

Let  $D$  be the testing set comprising  $n$  points in a  $d$  dimensional space, let  $\{c_1, c_2, \dots, c_k\}$  denote the set of  $k$  class labels, and let  $M$  be a classifier. For  $x_i \in D$ , let  $y_i$  denote its true class, and let  $\hat{y}_i = M(x_i)$  denote its predicted class.

#### Classification Accuracy and its Limitations

Classification accuracy is the ratio of correct predictions to total predictions made.

$$\text{classification accuracy} = \text{correct predictions} / \text{total predictions}$$

It is often presented as a percentage by multiplying the result by 100.

$$\text{classification accuracy} = (\text{correct predictions} / \text{total predictions}) * 100$$

Classification accuracy can also easily be turned into a misclassification rate or error rate by inverting the value, such as:

$$\text{error rate} = (1 - (\text{correct predictions} / \text{total predictions})) * 100$$

Classification accuracy is a great place to start, but often encounters problems in practice.

The main problem with classification accuracy is that it hides the detail you need to better understand the performance of your classification model. There are two examples where you are most likely to encounter this problem:

- When your data has more than 2 classes. With 3 or more classes you may get a classification accuracy of 80%, but you don't know if that is because all classes are being predicted equally well or whether one or two classes are being neglected by the model.
- When your data does not have an even number of classes. You may achieve accuracy of 90% or more, but this is not a good score if 90 records for every 100 belong to one class and you can achieve this score by always predicting the most common class value.

Classification accuracy can hide the detail you need to diagnose the performance of your model. But thankfully we can tease apart this detail by using a confusion matrix.

**Example:** Consider a Bayes classifier, which miss classifies 8 out of 30 test cases. Then the

$$accuracy = 22/30 = 0.733$$

$$errorrate = 8/30 = 0.267$$

### Contingency Table based Measures ( Confusion Matrix)

The error rate (and, thus also the accuracy) is a global measure in that it does not explicitly consider the classes that contribute to the error. More informative measures can be obtained by tabulating the class specific agreement and disagreement between the true and predicted labels over the testing set.

Let  $D = \{D_1, D_2, \dots, D_k\}$  denote a partitioning of the testing points based on their true class labels, where

$$D_j = \{x_i \in D | y_i = c_j\}$$

Let  $n_i = |D_i|$  denote the size of true class  $c_i$ .

Let  $R = \{R_1, R_2, \dots, R_k\}$  denote a partitioning of the testing points based on the predicted labels, that is,

$$R_j = \{x_i \in D | \hat{y}_i = c_j\}$$

Let  $m_j = |R_j|$  denote the size of the predicted class  $c_j$ .

$R$  and  $D$  induce a  $k \times k$  contingency table  $N$ , also called a **confusion matrix**, defined as follows:

$$N(i, j) = n_{ij} = |R_i \cap D_j| = |\{x_a \in D | \hat{y}_a = c_i \text{ and } y_a = c_j\}|$$

where  $1 \leq i, j \leq k$ . The count  $n_{ij}$  denotes the number of points with predicted class  $c_i$  whose true label is  $c_j$ . Thus,  $n_{ii}$  (for  $1 \leq i \leq k$ ) denotes the number of cases where the classifier agrees on the true label  $c_i$ . The remaining counts  $n_{ij}$ , with  $i \neq j$ , are cases where the classifier and true labels disagree.

### Accuracy/Precision

The class-specific accuracy or precision of the classifier  $M$  for class  $c_i$  is given as the fraction of correct predictions over all points predicted to be in class  $c_i$

$$acc_i = prec_i = \frac{n_{ii}}{m_i}$$

where  $m_i$  is the number of examples predicted as  $c_i$  by classifier  $M$ . The higher the accuracy on class  $c_i$  the better the classifier.

The overall precision or accuracy of the classifier is the weighted average of the class-specific accuracy:

$$Accuracy = Precision = \sum_{i=1}^k \frac{m_i}{n} acc_i = \frac{1}{n} \sum_{i=1}^k n_{ii}$$

### Coverage/Recall

The class-specific coverage or recall of  $M$  for class  $c_i$  is the fraction of correct predictions over all points in class  $c_i$  :

$$Coverage_i = Recall_i = \frac{n_{ii}}{n_i}$$

where  $n_i$  is the number of points in class  $c_i$  . The higher the coverage the better the classifier.

### F-measure

Often there is a trade-off between the precision and recall of a classifier. For example, it is easy to make  $recall_i = 1$ , by predicting all testing points to be in class  $c_i$  . However, in this case  $prec_i$  will be low. On the other hand, we can make  $prec_i$  very high by predicting only a few points as  $c_i$  , for instance, for those predictions where  $M$  has the most confidence, but in this case  $recall_i$  will be low. Ideally, we would like both precision and recall to be high. The class-specific F-measure tries to balance the precision and recall values, by computing their harmonic mean for class  $c_i$  :

$$F_i = \frac{2}{\frac{1}{Prec_i} + \frac{1}{Recall_i}} = 2 \cdot \frac{Prec_i * Recall_i}{Prec_i + Recall_i} = 2 \cdot \frac{n_{ii}}{n_i + m_i}$$

The higher the  $F_i$  value the better the classifier. The overall F-measure for the classifier  $M$  is the mean of the class-specific values:

$$F = \frac{1}{k} \sum_{i=1}^r F_i$$

For a perfect classifier, the maximum value of the F-measure is 1.

### Example:

Consider the 2-dimensional Iris dataset shown in Figure

Table Contingency table for Iris dataset: testing set

	True			
Predicted	Iris-setosa ( $c_1$ )	Iris-versicolor ( $c_2$ )	Iris-virginica( $c_3$ )	
Iris-setosa ( $c_1$ )	10	0	0	$m_1 = 10$
Iris-versicolor ( $c_2$ )	0	7	5	$m_2 = 12$
Iris-virginica ( $c_3$ )	0	3	5	$m_3 = 8$
	$n_1 = 10$	$n_2 = 10$	$n_3 = 10$	$n = 30$

From the confusion matrix we can compute the class-specific precision (or accuracy) values:

$$prec_1 = n_{11}/m_1 = 10/10 = 1.0$$

$$prec_2 = n_{22}/m_2 = 7/12 = 0.583$$

$$prec_3 = n_{33}/m_3 = 5/8 = 0.625$$

The overall accuracy

$$Accuracy = (n_{11} + n_{22} + n_{33})/n = (10 + 7 + 5)/30 = 22/30 = 0.733$$

The class-specific recall (or coverage) values are given as

$$recall_1 = n_{11}/n_1 = 10/10 = 1.0$$

$$recall_2 = n_{22}/n_2 = 7/10 = 0.7$$

$$recall_3 = n_{33}/n_3 = 5/10 = 0.5$$

From these we can compute the class-specific F-measure values:

$$F_1 = 2 \cdot n_{11}/(n_1 + m_1) = 20/20 = 1.0$$

$$F_2 = 2 \cdot n_{22}/(n_2 + m_2) = 14/22 = 0.636$$

$$F_3 = 2 \cdot n_{33}/(n_3 + m_3) = 10/18 = 0.556$$

Thus, the overall F-measure for the classifier is

$$F = \frac{1}{3}(1.0 + 0.636 + 0.556) = 2.192/3 = 0.731$$

### Binary Classification: Positive and Negative Class

When there are only  $k = 2$  classes, we call class  $c_1$  the positive class and  $c_2$  the negative class. The entries of the resulting  $2 \times 2$  confusion matrix, shown in Table , are given special names, as follows:

Table Confusion matrix for two classes

	True Class	
Predicted Class	Positive ( $c_1$ )	Negative ( $c_2$ )
Positive ( $c_1$ )	True Positive (TP)	False Positive (FP)
Negative ( $c_2$ )	False Negative (FN)	True Negative (TN)

**True Positives (TP):** The number of points that the classifier correctly predicts as positive:

$$TP = n_{11} = |\{x_i | \hat{y}_i = y_i = c_1\}|$$

**False Positives (FP) Type 1 error:** The number of points the classifier predicts to be positive, which in fact belong to the negative class:

$$FP = n_{12} = |\{x_i | \hat{y}_i = c_1 \text{ and } y_i = c_2\}|$$

**False Negatives (FN) Type 2 error:** The number of points the classifier predicts to be in the negative class, which in fact belong to the positive class:

$$FN = n_{21} = |\{x_i | \hat{y}_i = c_2 \text{ and } y_i = c_1\}|$$

**True Negatives (TN) :** The number of points that the classifier correctly predicts as negative:

$$TN = n_{22} = |\{x_i | \hat{y}_i = y_i = c_2\}|$$

### Error Rate

The error rate for the binary classification case is given as the fraction of mistakes (or false predictions):

$$ErrorRate = \frac{FP+FN}{n}$$

### Accuracy

The accuracy is the fraction of correct predictions:

$$Accuracy = \frac{TP+TN}{n}$$

The above are global measures of classifier performance. We can obtain class-specific measures as follows.

### Class-specific Precision

The precision for the positive and negative class is given as

$$prec_P = \frac{TP}{TP+FP} = \frac{TP}{m_1}$$

$$prec_N = \frac{TN}{TN+FN} = \frac{TN}{m_2}$$

where  $m_i = |R_i|$  is the number of points predicted by  $M$  as having class  $c_i$ .

### Sensitivity: True Positive Rate ( Recall)

The true positive rate, also called sensitivity, is the fraction of correct predictions with respect to all points in the positive class, that is, it is simply the recall for the positive class

$$TPR = sensitivity = recall_P = \frac{TP}{TP+FN} = \frac{TP}{n_1}$$

where  $n_1$  is the size of the positive class.

### Specificity: True Negative Rate

The true negative rate, also called specificity, is simply the recall for the negative class:

$TNR = specificity = recall_N = \frac{TN}{FP+TN} = \frac{TN}{n_2}$ , where  $n_2$  is the size of the negative class

### False Negative Rate

The false negative rate is defined as

$$FNR = \frac{FN}{TP+FN} = \frac{FN}{n_1} = 1 - sensitivity$$

### False Positive Rate

The false positive rate is defined as

$$FPR = \frac{FP}{FP+TN} = \frac{FP}{n_2} = 1 - specificity$$

### Example

Table Iris PC dataset: contingency table for binary classification

	True		
Predicted	Positive ( $c_1$ )	Negative ( $c_2$ )	
Positive ( $c_1$ )	$TP = 7$	$FP = 7$	$m_1 = 14$
Negative ( $c_2$ )	$FN = 3$	$TN = 13$	$m_2 = 16$
	$n_1 = 10$	$n_2 = 20$	$n = 30$

From this table, we can compute the various performance measures:

$$prec_P = \frac{TP}{TP+FP} = 7/14 = 0.5$$

$$prec_N = \frac{TN}{TN+FN} = 13/16 = 0.8125$$

$$recall_P = sensitivity = TPR = \frac{TP}{TP+FN} = 7/10 = 0.7$$

$$recall_N = specificity = TNR = \frac{TN}{TN+FP} = 13/20 = 0.65$$

$$FNR = 1 - sensitivity = 1 - 0.7 = 0.3$$

$$FPR = 1 - specificity = 1 - 0.65 = 0.35$$

We can observe that the precision for the positive class is rather low. The true positive rate is also low, and the false positive rate is relatively high. Thus, the naive Bayes classifier is not particularly effective on this testing dataset.

### Example ( University Question)

Imagine an application that is developed to recognize cats and dogs. This identifies eight dogs in a picture containing 12 dogs and some cats. Of the eight dogs identified, five actually are dogs while the rest are cats. Compute the True Positive, False Positive

and False Negative values.

$$TP = 5$$

$$FP = 8 - 5 = 3$$

$$FN = 12 - 5 = 7$$

### Example:(University Question)

For a classifier, the confusion matrix is given

What is the precision, recall and accuracy of that classifier?

	+	-
+	9	9
-	1	5

$$prec_P = \frac{TP}{TP+FP} = 9/(9+9) = 9/18 = 0.5$$

$$prec_N = \frac{TN}{TN+FN} = 5/6 = 0.83$$

$$recall_P = sensitivity = TPR = \frac{TP}{TP+FN} = 9/(9+1) = 9/10 = 0.9$$

$$recall_N = specificity = TNR = \frac{TN}{TN+FP} = 5/(5+9) = 5/14 = 0.357$$

$$Accuracy = \frac{TP+TN}{n} = 14/24 = 0.583$$

A confusion matrix is a tabular summary of the number of correct and incorrect predictions made by a classifier. It is used to measure the performance of a classification model. It can be used to evaluate the performance of a classification model through the calculation of performance metrics like accuracy, precision, recall, and F1-score.

### Example: University question

Suppose we train a model to predict whether an email is Spam or Not Spam. After training the model, we apply it to a test set of 500 new emails and the model produces the following contingency table.

		True Class	
		Spam	Not Spam
Predicted Class	Spam	70	30
	Not Spam	70	330

Compute the precision and recall of this model with respect to spam class.

$$\text{Precision} = TP / (TP + FP) = 70 / (70 + 30) = 0.70$$

$$\text{Recall} = TP / (TP + FN) = 70 / (70 + 70) = 0.5$$

### Precision vs. Recall for Imbalanced Classification

You may decide to use precision or recall on your imbalanced classification problem.

Maximizing precision will minimize the number false positives, whereas maximizing the recall will minimize the number of false negatives.

Precision: Appropriate when minimizing false positives is the focus.

Recall: Appropriate when minimizing false negatives is the focus.

Sometimes, we want excellent predictions of the positive class. We want high precision and high recall. This can be challenging, as often increases in recall often come at the expense of decreases in precision.

Instead of picking one measure or the other, we can choose a new metric that combines both precision and recall into one score.

### F-measure / F1-Score

The F1 score is a number between 0 and 1 and is the harmonic mean of precision and recall. We use harmonic mean because it is not sensitive to extremely large values, unlike simple averages.

F1 score sort of maintains a balance between the precision and recall for your classifier. If your precision is low, the F1 is low and if the recall is low again your F1 score is low.

There will be cases where there is no clear distinction between whether Precision is



more important or Recall. We combine them!

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. One can get a high F1 score only when both precision and recall are high.

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

F1 score gives the same weightage to recall and precision.

There is a weighted F1 score in which we can give different weightage to recall and precision. As discussed in the previous section, different problems give different weightage to recall and precision.

$$F_{\beta} = (1 + \beta^2) * \frac{(Precision * Recall)}{(\beta^2 * Precision) + Recall}$$

Beta represents how many times recall is more important than precision. If the recall is twice as important as precision, the value of Beta is 2.

### Example: University Question

Imagine an application that is developed to recognize cats and dogs. This identifies eight dogs in picture containing 12 dogs and some cats. Of the eight dogs identified, five actually are dogs while the rest are cats. Compute the following

predicted	true	
	Dogs ( 12)	Cats( x)
Dogs ( 8)	5 TP	3 FP
Cats ( y)	7 FN	x-3 TN

$$Precision = TP / (TP + FP) = 5/8$$

$$Recall = sensitivity = TP / (TP + FN) = 5/12$$

$$Specificity = TN / (FP + TN) = (x-3) / (3+x-3) = (x-3)/x$$

$$Accuracy = (5+x-3) / (5+7+3+x-3) = (2+x) / (12+x)$$

$$Error\ rate = 1 - accuracy = 1 - (2+x) / (12+x) = 10 / (12+x)$$

$$F\ measure = 2 * (Precision * recall) / (precision + recall) = 2 * 5/8 / (5/8 + 5/12) = 5/4 / (5/8 + 5/12)$$

### Support

Support is the number of actual occurrences of the class in the specified dataset. Imbalanced support in the training data may indicate structural weaknesses in the reported scores of the classifier and could indicate the need for stratified sampling or re balancing. Support doesn't change between models but instead diagnoses the evaluation process.

### When to use Accuracy / Precision / Recall / F1-Score?

- a. Accuracy is used when the True Positives and True Negatives are more important. Accuracy is a better metric for Balanced Data.
- b. Whenever False Positive is much more important use Precision.
- c. Whenever False Negative is much more important use Recall.
- d. F1-Score is used when the False Negatives and False Positives are important. F1-Score is a better metric for Imbalanced Data.

In the detection of spam mail, it is okay if any spam mail remains undetected (false negative), but what if we miss any critical mail because it is classified as spam (false positive). In this situation, False Positive should be as low as possible. Here, precision is more vital as compared to recall. Similarly, in the medical application, we don't want to miss any patient. Therefore we focus on having a high recall.

### Python Code( Confusion Matrix)

```
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
# actual values
actual = [1,1,0,1,1,0,0,1,1,0]
# predicted values
predicted = [0,1,0,1,0,0,1,1,0,0]
# confusion matrix
matrix = confusion_matrix(actual,predicted, labels=[1,0])
print('Confusion matrix : \n',matrix.T)

# outcome values order in sklearn
tp, fn, fp, tn = confusion_matrix(actual,predicted,labels=[1,0]).reshape(-1)
print('TP=%-5d FP=%-5d\nFN=%-5d TN=%-5d:\n'% (tp, fp, fn, tn))

# classification report for precision, recall f1-score and accuracy
matrix = classification_report(actual,predicted,labels=[1,0])
print('Classification report : \n',matrix)
output
Confusion matrix :
[[3 1]
 [3 3]]
```

TP=3 FP=1 FN=3 TN=3

Classification report :

	precision	recall	f1-score	support
1	0.75	0.50	0.60	6
0	0.50	0.75	0.60	4
accuracy			0.60	10
macro avg	0.62	0.62	0.60	10
weighted avg	0.65	0.60	0.60	10

## ROC and AUC Score

ROC is the short form of **Receiver Operating Curve**, which helps determine the optimum threshold value for classification. The threshold value is the floating-point value between two classes forming a boundary between those two classes. Here in our model, any predicted output above the threshold is classified as class 1 and below it is classified as class 0.

ROC is realized by visualizing it in a plot. The **area under ROC**, famously known as **AUC** is used as a metric to evaluate the classification model. ROC is drawn by taking **false positive rate in the x-axis** and **true positive rate in the y-axis**. The best value of AUC is 1 and the worst value is 0. However, AUC of 0.5 is generally considered the bottom reference of a classification model. Higher the AUC, better the model.

In Machine Learning, performance measurement is an essential task. So when it comes to a classification problem, we can count on an AUC - ROC Curve. When we need to check or visualize the performance of the multi-class classification problem, we use the **AUC (Area Under The Curve) ROC (Receiver Operating Characteristics) curve**. It is one of the most important evaluation metrics for checking any classification model's performance. It is also written as AUROC (Area Under the Receiver Operating Characteristics)

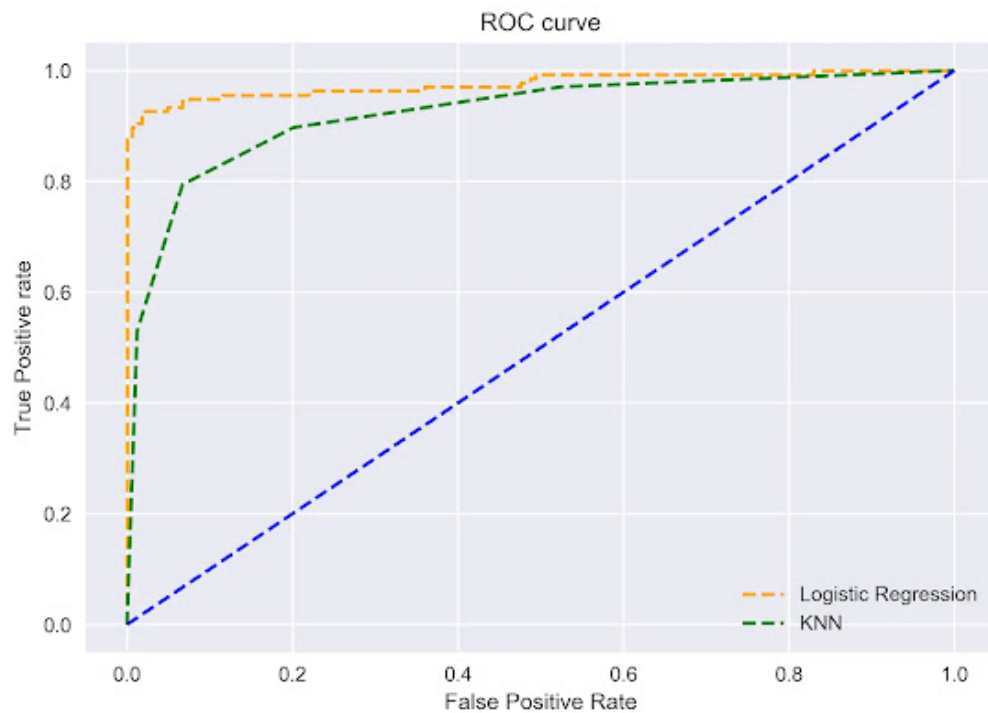
## What is the AUC - ROC Curve?

AUC - ROC curve is a performance measurement for the classification problems at various threshold settings. ROC is a probability curve and AUC represents 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 predicting 0 classes as 0 and 1 classes as 1. By analogy, the Higher the AUC, the better the model is at distinguishing between patients with the disease and no disease.

**The ROC curve is plotted with TPR against the FPR where TPR is on the y-axis and FPR is on the x-axis.**



the point (1,1) at the top right-hand corner in the ROC plot. An ideal classifier corresponds to the top left point (0,1), which corresponds to the case FPR=0 and TPR=1, that is, the classifier has no false positives, and identifies all true positives (as a consequence, it also correctly predicts all the points in the negative class). This case is shown in Table c. As such, a ROC curve indicates the extent to which the classifier ranks positive instances higher than the negative instances. An ideal classifier should score all positive points higher than any negative point. Thus, a classifier with a curve closer to the ideal case, that is, closer to the upper left corner, is a better classifier.



It is evident from the plot that the AUC for the Logistic Regression ROC curve is higher than that for the KNN ROC curve. Therefore, we can say that logistic regression did a better job of classifying the positive class in the dataset.

### Defining terms used in AUC and ROC Curve.

#### TPR (True Positive Rate) / Recall /Sensitivity

$$TPR/Recall/Sensitivity = \frac{TP}{TP+FN}$$

#### Specificity

$$Specificity = \frac{TN}{TN+FP}$$





#### False Positive Rate ( FPR)

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

If we use a random model to classify, it has a 50% probability of classifying the positive and negative classes correctly. Here, the AUC = 0.5. A perfect model has a 100% probability of classifying the positive and negative classes correctly. Here, the AUC = 1. So when we want to select the best model, we want a model that is closest to the perfect model. In other words, a model with AUC close to 1. When we say a model has a high AUC score, it means the model's ability to separate the classes is very high (high separability). This is a very important metric that should be checked while selecting a classification model.

### The relation between Sensitivity, Specificity, FPR, and Threshold.

Sensitivity and Specificity are inversely proportional to each other. So when we increase Sensitivity, Specificity decreases, and vice versa.

Sensitivity , Specificity  and Sensitivity , Specificity 

When we decrease the threshold, we get more positive values thus it increases the sensitivity and decreasing the specificity.

Similarly, when we increase the threshold, we get more negative values thus we get higher specificity and lower sensitivity.

As we know FPR is 1 - specificity. So when we increase TPR, FPR also increases and vice versa.

TPR , FPR  and TPR , FPR 

### Area Under ROC Curve

The area under the ROC curve, abbreviated AUC, can be used as a measure of classifier performance. Because the total area of the plot is 1, the AUC lies in the interval [0,1] – the higher the better. The AUC value is essentially the probability that the classifier will rank a random positive test case higher than a random negative test instance.

### ROC/AUC Algorithm

Algorithm shows the steps for plotting a ROC curve, and for computing the area under the curve. It takes as input the testing set  $D$ , and the classifier  $M$ . The first step is to predict the score  $S(x_i)$  for the positive class ( $c_1$ ) for each test point  $x_i \in D$ . Next, we sort the  $(S(x_i), y_i)$  pairs, that is, the score and the true class pairs, in decreasing order of the scores (line 3). Initially, we set the positive score threshold  $\rho = \infty$  (line 7). The for

loop (line 8) examines each pair  $(S(x_i), y_i)$  in sorted order, and for each distinct value of the score, it sets  $\rho = S(x_i)$  and plots the point  $(FPR, TPR) = \left(\frac{FP}{n_2}, \frac{TP}{n_1}\right)$

---

**ALGORITHM 22.1. ROC Curve and Area under the Curve**


---

**ROC-CURVE(D, M):**

```

1  $n_1 \leftarrow |\{x_i \in D \mid y_i = c_1\}|$  // size of positive class
2  $n_2 \leftarrow |\{x_i \in D \mid y_i = c_2\}|$  // size of negative class
   // classify, score, and sort all test points
3  $L \leftarrow$  sort the set  $\{(S(x_i), y_i) : x_i \in D\}$  by decreasing scores
4  $FP \leftarrow TP \leftarrow 0$ 
5  $FP_{prev} \leftarrow TP_{prev} \leftarrow 0$ 
6  $AUC \leftarrow 0$ 
7  $\rho \leftarrow \infty$ 
8 foreach  $(S(x_i), y_i) \in L$  do
9   if  $\rho > S(x_i)$  then
10     plot point  $\left(\frac{FP}{n_2}, \frac{TP}{n_1}\right)$ 
11      $AUC \leftarrow AUC + \text{TRAPEZOID-AREA}\left(\left(\frac{FP_{prev}}{n_2}, \frac{TP_{prev}}{n_1}\right), \left(\frac{FP}{n_2}, \frac{TP}{n_1}\right)\right)$ 
12      $\rho \leftarrow S(x_i)$ 
13      $FP_{prev} \leftarrow FP$ 
14      $TP_{prev} \leftarrow TP$ 
15   if  $y_i = c_1$  then  $TP \leftarrow TP + 1$ 
16   else  $FP \leftarrow FP + 1$ 
17 plot point  $\left(\frac{FP}{n_2}, \frac{TP}{n_1}\right)$ 
18  $AUC \leftarrow AUC + \text{TRAPEZOID-AREA}\left(\left(\frac{FP_{prev}}{n_2}, \frac{TP_{prev}}{n_1}\right), \left(\frac{FP}{n_2}, \frac{TP}{n_1}\right)\right)$ 

TRAPEZOID-AREA((x1, y1), (x2, y2)):
19  $b \leftarrow |x_2 - x_1|$  // base of trapezoid
20  $h \leftarrow \frac{1}{2}(y_2 + y_1)$  // average height of trapezoid
21 return  $(b \cdot h)$ 

```

---

As each test point is examined, the true and false positive values are adjusted based on the true class  $y_i$  for the test point  $x_i$ . If  $y_i = c_1$ , we increment the true positives, otherwise, we increment the false positives (lines 15-16). At the end of the for loop we plot the final point in the ROC curve (line 17). The AUC value is computed as each new point is added to the ROC plot. The algorithm maintains the previous values of the false and true positives,  $FP_{prev}$  and  $TP_{prev}$ , for the previous score threshold  $\rho$ . Given the current FP and TP values, we compute the area under the curve defined by the four points

$$(x_1, y_1) = \left(\frac{FP_{prev}}{n_2}, \frac{TP_{prev}}{n_1}\right)$$

$$(x_2, y_2) = \left(\frac{FP}{n_2}, \frac{TP}{n_1}\right)$$

$$(x_1, 0) = \left(\frac{FP_{prev}}{n_2}, 0\right)$$

$$(x_2, 0) = \left(\frac{FP}{n_2}, 0\right)$$

These four points define a trapezoid, whenever  $x_2 > x_1$  and  $y_2 > y_1$ , otherwise, they

define a rectangle (which may be degenerate, with zero area). The function TRAPEZOID-AREA computes the area under the trapezoid, which is given as  $b \cdot h$ , where  $b = |x_2 - x_1|$  is the length of the base of the trapezoid and  $h = \frac{1}{2}(y_2 + y_1)$  is the average height of the trapezoid.

### Example:

consider the following sorted scores, along with the true class, for some testing dataset with  $n = 5$ ,  $n_1 = 3$  and  $n_2 = 2$ .

$(0.9, c_1)$ ,  $(0.8, c_2)$ ,  $(0.8, c_1)$ ,  $(0.8, c_1)$ ,  $(0.1, c_2)$

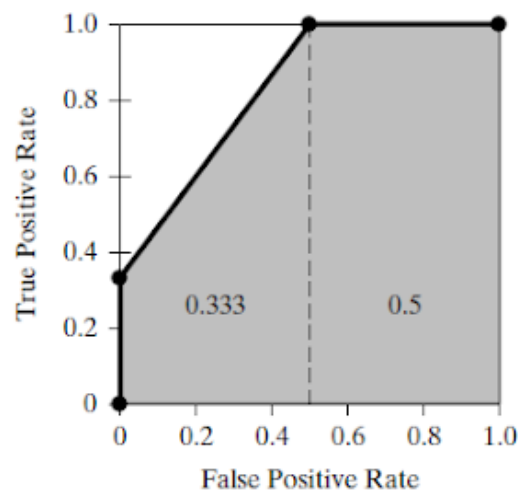


Figure 22.4. ROC plot and AUC: trapezoid region.

the following points that are added to the ROC plot, along with the running AUC:

$\rho$	FP	TP	(FPR, TPR)	AUC
$\infty$	0	0	(0, 0)	0
0.9	0	1	(0, 0.333)	0
0.8	1	3	(0.5, 1)	0.333
0.1	2	3	(1, 1)	0.833

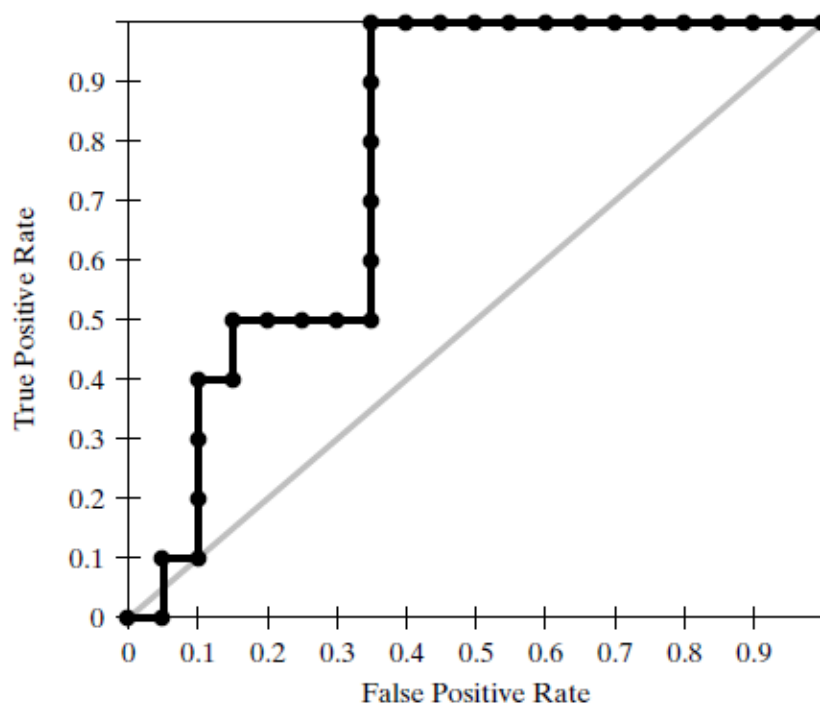
Figure shows the ROC plot, with the shaded region representing the AUC. We can observe that a trapezoid is obtained whenever there is at least one positive and one negative point with the same score. The total AUC is 0.833, obtained as the sum of the trapezoidal region on the left (0.333) and the rectangular region on the right (0.5).

### Random Classifier

It is interesting to note that a random classifier corresponds to a diagonal line in the ROC plot. To see this think of a classifier that randomly guesses the class of a point as positive half the time, and negative the other half. We then expect that half of the true positives and true negatives will be identified correctly, resulting in the point  $(\text{TPR}, \text{FPR}) = (0.5, 0.5)$  for the ROC plot. If, on the other hand, the classifier guesses the class of a point as positive 90% of the time and as negative 10% of the time, then we expect 90% of the true positives and 10% of the true negatives to be labeled correctly, resulting in  $\text{TPR} = 0.9$  and  $\text{FPR} = 1 - \text{TNR} = 1 - 0.1 = 0.9$ , that is, we get the point  $(0.9, 0.9)$  in the ROC plot.



In general, any fixed probability of prediction, say  $r$ , for the positive class, yields the point  $(r, r)$  in ROC space. The diagonal line thus represents the performance of a random classifier, over all possible positive class prediction thresholds  $r$ . It follows that if the ROC curve for any classifier is below the diagonal, it indicates performance worse than random guessing. For such cases, inverting the class assignment will produce a better classifier. As a consequence of the diagonal ROC curve, the AUC value for a random classifier is 0.5. Thus, if any classifier has an AUC value less than 0.5, that also indicates performance worse than random.

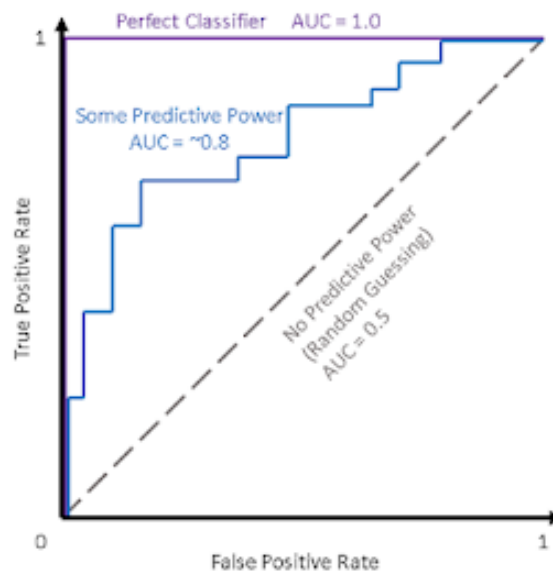


Example . In addition to the ROC curve for the naive Bayes classifier, Figure above also shows the ROC plot for the random classifier (the diagonal line in gray). We can see that the ROC curve for the naive Bayes classifier is much better than random. Its AUC value is 0.775, which is much better than the 0.5 AUC for a random classifier. However, at the very beginning naive Bayes performs worse than the random classifier because the highest scored point is from the negative class. As such, the ROC curve should be considered as a discrete approximation of a smooth curve that would be obtained for a very large (infinite) testing dataset.

The following figure demonstrates how some theoretical classifiers would plot on an ROC curve. The gray dotted line represents a classifier that is no better than random guessing – this will plot as a diagonal line. The purple line represents a perfect classifier – one with a true positive rate of 100% and a false positive rate of 0%. Nearly all real-world examples will fall somewhere between these two lines – not perfect, but providing more predictive power than random guessing. Typically, what we're looking for is a classifier that maintains a high true positive rate while also having a low false positive rate – this ideal classifier would "hug" the upper left corner of Figure 1, much like the

purple line.

AUC stands for area under the (ROC) curve. Generally, the higher the AUC score, the better a classifier performs for the given task. For a classifier with no predictive power (i.e., random guessing),  $AUC = 0.5$ , and for a perfect classifier,  $AUC = 1.0$ . Most classifiers will fall between 0.5 and 1.0, with the rare exception being a classifier performs worse than random guessing ( $AUC < 0.5$ ).



Suppose there are three classifiers A, B and C. The (FPR, TPR) measures of the three classifiers are as follows – A (0, 1), B (1, 1), C (1, 0.5). Which can be considered as a perfect classifier? Justify your answer.

A(0,1) is the perfect classifier

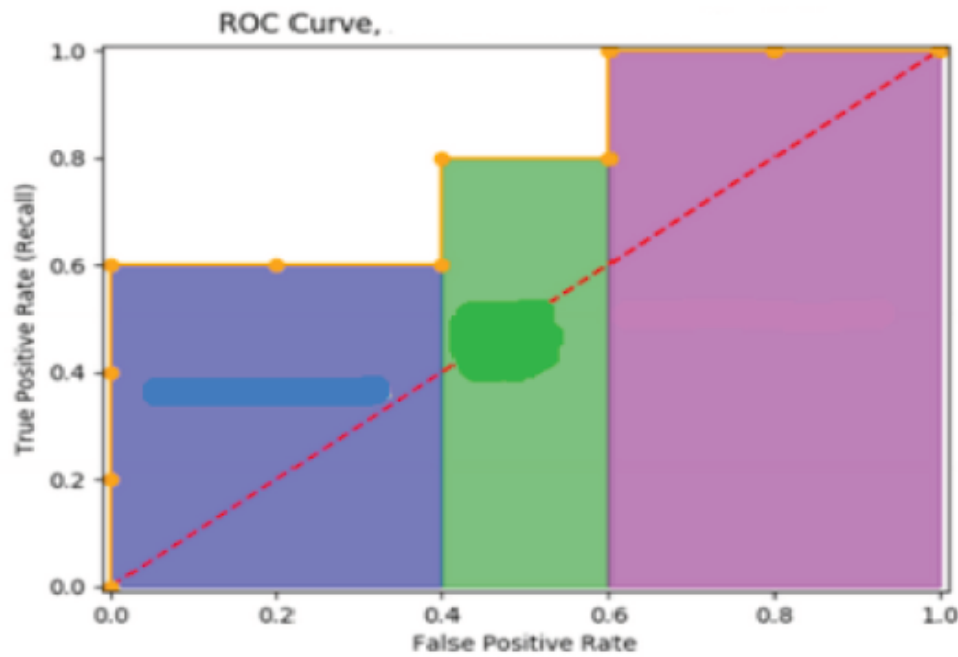
Additionally, ROC curves and AUC scores also allow us to compare the performance of different classifiers for the same problem.

### University question

What are ROC space and ROC curve in machine learning? In ROC space, illustrate which points correspond to perfect prediction, always positive prediction and always negative prediction with proper justification.

**Example: ( university question)**

Given the following ROC Curve? Find the AUC?



$$\text{Area} = (0.4-0) \cdot \frac{1}{2}(0.6+0.6) + (0.6-0.4) \cdot \frac{1}{2}(0.8+0.8) + (1.0-0.6) \cdot \frac{1}{2}(1+1)$$

$$\text{Area} = 0.4 \cdot 0.6 + 0.2 \cdot 0.8 + 0.4 \cdot 1 = 0.8$$

**Example:**

Given the following data, construct the ROC curve of the data. Compute the AUC.

(university question)

Threshold	TP	TN	FP	FN
1	0	25	0	29
2	7	25	0	22
3	18	24	1	11
4	26	20	5	3
5	29	11	14	0
6	29	0	25	0
7	29	0	25	0

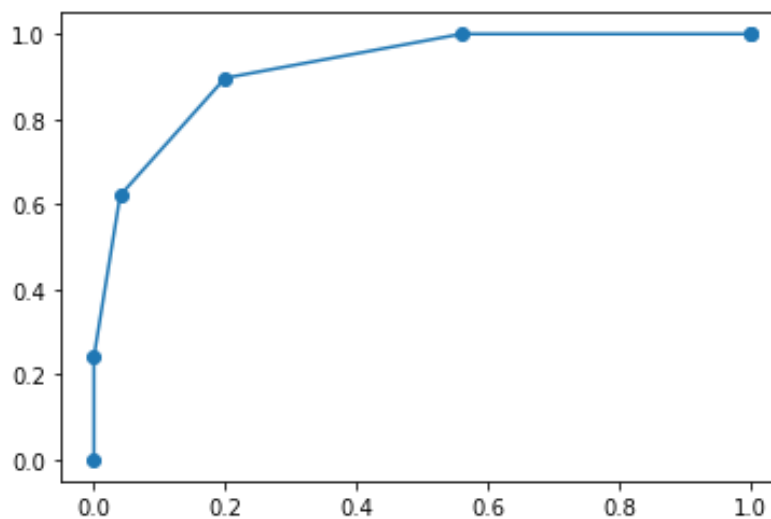
**Python code for plotting ROC and computing the area**

```
import matplotlib.pyplot as plt
import numpy as np
TP=np.array([0,7,18,26,29,29,29])
TN=np.array([25,25,24,20,11,0,0])
FP=np.array([0,0,1,5,14,25,25])
FN=np.array([29,22,11,3,0,0,0])
TPR=TP/(TP+FN)
FPR=FP/(FP+TN)
```

```
plt.plot(FPR,TPR)
plt.scatter(FPR,TPR)
x=zip(FPR,TPR)
for i in x:
    print(i)
area=0
for i in range(6):
    area=area+(FPR[i+1]-FPR[i])*((TPR[i]+TPR[i+1])/2)
print("Area under the curve=",area)
```

**Output**

```
(0.0, 0.0)
(0.0, 0.2413793103448276)
(0.04, 0.6206896551724138)
(0.2, 0.896551724137931)
(0.56, 1.0)
(1.0, 1.0)
(1.0, 1.0)
Area under the curve= 0.919999
```



## Popular posts from this blog

### Concepts in Machine Learning- CST 383 KTU Minor Notes- Dr Binu V P

February 28, 2022

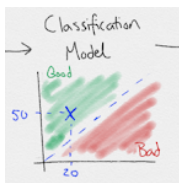
Powered by Blogger

About Me Syllabus Old Question Papers Module 1: Overview of Machine Learning Bayesian Formulation Maximum a Posteriori (MAP) a Bayesian method/Maximum Likelihood Estimation (MLE), Module 2: Supervised Learning Supervised ...

[READ MORE](#)

### Overview of Machine Learning

December 15, 2021



To solve a problem on a computer, we need an algorithm. An algorithm is a sequence of instructions that should be carried out to transform the input to output. For example, one can devi ...

[READ MORE](#)

### Supervised Learning-Linear Regression ,Logistic Regression

January 19, 2022

X	y	$x-x^2$
55	52	-11
60	54	-6
65	56	-1
70	58	4
80	62	16
56	56.4	
$\text{cov}(x,y)/\text{var}(x)$	0.4	
$\hat{y} = \text{beta}1x^2$	30	

Regression Linear Regression Linear regression is perhaps one of the most well known and well understood algorithms in statistics and machine learning. Linear regression was developed in th ...

[READ MORE](#)

Archive



[Report Abuse](#)