Performance Measures for Classification

Performance Measurement

Performance metrics in machine learning are used to evaluate the performance of a machine learning model. These metrics provide quantitative measures to assess how well a model is performing and to compare the performance of different models. Performance metrics are important because they help us understand how well our model is performing and whether it is meeting our requirements. Evaluating a classifier is often significantly trickier than evaluating a regressor. We will see some performance measuring techniques.

1. Confusion Matrix

The Confusion Matrix is a table used to describe the performance of a classification model. It shows the counts of predicted vs actual values. The matrix has four components:

	Predicted Positive	Predicted Negative
Actual Positive	True Positive (TP)	False Negative (FN)
Actual Negative	False Positive (FP)	True Negative (TN)

- True Positive (TP): Correctly predicted positive observations.
- True Negative (TN): Correctly predicted negative observations.
- False Positive (FP): Incorrectly predicted as positive when it was negative (Type I error).
- False Negative (FN): Incorrectly predicted as negative when it was positive (Type II error).

The confusion matrix helps in deriving most other classification metrics.

2. Classification Accuracy

Classification accuracy is a fundamental metric for evaluating the performance of a classification model, providing a quick snapshot of how well the model is performing in terms of correct predictions. This is calculated as the ratio of correct predictions to the total number of input Samples.

$$\text{Accuracy} = \frac{\text{No. of correct predictions}}{\text{Total number of input samples}} = \frac{TP + TN}{TP + TN + FP + FN}$$

It works great if there are an equal number of samples for each class. For example, we have a 90% sample of *class A* and a 10% sample of *class B* in our training set. Then, our model will predict with an accuracy of 90% by predicting all the training samples belonging to *class*

A. If we test the same model with a test set of 60% from class A and 40% from class B. Then the accuracy will fall, and we will get an accuracy of 60%.

Classification accuracy is good but it gives a False Positive sense of achieving high accuracy. The problem arises due to the possibility of misclassification of minor class samples being very high.

When to use: Accuracy is useful when the dataset is balanced (i.e., similar number of positives and negatives).

3. Precision (Positive Predictive Value)

Precision measures the proportion of correctly predicted positive observations out of all observations predicted as positive.

$$\text{Precision} = \frac{TP}{TP + FP}$$

When to use: Precision is important when the cost of false positives is high. For example, in spam detection, it's important to minimize the number of legitimate emails marked as spam.

4. Recall (Sensitivity or True Positive Rate)

Recall measures the proportion of actual positives that were correctly identified by the model.

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

When to use: Recall is important when we want to minimize false negatives. For example, in medical diagnostics, we want to ensure that we identify all patients with a certain disease, even at the risk of false positives.

5. F1 Score

The F1 Score is the **harmonic mean** of Precision and Recall, providing a balanced measure when there is an uneven class distribution.

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

When to use: F1 Score is particularly useful when you need a balance between Precision and Recall. It's especially valuable when the dataset is imbalanced, and neither Precision nor Recall alone is sufficient to describe the model's performance.

6. Specificity (True Negative Rate)

Specificity measures the proportion of actual negatives that were correctly identified as negative.

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

When to use: Specificity is useful when the cost of false positives is high, such as in security systems (e.g., minimizing the number of false alarms).

7. Receiver Operating Characteristic (ROC) Curve

The **ROC curve** is a graphical representation that shows the trade-off between **True Positive Rate (Recall)** and **False Positive Rate** (1 - Specificity) at various threshold settings.

True Positive Rate (TPR) = Recall = Sensitivity.

False Positive Rate (FPR) = $\frac{FP}{FP+TN}$.

The ROC curve is often summarized using the **Area Under the Curve (AUC)** metric, where:

- **AUC = 1** means a perfect classifier.
- **AUC = 0.5** means a random classifier.

When to use: ROC and AUC are particularly useful when you want to evaluate the trade-off between false positives and true positives across different classification thresholds.

8. Logarithmic Loss (Log Loss)

Log Loss penalizes incorrect classifications by using the probability estimates generated by the classifier.

$$\text{Log Loss} = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(\hat{p}_i) + (1 - y_i)\log(1 - \hat{p}_i)]$$

Where:

- y_i is the true label (0 or 1).
- $\hat{p}_i$ is the predicted probability for class 1.

When to use: Log Loss is useful when you want to evaluate a probabilistic classifier that outputs predicted probabilities instead of just class labels. Lower Log Loss values indicate better performance.

9. Cross Validation

Cross validation is a technique used in machine learning to evaluate the performance of a model on unseen data. It involves dividing the available data into multiple folds or subsets, using one of these folds as a validation set, and training the model on the remaining folds. This process is repeated multiple times, each time using a different fold as the validation set. Finally, the results from each validation step are averaged to produce a more robust estimate of the model's performance. Cross validation is an important step in the machine learning process and helps to ensure that the model selected for deployment is robust and generalizes well to new data.

Types of Cross-Validation

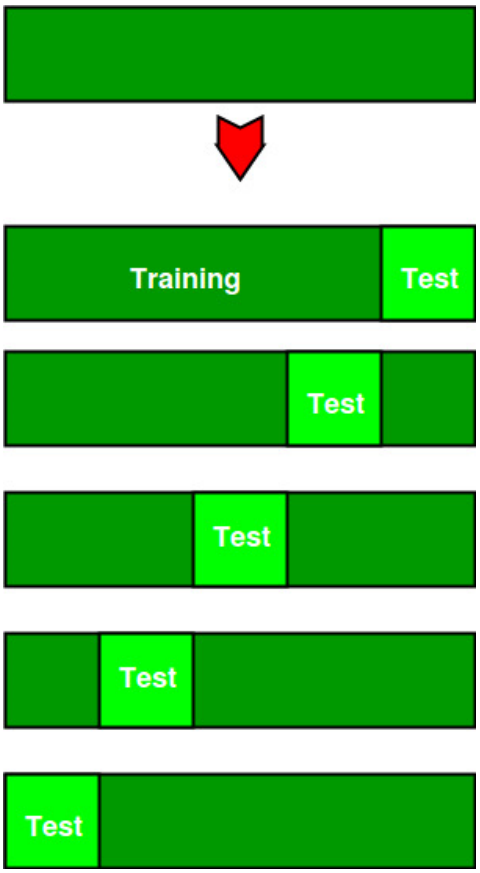
There are several types of cross validation techniques, including **k-fold cross validation, leave-one-out cross validation, and Holdout validation, Stratified Cross-Validation.** The choice of technique depends on the size and nature of the data, as well as the specific requirements of the modeling problem.

1. **Holdout Validation** In Holdout Validation, we perform training on the 50% of the given dataset and rest 50% is used for the testing purpose. It's a simple and quick way to evaluate a model. The major drawback of this method is that we perform training on the 50% of the dataset, it may possible that the remaining 50% of the data contains some important information which we are leaving while training our model i.e. higher bias.
2. **LOOCV (Leave one out CV)** In this method, we perform training on the whole dataset but leaves only one data-point of the available dataset and then iterates for each data-point. In LOOCV, the model is trained on n−1 samples and tested on the one omitted sample, repeating this process for each data point in the dataset.

   The major drawback of this method is that it leads to higher variation in the testing model as we are testing against one data point. If the data point is an outlier it can lead to higher variation. Another drawback is it takes a lot of execution time as it iterates over 'the number of data points' times.
3. **Stratified Cross-Validation** It is a technique used in machine learning to ensure that each fold of the cross-validation process maintains the same class distribution as the entire dataset. This is particularly important when dealing with imbalanced datasets, where certain classes may be underrepresented. In this method,
   - The dataset is divided into k folds while maintaining the proportion of classes in each fold.
   - During each iteration, one-fold is used for testing, and the remaining folds are used for training.
   - The process is repeated k times, with each fold serving as the test set exactly once.

   Stratified Cross-Validation is essential when dealing with classification problems where maintaining the balance of class distribution is crucial for the model to generalize well to unseen data.

4. **K Fold Cross Validation** In K-Fold Cross Validation, we split the dataset into k number of subsets (known as folds) then we perform training on the all the subsets but leave one(k-1) subset for the evaluation of the trained model. In this method, we iterate k times with a different subset reserved for testing purpose each time. *It is always suggested that the value of k should be 10 as the lower value of k takes towards validation and higher value of k leads to LOOCV method.*



When to Use Which Metric?

- **Accuracy**: Good for balanced datasets.
- **Precision**: Important when the cost of false positives is high.
- **Recall**: Critical when the cost of false negatives is high (e.g., medical diagnostics).
- **F1 Score**: A balanced metric when you need to weigh both Precision and Recall.
- **AUC-ROC**: Useful when working with probabilistic classifiers and when you want to evaluate performance at different classification thresholds.
- **Log Loss**: Best for evaluating models that output probabilities rather than discrete labels.
- **Cross Validation**: To avoid overfitting, tune hyperparameters, evaluate small or imbalanced datasets, or compare multiple models fairly.

Random sampling

Suppose you want to take a survey and decided to call 1000 people from a particular state, if you pick either 1000 males completely or 1000 females completely or 900 females and 100 males (randomly) to ask their opinion on a particular product. Then based on these 1000 opinions you can't decide the opinion of that entire state on your product. This is random sampling.

Stratified Sampling

Let the population for that state be 51.3% male and 48.7% female, then for choosing 1000 people from that state if you pick 513 male (51.3% of 1000) and 487 female (48.7% for 1000) i.e 513 male + 487 female (Total=1000 people) to ask their opinion. Then these groups of people represent the entire state. This is called Stratified Sampling.

Conclusion

Selecting the right performance metric depends on the **problem at hand** and the **impact of errors**. In tasks where missing a true positive (false negative) is highly costly, you would prioritize Recall, while in tasks where predicting a false positive is more problematic, Precision or Specificity might be more important.

It's crucial to understand not just how the metrics are calculated, but **what they represent** and **when they are relevant**, so you can interpret the results of your models accurately and apply the appropriate metric in your specific classification problem.