1. **What is the difference between precision and recall?**

Precision and recall are two important metrics used to evaluate the performance of classification algorithms, particularly in binary classification tasks:

**Precision**: Precision measures the accuracy of positive predictions made by a classifier. It calculates the ratio of true positive predictions to the total number of positive predictions made by the classifier. In other words, precision answers the question: "Of all the instances predicted as positive, how many are actually positive?"

**Formula**: Precision = TP / (TP + FP)

(TP = True Positives, FP = False Positives)

**Recall**: Recall, also known as sensitivity or true positive rate (TPR), measures the ability of a classifier to correctly identify all positive instances in the dataset. It calculates the ratio of true positive predictions to the total number of actual positive instances in the dataset. In other words, recall answers the question: "Of all the actual positive instances, how many were predicted correctly?"

**Formula**: Recall = TP / (TP + FN)

(FN = False Negatives)

In summary, precision focuses on the accuracy of positive predictions, while recall focuses on the completeness of positive predictions.

1. **What is cross-validation, and why is it important in binary classification?**

Cross-validation is a technique used to evaluate the performance of machine learning models by partitioning the dataset into multiple subsets, called folds. The model is trained on several combinations of these subsets and evaluated on the remaining subsets. This process is repeated multiple times, with each fold serving as both the training and testing dataset. The key idea behind cross-validation is to assess how well the model generalizes to unseen data by simulating the process of training on one subset and testing on another. Cross-validation is important in binary classification for several reasons:

It provides a more robust estimate of a model's performance by reducing the variance associated with a single train-test split.

It helps detect overfitting by evaluating the model's performance on multiple subsets of the data.

It ensures that the model's performance is not heavily influenced by the particular partitioning of the data into train and test sets.

It allows for a more reliable comparison of different models or algorithms by using the same data splits for evaluation.

It enables the identification of potential data leakage or bias issues that may arise from specific train-test splits.