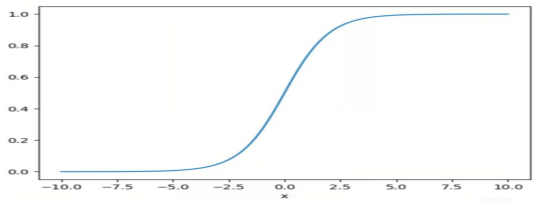
1. In logistic regression, what is the logistic function (sigmoid function) and how is it used to compute probabilities?

Ans:

Logistic regression is a powerful statistical technique widely used in machine learning and statistics for binary classification problems. One of its key components is the sigmoid function, which plays a crucial role in mapping inputs to probabilities.

The sigmoid function, also known as the logistic function, is a mathematical curve that has an S-shaped or sigmoidal curve. It is defined as follows:

$$f(x) = \frac{1}{1 + e^{-x}}$$



- f(x) represents the output of the sigmoid function.
- x is the input, which can take any real value.
- e is the base of the natural logarithm, approximately equal to 2.71828.

The sigmoid function forms an S shaped graph, which means as x approaches infinity, the probability becomes 1, and as x approaches negative infinity, the probability becomes 0. The model sets a threshold that decides what range of probability is mapped to which binary variable.

2. When constructing a decision tree, what criterion is commonly used to split nodes, and how is it calculated?

Ans:

A decision tree is one of the most powerful tools of supervised learning algorithms used for both classification and regression tasks. It builds a flowchart-like tree structure where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (terminal node) holds a class label.

It is one of the very powerful algorithms. And it is also used in Random Forest to train on different subsets of training data, which makes random forest one of the most powerful algorithms in machine learning.

- Parent and Child Node: A node that gets divided into sub-nodes is known as Parent Node, and these sub-nodes are known as Child Nodes. Since a node can be divided into multiple sub-nodes, it can act as a parent node of numerous child nodes.
- Root Node: The topmost node of a decision tree. It does not have any parent node. It represents the entire population or sample.
- Leaf / Terminal Nodes: Nodes of the tree that do not have any child node are known as Terminal/Leaf Nodes.

Node splitting, or simply splitting, divides a node into multiple sub-nodes to create relatively pure nodes. This is done by finding the best split for a node and can be done in multiple ways. The ways of splitting a node can be broadly divided into two categories based on the type of target variable:

- Continuous Target Variable: Reduction in Variance
- Categorical Target Variable: Gini Impurity, Information Gain, and Chi-Square

Attribute Selection Measures:

While implementing a Decision tree, the main issue arises that how to select the best attribute for the root node and for sub-nodes. So, to solve such problems there is a technique which is called as Attribute selection measure or ASM. By this measurement, we can easily select the best attribute for the nodes of the tree. There are two popular techniques for ASM, which are:

- Information Gain
- Gini Index

1. Information Gain:

Information gain is the measurement of changes in entropy after the segmentation of a dataset based on an attribute.
It calculates how much information a feature provides us about a class.
According to the value of information gain, we split the node and build the decision tree.

Information Gain= Entropy(S)- [(Weighted Avg) *Entropy(each feature)

Entropy: Entropy is a metric to measure the impurity in a given attribute. It specifies randomness in data. Entropy can be calculated as:

Entropy(s)= -P(yes)log2 P(yes)- P(no) log2 P(no)
Where,

S= Total number of samples
P(yes)= probability of yes
P(no)= probability of no

2. Gini Index:

Gini index is a measure of impurity or purity used while creating a decision tree in the CART(Classification and Regression Tree) algorithm.An attribute with the low Gini index should be preferred as compared to the high Gini index.It only creates binary splits, and the CART algorithm uses the Gini index to create binary splits.
Gini index can be calculated using the below formula:
Gini Index= 1- ∑jPj2

3. Explain the concept of entropy and information gain in the context of decision tree construction

Ans:

Entropy measures impurity in the data and information gain measures reduction in impurity in the data. The feature which has minimum impurity will be considered as the root node.

Information gain is used to decide which feature to split on at each step in building the tree. The term "entropy" comes from the study of thermodynamics, and it describes how chaotic or unpredictable a system is. Entropy is a measurement of a data set's impurity in the context of machine learning. In essence, it is a method of calculating the degree of uncertainty in a given dataset.

The following formula is used to compute entropy −

$$Entropy(S) = -p1\log_2 p1 - p2\log_2 p2 - \ldots - pn\log_2 pn$$

S is the data set, and p1 through pn are the percentages of various classes inside the data. The resultant entropy value is expressed in bits since the base 2 logarithm used in this method is typical.

- Entropy is a measurement of the disorder or impurity of a set of occurrences. It determines the usual amount of information needed to classify a sample taken from the collection.

- Entropy is calculated for a set of examples by calculating the probability of each class in the set and using that information in the entropy calculation

- Entropy is typically taken into account by decision trees for determining the best split.

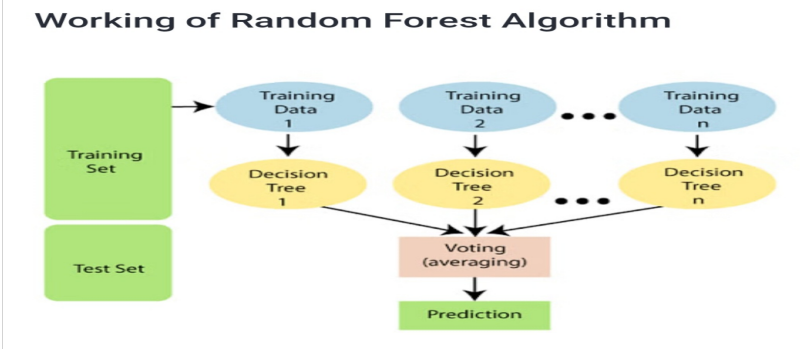
4. How does the random forest algorithm utilize bagging and feature randomization to

improve classification accuracy?

Ans:

Random Forest is a well-known machine learning algorithm from the supervised learning approach. It may be applied to both classification and regression issues in machine learning. It is built on the notion of ensemble learning, which is a method that involves integrating several classifiers to solve a complicated issue and enhance the model's performance.

"Random Forest is a classifier that comprises a number of decision trees on various subsets of the provided dataset and takes the average to enhance the predicted accuracy of that dataset ," as the name implies. Instead of depending on a single decision tree, the random forest collects the predictions from each tree and predicts the final output based on the majority vote of predictions.



The following steps explain the working Random Forest Algorithm:

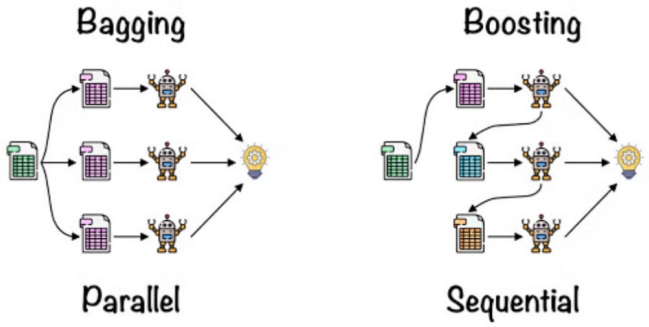
Step 1: Select random samples from a given data or training set.

Step 2: This algorithm will construct a decision tree for every training data.8

Step 3: Voting will take place by averaging the decision tree.

Step 4: Finally, select the most voted prediction result as the final prediction result. This

combination of multiple models is called Ensemble. Ensemble uses two methods:



Bagging: Creating a different training subset from sample training data with replacement is called Bagging. The final output is based on majority voting.

Boosting: Combing weak learners into strong learners by creating sequential models such that the final model has the highest accuracy is called Boosting. Example: ADA BOOST, XG BOOST.

5. What distance metric is typically used in k-nearest neighbors (KNN) classification, and how does it impact the algorithm's performance?
Ans:

Euclidean distance

The most intuitive and widely used distance metric for KNN is the Euclidean distance, which is the straight-line distance between two points in a vector space. It is calculated by taking the square root of the sum of the squared differences between the corresponding coordinates of the two points.
if you have two points A and B with coordinates (x1, y1) and (x2, y2), the Euclidean distance between them is: **d = sqrt((x1 - x2)^2 + (y1 - y2)^2)** The Euclidean distance is suitable for data that has continuous and numerical features with similar scales and ranges. It can also handle outliers and noise well, as it gives more weight to larger differences.

Manhattan distance

Another common distance metric for KNN is the Manhattan distance, which is the sum of the absolute differences between the corresponding coordinates of two points. It is also known as the city block or taxicab distance, as it represents the distance that a car would have to travel along a grid of streets to get from one point to another.
if you have two points A and B with coordinates (x1, y1) and (x2, y2), the Manhattan distance between them is**: d = abs(x1 - x2) + abs(y1 - y2)** The Manhattan distance is suitable for data that has discrete and categorical features, as it does not penalize small differences as much as the Euclidean distance

Minkowski distance

A more general distance metric for KNN is the Minkowski distance, which is a generalization of the Euclidean and Manhattan distances. It is defined by a parameter p that controls how much emphasis is given to larger or smaller differences between coordinates. For example, if you have two points A and B with coordinates (x1, y1) and (x2, y2), the Minkowski distance between them is: d **= (abs(x1 - x2)^p + abs(y1 - y2)^p)^(1/p)** The Minkowski distance can be seen as a family of distance metrics that includes the Euclidean distance (p = 2), the Manhattan distance (p = 1), and the Chebyshev distance (p = infinity), which is the maximum of the absolute differences between coordinates.

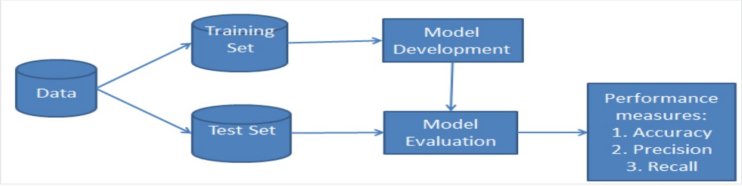
When selecting a distance metric to optimize your KNN algorithm, there is no one-size-fits-all solution. However, you can use some general guidelines to help you make the best choice. Analyzing your data and understanding the type, scale, range, and distribution of your features is essential. Experimenting with different distance metrics and comparing their results and performance on your data and problem is also key.

6. Describe the Naïve-Bayes assumption of feature independence and its implications for classification?
Ans:

Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.
It is mainly used in text classification that includes a high-dimensional training dataset.
Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.
It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.

The Naïve Bayes algorithm is comprised of two words Naïve and Bayes, Which can be described as:

- **Naïve:** It is called Naïve because it assumes that the occurrence of a certain feature is independent of the occurrence of other features. Such as if the fruit is identified on the bases of color, shape, and taste, then red, spherical, and sweet fruit is recognized as an apple. Hence each feature individually contributes to identify that it is an apple without depending on each other.
- **Bayes:** It is called Bayes because it depends on the principle of Bayes' Theorem.



Working of Naïve Bayes' Classifier:
Working of Naïve Bayes' Classifier can be understood with the help of the below example:

Suppose we have a dataset of weather conditions and corresponding target variable "Play". So using this dataset we need to decide that whether we should play or not on a particular day according to the weather conditions. So to solve this problem, we need to follow the below steps:

- Convert the given dataset into frequency tables.
- Generate Likelihood table by finding the probabilities of given features.
- Now, use Bayes theorem to calculate the posterior probability.

Naive Bayes classifier assumes that the effect of a particular feature in a class is independent of other features. For example, a loan applicant is desirable or not depending on his/her income, previous loan and transaction history, age, and location. Even if these features are interdependent, these features are still considered independently. This assumption simplifies

computation, and that's why it is considered as naive. This assumption is called class conditional independence.

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

- P(h): the probability of hypothesis h being true (regardless of the data). This is known as the prior probability of h.
- P(D): the probability of the data (regardless of the hypothesis). This is known as the prior probability.
- P(h|D): the probability of hypothesis h given the data D. This is known as posterior probability.
- P(D|h): the probability of data d given that the hypothesis h was true. This is known as posterior probability.

7. In SVMs, what is the role of the kernel function, and what are some commonly used kernel functions?
Ans:

Kernel Function is a method used to take data as input and transform it into the required form of processing data. “Kernel” is used due to a set of mathematical functions used in Support Vector Machine providing the window to manipulate the data. So, Kernel Function generally transforms the training set of data so that a non-linear decision surface is able to transform to a linear equation in a higher number of dimension spaces. Basically, It returns the inner product between two points in a standard feature dimension.

Support Vector Machines (SVMs) use kernel methods to transform the input data into a higher-dimensional feature space, which makes it simpler to distinguish between classes or generate predictions. Kernel approaches in SVMs work on the fundamental principle of implicitly mapping input data into a higher-dimensional feature space without directly computing the coordinates of the data points in that space.

The kernel function in SVMs is essential in determining the decision boundary that divides the various classes. In order to calculate the degree of similarity between any two points in the feature space, the kernel function computes their dot product.

Characteristics of Kernel Function

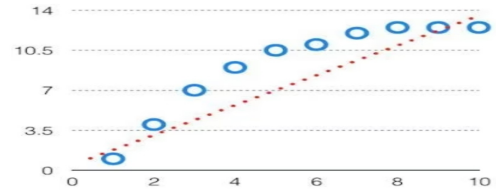
Kernel functions used in machine learning, including in SVMs (Support Vector Machines), have several important characteristics, including:

- **Mercer's condition**: A kernel function must satisfy Mercer's condition to be valid. This condition ensures that the kernel function is positive semi definite, which means that it is always greater than or equal to zero.
- **Positive definiteness**: A kernel function is positive definite if it is always greater than zero except for when the inputs are equal to each other.
- **Non-negativity**: A kernel function is non-negative, meaning that it produces non-negative values for all inputs.
- **Symmetry**: A kernel function is symmetric, meaning that it produces the same value regardless of the order in which the inputs are given.
- **Reproducing property**: A kernel function satisfies the reproducing property if it can be used to reconstruct the input data in the feature space.
- **Smoothness**: A kernel function is said to be smooth if it produces a smooth transformation of the input data into the feature space.
- **Complexity:** The complexity of a kernel function is an important consideration, as more complex kernel functions may lead to over fitting and reduced generalization performance.

8. Discuss the bias-variance tradeoff in the context of model complexity and overfitting.
Ans:

Bias:

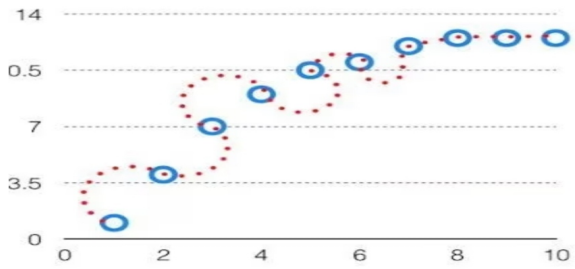
The bias is known as the difference between the prediction of the values by the Machine Learning model and the correct value. Being high in biasing gives a large error in training as well as testing data. It recommended that an algorithm should always be low-biased to avoid the problem of underfitting. By high bias, the data predicted is in a straight line format, thus not fitting accurately in the data in the data set. Such fitting is known as the Underfitting of Data. This happens when the hypothesis is too simple or linear in nature.



High Bias in the Model

variance

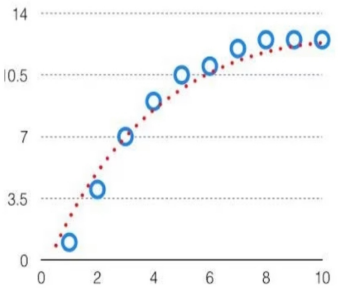
The variability of model prediction for a given data point which tells us the spread of our data is called the variance of the model. The model with high variance has a very complex fit to the training data and thus is not able to fit accurately on the data which it hasn't seen before. As a result, such models perform very well on training data but have high error rates on test data. When a model is high on variance, it is then said to as Overfitting of Data.



High Variance in the Model

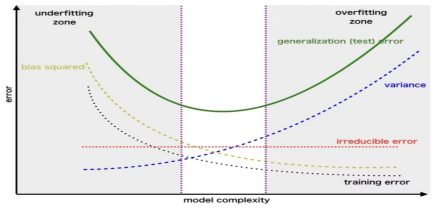
Bias Variance Tradeoff

If the algorithm is too simple (hypothesis with linear equation) then it may be on high bias and low variance condition and thus is error-prone. If algorithms fit too complex (hypothesis with high degree equation) then it may be on high variance and low bias. In the latter condition, the new entries will not perform well. Well, there is something between both of these conditions, known as a Trade-off or Bias Variance Trade-off. This tradeoff in complexity is why there is a tradeoff between bias and variance. An algorithm can't be more complex and less complex at the same time.



$$\text{Total Error} = \text{Bias}^2 + \text{Variance} + \text{Irreducible Error}$$

The best fit will be given by the hypothesis on the tradeoff point. The error to complexity graph to show trade-off is given as -



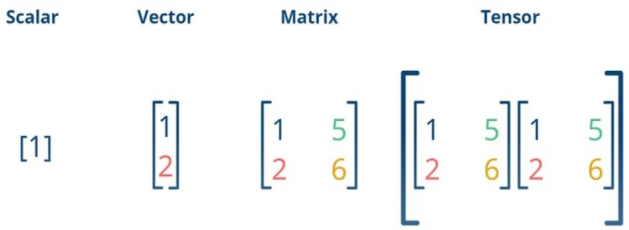
9. How does TensorFlow facilitate the creation and training of neural networks?
Ans:

TensorFlow, developed by Google, has emerged as one of the most popular and powerful open-source libraries for machine learning and artificial intelligence applications. With its versatile architecture and extensive set of tools, TensorFlow has become the go-to choice for both researchers and developers alike.

TensorFlow is an open-source programming language specifically designed for creating and training machine learning models. It was developed by Google and is widely used in various industries for a wide range of artificial intelligence (AI) applications. TensorFlow provides a flexible and efficient framework for building and deploying AI models, making it a popular choice among researchers and developers.

The name TensorFlow may seem a bit strange at first since there is no direct connection to Machine Learning. However, the name comes from the so-called tensors, which are used to train Deep Learning models and therefore form the core of TF.

The tensor is a mathematical function from linear algebra that maps a selection of vectors to a numerical value. The concept originated in physics and was subsequently used in mathematics. Probably the most prominent example that uses the concept of tensors is general relativity.



TensorFlow provides a comprehensive framework for building and training neural networks. It offers high-level APIs like Keras for easy model building and training, as well as lower-level operations for fine-grained control. TensorFlow's computational graph allows efficient execution of operations, and its automatic differentiation simplifies the process of calculating gradients for training. The extensive community support and pre-built models make it a powerful tool for both beginners and experts in deep learning.

10. Explain the concept of cross-validation and its importance in evaluating model performance.
Ans:

Cross-validation is a technique for validating the model efficiency by training it on the subset of input data and testing on previously unseen subset of the input data. We can also say that it is a technique to check how a statistical model generalizes to an independent dataset.

- Reserve a subset of the dataset as a validation set.
- Provide the training to the model using the training dataset.
- Now, evaluate model performance using the validation set. If the model performs well with the validation set, perform the further step, else check for the issues.

Methods used for Cross-Validation

There are some common methods that are used for cross-validation. These methods are given below:

- Validation Set Approach
- Leave-P-out cross-validation
- Leave one out cross-validation
- K-fold cross-validation
- Stratified k-fold cross-validation
- Hold-out cross-validation

Hold-out cross-validation

Hold-out cross-validation is the simplest and most common technique. You might not know that it is a hold-out method but you certainly use it every day.

k-Fold cross-validation
k-Fold cross-validation is a technique that minimizes the disadvantages of the hold-out method. k-Fold introduces a new way of splitting the dataset which helps to overcome the “test only once bottleneck”.

Leave-one-out cross-validation
Leave-one-out cross-validation (LOOCV) is an extreme case of k-Fold CV. Imagine if k is equal to n where n is the number of samples in the dataset. Such k-Fold case is equivalent to Leave-one-out technique.

Leave-p-out cross-validation
Leave-p-out cross-validation (LpOC) is similar to Leave-one-out CV as it creates all the possible training and test sets by using p samples as the test set. All mentioned about LOOCV is true and for LpOC.

Stratified k-Fold cross-validation
Stratified k-Fold is a variation of the standard k-Fold CV technique which is designed to be effective in such cases of target imbalance.

11. What techniques can be employed to handle overfitting in machine learning models?
Ans:

Overfitting occurs when the model fits more data than required, and it tries to capture each and every datapoint fed to it. Hence it starts capturing noise and inaccurate data from the dataset, which degrades the performance of the model.
An overfitted model doesn't perform accurately with the test/unseen dataset and can’t generalize well.
An overfitted model is said to have low bias and high variance.

Overfitting in the model can only be detected once you test the data. To detect the issue, we can perform Train/test split.

In the train-test split of the dataset, we can divide our dataset into random test and training datasets. We train the model with a training dataset which is about 80% of the total dataset. After training the model, we test it with the test dataset, which is 20 % of the total dataset.

Ways to prevent the Overfitting

- Early Stopping

- Train with more data
- Feature Selection
- Cross-Validation
- Data Augmentation
- Regularization
-

Early Stopping

Stopping the training process before the model starts capturing noise from the data is known as early stopping.

Train with More data

Increasing the training set by including more data can enhance the accuracy of the model, as it provides more chances to discover the relationship between input and output variables.

Feature Selection

In the feature selection process, we identify the most important features within training data, and other features are removed. Further, this process helps to simplify the model and reduces noise from the data. Some algorithms have the auto-feature selection, and if not, then we can manually perform this process.

Cross-Validation

Cross-validation is one of the powerful techniques to prevent overfitting.

In the general k-fold cross-validation technique, we divided the dataset into k-equal-sized subsets of data; these subsets are known as folds.

Data Augmentation

Data Augmentation is a data analysis technique, which is an alternative to adding more data to prevent overfitting. In this technique, instead of adding more training data, slightly modified copies of already existing data are added to the dataset.

Regularization

Regularization is the most popular technique to prevent overfitting. It is a group of methods that forces the learning algorithms to make a model simpler. Applying the regularization technique may slightly increase the bias but slightly reduces the variance

12.What is the purpose of regularization in machine learning, and how does it work?
Ans:

Regularization refers to techniques that are used to calibrate machine learning models in order to minimize the adjusted loss function and prevent overfitting or underfitting.
Using Regularization, we can fit our machine learning model appropriately on a given test set and hence reduce the errors in it.

Regularization Techniques
There are two main types of regularization techniques:

- Ridge Regularization and
- Lasso Regularization.

Ridge Regularization :

- Also known as Ridge Regression, it modifies the over-fitted or under fitted models by adding the penalty equivalent to the sum of the squares of the magnitude of coefficients.
- Ridge regression is a regularization technique, which is used to reduce the complexity of the model. It is also called as L2 regularization
- In this technique, the cost function is altered by adding the penalty term to it. The amount of bias added to the model is called Ridge Regression penalty. We can calculate it by multiplying with the lambda to the squared weight of each individual feature

Lasso Regression

- It modifies the over-fitted or under-fitted models by adding the penalty equivalent to the sum of the absolute values of coefficients
- Lasso regression is another regularization technique to reduce the complexity of the model. It stands for Least Absolute and Selection Operator
- It is also called as L1 regularization. The equation for the cost function of Lasso regression.

13. Describe the role of hyper-parameters in machine learning models and how they are tuned for optimal performance.
Ans:

Hyperparameters are parameters whose values control the learning process and determine the values of model parameters that a learning algorithm ends up learning. The prefix ‘hyper_’ suggests that they are ‘top-level’ parameters that control the learning process and the model parameters that result from it.

Some examples of Hyperparameters in Machine Learning

- The k in kNN or K-Nearest Neighbour algorithm
- Learning rate for training a neural network
- Train-test split ratio
- Batch Size

- Number of Epochs
- Branches in Decision Tree
- Number of clusters in Clustering Algorithm

Categories of Hyperparameters

Broadly hyperparameters can be divided into two categories, which are given below:

- Hyperparameter for Optimization
- Hyperparameter for Specific Models

Hyperparameter for Optimization

The process of selecting the best hyperparameters to use is known as hyperparameter tuning, and the tuning process is also known as hyperparameter optimization. Optimization parameters are used for optimizing the model.

Hyperparameter for Specific Models

Hyperparameters that are involved in the structure of the model are known as hyperparameters for specific models.

14.What are precision and recall, and how do they differ from accuracy in classification evaluation?

Ans:

Precision

Precision is a pivotal metric in classification tasks, especially in scenarios with a high cost of false positives. It provides insights into the model's ability to correctly predict positive instances while minimizing the risk of false alarms.

Precision, often referred to as the positive predictive value, quantifies the proportion of true positive predictions among all positive predictions made by the model. It answers the question: "Of all the instances predicted as positive, how many were positive?"

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

Where:

TP = True Positives

FP = False Positives

Significance

Precision is important when false positives are costly.

Recall

Recall, also known as sensitivity or true positive rate, is a crucial metric in classification that emphasizes the model's ability to identify all relevant instances.

Recall measures the proportion of actual positive cases correctly identified by the model. It answers the question: "Of all the actual positive instances, how many were correctly predicted by the model?"

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

Where:

TP = True Positives
FN = False Negatives

Significance

Recall is important in scenarios where False Negatives are costly.

- Accuracy shows how often a classification ML model is correct overall.
- Precision shows how often an ML model is correct when predicting the target class.
- Recall shows whether an ML model can find all objects of the target class.
- Consider the class balance and costs of different errors when choosing the suitable metric.

precision focuses on the accuracy of positive predictions, recall emphasizes capturing all actual positive instances, and accuracy assesses overall correctness. The choice between precision and recall depends on the specific goals of the classification task and the consequences of false positives and false negatives in the application.

15. Explain the ROC curve and how it is used to visualize the performance of binary classifiers?

Ans:

ROC curve

An ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters:

- True Positive Rate
- False Positive Rate

True Positive Rate (TPR) is a synonym for recall and is therefore defined as follows:

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

- It represents the proportion of actual positive instances correctly classified by the model

False Positive Rate (FPR) is defined as follows:

- It indicates the proportion of actual negative instances incorrectly classified as positive

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

An ROC curve plots TPR vs. FPR at different classification thresholds. Lowering the classification threshold classifies more items as positive, thus increasing both False Positives and True Positives.

By varying the classification threshold , you can observe how the trade-off between sensitivity and specificity changes . A diagonal line from (0,0) to (1,1) represents random guessing , and a good classifier's ROC curve should be situated towards the upper-left corner, indicating higher sensitivity and lower false positive rate.

The Area Under the ROC Curve (AUC-ROC) is a single metric derived from the curve. A higher AUC-ROC value (closer to 1) suggests better classifier performance, as it signifies a better ability to distinguish between positive and negative instances.