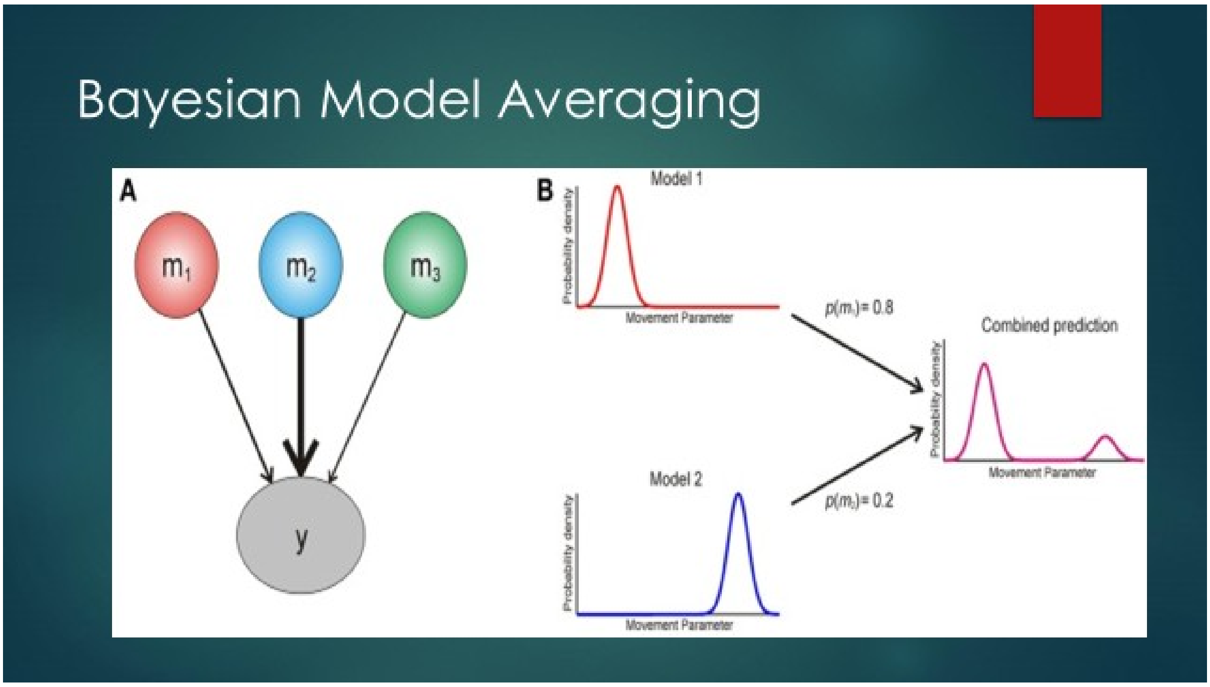
MACHINE LEARNING(21CSC305P)

UNIT 5 : SYLLABUS

Bayesian model averaging, Boosting, Adaptive basis function models, CART, generalized additive models, Ensemble learning

1. BMA (Bayesian Model Averaging (BMA))

Bayesian Model Averaging (BMA) is a statistical method used to account for model uncertainty in predictive modeling. It involves averaging over multiple models to make predictions rather than relying on a single "best" model. Each model is weighted according to its posterior probability, which is derived using Bayesian inference based on the model's fit to the data and its prior probability. The resulting predictions reflect a combination of different models, capturing uncertainty more effectively.



Bayesian Model Averaging

Steps of BMA:

Model Specification: Start with candidate models, each representing a different hypothesis about the data.

Model Fitting: Each model is fitted to the data, usually using Bayesian inference techniques, which produce posterior distributions of the model parameters given the data.

Model Comparison: Bayesian model comparison methods, such as Bayes factors are used to assess the relative fit of each model to the data.

Model Averaging: Once the posterior probabilities of the models are calculated, predictions from each model are combined, weighted by their posterior probabilities.

This weighted average provides a more robust prediction than any individual model.

Advantages of BMA:

MACHINE LEARNING(21CSC305P)

UNIT 5 : SYLLABUS

Bayesian model averaging, Boosting, Adaptive basis function models, CART, generalized additive models, Ensemble learning

1. **Reduces Overfitting:** By averaging over several models, BMA avoids the risk of overfitting associated with choosing a single model that may not generalize well to new data.

2. **Incorporates Model Uncertainty:** Traditional approaches often neglect model uncertainty, which can lead to underestimation of prediction error. BMA addresses this by integrating predictions from multiple models.

3. **Improved Predictive Performance:** BMA can enhance predictive accuracy by combining the strengths of different models. When some models perform well in certain scenarios and poorly in others, averaging can lead to better overall performance.

4. **Automatic Model Selection:** BMA can implicitly perform model selection through the process of weighting models according to their posterior probabilities.

Examples of Scenarios Where BMA is Beneficial:

1. **Medical Decision Making:** In predicting patient outcomes based on various risk factors, multiple models can be considered, each using different subsets of risk factors. BMA can combine these models to produce more reliable predictions, reflecting uncertainties in model choice.

2. **Economic Forecasting:** When predicting economic indicators, various models based on different economic theories or assumptions may exist. BMA can help aggregate these models to account for uncertainty and provide a more accurate forecast.

3. **Climate Modeling:** Given the complexity of climate systems, multiple models are often developed to predict climate changes. BMA can help combine the output of different models to provide a more comprehensive and robust prediction.

4. **Genomics and Proteomics:** In biological data analysis, there are often many potential predictors (genes, proteins). BMA can be used to combine different models that use various subsets of these predictors, leading to better predictions of disease risk or drug response.

Bayesian Model Averaging (BMA) offers several advantages for prediction compared to selecting a single best model:

1. **Uncertainty Quantification**: BMA provides a systematic way to account for uncertainty in model selection. Instead of committing to a single model, it averages over multiple models weighted by their posterior probabilities, allowing for more robust predictions that consider model uncertainty.

2. **Improved Predictive Performance**: By averaging predictions from multiple models, BMA can improve overall predictive accuracy, especially in situations where no single model consistently outperforms the others. This is particularly useful in complex datasets where different models may capture different aspects of the data.

3. **Bias Reduction**: Relying on a single model can lead to bias if that model is not representative of the underlying data-generating process. BMA helps mitigate this bias by incorporating information from multiple models.

4. **Flexibility**: BMA can incorporate a variety of models, allowing it to adapt to different aspects of the data. This is beneficial in scenarios where the underlying relationships are complex or unknown.

5. **Robustness to Overfitting**: A single best model may overfit the training data, leading to poor generalization on new data. BMA tends to be more robust against overfitting since it considers multiple models and their respective performances.

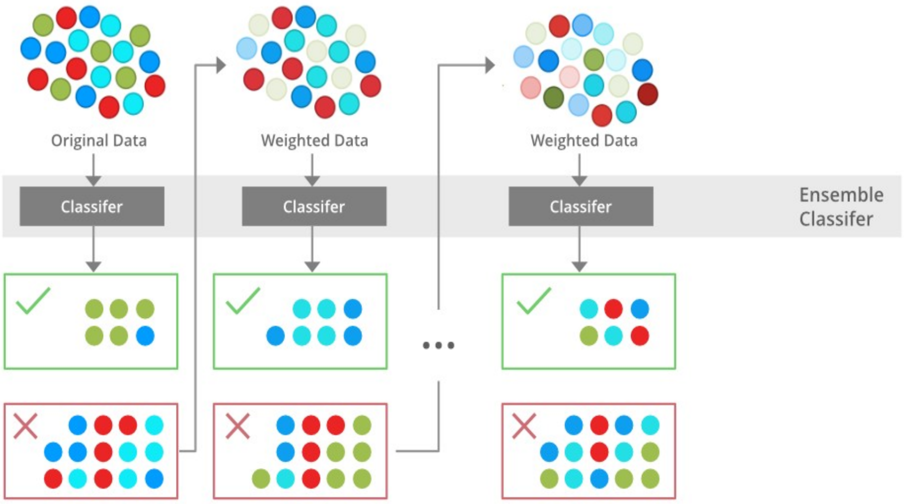
6. **Model Comparison**: BMA naturally incorporates model comparison into the prediction process. It allows for the evaluation of how well different models perform and helps identify the most relevant models for prediction.

7. **Combining Strengths of Multiple Models**: Different models may excel in different regions of the input space. BMA leverages this by combining models, which can lead to better overall performance compared to a single best model that may not perform well across all regions.

8. **Ease of Interpretation**: BMA provides a clear interpretation of how different models contribute to predictions, as it quantifies the importance of each model in the averaging process.

2. BOOSTING

Boosting and bagging are both ensemble learning techniques used in machine learning to improve the performance of models by combining multiple weak learners, but they do so in fundamentally different ways. Here's a breakdown of each concept and their key differences:



MACHINE LEARNING(21CSC305P)

UNIT 5 : SYLLABUS

Bayesian model averaging, Boosting, Adaptive basis function models, CART, generalized additive models, Ensemble learning

Boosting

Concept:

- Boosting is an iterative ensemble technique that combines multiple weak learners (often decision trees) to create a strong predictive model. The main idea is to train models sequentially, where each new model focuses on the errors made by the previous ones.

How it Works:

1. **Initialization**: Start with a base model (often a decision tree with low depth) and train it on the dataset.
2. **Error Focus**: Evaluate the model's performance and identify the instances it misclassified.
3. **Re-weighting**: Adjust the weights of these misclassified instances, giving them more importance in the training of the next model.
4. **Model Training**: Train a new model on the re-weighted dataset, which allows it to focus on the previously misclassified instances.
5. **Combining Predictions**: The final prediction is made by aggregating the predictions of all models, often using a weighted majority vote.

Popular Algorithms: AdaBoost, Gradient Boosting, XGBoost, LightGBM.

BAGGING

Bagging

Concept:

- Bagging (Bootstrap Aggregating) is a parallel ensemble technique that builds multiple models independently and then combines their predictions. The primary goal is to reduce variance and improve stability.

How it Works:

1. **Bootstrap Sampling**: Create multiple subsets of the training data by randomly sampling with replacement (bootstrapping). Each subset is typically of the same size as the original dataset but will contain some duplicate instances and leave out others.
2. **Model Training**: Train a separate model (e.g., decision tree) on each of these subsets independently.
3. **Combining Predictions**: Aggregate the predictions from all models, usually by averaging (for regression) or majority voting (for classification).

MACHINE LEARNING(21CSC305P)

UNIT 5 : SYLLABUS

Bayesian model averaging, Boosting, Adaptive basis function models, CART, generalized additive models, Ensemble learning

Popular Algorithms: Random Forest (which is essentially an extension of bagging with decision trees).

Key Differences

Feature	**Boosting**	**Bagging**
Model Training	Sequential (models are trained one after another)	Parallel (models are trained independently)
Focus	Emphasizes correcting the errors of previous models	Reduces variance by averaging multiple models
Data Sampling	Uses the entire dataset, but changes the instance weights	Uses bootstrap samples (subsets of the dataset)
Model Contribution	Weights models based on their performance	All models contribute equally to the final prediction
Overfitting	More prone to overfitting, but can model complex relationships	Generally reduces overfitting by averaging out noise
Interpretability	Can be harder to interpret due to the sequential nature	Easier to interpret since it aggregates independently trained models

3. ADAPTIVE BASIS FUNCTION MODEL

An adaptive basis function model is a type of mathematical framework used in various fields, such as machine learning, signal processing, and function approximation. The core idea is to represent a target function as a linear combination of basis functions that can adapt based on the data.

Key Concepts:

1. **Basis Functions**: These are functions that serve as building blocks for constructing more complex functions. Common types include polynomials, wavelets, and radial basis functions.
2. **Adaptivity**: The model adapts its basis functions based on the characteristics of the data. This can involve selecting the most relevant basis functions or adjusting their parameters to better fit the target function.
3. **Applications**:
   - **Regression Analysis**: Used for fitting complex relationships in data.
   - **Neural Networks**: Can be viewed as a type of adaptive basis function model where neurons act as basis functions.

    - **Function Approximation**: Helps in approximating unknown functions based on observed data.

4. **Training Process**: The model typically involves a training phase where parameters are optimized using techniques like gradient descent, regularization, or other optimization methods.
5. **Flexibility**: The adaptivity allows the model to capture complex patterns and trends in the data, making it versatile for various applications.

Advantages:

- Can model nonlinear relationships effectively.
- Generally provides better generalization for complex data sets compared to fixed basis models.

Challenges:

- Computational complexity can increase with the number of basis functions.
- Overfitting can occur if the model becomes too flexible.

Overall, adaptive basis function models are powerful tools for capturing intricate patterns in data and are widely used in modern computational methods.

A common example of an adaptive basis function model is Radial Basis Function (RBF) networks. Here's how it works:

Radial Basis Function Network

1.Structure:An RBF network consists of an input layer, a hidden layer with radial basis functions as neurons, and a linear output layer.

2. Basis Functions:The hidden layer typically uses Gaussian functions as basis functions. Each neuron in this layer computes the distance from the input to a center point (the neuron's weight) and applies the Gaussian function.

3. Adaptivity:The network adapts by adjusting the centers and widths of the Gaussian functions based on the training data. This is usually done through a process called K-means clustering to find suitable centers, followed by optimization techniques (like gradient descent) to fine-tune the parameters.

4. Output Layer:The output is a linear combination of the outputs from the hidden layer, allowing the network to model complex functions.

Example Application

Function Approximation: Suppose you want to approximate a nonlinear function, such as: f(x) = sin(2pi x)

1. Data Generation: Generate a set of input-output pairs by sampling the function at various points.

2. Model Training: Use an RBF network to fit this data. During training, the network will adapt its basis functions based on the sampled points, effectively capturing the behaviour of the sine function.

3. Result: Once trained, the RBF network can predict values for new inputs by computing the weighted sum of the Gaussian outputs, providing a smooth approximation of the sine function.

Benefits

- Nonlinearity: RBF networks can handle complex, nonlinear relationships.
- Flexibility: The adaptively allows the model to generalize well to new data.
- This makes RBF networks a practical example of an adaptive basis function model in action!

4. CART ALGORITHM

CART (Classification and Regression Trees) is a machine learning algorithm used for predictive modeling. It constructs decision trees based on the features of the input data and is capable of performing both classification and regression tasks. Here's a detailed overview of the CART algorithm:

Key Features of Cart algorithm

1. **Decision Trees**: CART creates a tree-like model where internal nodes represent tests on features, branches represent outcomes of these tests, and leaf nodes represent predicted outcomes (either class labels in classification or continuous values in regression).
2. **Splitting**: The algorithm recursively splits the data into subsets based on the feature values that result in the most significant reduction in impurity. This process continues until a stopping criterion is met (e.g., maximum depth of the tree, minimum number of samples in a node).
3. **Impurity Measures**:
   - For classification tasks, CART typically uses measures like Gini impurity or entropy to determine the quality of a split.

   - For regression tasks, it often uses mean squared error (MSE) to evaluate splits.

4. **Binary Trees**: CART produces binary trees, meaning each internal node has at most two branches (left and right).
5. **Pruning**: To avoid overfitting, CART may prune the tree after it has been built. This involves removing sections of the tree that provide little predictive power, based on cross-validation.

Steps of the CART Algorithm

1. **Select the Best Feature to Split**: Evaluate each feature and determine the best threshold that minimizes impurity for the resulting subsets.
2. **Create Subsets**: Split the dataset into subsets based on the chosen feature and threshold.
3. **Repeat**: For each subset, repeat the process recursively until a stopping criterion is reached.
4. **Assign Predictions**:
   - In classification, assign the most common class label in the leaf node.
   - In regression, assign the mean value of the target variable for the samples in the leaf node.
5. **Prune the Tree (if necessary)**: Evaluate the tree’s performance and prune it to enhance generalization.

Advantages of CART

- **Interpretability**: Decision trees are easy to visualize and understand, making them interpretable.
- **Handling Non-linearity**: CART can model complex relationships without requiring transformation of the features.
- **Feature Importance**: CART can provide insights into which features are most important for predictions.

Disadvantages of CART

- **Overfitting**: If not pruned properly, CART can fit noise in the data, leading to poor generalization.
- **Instability**: Small changes in the data can lead to a completely different tree structure, making CART sensitive to data variations.

Applications of CART

- Medical diagnosis (classifying patients based on symptoms)
- Credit scoring (predicting the likelihood of default)
- Marketing (targeting customer segments)

MACHINE LEARNING(21CSC305P)

UNIT 5 : SYLLABUS

Bayesian model averaging, Boosting, Adaptive basis function models, CART, generalized additive models, Ensemble learning

CART (Classification and Regression Trees) is designed to handle both classification and regression tasks, adapting its approach based on the nature of the output variable. Here's how it works for each type of task:

1. CART for Classification Tasks

In classification, the goal is to predict a categorical outcome (class label) based on input features. The process involves the following steps:

- **Data Preparation**:
  - Input features can be numerical or categorical.
  - The target variable is a categorical variable representing different classes.
- **Building the Tree**:
  - The algorithm evaluates splits based on impurity measures like Gini impurity or entropy.
  - For each candidate split, CART calculates how well the split separates the classes. The split that results in the highest reduction in impurity is chosen.
- **Creating Leaf Nodes**:
  - Once the tree reaches a stopping criterion (e.g., maximum depth, minimum samples per leaf), the final leaf nodes are assigned class labels.
  - The class label for a leaf node is usually determined by majority voting; that is, the most frequent class among the samples in that node.
- **Prediction**:
  - To make predictions, new input data is passed through the decision tree, following the splits until reaching a leaf node. The predicted class is the label of that leaf node.

Example:

- If a dataset consists of features such as age, income, and education level, and the target variable is whether a person buys a product (yes/no), CART can create a decision tree that splits the data based on these features to classify new individuals.

2. CART for Regression Tasks

In regression, the goal is to predict a continuous outcome based on input features. The steps involved are similar but focus on numerical values:

- **Data Preparation**:
  - Input features are still numerical or categorical.

  - The target variable is a continuous variable (e.g., price, temperature).

- **Building the Tree**:
  - CART uses mean squared error (MSE) to evaluate potential splits. It looks for the split that minimizes the variance within the resulting subsets.
  - The process is similar to classification, but instead of calculating impurity based on class labels, it focuses on reducing the difference between the predicted and actual values.
- **Creating Leaf Nodes**:
  - When the tree construction is complete, each leaf node is assigned a value, typically the mean of the target variable for the samples that fall into that node.
- **Prediction**:
  - For new input data, the process is the same: the data is passed through the decision tree until a leaf node is reached, and the predicted value is the mean value of that node.

ART (Classification and Regression Trees) is designed to handle both classification and regression tasks, adapting its approach based on the nature of the output variable. Here’s how it works for each type of task:

1. CART for Classification Tasks

In classification, the goal is to predict a categorical outcome (class label) based on input features. The process involves the following steps:

- **Data Preparation**:
  - Input features can be numerical or categorical.
  - The target variable is a categorical variable representing different classes.
- **Building the Tree**:
  - The algorithm evaluates splits based on impurity measures like Gini impurity or entropy.
  - For each candidate split, CART calculates how well the split separates the classes. The split that results in the highest reduction in impurity is chosen.
- **Creating Leaf Nodes**:
  - Once the tree reaches a stopping criterion (e.g., maximum depth, minimum samples per leaf), the final leaf nodes are assigned class labels.
  - The class label for a leaf node is usually determined by majority voting; that is, the most frequent class among the samples in that node.

- **Prediction**:
    - To make predictions, new input data is passed through the decision tree, following the splits until reaching a leaf node. The predicted class is the label of that leaf node.

2. CART for Regression Tasks

In regression, the goal is to predict a continuous outcome based on input features. The steps involved are similar but focus on numerical values:

- **Data Preparation**:
    - Input features are still numerical or categorical.
    - The target variable is a continuous variable (e.g., price, temperature).
- **Building the Tree**:
    - CART uses mean squared error (MSE) to evaluate potential splits. It looks for the split that minimizes the variance within the resulting subsets.
    - The process is similar to classification, but instead of calculating impurity based on class labels, it focuses on reducing the difference between the predicted and actual values.
- **Creating Leaf Nodes**:
    - When the tree construction is complete, each leaf node is assigned a value, typically the mean of the target variable for the samples that fall into that node.
- **Prediction**:
    - For new input data, the process is the same: the data is passed through the decision tree until a leaf node is reached, and the predicted value is the mean value of that node.
    - ❖ **Pruning** in CART (Classification and Regression Trees) is a technique used to reduce the size of a decision tree after it has been fully grown. The primary purpose of pruning is to prevent **overfitting**, which occurs when a model learns not only the underlying patterns in the training data but also the noise, resulting in poor generalization to unseen data.

Importance of Pruning in Preventing Overfitting

1. **Simplicity**:
    - A pruned tree is often simpler and more interpretable. Simpler models are easier to understand and communicate, which is valuable in many applications.
2. **Generalization**:

- Pruning helps to improve the model's ability to generalize to new, unseen data. By removing parts of the tree that are too specific to the training data (e.g., capturing noise), the model can perform better on test datasets.

3. **Reduced Variance**:
    - Overfitting typically results in high variance in the model, meaning it will perform well on training data but poorly on validation or test data. Pruning helps to lower the variance by simplifying the model, making it more robust.
4. **Improved Predictive Performance**:
    - Through pruning, the model can maintain accuracy while becoming less complex. In many cases, a pruned tree may achieve similar or better performance compared to an unpruned tree on validation datasets.

5. GENERALIZED ADDITIVE MODEL (GAM)

A **Generalized Additive Model (GAM)** is a statistical learning method that extends traditional linear models by allowing for a more flexible relationship between the dependent and independent variables. In a traditional linear model, the relationship between the input features and the output is assumed to be linear:

$y=\beta0+f1x1+f2x2+\ldots+fnXn+\epsilon$

Here β coefficients represent the linear relationship, and ϵ\epsilonϵ is the error term.

In contrast, **GAM** introduces non-linear functions for each predictor, making it possible to model more complex relationships:

$y=\beta0+f1(x1)+f2(x2)+\ldots+fn(xn)+\epsilon$

Here, fi(xi) are smooth, non-linear functions of the predictor variables, which allows GAM to capture non-linear patterns in the data. The functions fi are often estimated using techniques like splines or smoothing functions.

Key Differences Between GAM and Traditional Linear Models:

1. **Flexibility**: GAMs are more flexible than traditional linear models because they can model non-linear relationships between the predictors and the response variable, whereas linear models assume linearity.
2. **Interpretability**: Even though GAMs are non-linear, they retain some interpretability because the effect of each predictor can be visualized individually, making it possible to understand how each predictor influences the response.
3. **Regularization**: In GAMs, smoothness penalties can be applied to the functions fif_ifi, controlling the complexity of the model and preventing overfitting.

4. **Use of Non-parametric Functions**: Traditional linear models use parametric coefficients to describe relationships, whereas GAMs use non-parametric functions, providing more flexibility in capturing different types of patterns.

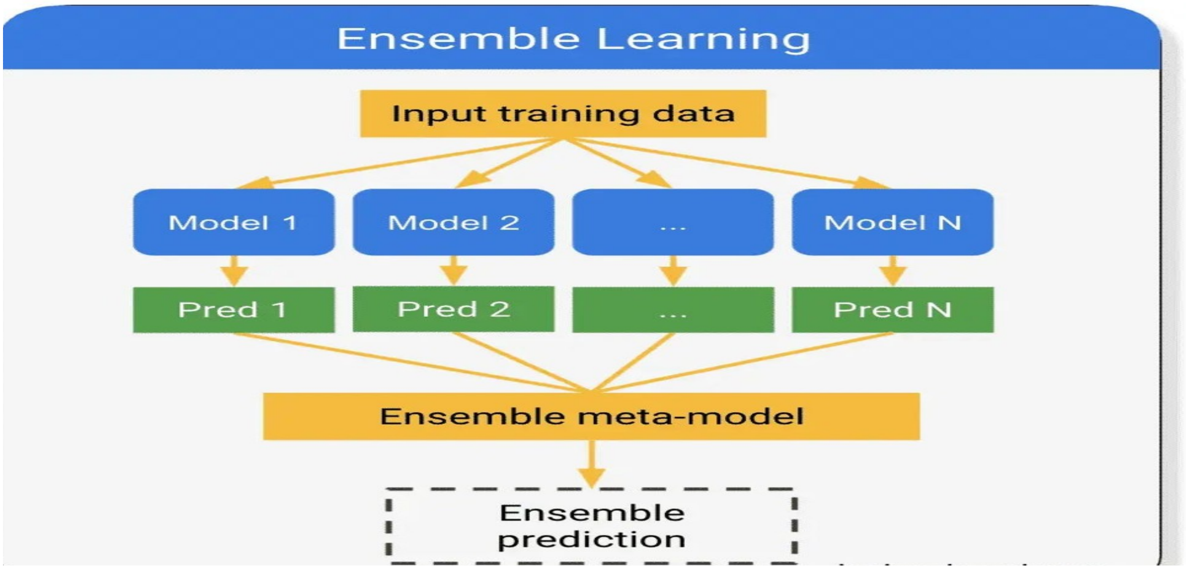
Example

For example, in a dataset where the response variable depends non-linearly on predictors like temperature and humidity, a traditional linear model might not fit well. A GAM can be used to model the non-linear relationship, resulting in better predictive accuracy while maintaining interpretability.

In summary, GAMs generalize linear models by allowing non-linear relationships for each predictor, thus providing more modelling flexibility while maintaining the interpretability of linear models.

6. ENSEMBLE LEARNING

Ensemble learning is a machine learning technique where multiple models, often referred to as "weak learners" or "base models," are combined to solve a particular problem and improve the performance of a single model. The goal is to create a more robust and accurate model by leveraging the strengths of each individual model while compensating for their weaknesses. Ensemble methods are particularly useful for improving the predictive accuracy and robustness of models in both classification and regression tasks.



Types of Ensemble Learning Methods

1. **Bagging (Bootstrap Aggregating)**:

UNIT 5 : SYLLABUS

Bayesian model averaging, Boosting, Adaptive basis function models, CART, generalized additive models, Ensemble learning

- In bagging, multiple models are trained independently on different subsets of the training data, which are created using bootstrapping (random sampling with replacement). Each model is trained separately on these different samples, and their predictions are combined using averaging (for regression) or voting (for classification).
- **Example**: Random Forest is an example of a bagging-based ensemble algorithm, where multiple decision trees are trained on different subsets of the data, and the final prediction is made by averaging the output of all trees (regression) or majority voting (classification).

2. **Boosting**:
   - Boosting focuses on training models sequentially, where each new model attempts to correct the errors made by the previous models. The final model is a weighted sum of all the models, giving more emphasis to models that performed well.
   - **Example**: Gradient Boosting Machines (GBM), AdaBoost, and XGBoost are popular boosting algorithms where subsequent models are trained with a focus on the mistakes made by the earlier models.
3. **Stacking (Stacked Generalization)**:
   - Stacking involves training multiple base models on the original dataset and then using another model, known as a "meta-learner" or "stacker," to combine their predictions. The meta-learner tries to find the optimal way to blend the predictions of the base models.
   - **Example**: In a classification task, the base models might be logistic regression, decision tree, and support vector machine (SVM), and the meta-learner could be another model that takes the outputs of these models as input features to make the final prediction.
4. **Voting**:
   - In voting-based ensemble methods, the predictions of multiple models are aggregated by taking a majority vote (classification) or averaging (regression).
   - **Types of Voting**:
     - **Hard Voting**: In classification, the class label that gets the majority vote is the final prediction.
     - **Soft Voting**: Uses the average predicted probabilities and selects the class label with the highest average probability.

Advantages of Ensemble Learning

- **Improved Accuracy**: By combining the strengths of multiple models, ensemble methods can achieve higher predictive accuracy.

MACHINE LEARNING(21CSC305P)

UNIT 5 : SYLLABUS

Bayesian model averaging, Boosting, Adaptive basis function models, CART, generalized additive models, Ensemble learning

- **Robustness**: They help reduce overfitting and increase the model's generalization capability.
- **Versatility**: Can be applied to different types of machine learning algorithms.

Disadvantages of Ensemble Learning

- **Complexity**: Ensemble models are more complex to implement and interpret than individual models.
- **Computational Cost**: Training multiple models requires more computational resources.