NAIVE BAYES ALGORITHM CLASSIFIER IN MACHINE LEARNING

What is Naive Bayes Classifier?

- The Naive Bayes Classifier is a probabilistic machine learning algorithm based on Bayes' Theorem.
- It is widely used for classification tasks and assumes that all features are independent of each other, an assumption that simplifies calculations and speeds up computation.
- Despite the independence assumption being rarely true in real-world data, the algorithm performs exceptionally well in many scenarios.

Why is it called Naive Bayes?

- The Naive Bayes algorithm is called “naive” because it makes a strong assumption: all the features in the dataset are independent of each other. This means that the presence of one feature does not affect the presence of another.
- In reality, many features are correlated, and this assumption may not always hold. However, despite this simplification, the Naive Bayes algorithm often performs well in practice. Its simplicity allows for quick computation, making it a popular choice for tasks like text classification and spam detection.

Dataset

- Consider a fictional dataset that describes the weather conditions for playing a game of golf. Each row in the dataset records the weather, temperature, and whether the conditions were suitable for playing. The goal is to predict whether it’s fit (“Yes”) or unfit (“No”) for playing golf based on the weather conditions. Below is a tabular representation of our dataset.

Weather	Temperature	Play?
Sunny	Hot	No
Sunny	Hot	No
Overcast	Hot	Yes
Rainy	Mild	Yes
Rainy	Cool	Yes
Rainy	Cool	No
Overcast	Cool	Yes
Sunny	Mild	No
Sunny	Cool	Yes
Rainy	Mild	Yes
Sunny	Mild	Yes
Overcast	Mild	Yes
Overcast	Hot	Yes
Rainy	Mild	No

Bayes' Theorem

- Bayes' Theorem is a fundamental concept that forms the basis of the Naive Bayes algorithm.
- It calculates the posterior probability of an event occurring based on prior knowledge and evidence.
- In simple terms, it helps answer: "Given some observations, how likely is this event to belong to a particular class?"
- The mathematical formula for Bayes' Theorem is:

- Where:
- P(A | B): Posterior probability – probability of hypothesis A given that B is true.
- P(B | A): Likelihood – probability of observing B given that A is true.
- P(A): Prior probability – the initial probability of A before seeing any evidence.
- P(B): Marginal probability of observing B.

$$P(A \mid B) = \frac{P(B|A) \times P(A)}{P(B)}$$

How Bayes' Theorem Works in Naive Bayes Classifier

- In a Naive Bayes classifier, Bayes' Theorem is used to calculate the probability of a class based on given features. For instance, if we want to predict whether it is fit to play golf on a Sunny day with Mild temperature, the classifier will compute two probabilities:
- P(Play = Yes | Sunny, Mild) – Probability of playing on a sunny, mild day.
- P(Play = No | Sunny, Mild) – Probability of not playing on a sunny, mild day.
- Whichever probability is higher becomes the final prediction.

Naive Assumption

- In the Naive Bayes algorithm, we assume that all features are independent of each other, which simplifies the computation.
- This is known as the "naive" assumption. Although this assumption may not always be true in practice, it often leads to good results.

Simplified Example Calculation

- Let's consider our dataset. To compute the probability of Play = Yes on a Sunny day with Mild temperature, we need the following:

- Similarly, the algorithm will calculate P(Play = No | Sunny, Mild). The class with the higher probability will be the predicted outcome.

$$P(\text{Play} = \text{Yes} \mid \text{Sunny}, \text{Mild}) = \frac{P(\text{Sunny}|\text{Play}=\text{Yes}) \times P(\text{Mild}|\text{Play}=\text{Yes}) \times P(\text{Play}=\text{Yes})}{P(\text{Sunny}, \text{Mild})}$$

Bayes' Theorem: Implementation (Example)

- To demonstrate how Bayes' Theorem works, let's predict whether we can play outside on a sunny day with mild temperature using the dataset.
- The dataset has two features: Weather (Sunny, Overcast, Rainy) and Temperature (Hot, Mild, Cool). The target variable is Play (Yes/No). We'll calculate the probabilities step by step.
- Goal:
- Predict if Play = Yes or Play = No for Weather = Sunny and Temperature = Mild.

Step 1: Calculate Prior Probabilities

- We first calculate the prior probabilities for each class:

$$P(\text{Play} = \text{Yes}) = \frac{\text{Number of Yes outcomes}}{\text{Total number of outcomes}} = \frac{9}{14}$$

$$P(\text{Play} = \text{No}) = \frac{5}{14}$$

Step 2: Calculate Likelihood Probabilities

- Next, we calculate the likelihood probabilities for each feature value based on the given class:
- For Weather = Sunny:
- For Temperature = Mild:

$$P(\text{Sunny} \mid \text{Play} = \text{Yes}) = \frac{\text{Sunny and Play} = \text{Yes}}{\text{Total Yes outcomes}} = \frac{2}{9}$$

$$P(\text{Sunny} \mid \text{Play} = \text{No}) = \frac{3}{5}$$

$$P(\text{Mild} \mid \text{Play} = \text{Yes}) = \frac{\text{Mild and Play} = \text{Yes}}{\text{Total Yes outcomes}} = \frac{4}{9}$$

$$P(\text{Mild} \mid \text{Play} = \text{No}) = \frac{2}{5}$$

Step 3: Apply Bayes' Theorem

- Now, we calculate the posterior probabilities for both classes using Bayes' Theorem.
- Similarly, we calculate the probability for Play = No:

$$P(\text{Play} = \text{Yes} \mid \text{Sunny}, \text{Mild}) = \frac{2}{9} \times \frac{4}{9} \times \frac{9}{14}$$

$$P(\text{Play} = \text{Yes} \mid \text{Sunny}, \text{Mild}) = \frac{8}{126} = 0.0635$$

$$P(\text{Play} = \text{No} \mid \text{Sunny}, \text{Mild}) = P(\text{Sunny} \mid \text{Play} = \text{No}) \times P(\text{Mild} \mid \text{Play} = \text{No}) \times P(\text{Play} = \text{No})$$

$$P(\text{Play} = \text{No} \mid \text{Sunny}, \text{Mild}) = \frac{3}{5} \times \frac{2}{5} \times \frac{5}{14} = \frac{30}{350} = 0.0857$$

Step 4: Make the Prediction

- Since P(Play = No | Sunny, Mild) is greater than P(Play = Yes | Sunny, Mild), the classifier will predict:
- Prediction: Play = No

Working of Naive Bayes Classifier

- The Naive Bayes classifier works in two main stages: training and prediction.
- Below is a step-by-step breakdown of how the algorithm classifies new data using the dataset we introduced earlier.

Step 1: Training the Model

- In the training phase, the algorithm calculates:
- 1. Prior Probability:
- The probability of each class in the dataset.

Example:

$$P(\text{Play} = \text{Yes}) = \frac{\text{Number of Yes outcomes}}{\text{Total number of outcomes}} = \frac{9}{14}$$

$$P(\text{Play} = \text{No}) = \frac{5}{14}$$

2. Conditional Probability:

- The probability of each feature value given a class.

Example:

$$P(\text{Sunny} \mid \text{Play} = \text{Yes}) = \frac{\text{Number of Sunny days when Play=Yes}}{\text{Total Yes outcomes}} = \frac{2}{9}$$

$$P(\text{Sunny} \mid \text{Play} = \text{No}) = \frac{3}{5}$$

- **3. Smoothing (Optional):**
- If a feature value has zero occurrences for a class, it can make the probability zero. Smoothing techniques like Laplace Smoothing are used to handle this.

Step 2: Making Predictions

- To predict the class of a new data point (e.g., a Sunny day with Mild temperature), the classifier calculates the posterior probability for each class.

$$P(\text{Play} = \text{Yes} \mid \text{Sunny}, \text{Mild}) = P(\text{Sunny} \mid \text{Play} = \text{Yes}) \times P(\text{Mild} \mid \text{Play} = \text{Yes}) \times P(\text{Play} = \text{Yes})$$

Similarly:

$$P(\text{Play} = \text{No} \mid \text{Sunny}, \text{Mild}) = P(\text{Sunny} \mid \text{Play} = \text{No}) \times P(\text{Mild} \mid \text{Play} = \text{No}) \times P(\text{Play} = \text{No})$$

- The classifier will select the class with the highest probability as the prediction.

Summary of the Workflow

- Training Phase: Calculate prior and conditional probabilities from the dataset.
- Prediction Phase: Use Bayes' Theorem to find the class with the highest probability.
- Output: The predicted class is assigned to the new data point.

Types of Naive Bayes Model

- **1. Gaussian Naive Bayes**
- In Gaussian Naive Bayes, continuous values associated with each feature are assumed to be distributed according to a Gaussian distribution. A Gaussian distribution is also called Normal distribution When plotted, it gives a bell shaped curve which is symmetric about the mean of the feature values

- **2. Multinomial Naive Bayes**
- Multinomial Naive Bayes is used when features represent the frequency of terms (such as word counts) in a document. It is commonly applied in text classification, where term frequencies are important.
- **3. Bernoulli Naive Bayes**
- Bernoulli Naive Bayes deals with binary features, where each feature indicates whether a word appears or not in a document. It is suited for scenarios where the presence or absence of terms is more relevant than their frequency. Both models are widely used in document classification tasks

Advantages of Naive Bayes Classifier

- Simplicity and Ease of Implementation: Naive Bayes is straightforward to implement and requires less computational power compared to other algorithms.
- Fast Training and Prediction: Since it involves simple probability calculations, Naive Bayes is efficient for both training and making predictions, even on large datasets.
- Performs Well with High-Dimensional Data: The algorithm works effectively even with datasets that contain many features, such as text classification where each word becomes a feature.
- Works Well with Small Datasets: Naive Bayes can perform reliably even with limited training data, as it generalizes well using prior probabilities.
- Handles Categorical and Text Data: It is widely used in spam filtering, sentiment analysis, and news classification due to its efficiency with categorical data.

Disadvantages of Naive Bayes Classifier

- Naive Assumption of Feature Independence: The algorithm assumes that all features are independent of each other, which is rarely true in real-world datasets. This can reduce its accuracy if the features are highly correlated.
- Zero Probability Issue: If a feature value does not appear in the training data for a given class, the algorithm assigns zero probability to that class. This can be mitigated using Laplace Smoothing.

- Limited with Continuous Features: Naive Bayes struggles with continuous features unless they follow a normal distribution, as required in Gaussian Naive Bayes. Otherwise, data preprocessing may be necessary.
- Sensitive to Irrelevant Features: Including irrelevant or noisy features can negatively impact the classifier's performance, as the algorithm gives equal importance to all features.
- Not Suitable for Complex Relationships: The algorithm may not perform well when there are complex relationships between features, as it lacks the capability to model feature interactions.

Applications

- Spam Email Filtering: Classifies emails as spam or non-spam based on features.
- Text Classification: Used in sentiment analysis, document categorization and topic classification.
- Medical Diagnosis: Helps in predicting the likelihood of a disease based on symptoms.
- Credit Scoring: Evaluates creditworthiness of individuals for loan approval.
- Weather Prediction: Classifies weather conditions based on various factors.