ASSIGNMENT 6 :- Naive Bayes

Name: Sahil Vishwas Shendkar  
Roll No: TYITB118  
Assignment No.: 4  
AIM: Assignment on Naive Bayes  
PREREQUISITE: Python programming

**THEORY:**

**Naïve Bayes** is a fundamental and powerful **supervised machine learning algorithm** used for **classification tasks**. It is based on **Bayes’ Theorem** from probability theory and is called "naïve" because it assumes that all features (variables) in the dataset are **independent** of each other given the class label—an assumption that is rarely true in reality but simplifies the computation.

Despite its simplicity and the unrealistic independence assumption, Naïve Bayes often performs **exceptionally well** in practice, especially for problems involving **high-dimensional data** such as text classification, spam detection, sentiment analysis, and medical diagnoses.

Naïve Bayes is particularly effective when the goal is to classify input data into predefined categories, based on the probabilities of features occurring within each class.

**DATABASE: Salary Dataset**

For this assignment, we use the **Salary Dataset**, a structured dataset often employed in classification and predictive analytics. The objective is to **predict salary categories** (such as “Above 50K” or “Below 50K”) based on various personal and professional features.

**Typical Features in the Salary Dataset Include:**

* **Age** – Numeric value indicating the age of the individual
* **Education Level** – Categorical feature indicating the highest qualification
* **Workclass** – Type of employment (e.g., Private, Government, Self-employed)
* **Occupation** – Job type (e.g., Manager, Technician, Sales)
* **Marital Status** – Married, Single, Divorced, etc.
* **Hours Worked per Week** – Numeric value
* **Gender** – Male or Female
* **Native Country** – Country of residence
* **Salary** – The target variable, usually categorized (e.g., >50K or <=50K)

In this case, the goal of using Naïve Bayes is to **classify an individual’s salary category** based on the other attributes. Since many features are categorical and independent in nature, Naïve Bayes is a suitable algorithm for this dataset.

**Concept of Naïve Bayes Classification**

Naïve Bayes works by calculating the **probability** that a given input belongs to a particular class based on the observed values of the features. It does this for all classes and selects the class with the **highest probability**.

Here’s how it operates conceptually:

1. It **learns the frequency** of feature values in each class from the training data.
2. Then, it **applies probability** rules to estimate the likelihood that a new data point belongs to each class.
3. Finally, it **assigns the class** with the highest estimated probability.

In the context of the **Salary Dataset**, Naïve Bayes calculates the probability that a person earns >50K or <=50K based on features like education, hours worked, and occupation.

**Types of Naïve Bayes Models**

There are different variations of the Naïve Bayes algorithm depending on the type of input data:

**1. Gaussian Naïve Bayes**

Used when features are **continuous (numeric)** and assumed to follow a **normal distribution** (e.g., age, hours per week).

**2. Multinomial Naïve Bayes**

Used for **discrete count data**, like word frequencies in text documents.

**3. Bernoulli Naïve Bayes**

Best suited for **binary features**, where variables represent true/false or 1/0 type data.

In the case of the **Salary Dataset**, a combination of **Gaussian and Bernoulli models** may be used, depending on how the features are encoded.

**Working Mechanism of Naïve Bayes**

The Naïve Bayes classifier works in the following steps:

1. **Data Preparation**:
   * Convert categorical variables into a suitable format using techniques like label encoding or one-hot encoding.
   * Divide the dataset into training and testing sets.
2. **Calculate Prior Probabilities**:
   * Determine the overall proportion of each class (e.g., the proportion of people earning >50K vs. <=50K).
3. **Calculate Likelihood for Each Feature**:
   * For each feature, calculate the probability of each value occurring within each class.
4. **Predict on New Data**:
   * For any new record, compute the probability for each class based on the feature values and assign the class with the highest probability.

This process is **fast**, **scalable**, and can handle **large datasets with many features** efficiently.

**Application of Naïve Bayes to the Salary Dataset**

Let’s consider an example:

A record represents a 35-year-old individual with a Master's degree, working in the private sector, with 45 hours per week. Based on the training data:

* Naïve Bayes calculates the probability that this person falls into the >50K and <=50K categories.
* The class with the **higher probability** is chosen as the prediction.

This approach is particularly effective in understanding **what combinations of features** are most associated with high or low salaries. For example:

* People with higher education levels and longer working hours may be more likely to earn above 50K.
* Certain job types may have a strong correlation with income brackets.

**Advantages of Naïve Bayes**

* **Simple and Fast**: Quick to train and predict, even with large datasets.
* **Performs Well with Categorical Data**: Efficient with discrete features.
* **Scales Well**: Handles high-dimensional data better than many complex algorithms.
* **Robust to Irrelevant Features**: Performs well even when some features do not contribute significantly.
* **Effective with Small Training Data**: Requires less training data compared to other models.

**Disadvantages of Naïve Bayes**

* **Assumption of Feature Independence**: Assumes features are independent, which is rarely true in real-life data.
* **Zero-Frequency Problem**: If a feature value was not seen in the training set, it assigns a probability of zero. This is usually addressed using techniques like **Laplace Smoothing**.
* **Not Suitable for Continuous Feature Relationships**: Doesn't capture interactions between continuous variables as well as other models like Decision Trees or SVM.
* **May Oversimplify Relationships**: Useful for initial classification, but more complex models may outperform it in precision.

Despite its limitations, Naïve Bayes is a **strong baseline classifier** and a great tool for quick prototyping and understanding data relationships.

**Conclusion:**

The **Naïve Bayes Algorithm** is a simple yet powerful classification technique that uses probability to make predictions based on feature values. When applied to the **Salary Dataset**, it helps classify whether a person is likely to earn more or less than 50K based on their age, education, job type, and working hours. Its speed, simplicity, and effectiveness with high-dimensional data make it an ideal choice for early-stage classification tasks and feature analysis.

While the model assumes independence between features—which may not always hold true—it still provides **accurate and interpretable results** for many real-world problems. It is especially effective in domains such as **HR analytics, income prediction, and financial risk modeling**. As a result, Naïve Bayes remains one of the most essential and widely used tools in the machine learning toolbox.