ASSIGNMENT NO. 6

AIM: Assignment on Naive Bayes Classification using the Adult Income Dataset.

OBJECTIVES:

- To understand the principles of the Naive Bayes classification algorithm.
- To implement the Naive Bayes algorithm using Python and scikit-learn.
- To perform classification on the Adult Income dataset.
- To evaluate the model's performance using accuracy, precision, recall, and F1-score.

THEORY: Naive Bayes is a probabilistic classification technique based on Bayes' Theorem with the assumption of independence between predictors. It is called "naive" because it assumes that the presence of one feature in a class is independent of the presence of other features.

Despite its simplicity, Naive Bayes can perform surprisingly well in many real-world situations, especially for large datasets with high-dimensional features such as text classification and spam filtering.

Bayes' Theorem:

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

Where:

- $P(A|B)$: Posterior probability
- $P(B|A)$: Likelihood
- $P(A)$: Prior probability
- $P(B)$: Evidence

Types of Naive Bayes:

1. Gaussian Naive Bayes: Assumes continuous values follow a Gaussian (normal) distribution.
2. Multinomial Naive Bayes: Used for discrete counts (e.g., word counts in documents).
3. Bernoulli Naive Bayes: Used for binary/boolean features.

Dataset:

The dataset used is adult.csv, also known as the Adult Income dataset or Census Income dataset, which contains demographic information and income labels of individuals. The goal is to classify whether a person earns more or less than $50K annually.

Attributes include

- Age
- Workclass
- Education
- Marital Status
- Occupation
- Race
- Gender
- Hours-per-week
- Native Country
- Income (target class)

Methodology:

1. **Import Libraries**
   Import necessary libraries like pandas, NumPy, matplotlib, seaborn, and scikit-learn modules.

2. **Load the Dataset**
   Load the adult.csv dataset using pandas and inspect the data structure.

3. **Exploratory Data Analysis (EDA)**
   Analyze feature distributions, class balance, and detect any inconsistencies or missing values.

4. **Data Preprocessing**
   Handle missing values, encode categorical features, and clean irrelevant data if needed.

5. **Train-Test Split (70:30)**
   Split the dataset into training and testing sets using 70% for training and 30% for testing.

6. **Train the Naive Bayes Model**
   Use GaussianNB or CategoricalNB from scikit-learn to train the model.

7. **Make Predictions**
   Predict income class on the test dataset.

8. **Evaluate the Model**
   Use accuracy score, precision, recall, F1-score, and confusion matrix.

9. **Visualize Confusion Matrix**
   Plot the confusion matrix using seaborn's heatmap.

10. **Test the Model on Sample Inputs**
    Run the model on new sample inputs to predict income class.

Model Implementation:

- Used GaussianNB from the sklearn.naive_bayes module.
- Trained the model on the training data.
- Predicted income category on the test data.
- Evaluated performance using metrics such as:
  - Accuracy
  - Precision
  - Recall
  - F1-Score
  - Confusion Matrix

Advantages of Naive Bayes:

1. Simple and fast to implement.
2. Works well with high-dimensional data.
3. Performs well even with a small amount of training data.

Limitations:

1. Assumes feature independence, which may not always hold true.
2. Not suitable for datasets with highly correlated features.

Conclusion:

Naive Bayes is an effective and efficient classification algorithm, particularly for categorical datasets. On the Adult Income dataset, it successfully classifies whether individuals earn more or less than $50K per year. While simple in its assumptions, the algorithm performs competitively with more complex models when applied properly with preprocessing and clean data.