Experiment Number 4

**Aim:**Implementation of Bayesian Algorithm

**Theory:**To implement the **Bayesian Algorithm** on your Sales Management System, you would typically apply **Naive Bayes** or a similar variant of Bayesian inference for classification and predictive analysis. Bayesian algorithms work well when you need to predict the likelihood of a certain outcome based on historical data, which is common in sales prediction, customer segmentation, or inventory forecasting.

In this case, you can apply the Bayesian algorithm to predict customer purchases, classify sales trends, or predict inventory needs based on sales data. Here's how you can implement it:

### **Steps for Implementing Bayesian Algorithm**

1. **Prepare the Dataset**
   * Extract relevant data from the OLAP cube or data warehouse. For example, you may want to focus on the CustomerDim, SalesFact, and CarDim tables.
   * Key features could be:
     + **Customer Attributes**: Age, Region, Gender
     + **Car Attributes**: Car Brand, Car Model, Price
     + **Sales Attributes**: Quantity Sold, Sales Amount, Date
2. **Define the Problem**
   * You might want to predict the likelihood of a customer purchasing a certain car based on their profile, or predict which car model is likely to sell more in the next quarter based on historical sales data.
   * Let’s say the target variable is **Car Purchase (yes/no)** or **Sales Amount** in a probabilistic sense.
3. **Calculate Probabilities**
   * Apply Bayes’ Theorem: P(A∣B)=P(B∣A)⋅P(A)P(B)P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}P(A∣B)=P(B)P(B∣A)⋅P(A)​
     + **P(A|B)**: Posterior probability of event A (e.g., customer buys a car) given B (e.g., age, gender, region, etc.)
     + **P(B|A)**: Likelihood of feature B given A (i.e., customer’s attributes given they bought a car).
     + **P(A)**: Prior probability (e.g., overall probability of a car purchase in the dataset).
     + **P(B)**: Normalizing constant, representing the marginal likelihood of observing the data.
4. **Training the Naive Bayes Classifier**
   * Assuming feature independence, which is the hallmark of Naive Bayes, you can train the algorithm using customer and sales data.

**Code:**

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.naive\_bayes import GaussianNB

from sklearn.metrics import accuracy\_score

from sklearn.preprocessing import LabelEncoder

# Sample dataset

data = {

'CustomerAge': [25, 34, 45, 30, 52, 28, 40, 23, 33, 38],

'CustomerRegion': ['Urban', 'Rural', 'Urban', 'Suburban', 'Urban', 'Rural', 'Suburban', 'Urban', 'Rural', 'Suburban'],

'CustomerGender': ['Male', 'Female', 'Male', 'Female', 'Female', 'Male', 'Male', 'Female', 'Female', 'Male'],

'CarPrice': [25000, 32000, 28000, 45000, 22000, 35000, 41000, 27000, 29000, 31000],

'QuantitySold': [1, 2, 1, 3, 1, 2, 1, 1, 2, 1],

'CarPurchase': ['Yes', 'Yes', 'No', 'Yes', 'No', 'Yes', 'No', 'Yes', 'Yes', 'No']

}

# Convert to DataFrame

df = pd.DataFrame(data)

# Encode categorical variables using LabelEncoder

label\_encoder = LabelEncoder()

df['CustomerRegion'] = label\_encoder.fit\_transform(df['CustomerRegion'])

df['CustomerGender'] = label\_encoder.fit\_transform(df['CustomerGender'])

df['CarPurchase'] = label\_encoder.fit\_transform(df['CarPurchase']) # Target column

# Features and target

X = df[['CustomerAge', 'CustomerRegion', 'CustomerGender', 'CarPrice', 'QuantitySold']]

y = df['CarPurchase']

# Split the dataset into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)

# Instantiate the model

model = GaussianNB()

# Train the model

model.fit(X\_train, y\_train)

# Predict on the test set

y\_pred = model.predict(X\_test)

# Evaluate the model's accuracy

accuracy = accuracy\_score(y\_test, y\_pred)

print(f'Model Accuracy: {accuracy}')

**Output:**



**Dataset:**



**Conclusion:**

In summary, the Naive Bayes classifier, combined with proper data preprocessing, provides a useful and interpretable approach for predicting customer decisions in sales management systems. Future work could involve refining the model with more features, exploring other machine learning algorithms, and testing with larger datasets to improve prediction accuracy and generalization.