**Experiment-No.11**

**Objective:** Write a program to implement naive bayes classification problem

|  |  |  |
| --- | --- | --- |
| **Scheduled Date:** | **Compiled Date:** | **Submitted Date:** |
| 26 Nov 2024 | 26 Nov 2024 | 26 Nov 2024 |

### ****Description of Naive Baye’s Classification:****

Naive Bayes classifiers are a family of classification algorithms based on Bayes' Theorem. They assume independence between every pair of features while classifying. Despite this naive assumption, they perform well in many real-world scenarios.

For example, consider a dataset with weather conditions (sunny, rainy, overcast), temperatures (hot, mild, cool), and a target (play) indicating whether it is suitable to play. Naive Bayes can predict the outcome (yes or no) for given weather and temperature values.

### ****Algorithm for Naive Bayes Classification:****

1. **Data Preparation:**
   * Define three lists: weather, temp, and play (target variable).
   * Encode categorical variables (weather, temp, play) into numerical form using LabelEncoder.
2. **Feature Combination:**
   * Combine encoded weather and temp data into a list of tuples called features.
3. **Model Training:**
   * Import GaussianNB from sklearn.naive\_bayes.
   * Train the Naive Bayes model using features and the encoded play data.
4. **Prediction:**
   * Provide new feature inputs to the model using predict() to obtain the predicted class (yes or no).
5. **Output Result:**
   * Display the encoded data, feature combinations, and predictions for the provided inputs.

**Python Code for naive bayes classification problem :**

weather = ['sunny', 'sunny', 'overcast', 'rainy', 'rainy', 'rainy', 'overcast',

           'sunny', 'rainy', 'sunny', 'overcast', 'overcast', 'rainy']

temp = ['hot', 'hot', 'hot', 'mild', 'cool', 'cool', 'cool', 'mild',

        'cool', 'mild', 'mild', 'hot', 'hot']

play = ['no', 'no', 'yes', 'yes', 'no', 'yes', 'no', 'yes',

        'yes', 'no', 'yes', 'no', 'no']

from sklearn import preprocessing

le = preprocessing.LabelEncoder()

weather\_encoded = le.fit\_transform(weather)

temp\_encoded = le.fit\_transform(temp)

play\_encoded = le.fit\_transform(play)

print("Encoded Weather:", weather\_encoded)

print("Encoded Temperature:", temp\_encoded)

print("Encoded Play Decision:", play\_encoded)

features = tuple(zip(weather\_encoded, temp\_encoded))

print("Features:", features)

from sklearn.naive\_bayes import GaussianNB

model = GaussianNB()

model.fit(features, play\_encoded)

predicted = model.predict([[1, 2]])

print("Predicted Value for [1, 2]:", "Yes" if predicted[0] == 1 else "No")

predicted = model.predict([[2, 2]])

print("Predicted Value for [2, 2]:", "Yes" if predicted[0] == 1 else "No")

Output :

Encoded Weather: [2, 2, 0, 1, 1, 1, 0, 2, 1, 2, 0, 0, 1]

Encoded Temperature: [1, 1, 1, 2, 0, 0, 0, 2, 0, 2, 2, 1, 1]

Encoded Play Decision: [0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0]

Features: [(2, 1), (2, 1), (0, 1), (1, 2), (1, 0), (1, 0), (0, 0), (2, 2), (1, 0), (2, 2), (0, 2), (0, 1), (1, 1)]

Predicted Value for [1, 2]: Yes

Predicted Value for [2, 2]: No