Naïve Bayes Classifier

Aim:
1. To implement the Naïve Bayes' algorithm for classification.
2. Train a Naïve Bayes' model using a given dataset and calculate class probabilities.
3. Evaluate the accuracy of the model on test data and analyze the results.

Theory:

Naïve Bayes is a family of probabilistic machine learning algorithms based on Bayes' Theorem. It's particularly useful for classification tasks, where you want to predict the class label of an input based on its features. Despite its "Naïve" assumption, Naïve Bayes has been shown to work surprisingly well in many real-world scenarios, especially in text classification and other high-dimensional datasets.

1. Bayes' Theorem:
Naïve Bayes is built upon Bayes' Theorem, which is a fundamental concept in probability theory. Bayes' Theorem describes how to update the probability of a hypothesis (an event) based on new evidence. It's written as:

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

Where:
P(A | B) is the probability of event A given event B.
P(B | A) is the probability of event B given event A.
P(A) is the prior probability of event A.
P(B) is the evidence or marginal probability of event B.

2. Naïve Assumption:
The "Naïve" in Naïve Bayes refers to the assumption that the features (attributes) used to predict the class are conditionally independent of each other given the class label. In other words, the presence or absence of a particular feature doesn't affect the presence or absence of any other feature. This simplifies the calculations significantly and makes the algorithm computationally efficient.

3. Training:
During the training phase, Naïve Bayes estimates the probabilities needed for classification. It calculates the prior probabilities of each class based on the training data and then calculates the likelihood of each feature given each class.

For example, if you're classifying emails as "spam" or "not spam," you'd estimate the probability of certain words appearing in spam emails versus non-spam emails.

4. Prediction:
When making a prediction for a new input, Naïve Bayes calculates the probability of the input belonging to each class based on the features. It uses Bayes' Theorem to compute the posterior probability for each class, and the class with the highest posterior probability is chosen as the predicted class.

Types of Naïve Bayes Classifiers:
There are several variants of Naïve Bayes classifiers, including:

	- Gaussian Naïve Bayes: Assumes that the features follow a Gaussian distribution. - Multinomial Naïve Bayes: Used for discrete data, often in text classification (e.g., counting word occurrences). - Bernoulli Naïve Bayes: Used for binary data, such as presence or absence of features. - Categorical Naïve Bayes: Used for categorical data, where features have discrete values.

The following is the Python code to implement Naïve Bayes Classifier

(To run this code, Scikit-Learn must be installed (pip install scikit-learn).

Python Code

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score

# Load the Iris dataset
iris = load_iris()
X = iris.data
y = iris.target

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Create a Naive Bayes classifier (Gaussian Naive Bayes for continuous
features)
clf = GaussianNB()

# Train the classifier on the training data
clf.fit(X_train, y_train)

# Make predictions on the test data
y_pred = clf.predict(X_test)

# Calculate and print the accuracy
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

Another example with the following datasets

Match	Argentina	France	Result
1	1	0	Argentina
2	0	0	Draw
3	3	4	France
4	2	0	Argentina
5	0	0	Draw
6	1	0	Argentina
7	0	0	Draw
8	2	1	Argentina
9	0	2	France
10	1	0	Argentina
11	2	0	Argentina
12	3	4	France
13	4	2	Argentina

(To run this code, Pandas must be installed (pip install pandas).

```
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score
import pandas as pd

# Load the football dataset from CSV
dataset = pd.read_csv('argfrc_dataset.csv')

# Separate features (X) and labels (y)
X = dataset[['Argentina', 'France']].values
y = dataset['Result'].values

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Create a Naïve Bayes classifier (Gaussian Naïve Bayes for continuous
features)
clf = GaussianNB()

# Train the classifier on the training data
clf.fit(X_train, y_train)

# Make predictions on the test data
y_pred = clf.predict(X_test)

# Calculate and print the accuracy
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

Output	

For Video demonstration of the practical click on the link or scan the QR-code

https://youtu.be/_tcjQjh2rEE	