NAÏVE BAYES CLASSIFIER :

NAME: NAGAVENI L G

SRN: PES2UG21CS315

SEC:5F

Code:

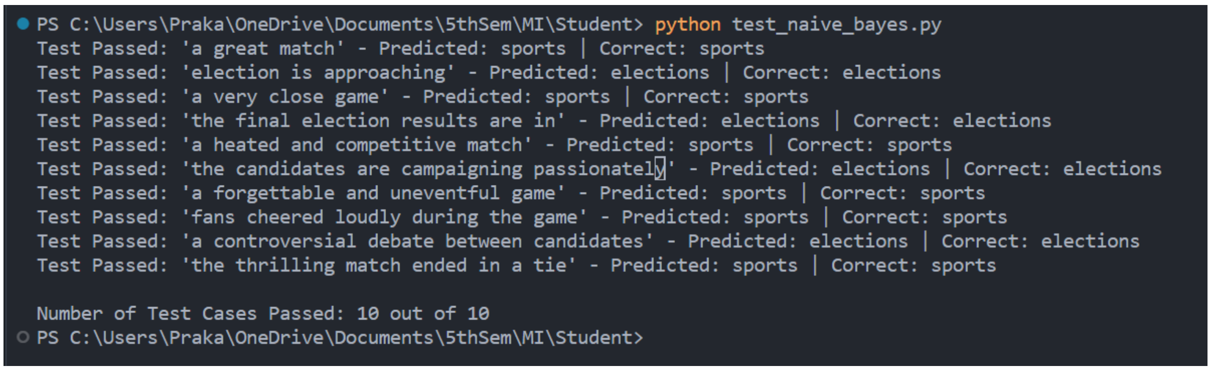
```
import numpy as np
import warnings
warnings.filterwarnings("ignore", category = RuntimeWarning)


class NaiveBayesClassifier:
    """
    A simple implementation of the Naive Bayes Classifier for text
classification.
    This classifier assumes that the input features are binary word
occurrences in documents.

    Attributes:
        None

    Methods:
        fit(X, y):
            Trains the Naive Bayes Classifier using the provided training
data.

            Args:
                X (numpy.ndarray): The training data matrix where each row
represents a document
                                  and each column represents the presence (1)
or absence (0) of a word.
                y (numpy.ndarray): The corresponding labels for the training
documents.

            Returns:
                tuple: A tuple containing two dictionaries:
                    - class_probs (dict): Prior probabilities of each class in
the training set.
```

```
                    - word_probs (dict): Conditional probabilities of words
given each class.

        predict(X, class_probs, word_probs, classes):
            Predicts the classes for the given test data using the trained
classifier.

            Args:
                X (numpy.ndarray): The test data matrix where each row
represents a document
                                  and each column represents the presence (1)
or absence (0) of a word.
                class_probs (dict): Prior probabilities of each class obtained
from the training phase.
                word_probs (dict): Conditional probabilities of words given
each class obtained from training.
                classes (numpy.ndarray): The unique classes in the dataset.

            Returns:
                list: A list of predicted class labels for the test documents.

    Usage Example:
    The functions you write will be called as given below:

        # Instantiate the classifier
        nb_classifier = NaiveBayesClassifier()

        # Train the classifier
        class_probs, word_probs = nb_classifier.fit(X_train, y_train)

        # Predict using the trained classifier
        predicted_labels = nb_classifier.predict(X_test, class_probs,
word_probs, classes)

    Note: You do not have to call these functions. They will be called when
you run the test_naive_bayes.py
    """

    @staticmethod
    def fit(X, y):
        """
        Trains the Naive Bayes Classifier using the provided training data.

        Args:
            X (numpy.ndarray): The training data matrix where each row
represents a document
                              and each column represents the presence (1) or
absence (0) of a word.
```

```
            y (numpy.ndarray): The corresponding labels for the training
documents.

        Returns:
            tuple: A tuple containing two dictionaries:
                - class_probs (dict): Prior probabilities of each class in the
training set.
                - word_probs (dict): Conditional probabilities of words given
each class.
        """

        # Write your code here
        classes = np.unique(y)
        class_probs = {}
        word_probs = {}
        total_documents = len(y)

        y=np.array(y)

        # Calculate class probabilities
        for c in classes:
            class_count = np.sum(y == c)
            class_probs[c] = class_count / total_documents

        # Calculate word probabilities
        for c in classes:
            # Filter documents of class c
            X_c = X[y == c]

            # Calculate the conditional probabilities of each word for class c
            word_probs[c] = (np.sum(X_c, axis=0) + 1) / (np.sum(X_c) + 2)

        return class_probs, word_probs

    @staticmethod
    def predict(X, class_probs, word_probs, classes):
        """
        Predicts the classes for the given test data using the trained
classifier.

        Args:
            X (numpy.ndarray): The test data matrix where each row represents
a document
                              and each column represents the presence (1) or
absence (0) of a word.
            class_probs (dict): Prior probabilities of each class obtained
from the training phase.
```

```
            word_probs (dict): Conditional probabilities of words given each
class obtained from training.
            classes (numpy.ndarray): The unique classes in the dataset.

        Returns:
            list: A list of predicted class labels for the test documents.
        """
        predictions = []

        # Write your code here
        for document in X:
            max_prob = -1
            predicted_class = None

            for c in classes:
                # Calculate the probability of the document belonging to class
c
                prob_c = np.log(class_probs[c]) + np.sum(np.log(word_probs[c]
** document))

                # Update the predicted class if a higher probability is found
                if prob_c > max_prob or predicted_class is None:
                    max_prob = prob_c
                    predicted_class = c

            predictions.append(predicted_class)

        return predictions
```

OUTPUT :



```
● PS C:\Users\Praka\OneDrive\Documents\5thSem\MI\Student> python test_naive_bayes.py
  Test Passed: 'a great match' - Predicted: sports | Correct: sports
  Test Passed: 'election is approaching' - Predicted: elections | Correct: elections
  Test Passed: 'a very close game' - Predicted: sports | Correct: sports
  Test Passed: 'the final election results are in' - Predicted: elections | Correct: elections
  Test Passed: 'a heated and competitive match' - Predicted: sports | Correct: sports
  Test Passed: 'the candidates are campaigning passionately' - Predicted: elections | Correct: elections
  Test Passed: 'a forgettable and uneventful game' - Predicted: sports | Correct: sports
  Test Passed: 'fans cheered loudly during the game' - Predicted: sports | Correct: sports
  Test Passed: 'a controversial debate between candidates' - Predicted: elections | Correct: elections
  Test Passed: 'the thrilling match ended in a tie' - Predicted: sports | Correct: sports

  Number of Test Cases Passed: 10 out of 10
○ PS C:\Users\Praka\OneDrive\Documents\5thSem\MI\Student>
```