exp5.py

```python
'''
5) Write a program to implement the naïve Bayesian classifier for a sample training data set stored as a
.CSV file. Compute the accuracy of the classifier, considering few test data sets.
'''
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from collections import defaultdict

class NaiveBayesClassifier:
    def __init__(self):
        self.priors = {}
        self.likelihoods = defaultdict(dict)

    def fit(self, X, y):
        self.classes = np.unique(y)
        total_samples = len(y)

        for cls in self.classes:
            X_cls = X[y == cls]
            self.priors[cls] = len(X_cls) / total_samples
            for column in X.columns:
                self.likelihoods[column][cls] = X_cls[column].value_counts(normalize=True).to_dict()

    def predict(self, X):
        results = []
        for i in range(len(X)):
            posteriors = {}
            for cls in self.classes:
                prior = np.log(self.priors[cls])
                likelihood = sum(np.log(self.likelihoods[col].get(cls, {}).get(X.iloc[i][col], 1e-6)) for col in X.columns)
                posteriors[cls] = prior + likelihood
            results.append(max(posteriors, key=posteriors.get))
        return results

    def accuracy(self, y_true, y_pred):
        return np.mean(np.array(y_true) == np.array(y_pred))

# Load data from CSV file
data = pd.read_csv('iris.csv')

# Separate features and target variable
X = data.iloc[:, :-1]
y = data.iloc[:, -1]

# Split data 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)

# Initialize and train the Naive Bayes Classifier
```

```python
nb_classifier = NaiveBayesClassifier()
nb_classifier.fit(X_train, y_train)

# Make predictions on the test set
y_pred = nb_classifier.predict(X_test)

# Compute the accuracy
accuracy = nb_classifier.accuracy(y_test, y_pred)
print(f'Accuracy: {accuracy * 100:.2f}%')

'''OUTPUT
Accuracy: 93.33%
'''
```