

**School of Computing Science & Engineering**

**Laboratory Record**

**Deep Learning – CSA3007**

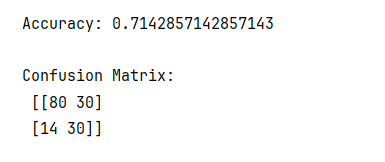
|  |  |
| --- | --- |
| Slot | A24+D23+D24 |
| Class ID | BL202324050470 |
| Semester | Winter Semester 2023-2024 |
| Name of the student | Harshit Mittal |
| Register Number | 21BAI10058 |

Experiment -1 Implementing Naïve Bayes Algorithm

**Aim:**  Implement Naïve Bayes Algorithm on diabetes dataset.

**Code:** import pandas as pd  
from sklearn.model\_selection import train\_test\_split  
from sklearn.naive\_bayes import GaussianNB *# For numerical features*from sklearn.naive\_bayes import MultinomialNB *# For text features (categorical counts)*from sklearn.preprocessing import LabelEncoder *# For categorical features*from sklearn.feature\_extraction.text import CountVectorizer *# For text data*from sklearn.metrics import accuracy\_score, confusion\_matrix  
  
def naive\_bayes(data\_path, target\_column, text\_column=None):  
 *"""  
 Implements Naive Bayes classification on a Covid-19 dataset,  
 calculates the confusion matrix, and returns accuracy.  
  
 Args:  
 data\_path (str): Path to the CSV file containing the Covid-19 data.  
 target\_column (str): Name of the column containing the target variable (e.g., 'Covid\_positive').  
 text\_column (str, optional): Name of the column containing text data (if applicable).  
  
 Returns:  
 tuple: (float, numpy.ndarray) Accuracy score of the trained Naive Bayes model,  
 and the confusion matrix.  
 """  
  
 # Load data* data = pd.read\_csv(data\_path)  
  
 *# Preprocess data* if text\_column:  
 *# Text data preparation (assuming categorical counts are important)* vectorizer = CountVectorizer()  
 text\_features = vectorizer.fit\_transform(data[text\_column])  
 data.drop(text\_column, axis=1, inplace=True) *# Remove raw text column* data = pd.concat([data, text\_features], axis=1) *# Add vectorized features* else:  
 *# Handle categorical features (if any)* for col in data.columns:  
 if data[col].dtype == 'object': *# Check for categorical data* le = LabelEncoder()  
 data[col] = le.fit\_transform(data[col])  
  
 *# Split data* X\_train, X\_test, y\_train, y\_test = train\_test\_split(data.drop(target\_column, axis=1), data[target\_column], test\_size=0.2)  
  
 *# Choose Naive Bayes variant* if text\_column:  
 clf = MultinomialNB() *# For text data (categorical counts)* else:  
 clf = GaussianNB() *# For numerical features  
  
 # Train model* clf.fit(X\_train, y\_train)  
  
 *# Make predictions* y\_pred = clf.predict(X\_test)  
  
 *# Evaluate performance* accuracy = accuracy\_score(y\_test, y\_pred)  
 print("Accuracy:", accuracy)

**Output:**

****