**КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ**

**ІМЕНІ ТАРАСА ШЕВЧЕНКА**

**ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ**

**КАФЕДРА ІНТЕЛЕКТУАЛЬНИХ ТЕХНОЛОГІЙ**

**Лабораторна робота №**6

з дисципліни «Інтелектуальний аналіз та візуалізація даних»

тема «КЛАСИФІКАЦІЯ, МЕТОД НАЇВНОГО КЛАСИФІКАТОРА БАЙЄСА»

**Варіант №8**

Виконала студентка

групи КН-21

Іванова А.О.

Перевірив(-ла):

МІнаєва Ю.І.

**Київ – 2025**

**Мета роботи:**

Алгоритм побудови наївного класифікатора Байєса. Мова Python.

Завантаження бібліотек

import numpy as np

from sklearn import datasets

import matplotlib.pyplot as plt

from sklearn.model\_selection import train\_test\_split

from sklearn.svm import SVC

from sklearn.metrics import accuracy\_score, classification\_report, confusion\_matrix

import pandas as pd

from google.colab import files

from sklearn.feature\_selection import SelectKBest, f\_classif

from mlxtend.plotting import plot\_decision\_regions

# Завантаження датасету Wine

wine = datasets.load\_wine()

X = wine.data

y = wine.target

# Виведення статистик

statistics = pd.DataFrame(X, columns=wine.feature\_names)

statistics['Target'] = y

statistics\_desc = statistics.describe()

print(statistics\_desc)

alcohol malic\_acid ash alcalinity\_of\_ash magnesium \

count 178.000000 178.000000 178.000000 178.000000 178.000000

mean 13.000618 2.336348 2.366517 19.494944 99.741573

std 0.811827 1.117146 0.274344 3.339564 14.282484

min 11.030000 0.740000 1.360000 10.600000 70.000000

25% 12.362500 1.602500 2.210000 17.200000 88.000000

50% 13.050000 1.865000 2.360000 19.500000 98.000000

75% 13.677500 3.082500 2.557500 21.500000 107.000000

max 14.830000 5.800000 3.230000 30.000000 162.000000

total\_phenols flavanoids nonflavanoid\_phenols proanthocyanins \

count 178.000000 178.000000 178.000000 178.000000

mean 2.295112 2.029270 0.361854 1.590899

std 0.625851 0.998859 0.124453 0.572359

min 0.980000 0.340000 0.130000 0.410000

25% 1.742500 1.205000 0.270000 1.250000

50% 2.355000 2.135000 0.340000 1.555000

75% 2.800000 2.875000 0.437500 1.950000

max 3.880000 5.080000 0.660000 3.580000

color\_intensity hue od280/od315\_of\_diluted\_wines proline \

count 178.000000 178.000000 178.000000 178.000000

mean 5.058090 0.957449 2.611685 746.893258

std 2.318286 0.228572 0.709990 314.907474

min 1.280000 0.480000 1.270000 278.000000

25% 3.220000 0.782500 1.937500 500.500000

50% 4.690000 0.965000 2.780000 673.500000

75% 6.200000 1.120000 3.170000 985.000000

max 13.000000 1.710000 4.000000 1680.000000

Target

count 178.000000

mean 0.938202

std 0.775035

min 0.000000

25% 0.000000

50% 1.000000

75% 2.000000

max 2.000000

# Поділ вибірки на навчальну та тестову

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3,

random\_state=42)

# Застосування класифікатора

clf = GaussianNB()

clf.fit(X\_train, y\_train)

y\_pred = clf.predict(X\_test)

# Виведення матриці невідповідностей та звіту classification report:

confusion = confusion\_matrix(y\_test, y\_pred)

report = classification\_report(y\_test, y\_pred, target\_names=class\_names,

output\_dict=True)

print(" The result report:\n", classification\_report(y\_test, y\_pred,

target\_names=class\_names))

The result report:

precision recall f1-score support

class\_0 1.00 1.00 1.00 19

class\_1 1.00 1.00 1.00 21

class\_2 1.00 1.00 1.00 14

accuracy 1.00 54

macro avg 1.00 1.00 1.00 54

weighted avg 1.00 1.00 1.00 54

# Обчислення точності класифікації

accuracy = accuracy\_score(y\_test, y\_pred)

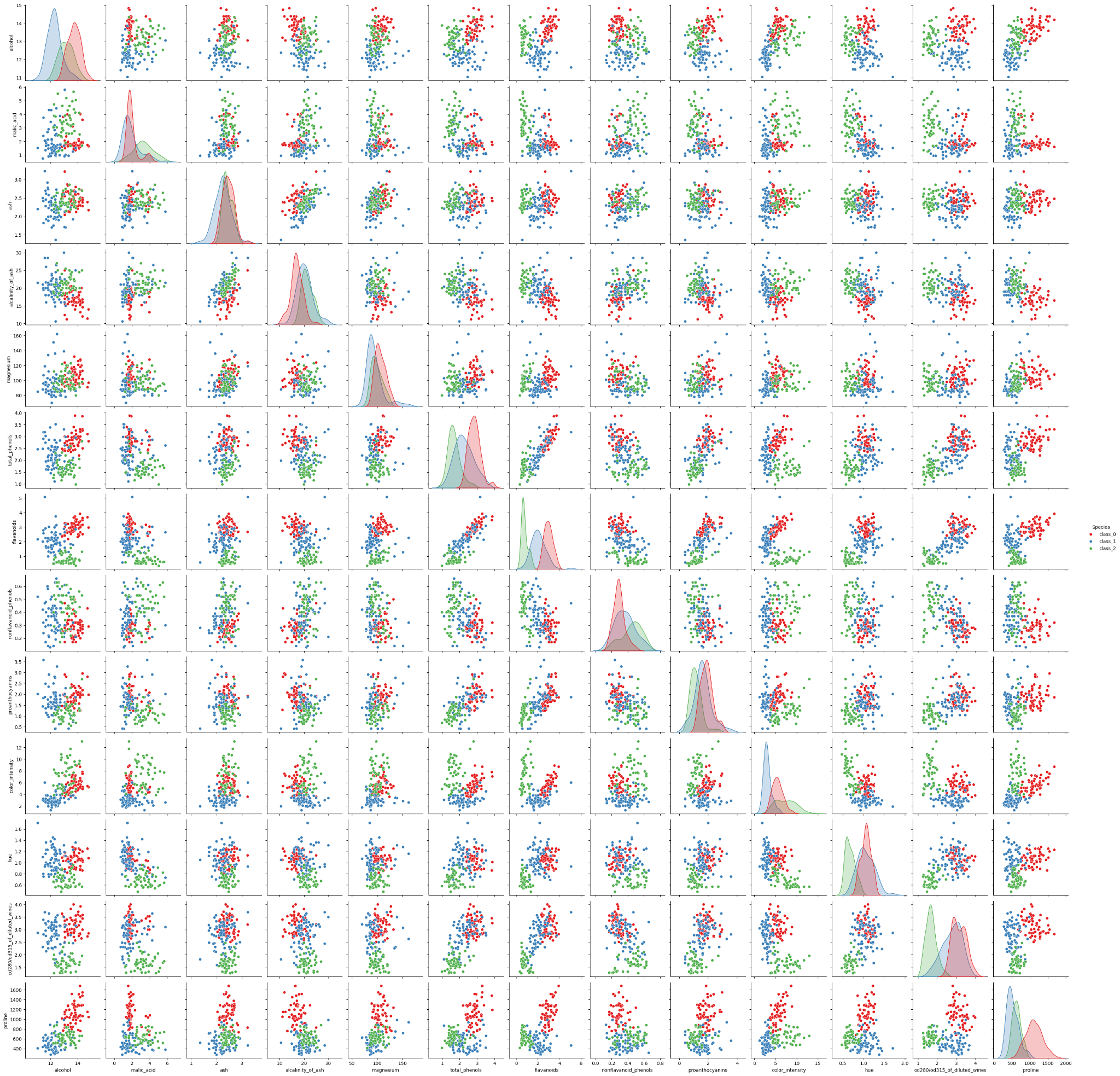
print("The accuracy is:", accuracy)

The accuracy is: 1.0

df = pd.DataFrame(X, columns=wine.feature\_names)

df['Species'] = wine.target\_names[y]

sns.pairplot(df, hue='Species', palette='Set1', diag\_kind='kde')



plt.figure(figsize=(12, 5))

for i in range(3):

plt.subplot(1, 3, i + 1)

sns.histplot(X[y == i][:, 0], bins=15, kde=True, label=class\_names[i],

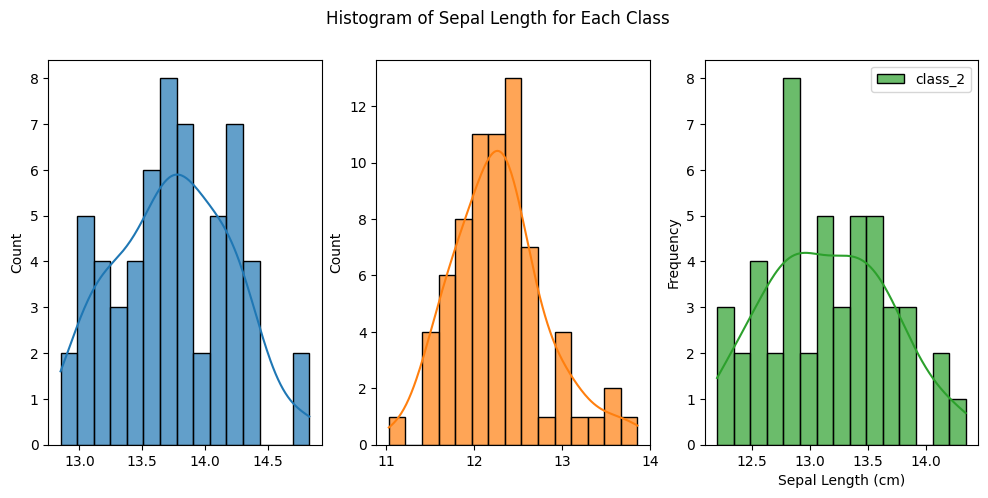
color=f'C{i}', alpha=0.7)

plt.xlabel('Sepal Length (cm)')

plt.ylabel('Frequency')

plt.legend()

plt.suptitle('Histogram of Sepal Length for Each Class');



plt.figure(figsize=(8, 6))

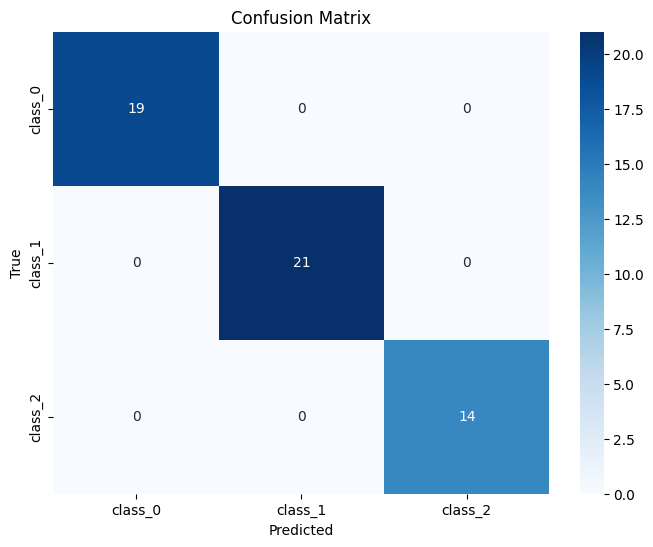
sns.heatmap(confusion, annot=True, fmt="d", cmap="Blues",

xticklabels=class\_names, yticklabels=class\_names)

plt.xlabel("Predicted")

plt.ylabel("True")

plt.title("Confusion Matrix");

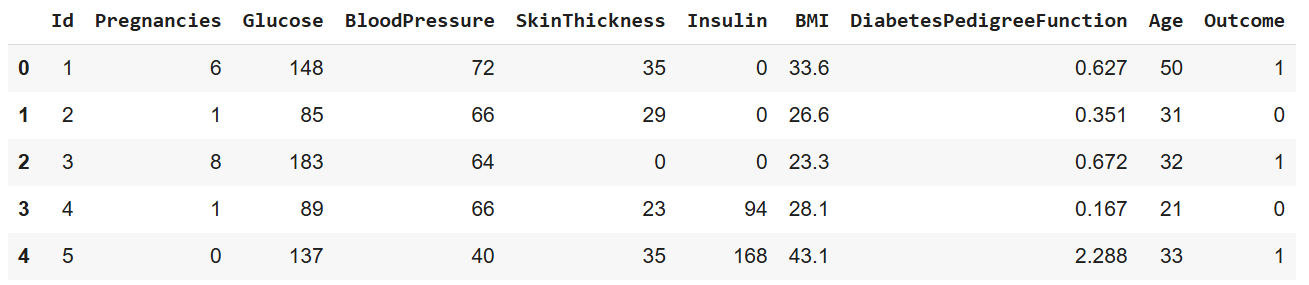


# Завантаження датасету Healthcare Diabetse

uploaded = files.upload()

df\_healthcare = pd.read\_csv("Healthcare-Diabetes.csv")

df\_healthcare.head()



Id Pregnancies Glucose BloodPressure SkinThickness \

count 2768.000000 2768.000000 2768.000000 2768.000000 2768.000000

mean 1384.500000 3.742775 121.102601 69.134393 20.824422

std 799.197097 3.323801 32.036508 19.231438 16.059596

min 1.000000 0.000000 0.000000 0.000000 0.000000

25% 692.750000 1.000000 99.000000 62.000000 0.000000

50% 1384.500000 3.000000 117.000000 72.000000 23.000000

75% 2076.250000 6.000000 141.000000 80.000000 32.000000

max 2768.000000 17.000000 199.000000 122.000000 110.000000

Insulin BMI DiabetesPedigreeFunction Age \

count 2768.000000 2768.000000 2768.000000 2768.000000

mean 80.127890 32.137392 0.471193 33.132225

std 112.301933 8.076127 0.325669 11.777230

min 0.000000 0.000000 0.078000 21.000000

25% 0.000000 27.300000 0.244000 24.000000

50% 37.000000 32.200000 0.375000 29.000000

75% 130.000000 36.625000 0.624000 40.000000

max 846.000000 80.600000 2.420000 81.000000

Target

count 2768.000000

mean 0.343931

std 0.475104

min 0.000000

25% 0.000000

50% 0.000000

75% 1.000000

max 1.000000

# Поділ вибірки на навчальну та тестову

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3,

random\_state=42)

# Застосування класифікатора

clf = GaussianNB()

clf.fit(X\_train, y\_train)

y\_pred = clf.predict(X\_test)

# Виведення матриці невідповідностей та звіту classification report:

confusion = confusion\_matrix(y\_test, y\_pred)

report = classification\_report(y\_test, y\_pred, target\_names=class\_names, output\_dict=True)

print(" The result report:\n", classification\_report(y\_test, y\_pred, target\_names=class\_names))

The result report:

precision recall f1-score support

Non-Diabetic 0.82 0.86 0.84 562

Diabetic 0.67 0.60 0.64 269

accuracy 0.78 831

macro avg 0.75 0.73 0.74 831

weighted avg 0.77 0.78 0.77 831

# Обчислення точності класифікації

accuracy = accuracy\_score(y\_test, y\_pred)

print("The accuracy is:", accuracy)

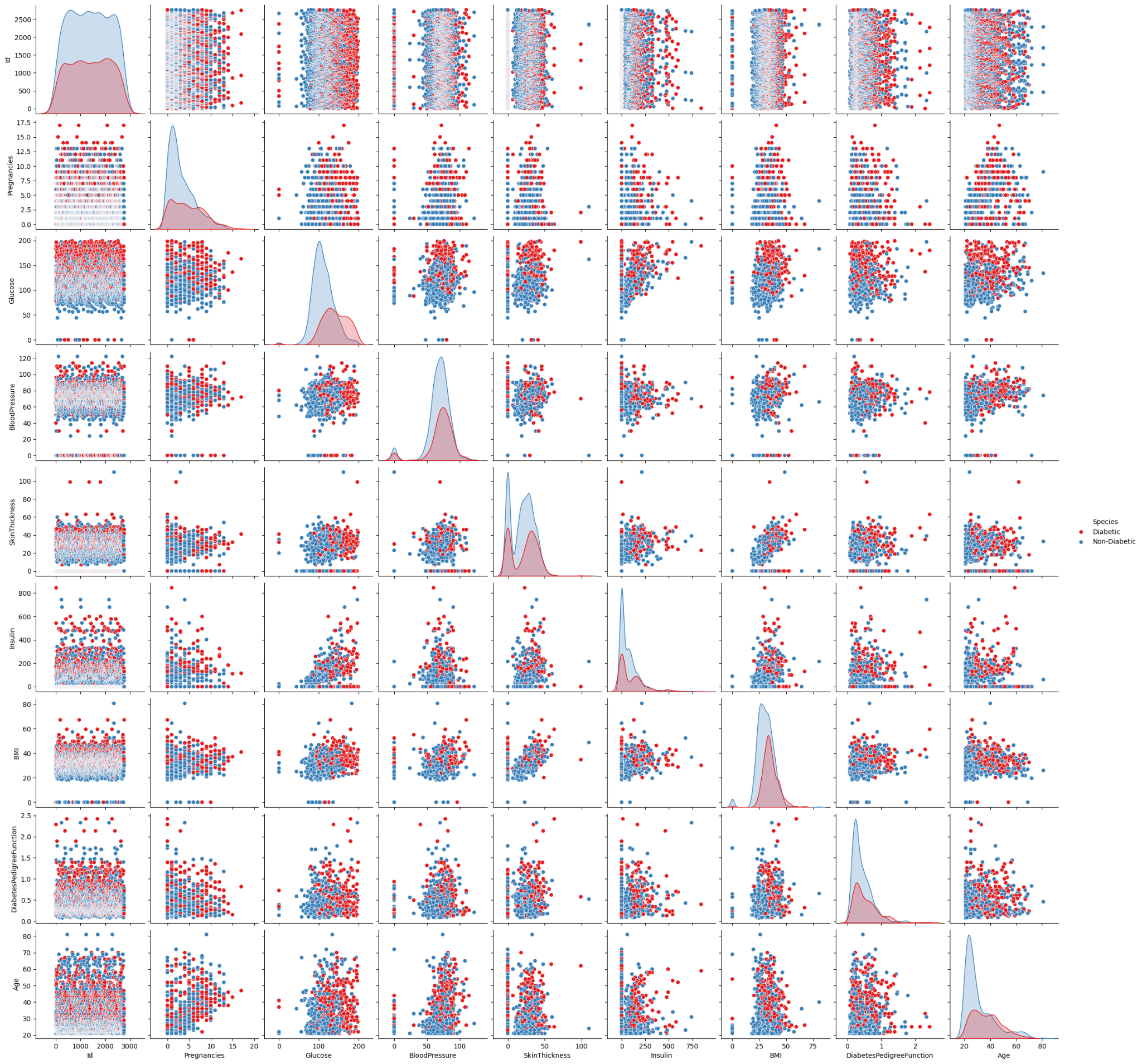
The accuracy is: 0.776173285198556

df = pd.DataFrame(X, columns=X.columns)

species = ['Non-Diabetic', 'Diabetic']

df['Species'] = [species[i] for i in y]

sns.pairplot(df, hue='Species', palette='Set1', diag\_kind='kde')



plt.figure(figsize=(12, 5))

for i in range(2):

plt.subplot(1, 2, i + 1)

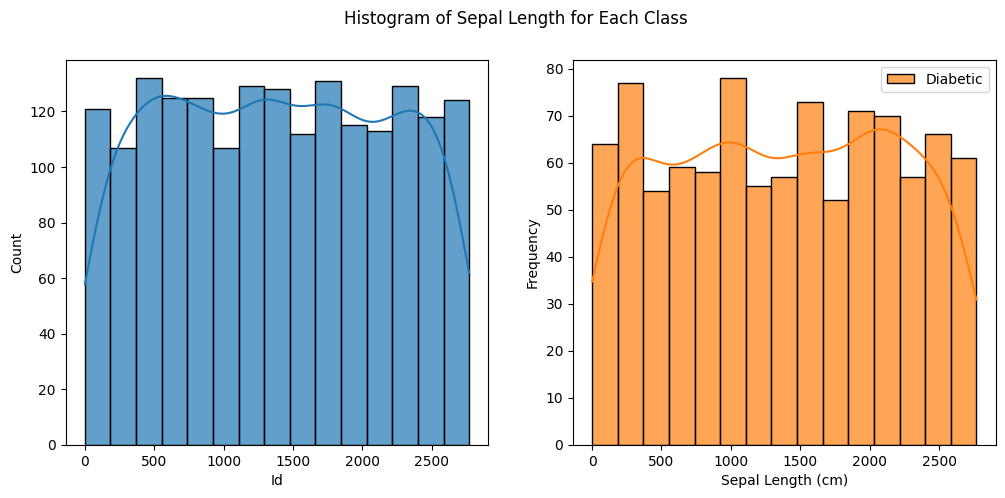
sns.histplot(X[y == i].iloc[:, 0], bins=15, kde=True, label=class\_names[i], color=f'C{i}', alpha=0.7)

plt.xlabel('Sepal Length (cm)')

plt.ylabel('Frequency')

plt.legend()

plt.suptitle('Histogram of Sepal Length for Each Class');



plt.figure(figsize=(8, 6))

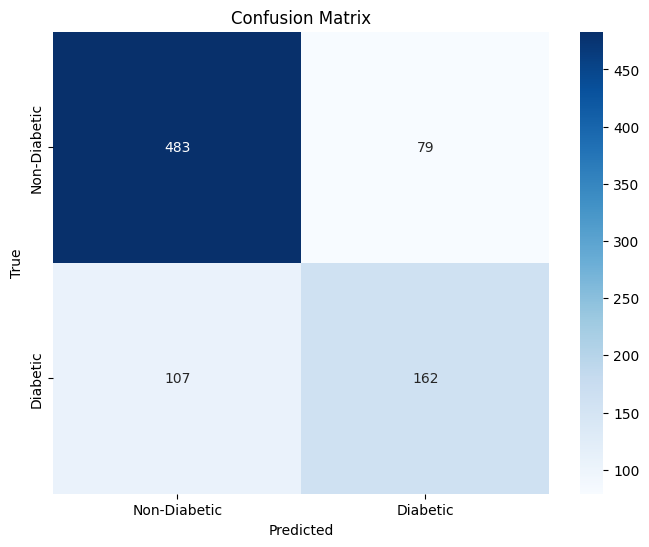
sns.heatmap(confusion, annot=True, fmt="d", cmap="Blues",

xticklabels=class\_names, yticklabels=class\_names)

plt.xlabel("Predicted")

plt.ylabel("True")

plt.title("Confusion Matrix");



**Висновок:** лабораторній роботі було вивчено алгоритм побудови наївного класифікатора Байєса.