Міністерство освіти і науки України

Національний університет «Одеська політехніка»

Інститут комп'ютерних систем

Кафедра інформаційних систем

Лабораторна робота №1

Частина друга

З дисципліни: "Інтелектуальний аналіз даних"

Тема: "Вступ до наукової мови програмування Python"

Підготувала студентка

Групи АІ – 212

Козуб К.О

Перевірили:

Антощук С.Г

Кошутіна Д.В

Одеса 2023

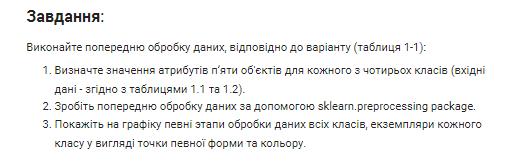
**Мета:** Ознайомлення з основами програмування Python та Google Colab з прикладами інтерактивних завдань у Colab; проведення вправ, контрольних запитань, а також написання власних базових програм мовою програмування Python.

**Завдання**:

Виконайте попередню обробку даних, відповідно до варіанту (таблиця 1-1):

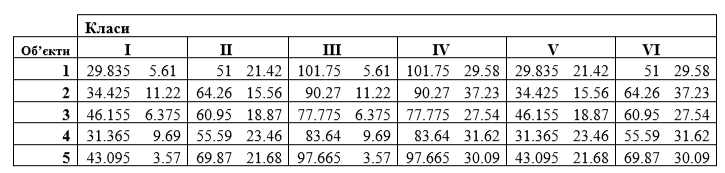
1. Визначте значення атрибутів п’яти об'єктів для кожного з чотирьох класів (вхідні дані - згідно з таблицями 1.1 та 1.2).
2. Зробіть попередню обробку даних для всіх класів за допомогою sklearn.preprocessing package (бінарізація, вийняток середнього, маштабування, нормалізація).
3. Покажіть на графіку певні этапи обробки даних всіх класів, екземпляри кожного класу у вигляді точки певної форми та кольору.

Посилання на google colab: <https://colab.research.google.com/drive/1fkE_pGqTM0GAAg2IGMVeK51zbegG-k_n?usp=sharing>



Варіант 7:





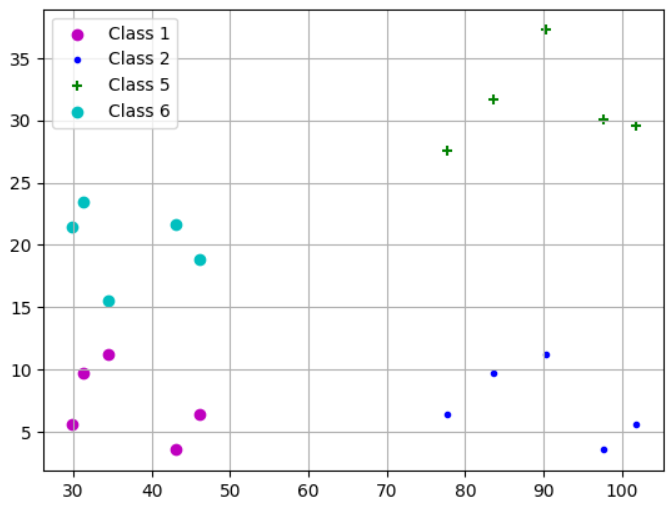


Рисунок 1 – Графік

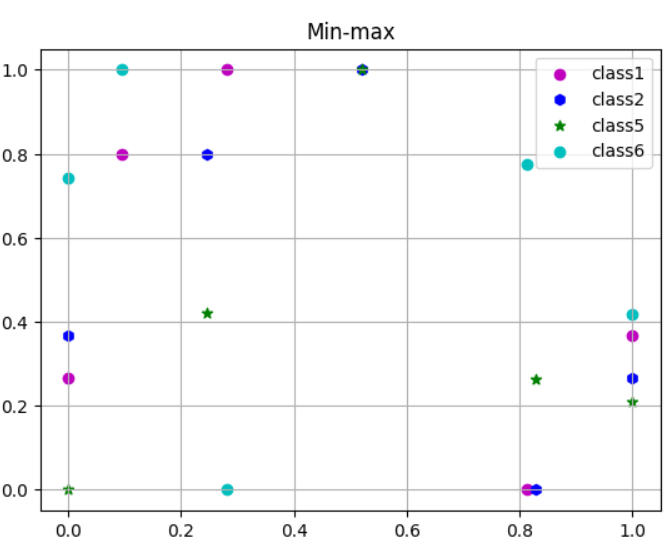


Рисунок 2 – Min-max

Висновок: На лабораторнії робота № 1(частина 2). Я познайомилися з попередньою обробкою даних та підтримуючими бібліотеками Python та виконала завдання з обробки даних за допомогою рекомендованих бібліотек Python

Додаток А

import numpy as np

from sklearn import preprocessing

import math

import matplotlib.pyplot as plt

class1 = np.array([[ 29.835, 5.61],[ 34.425, 11.22],[ 46.155, 6.375],[ 31.365, 9.69],[ 43.095, 3.57]])

class2 = np.array([[ 101.75, 5.61],[ 90.27, 11.22],[ 77.775, 6.375],[ 83.64, 9.69],[ 97.665, 3.57]])

class5 = np.array([[ 101.75, 29.58],[ 90.27, 37.23],[ 77.777, 27.54],[ 83.64, 31.62],[ 97.665, 30.09]])

class6 = np.array([[ 29.835, 21.42],[ 34.425, 15.56],[ 46.155, 18.87],[ 31.365, 23.46],[ 43.095, 21.68]])

data\_binarized1 = preprocessing.Binarizer(threshold=45).transform(class1)

print(data\_binarized1 )

data\_binarized2 = preprocessing.Binarizer(threshold=45).transform(class2)

print(data\_binarized2 )

data\_binarized5 = preprocessing.Binarizer(threshold=45).transform(class5)

print(data\_binarized5 )

data\_binarized6 = preprocessing.Binarizer(threshold=45).transform(class6)

print(data\_binarized6 )

print("\nClass1: ")

print("Mean =", class1.mean(axis=0))

print( "Std deviation =", class1.std(axis=0))

data\_scaler = preprocessing.scale(class1)

print ( "\nAFTER: ")

print('\n Scaled data:\n', data\_scaler)

print("Mean =", data\_scaler.mean(axis=0))

print("Std deviation =", data\_scaler.std(axis=0))

print("\nClass2: ")

print("Mean =", class2.mean(axis=0))

print( "Std deviation =", class2.std(axis=0))

data\_scaler = preprocessing.scale(class2)

print ( "\nAFTER: ")

print('\n Scaled data:\n', data\_scaler)

print("Mean =", data\_scaler.mean(axis=0))

print("Std deviation =", data\_scaler.std(axis=0))

print("\nClass5: ")

print("Mean =", class5.mean(axis=0))

print( "Std deviation =", class5.std(axis=0))

data\_scaler = preprocessing.scale(class5)

print ( "\nAFTER: ")

print('\n Scaled data:\n', data\_scaler)

print("Mean =", data\_scaler.mean(axis=0))

print("Std deviation =", data\_scaler.std(axis=0))

print("\nClass6: ")

print("Mean =", class6.mean(axis=0))

print( "Std deviation =", class6.std(axis=0))

data\_scaler = preprocessing.scale(class6)

print ( "\nAFTER: ")

print('\n Scaled data:\n', data\_scaler)

print("Mean =", data\_scaler.mean(axis=0))

print("Std deviation =", data\_scaler.std(axis=0))

data\_minmax\_scaler = preprocessing.MinMaxScaler(feature\_range=(0, 1))

data\_minmax\_scaler = data\_minmax\_scaler.fit\_transform(class1)

print("\nМin max scaled data:\n", data\_minmax\_scaler)

data\_maxabs\_scaler = preprocessing.MaxAbsScaler()

data\_maxabs\_scaler = data\_maxabs\_scaler.fit\_transform(class1)

print("\nМax Abs scaled data:\n", data\_maxabs\_scaler)

data\_minmax\_scaler = preprocessing.MinMaxScaler(feature\_range=(0, 1))

data\_minmax\_scaler = data\_minmax\_scaler.fit\_transform(class2)

print("\nМin max scaled data:\n", data\_minmax\_scaler)

data\_maxabs\_scaler = preprocessing.MaxAbsScaler()

data\_maxabs\_scaler = data\_maxabs\_scaler.fit\_transform(class2)

print("\nМax Abs scaled data:\n", data\_maxabs\_scaler)

data\_minmax\_scaler = preprocessing.MinMaxScaler(feature\_range=(0, 1))

data\_minmax\_scaler = data\_minmax\_scaler.fit\_transform(class5)

print("\nМin max scaled data:\n", data\_minmax\_scaler)

data\_maxabs\_scaler = preprocessing.MaxAbsScaler()

data\_maxabs\_scaler = data\_maxabs\_scaler.fit\_transform(class5)

print("\nМax Abs scaled data:\n", data\_maxabs\_scaler)

data\_minmax\_scaler = preprocessing.MinMaxScaler(feature\_range=(0, 1))

data\_minmax\_scaler = data\_minmax\_scaler.fit\_transform(class6)

print("\nМin max scaled data:\n", data\_minmax\_scaler)

data\_maxabs\_scaler = preprocessing.MaxAbsScaler()

data\_maxabs\_scaler = data\_maxabs\_scaler.fit\_transform(class6)

print("\nМax Abs scaled data:\n", data\_maxabs\_scaler)

data\_normalized\_ll = preprocessing.normalize(class1, norm='l1')

data\_normalized\_l2 = preprocessing.normalize(class1, norm='l2')

print("\nLl normalized data:\n", data\_normalized\_ll)

print("\nL2 normalized data:\n", data\_normalized\_l2)

data\_normalized\_ll = preprocessing.normalize(class2, norm='l1')

data\_normalized\_l2 = preprocessing.normalize(class2, norm='l2')

print("\nLl normalized data:\n", data\_normalized\_ll)

print("\nL2 normalized data:\n", data\_normalized\_l2)

data\_normalized\_ll = preprocessing.normalize(class5, norm='l1')

data\_normalized\_l2 = preprocessing.normalize(class5, norm='l2')

print("\nLl normalized data:\n", data\_normalized\_ll)

print("\nL2 normalized data:\n", data\_normalized\_l2)

data\_normalized\_ll = preprocessing.normalize(class6, norm='l1')

data\_normalized\_l2 = preprocessing.normalize(class6, norm='l2')

print("\nLl normalized data:\n", data\_normalized\_ll)

print("\nL2 normalized data:\n", data\_normalized\_l2)

print("\n")

plt.scatter(class1[:,0], class1[:,1], color='m', marker = "o", label='Class 1')

plt.scatter(class2[:,0], class2[:,1], color='b', marker = ".", label='Class 2')

plt.scatter(class5[:,0], class5[:,1], color='g', marker = "+", label='Class 5')

plt.scatter(class6[:,0], class6[:,1], color='c', label='Class 6')

plt.legend()

plt.grid()

plt.show()

data\_minmax\_scaler1 = preprocessing.MinMaxScaler(feature\_range=(0, 1)).fit\_transform(class1)

data\_minmax\_scaler2 = preprocessing.MinMaxScaler(feature\_range=(0, 1)).fit\_transform(class2)

data\_minmax\_scaler5 = preprocessing.MinMaxScaler(feature\_range=(0, 1)).fit\_transform(class5)

data\_minmax\_scaler6 = preprocessing.MinMaxScaler(feature\_range=(0, 1)).fit\_transform(class6)

plt.title("Min-max")

plt.scatter(data\_minmax\_scaler1[:, 0], data\_minmax\_scaler1[:, 1], color='m', marker='o', label='class1')

plt.scatter(data\_minmax\_scaler2[:, 0], data\_minmax\_scaler2[:, 1], color='b', marker='h', label='class2')

plt.scatter(data\_minmax\_scaler5[:, 0], data\_minmax\_scaler5[:, 1], color='g', marker='\*', label='class5')

plt.scatter(data\_minmax\_scaler6[:, 0], data\_minmax\_scaler6[:, 1], color='c', label='class6')

plt.grid()

plt.legend()

plt.show()