Модульна робота 2

Студента КНм-11

Ділай Р.І.

Варіант 7

Посилання на GitHub https://github.com/RomaPudgeAndInvoker/second\_module\_intelligence

%pip install numpy scipy scikit-learn matplotlib

import numpy as np

from scipy.io import loadmat

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

from sklearn.preprocessing import StandardScaler

from sklearn.neural\_network import MLPClassifier

from sklearn.svm import SVC

from sklearn.metrics import accuracy\_score

import matplotlib.pyplot as plt

import pandas as pd

# 1. Завантаження та огляд даних

data = loadmat('../data/digits.mat')

data, labels, etc.

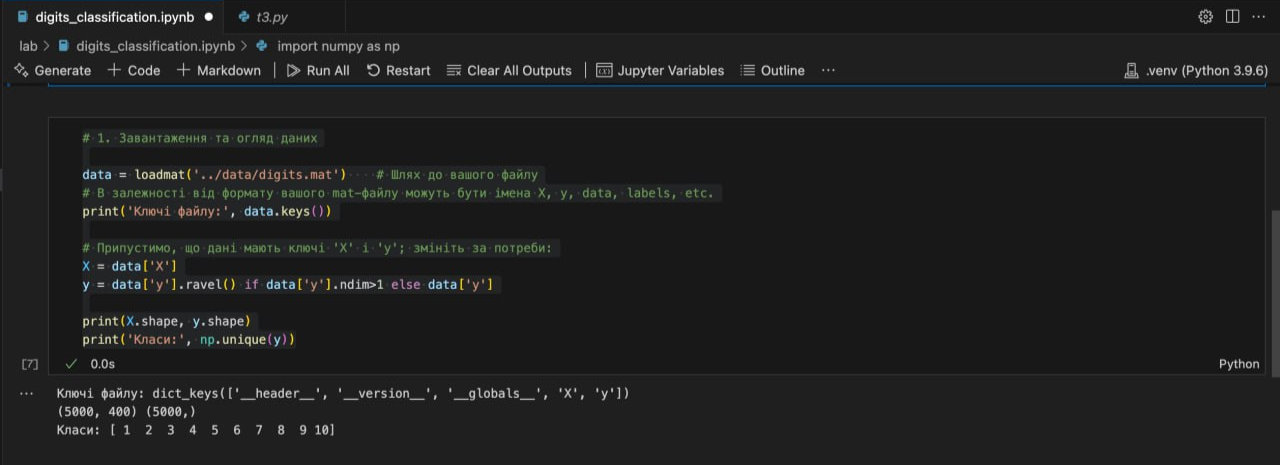
print('Ключі файлу:', data.keys())

X = data['X']

y = data['y'].ravel() if data['y'].ndim>1 else data['y']

print(X.shape, y.shape)

print('Класи:', np.unique(y))



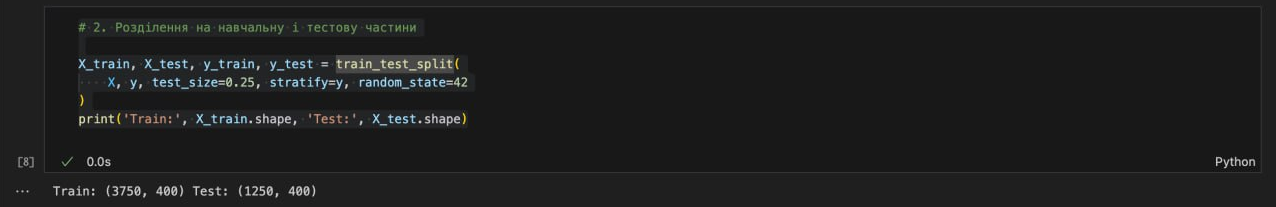
# 2. Розділення на навчальну і тестову частини

X\_train, X\_test, y\_train, y\_test = train\_test\_split(

X, y, test\_size=0.25, stratify=y, random\_state=42

)

print('Train:', X\_train.shape, 'Test:', X\_test.shape)

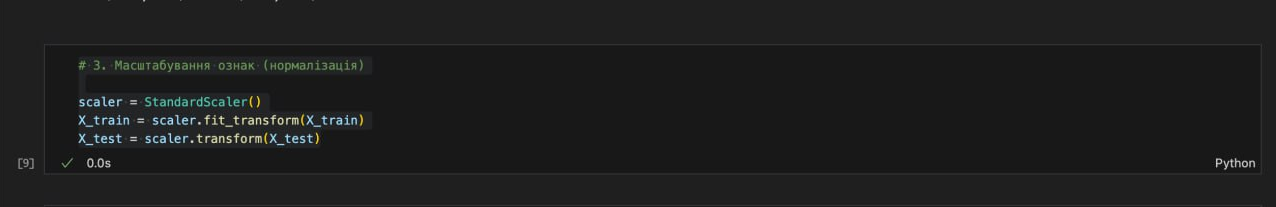


# 3. Масштабування ознак

scaler = StandardScaler()

X\_train = scaler.fit\_transform(X\_train)

X\_test = scaler.transform(X\_test)



# 4. Побудова і навчання моделей

results = []

# Одношарова MLP (100 нейронів)

mlp1 = MLPClassifier(hidden\_layer\_sizes=(100,),

solver='adam', activation='tanh', max\_iter=100, random\_state=1)

mlp1.fit(X\_train, y\_train)

results.append({

'Модель': 'MLP (100)',

'Точність train': accuracy\_score(y\_train, mlp1.predict(X\_train)),

'Точність test': accuracy\_score(y\_test, mlp1.predict(X\_test))

})

# Двошарова MLP [3,3] з tanh та relu

for act in ['tanh', 'relu']:

mlp2 = MLPClassifier(hidden\_layer\_sizes=(3,3),

solver='adam', activation=act, max\_iter=100, random\_state=1)

mlp2.fit(X\_train, y\_train)

results.append({

'Модель': f"MLP (3,3) {act}",

'Точність train': accuracy\_score(y\_train, mlp2.predict(X\_train)),

'Точність test': accuracy\_score(y\_test, mlp2.predict(X\_test))

})

# Тришарова MLP [20,7,10] з tanh та relu

for act in ['tanh', 'relu']:

mlp3 = MLPClassifier(hidden\_layer\_sizes=(20,7,10),

solver='adam', activation=act, max\_iter=100, random\_state=1)

mlp3.fit(X\_train, y\_train)

results.append({

'Модель': f"MLP (20,7,10) {act}",

'Точність train': accuracy\_score(y\_train, mlp3.predict(X\_train)),

'Точність test': accuracy\_score(y\_test, mlp3.predict(X\_test))

})

# SVM (RBF)

svc = SVC(kernel='rbf', gamma='scale', random\_state=1)

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

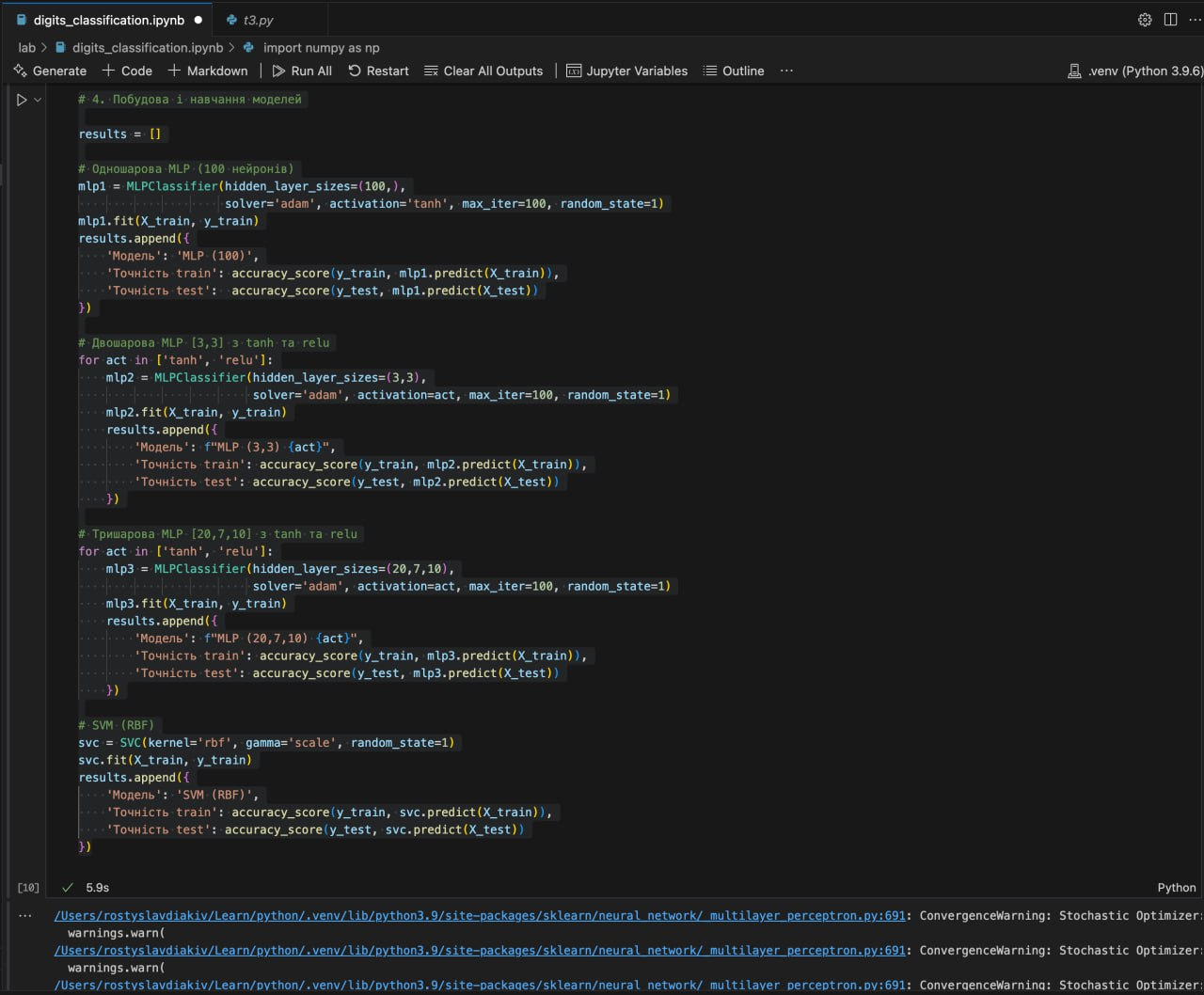
results.append({

'Модель': 'SVM (RBF)',

'Точність train': accuracy\_score(y\_train, svc.predict(X\_train)),

'Точність test': accuracy\_score(y\_test, svc.predict(X\_test))

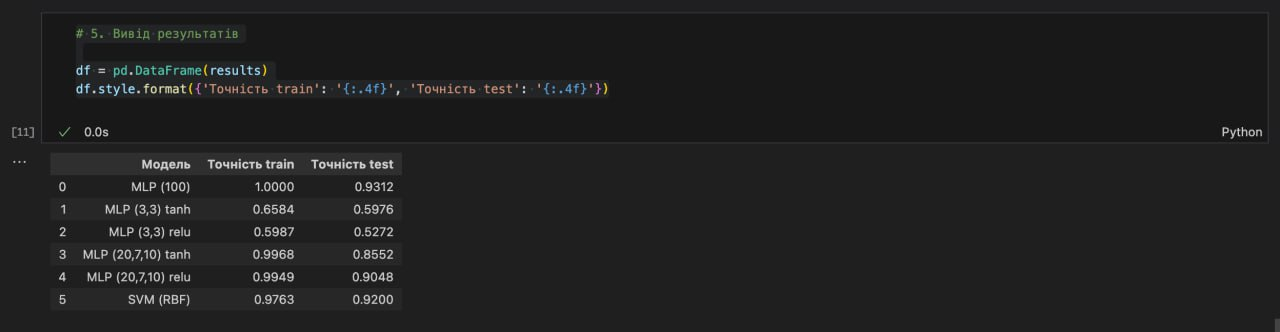
})



# 5. Вивід результатів

df = pd.DataFrame(results)

df.style.format({'Точність train': '{:.4f}', 'Точність test': '{:.4f}'})



# 6. Візуалізація порівняння

plt.figure(figsize=(10,5))

x = np.arange(len(results))

plt.bar(x-0.15, [r['Точність train'] for r in results], width=0.3, label='Train')

plt.bar(x+0.15, [r['Точність test'] for r in results], width=0.3, label='Test')

plt.xticks(x, [r['Модель'] for r in results], rotation=25)

plt.ylim(0.5, 1.05)

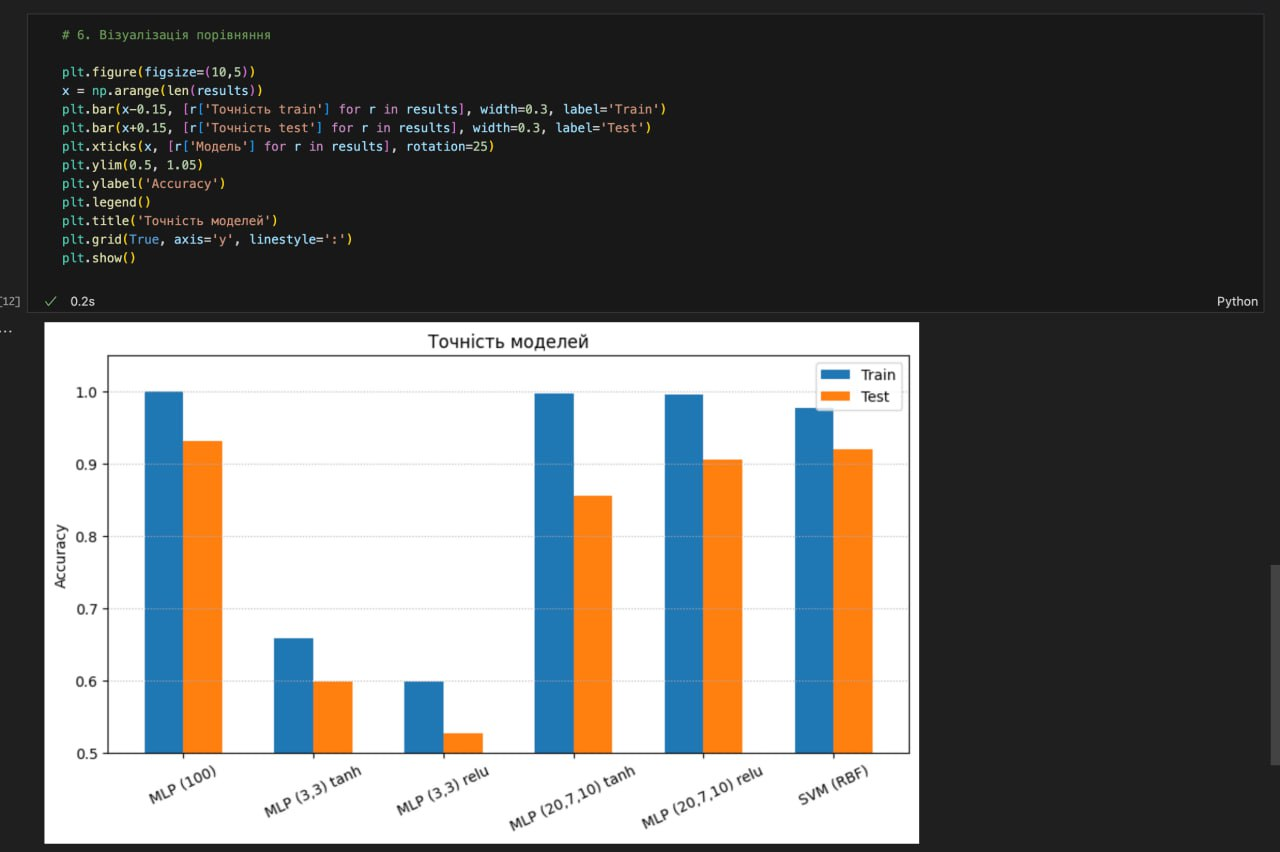
plt.ylabel('Accuracy')

plt.legend()

plt.title('Точність моделей')

plt.grid(True, axis='y', linestyle=':')

plt.show()



python file execution

import numpy as np

from scipy.io import loadmat

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

from sklearn.preprocessing import StandardScaler

from sklearn.neural\_network import MLPClassifier

from sklearn.svm import SVC

from sklearn.metrics import accuracy\_score

# Завантаження даних

data = loadmat('../data/digits.mat')

# Залежить від структури матфайлу - треба уточнити ключі, імовірно 'X','y'

X = data['X'] # приклади

y = data['y'].ravel() # цільові значення

# Розділення на train/test

X\_train, X\_test, y\_train, y\_test = train\_test\_split(

X, y, test\_size=0.25, random\_state=42, stratify=y

)

# Нормалізація

scaler = StandardScaler()

X\_train = scaler.fit\_transform(X\_train)

X\_test = scaler.transform(X\_test)

results = []

# Одношарова (100 нейронів)

clf1 = MLPClassifier(

hidden\_layer\_sizes=(100,),

solver='adam',

activation='tanh',

max\_iter=100,

random\_state=42

)

clf1.fit(X\_train, y\_train)

results.append({

'model': 'MLP (100)',

'train\_acc': accuracy\_score(y\_train, clf1.predict(X\_train)),

'test\_acc': accuracy\_score(y\_test, clf1.predict(X\_test))

})

# Двошарова [3,3] (активації нема як різні в шарі; тільки одна на все)

# Вибираємо по черзі 'tanh' та 'relu'

for act in ['tanh', 'relu']:

clf2 = MLPClassifier(

hidden\_layer\_sizes=(3,3),

solver='adam',

activation=act,

max\_iter=100,

random\_state=42

)

clf2.fit(X\_train, y\_train)

results.append({

'model': f'MLP (3,3) {act}',

'train\_acc': accuracy\_score(y\_train, clf2.predict(X\_train)),

'test\_acc': accuracy\_score(y\_test, clf2.predict(X\_test))

})

# Тришарова [20,7,10] (тільки одна активація у sklearn)

# Візьмемо 'tanh', 'relu'

for act in ['tanh', 'relu']:

clf3 = MLPClassifier(

hidden\_layer\_sizes=(20,7,10),

solver='adam',

activation=act,

max\_iter=100,

random\_state=42

)

clf3.fit(X\_train, y\_train)

results.append({

'model': f'MLP (20,7,10) {act}',

'train\_acc': accuracy\_score(y\_train, clf3.predict(X\_train)),

'test\_acc': accuracy\_score(y\_test, clf3.predict(X\_test))

})

svc = SVC(kernel='rbf', gamma='scale', random\_state=42)

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

results.append({

'model': 'SVM (RBF)',

'train\_acc': accuracy\_score(y\_train, svc.predict(X\_train)),

'test\_acc': accuracy\_score(y\_test, svc.predict(X\_test))

})

print("{:20} | Train Acc | Test Acc".format("Model"))

print("-"\*50)

for res in results:

print("{:20} | {:.4f} | {:.4f}".format(

res['model'], res['train\_acc'], res['test\_acc']

))

