Міністерство освіти і науки України
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 4 з дисципліни
«Прикладні задачі машинного навчаннях»

«Класифікація методом k найближчих сусідів і набір даних Digits, частина 1»

Варіант 7

Виконав студент: ІП-12 Васильєв Єгор Костянтинович

Перевірив: Нестерук Андрій Олександрович

Київ 2023

Лабораторна робота №4

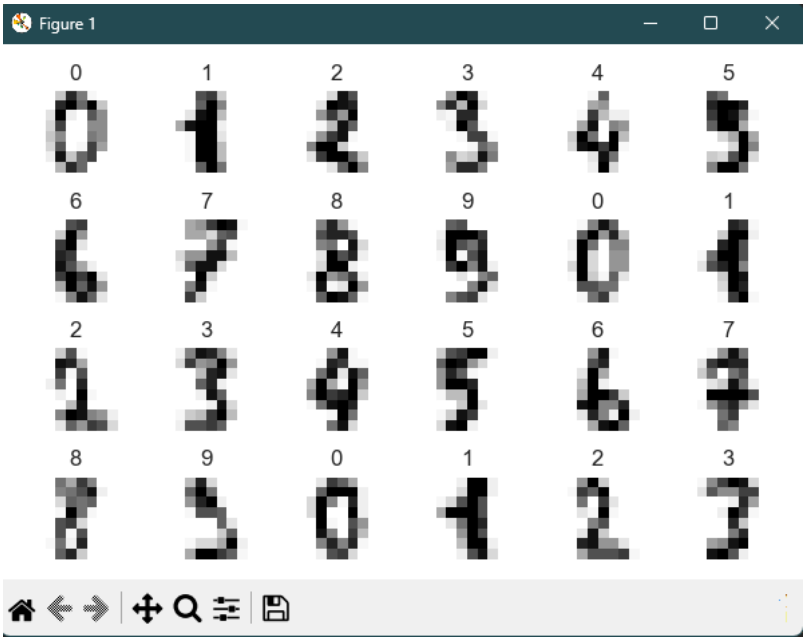
Тема: Класифікація методом k найближчих сусідів і набір даних Digits, частина 1.

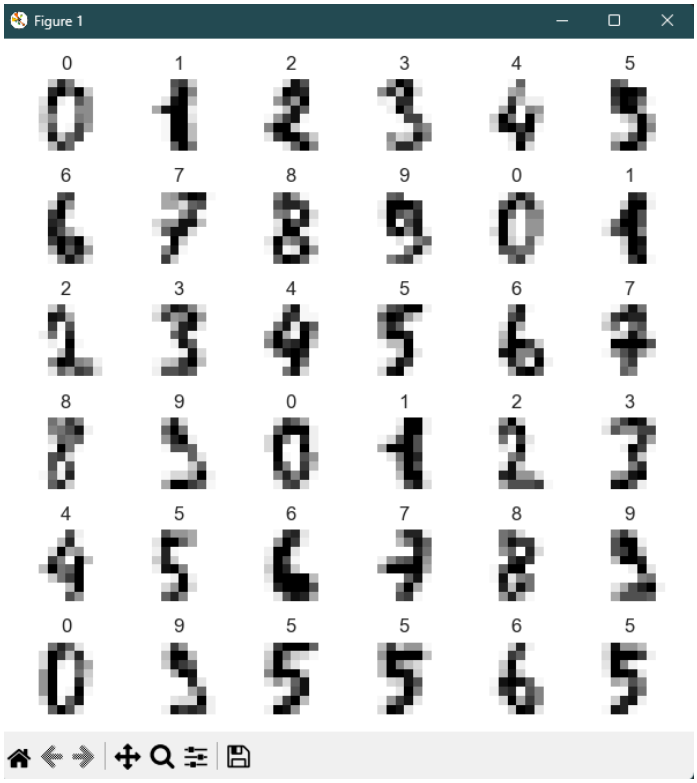
Постановка завдання:

1) Візуалізація перших 24 і 36 цифр набору.
2) Розбиття даних на навчальні та тестові.
3) Порівняння прогнозованих та очікуваних цифр для перших 36 зразків.
4) Оцінка точності трьох різних класифікаторів.
5) Виведення матриці невідповідностей для KNN класифікатора.
6) Виведення звіту класифікації KNN.
7) Підбір гіперпараметра k в KNN класифікаторі.

Хід роботи:

- Графічне представлення цифр набору





- Ділення початкових даних на навчальні і тестові

```
X_test size:  (360, 64)
X_train size:  (1437, 64)
```

- Порівняння передбачених та фактичних цифр

```
predicted:  [0 4 9 9 3 1 4 1 5 0 4 9 4 1 5 3 3 8 5 6 9 6 0 6 9 3 2 1 8 1 7 0 4 4 1 5]
expected:   [0 4 9 9 3 1 4 1 5 0 4 9 4 1 5 3 3 8 3 6 9 6 0 6 9 3 2 1 8 1 7 0 4 4 1 5]
```

- Оцінка точності різних класифікаторів

```
KNN Accuracy: 98.611%
SVC Accuracy: 98.611%
GNB Accuracy: 85.556%
```

- Виведення матриці невідповідностей для KNN класифікатора

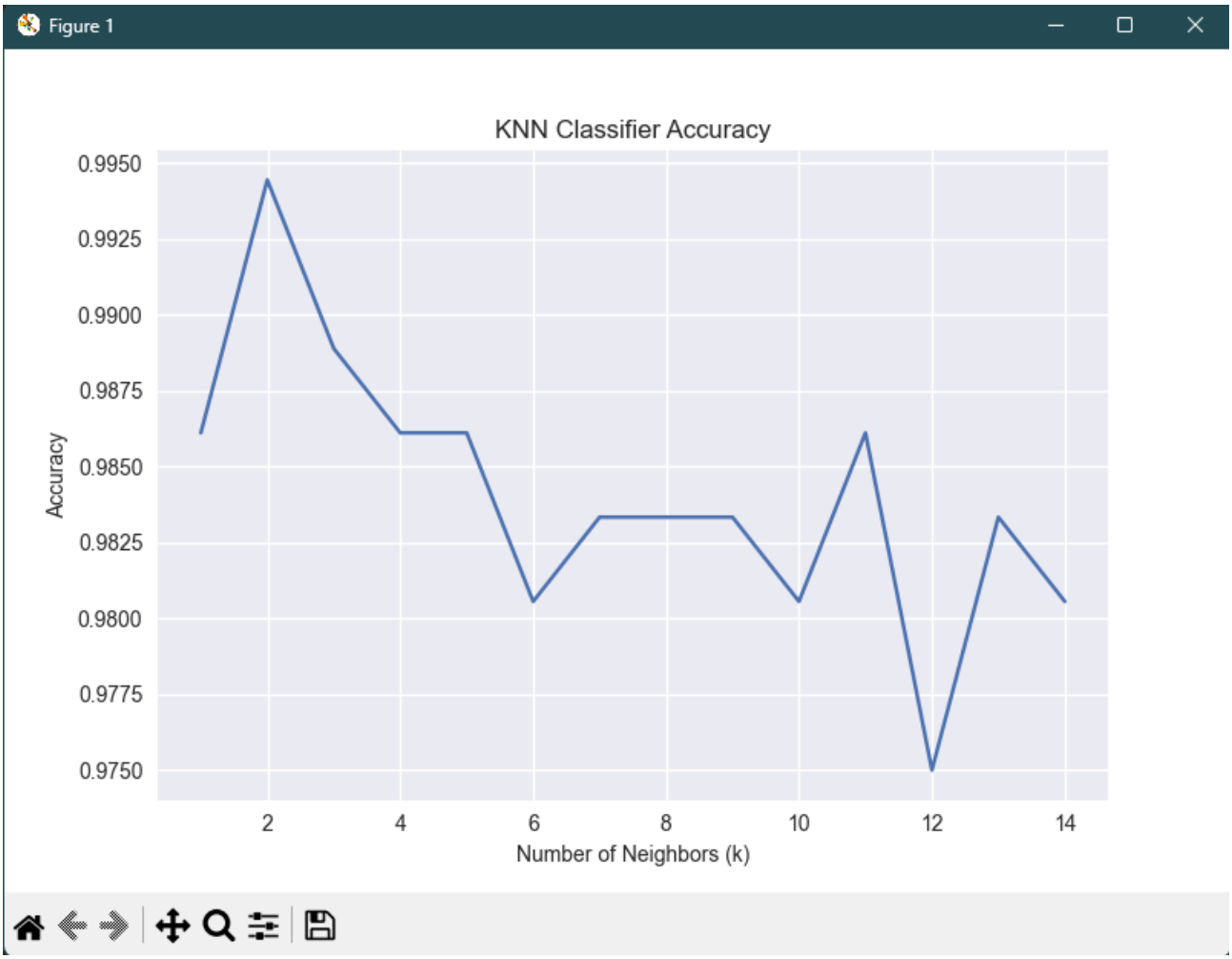
```
[[38  0  0  0  0  0  0  0  0  0]
 [ 0 37  0  0  0  0  0  0  0  0]
 [ 0  0 39  0  0  0  0  0  0  0]
 [ 0  0  0 39  0  1  0  1  0  0]
 [ 0  0  0  0 40  0  0  1  0  0]
 [ 0  0  0  0  0 27  0  0  0  0]
 [ 0  0  0  0  0  0 30  0  0  0]
 [ 0  0  0  0  0  0  0 36  0  0]
 [ 0  0  0  0  0  0  0  0 34  0]
 [ 0  0  0  0  1  0  0  0  1 35]]
```

- Виведення звіту класифікації KNN

	precision	recall	f1-score	support
0	1.00	1.00	1.00	38
1	1.00	1.00	1.00	37
2	1.00	1.00	1.00	39
3	1.00	0.95	0.97	41
4	0.98	0.98	0.98	41
5	0.96	1.00	0.98	27
6	1.00	1.00	1.00	30
7	0.95	1.00	0.97	36
8	0.97	1.00	0.99	34
9	1.00	0.95	0.97	37
accuracy			0.99	360
macro avg	0.99	0.99	0.99	360
weighted avg	0.99	0.99	0.99	360

- Підбір оптимальної кількості сусідів для KNN класифікатора

```
Best accuracy: 99.44 with 2 neighbours
```



Висновок

Було поглиблено знання бібліотек pandas, matplotlib, numpy та досліджено різні типи класифікаторів Scikit-Learn; було порівняно декілька класифікаційних оцінювачів: KNeighborsClassifier, SVC та GaussianNB для вбудованого в scikit-learn digits датасету та досліджено, що при параметрах за замовчуванням, метод опорних векторів та метод k найближчих сусідів виконують класифікацію найкраще; було детально досліджено результат роботи класифікатора KNN, використовуючи метрики точності моделі: метод score, матриця невідповідностей, звіт класифікації; було порівняно ефективність оцінювача при різних значеннях гіперпараметра k (кількість сусідів) та підібрано його оптимальне значення і досягнуто точності 99,44%; таким чином було вивчену тему «Класифікація методом k найближчих сусідів і набір даних Digits»

Вихідний код

```
import ...

pd.set_option('display.precision', 2)
plt.style.use('seaborn-v0_8')
digits = load_digits()

def show_digits(row_amount, col_amount):
    figure, axes = plt.subplots(nrows=row_amount, ncols=col_amount, figsize=(col_amount, row_amount))
    for item in zip(axes.ravel(), digits.images, digits.target):
        axes, image, target = item
        axes.imshow(image)
        axes.set_xticks([])
        axes.set_yticks([])
        axes.set_title(target)
    plt.tight_layout()
    plt.show()

show_digits(4, 6)
show_digits(6, 6)
X_train, X_test, y_train, y_test = train_test_split(digits.data, digits.target, random_state=11, train_size=0.8)
print('X_test size: ', X_test.shape)
print('X_train size: ', X_train.shape)

knn = KNeighborsClassifier(n_neighbors=5)
svc = SVC()
gnb = GaussianNB()

knn.fit(X_train, y_train)
svc.fit(X_train, y_train)
gnb.fit(X_train, y_train)

knn_predicted = knn.predict(X_test)
expected = y_test
print('predicted: ', knn_predicted[:36])
print('expected:  ', expected[:36])

print(f'KNN Accuracy: {knn.score(X_test, y_test) * 100:.3f}%')
print(f'SVC Accuracy: {svc.score(X_test, y_test) * 100:.3f}%')
print(f'GNB Accuracy: {gnb.score(X_test, y_test) * 100:.3f}%')

confusion = confusion_matrix(y_true=expected, y_pred=knn_predicted)
print(confusion)

names = [str(digit) for digit in digits.target_names]
print(classification_report(expected, knn_predicted, target_names=names))

k_range = range(1, 15)
accuracy_scores = []
for k in k_range:
    knn = KNeighborsClassifier(n_neighbors=k)
    knn.fit(X_train, y_train)  # Train the classifier using the current value of k
    accuracy = knn.score(X_test, y_test)  # Evaluate the accuracy of the classifier on the test set
    accuracy_scores.append(accuracy)  # Append the accuracy score to the list

# Generate a plot of the accuracy scores as a function of k
plt.plot(k_range, accuracy_scores)
plt.xlabel('Number of Neighbors (k)')
plt.ylabel('Accuracy')
plt.title('KNN Classifier Accuracy')
plt.show()
print(f'Best accuracy: {max(accuracy_scores) * 100:.2f} with {accuracy_scores.index(max(accuracy_scores)) + 1} neighbours')
```