**ЛАБОРАТОРНА РОБОТА № 2**

**Мета роботи**: використовуючи спеціалізовані бібліотеки та мову програмування Python дослідити різні методи класифікації даних та навчитися їх порівнювати.

Github: [link](https://github.com/Ajoke32/AILabs)

**Завдання 1.** Класифікація за допомогою машин опорних векторів (SVM)

***Випишіть у звіт всі 14 ознак з набору даних – їх назви та що вони позначають та вид (числові чи категоріальні).***

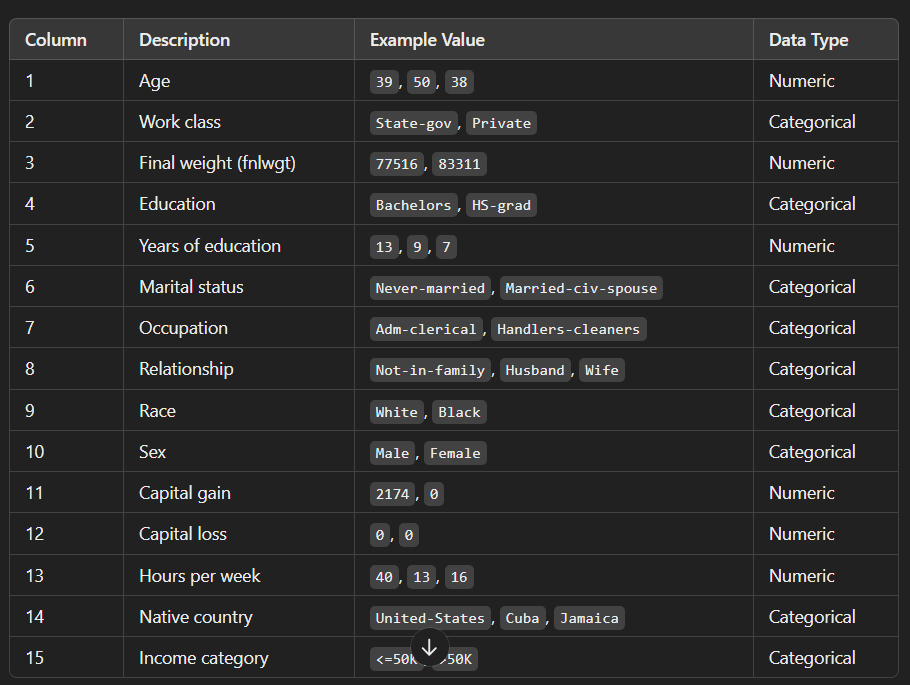


Рис 1. Ознаки

***Обчисліть значення інших показників якості класифікації (акуратність, повнота, точність) та разом з F1 занесіть їх у звіт.***

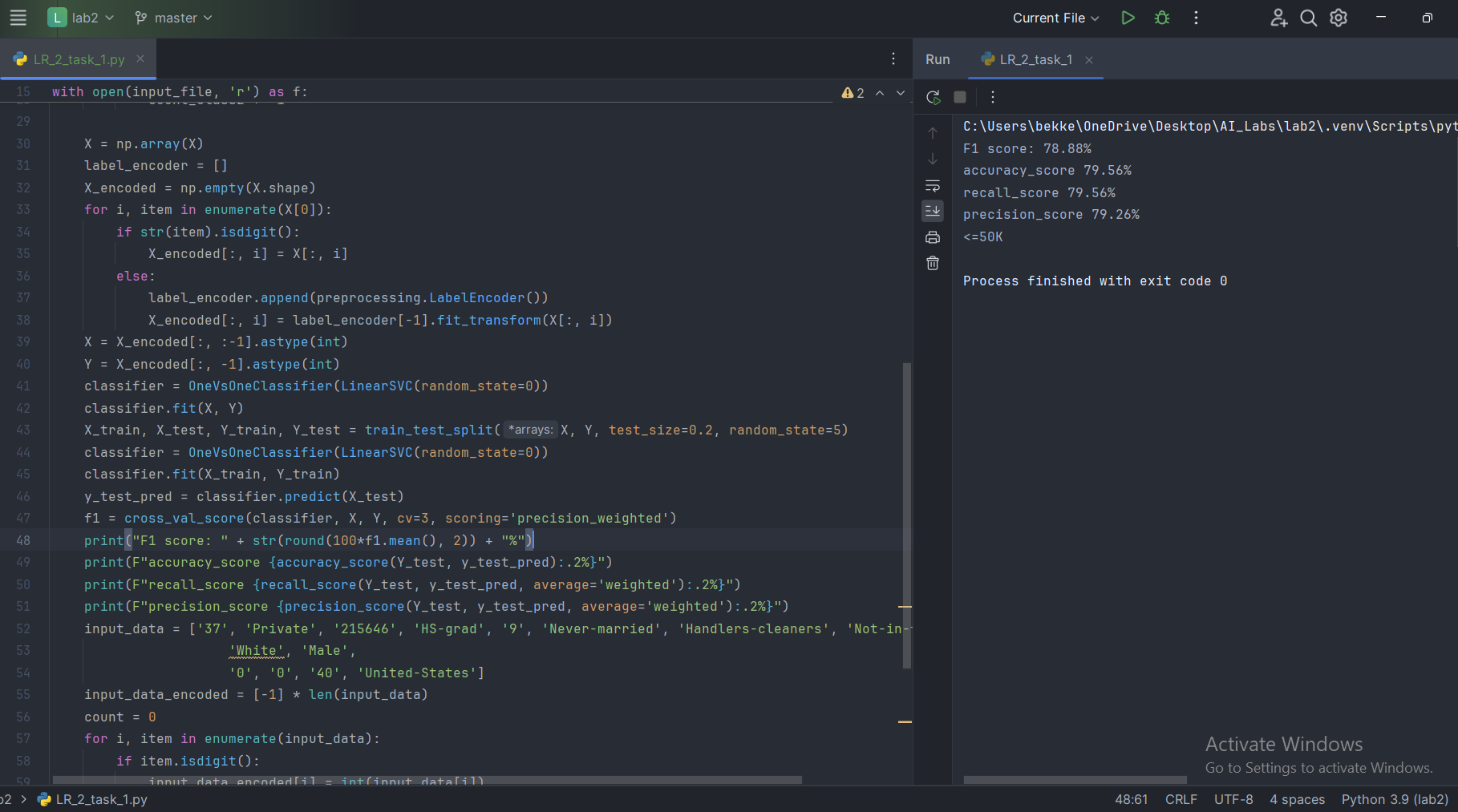


Рис 2. Розрахунки

***Код програми занесіть у звіт.***

1. import numpy as np  
   from sklearn import preprocessing  
   from sklearn.svm import LinearSVC  
   from sklearn.multiclass import OneVsOneClassifier  
   from sklearn.model\_selection import train\_test\_split, cross\_val\_score  
   from sklearn.metrics import accuracy\_score, recall\_score, precision\_score  
     
   input\_file = 'income\_data.txt'  
     
   X = []  
   Y = []  
   count\_class1 = 0  
   count\_class2 = 0  
   max\_datapoints = 25\_000  
   with open(input\_file, 'r') as f:  
    for line in f.readlines():  
    if count\_class1 >= max\_datapoints and count\_class2 >= max\_datapoints:  
    break  
    if '?' in line:  
    continue  
    data = line[:-1].split(', ')  
    if data[-1] == '<=50K' and count\_class1 < max\_datapoints:  
    X.append(data)  
    count\_class1 += 1  
     
    if data[-1] == '>50K' and count\_class2 < max\_datapoints:  
    X.append(data)  
    count\_class2 += 1  
     
    X = np.array(X)  
    label\_encoder = []  
    X\_encoded = np.empty(X.shape)  
    for i, item in enumerate(X[0]):  
    if str(item).isdigit():  
    X\_encoded[:, i] = X[:, i]  
    else:  
    label\_encoder.append(preprocessing.LabelEncoder())  
    X\_encoded[:, i] = label\_encoder[-1].fit\_transform(X[:, i])  
    X = X\_encoded[:, :-1].astype(int)  
    Y = X\_encoded[:, -1].astype(int)  
    classifier = OneVsOneClassifier(LinearSVC(random\_state=0))  
    classifier.fit(X, Y)  
    X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, test\_size=0.2, random\_state=5)  
    classifier = OneVsOneClassifier(LinearSVC(random\_state=0))  
    classifier.fit(X\_train, Y\_train)  
    y\_test\_pred = classifier.predict(X\_test)  
    f1 = cross\_val\_score(classifier, X, Y, cv=3, scoring='precision\_weighted')  
    print(print("F1 score: " + str(round(100\*f1.mean(), 2)) + "%"))  
    print(F"accuracy\_score {accuracy\_score(Y\_test, y\_test\_pred):.2%}")  
    print(F"recall\_score {recall\_score(Y\_test, y\_test\_pred, average='weighted'):.2%}")  
    print(F"precision\_score {precision\_score(Y\_test, y\_test\_pred, average='weighted'):.2%}")  
    input\_data = ['37', 'Private', '215646', 'HS-grad', '9', 'Never-married', 'Handlers-cleaners', 'Not-in-family',  
    'White', 'Male',  
    '0', '0', '40', 'United-States']  
    input\_data\_encoded = [-1] \* len(input\_data)  
    count = 0  
    for i, item in enumerate(input\_data):  
    if item.isdigit():  
    input\_data\_encoded[i] = int(input\_data[i])  
    else:  
    input\_data\_encoded[i] = int(label\_encoder[count].transform([input\_data[i]])[0])  
    count += 1  
    input\_data\_encoded = np.array(input\_data\_encoded).reshape(1, 14)  
    predicted\_class = classifier.predict(input\_data\_encoded)  
    print(label\_encoder[-1].inverse\_transform(predicted\_class)[0])

***Зробіть висновок до якого класу належить тестова точка***

**Тестова точка, відноситься до класу людей, у яких дохід менше або рівне 50К**

**Завдання 2.** Порівняння якості класифікаторів SVM з нелінійними ядрами

**Поліноміальне ядро**

Розрахунки проводяться довго, результатів досягнути не вдалося.

**Гаусове ядро**

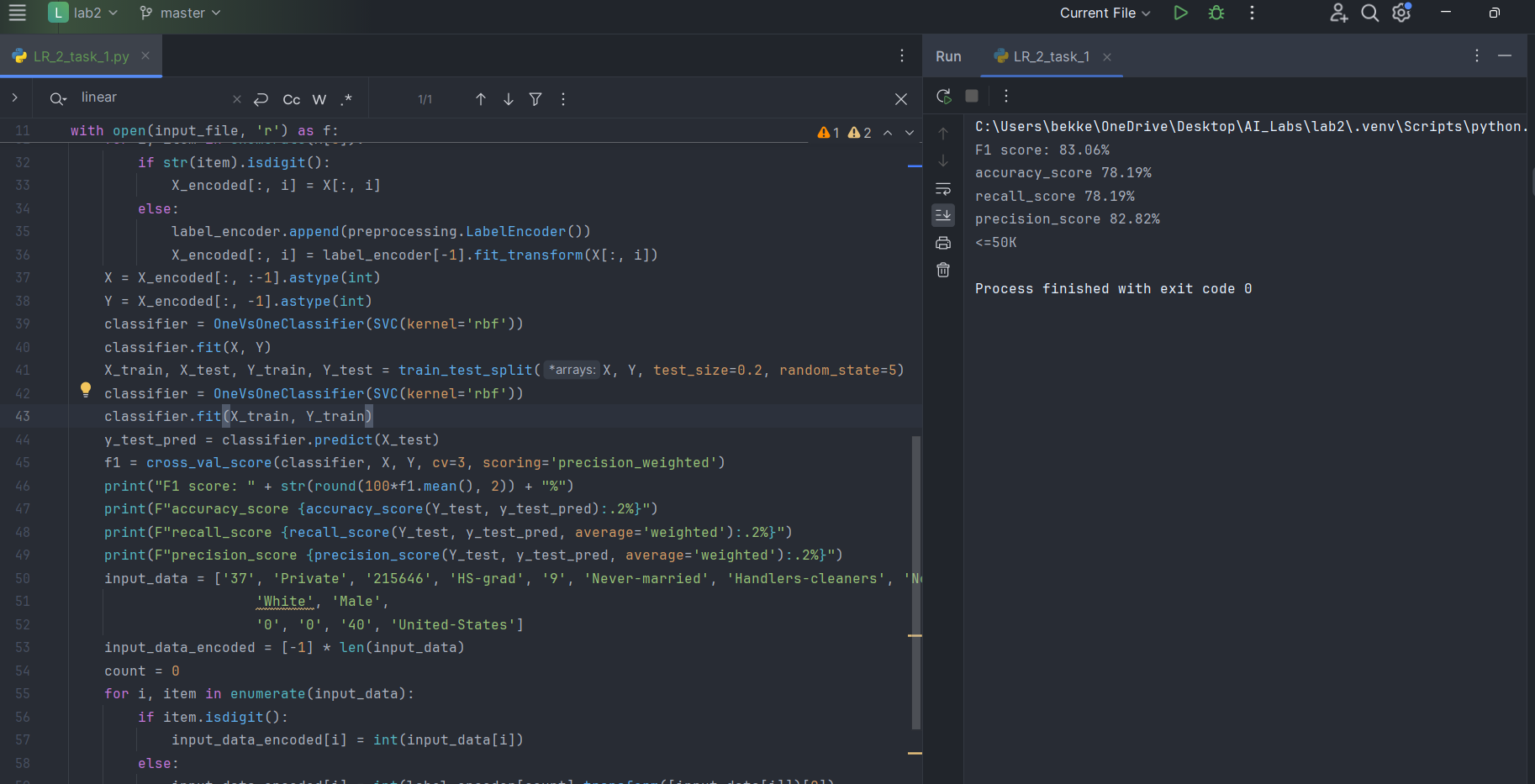


Рис 3. Гаусове ядро

**Сигмоїдальне ядро**

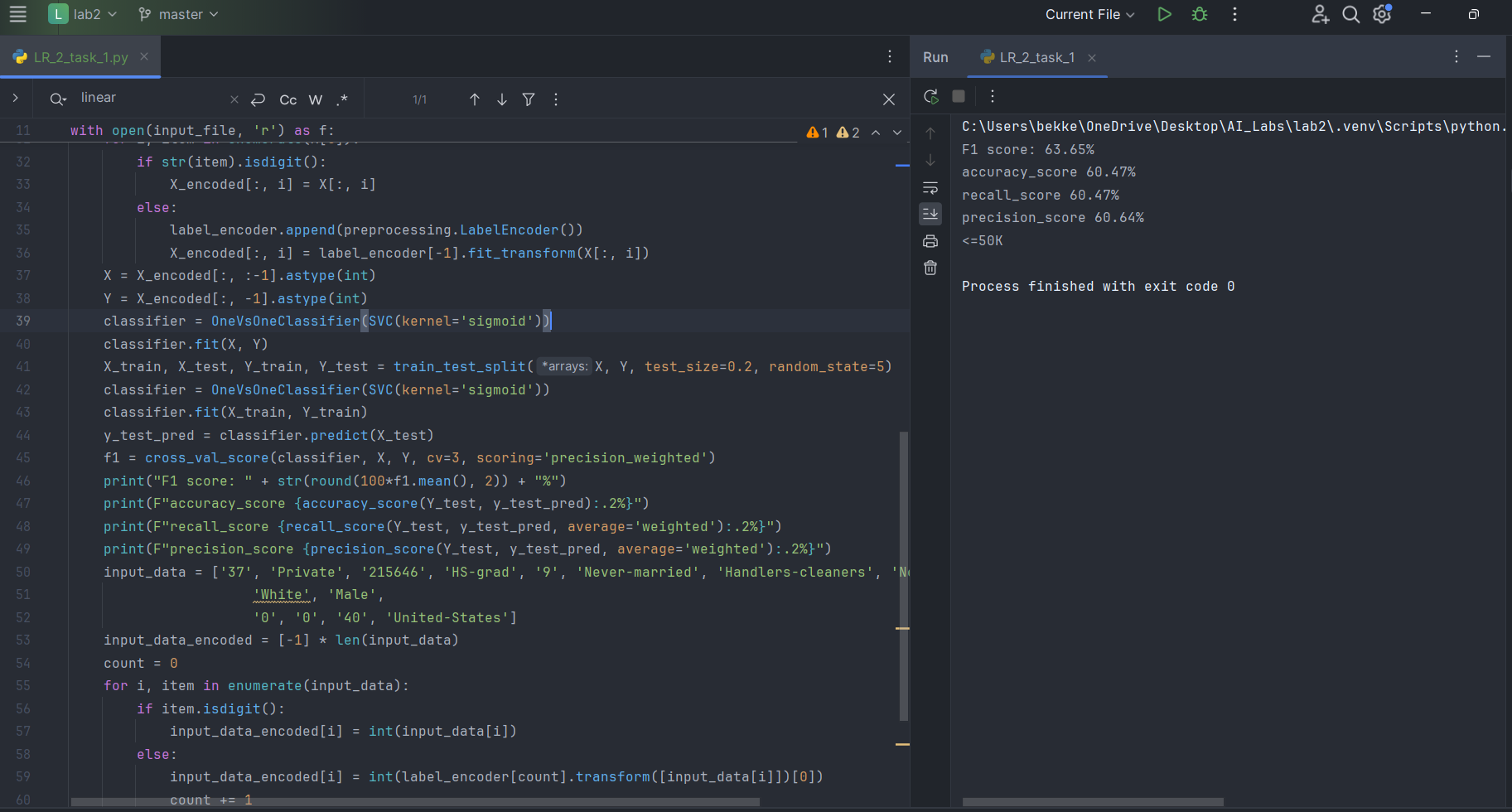
****

Рис 4. Сигмоїдальне ядро

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Kernel** | **F1** | **Accuracy** | **Recall** | **Precision** | **Predict** |
| Linear | 78.88 | 79.56 | 79.56 | 79.26 | <= 50 |
| Poly | None | None | None | None | None |
| Rbf | 83.06 | 78.19 | 78.19 | 82.82 | <= 50 |
| Sigmoid | 63.65 | 60.47 | 60.47 | 60.64 | <= 50 |

**Висновок:** відповідно до показників, найкраще виконує завдання Гаусовське та Лінійне ядра, але всі ці ядра мають однакові результати: <= 50

**Завдання 3.** Порівняння якості класифікаторів на прикладі класифікації сортів ірисів.

**Ознаки та дані**

***Виведіть значення ознак для перших п'яти прикладів:***

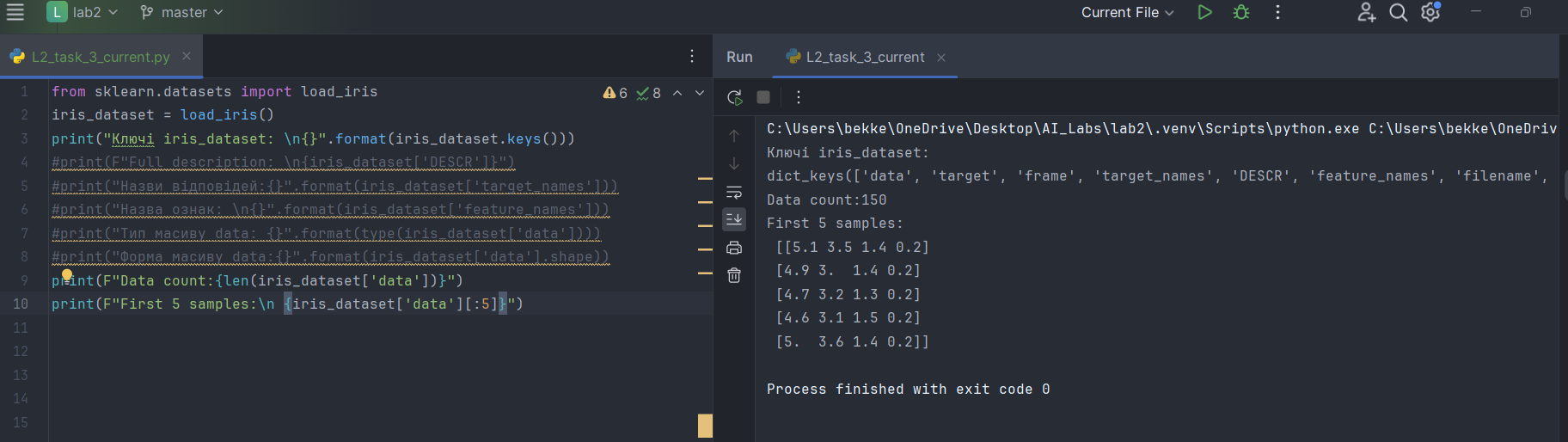
****

Рис 5. Перші п'ять прикладів

***Код для ознайомлення зі структурою даних та результати його виконання занесіть у звіт:***

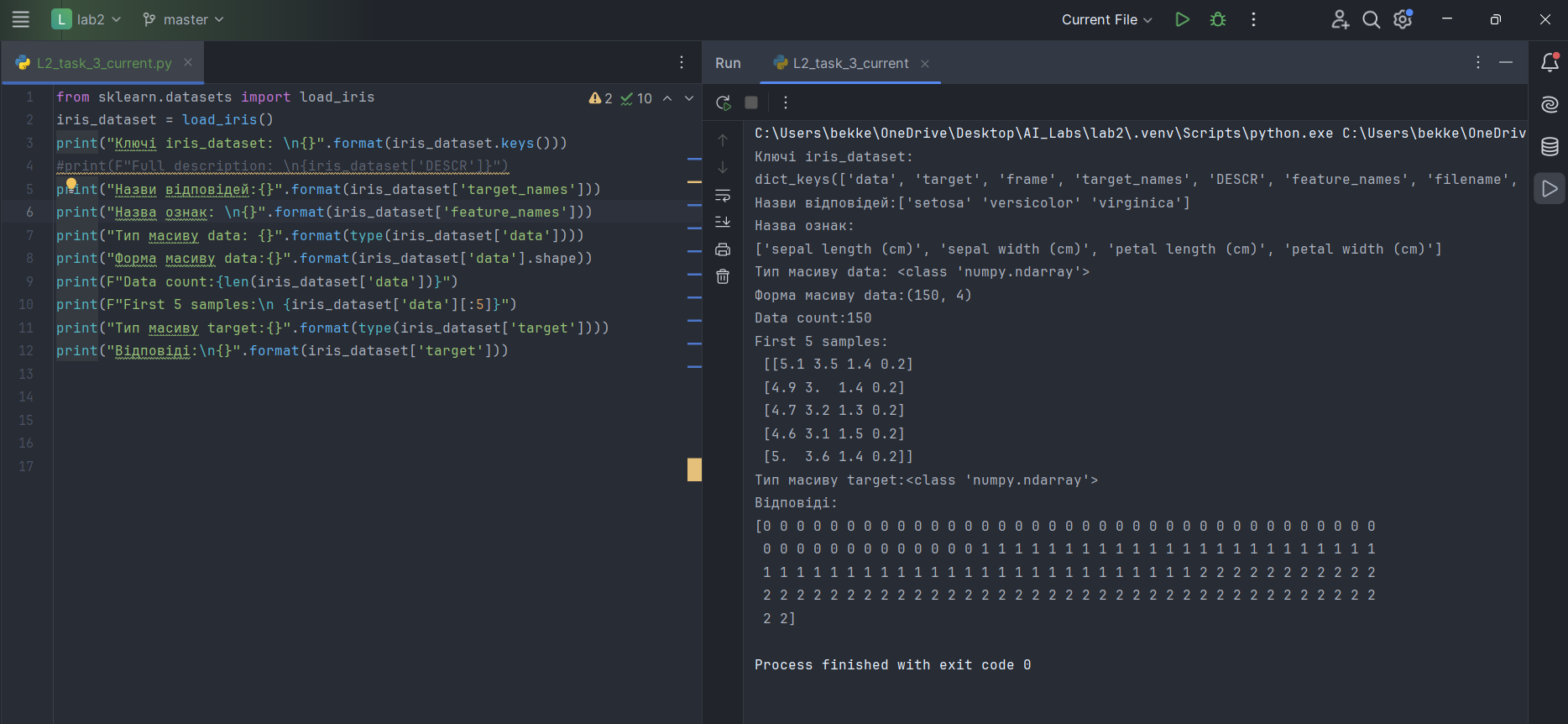
****

Рис 6. Структура даних та результати виконання

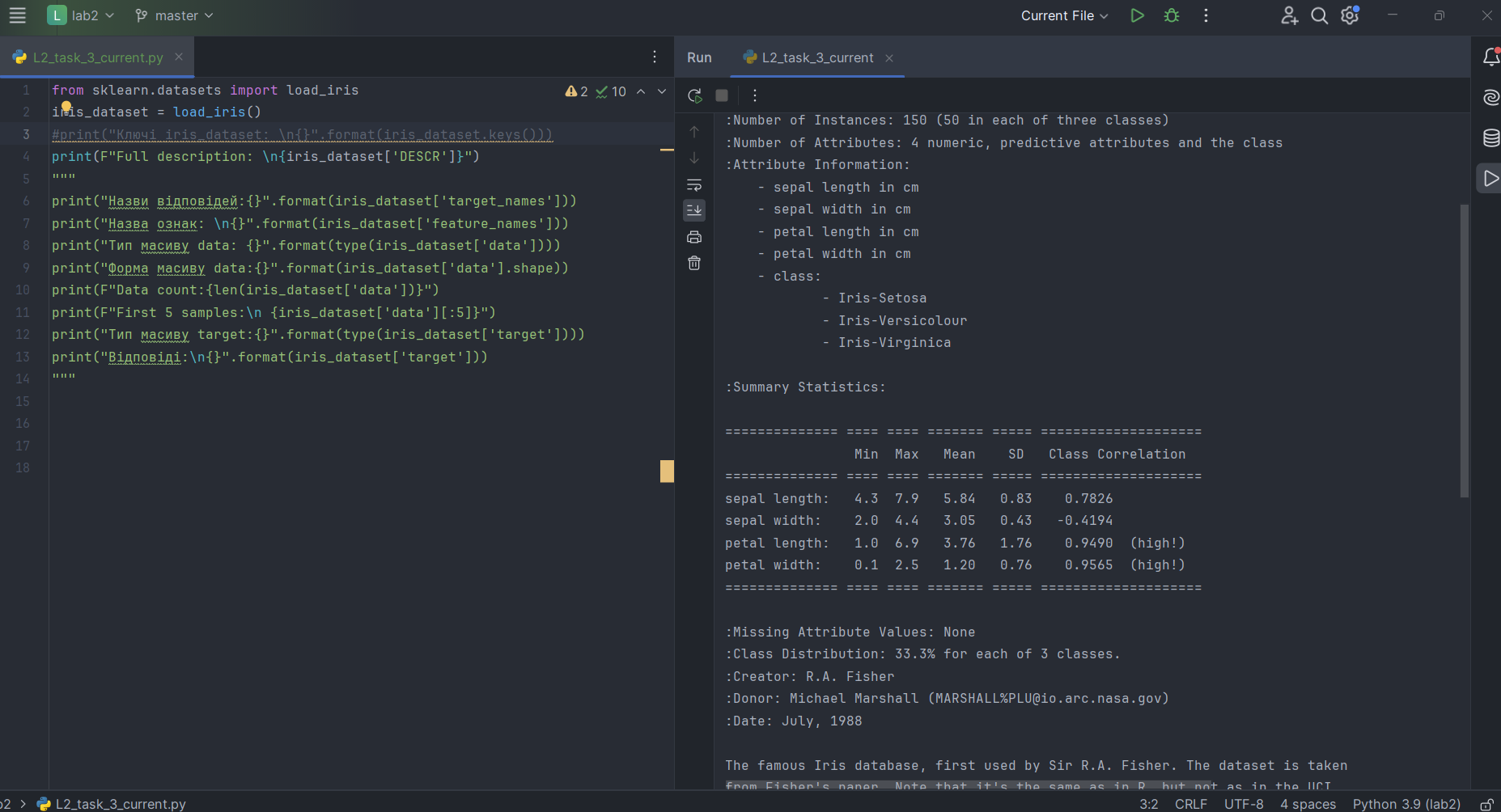
****

Рис 7. Ознайомлення зі структурою даних

***Створіть новий файл Python та імпортуйте такі пакети. Тут записані всі необхідні бібліотеки***

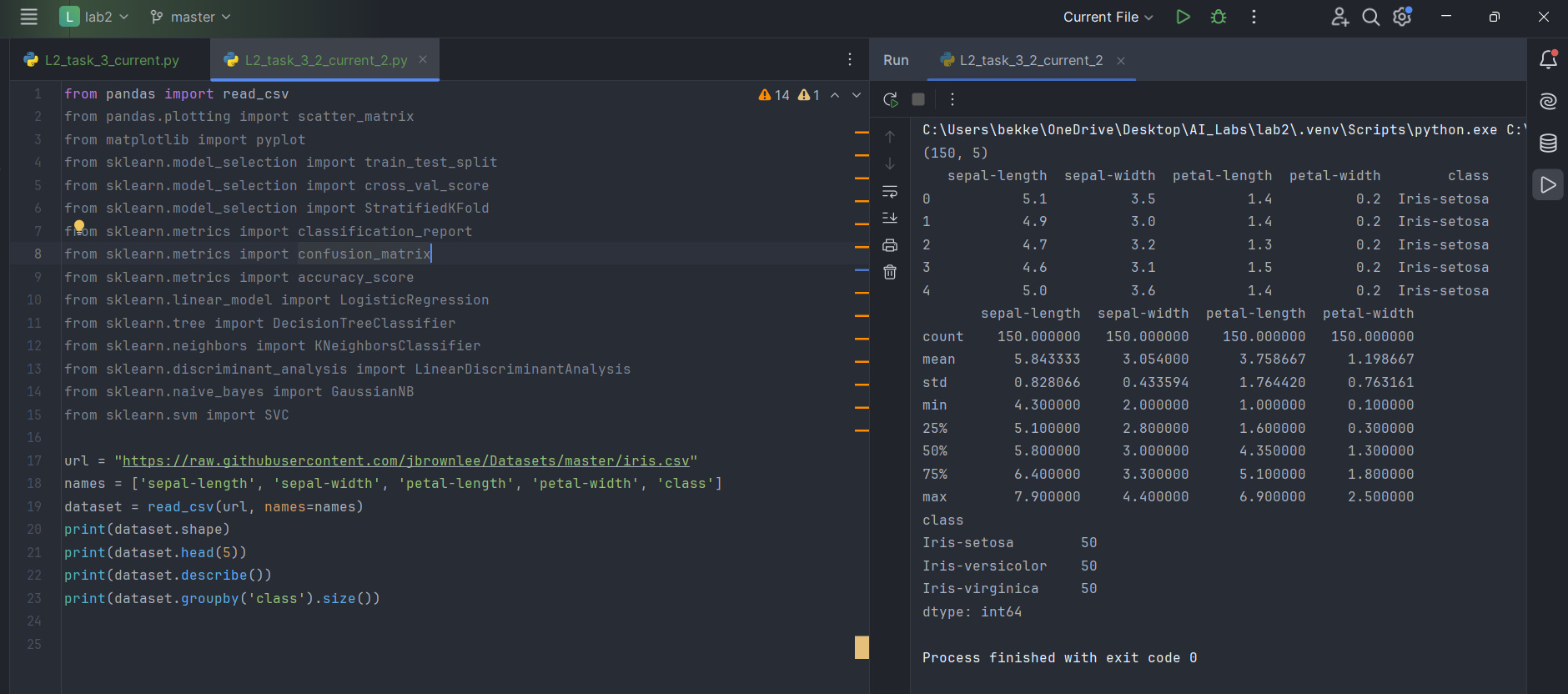
****

Рис 8. Ознайомлення зі структурою даних за допомогою нових методів

**Візуалізація даних**

*Одновимірні графіки*

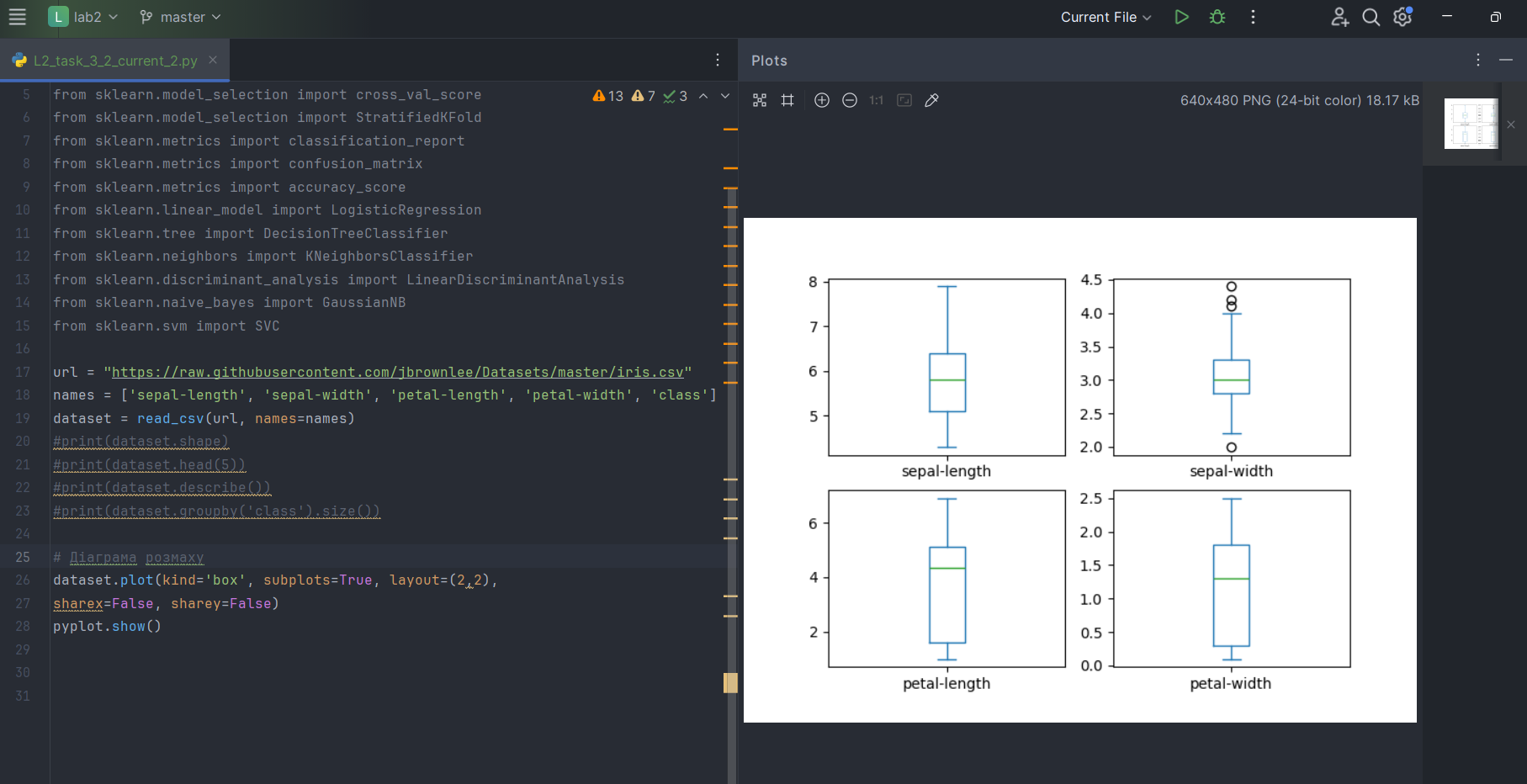
****

Рис 9. Box and whiskers diagram

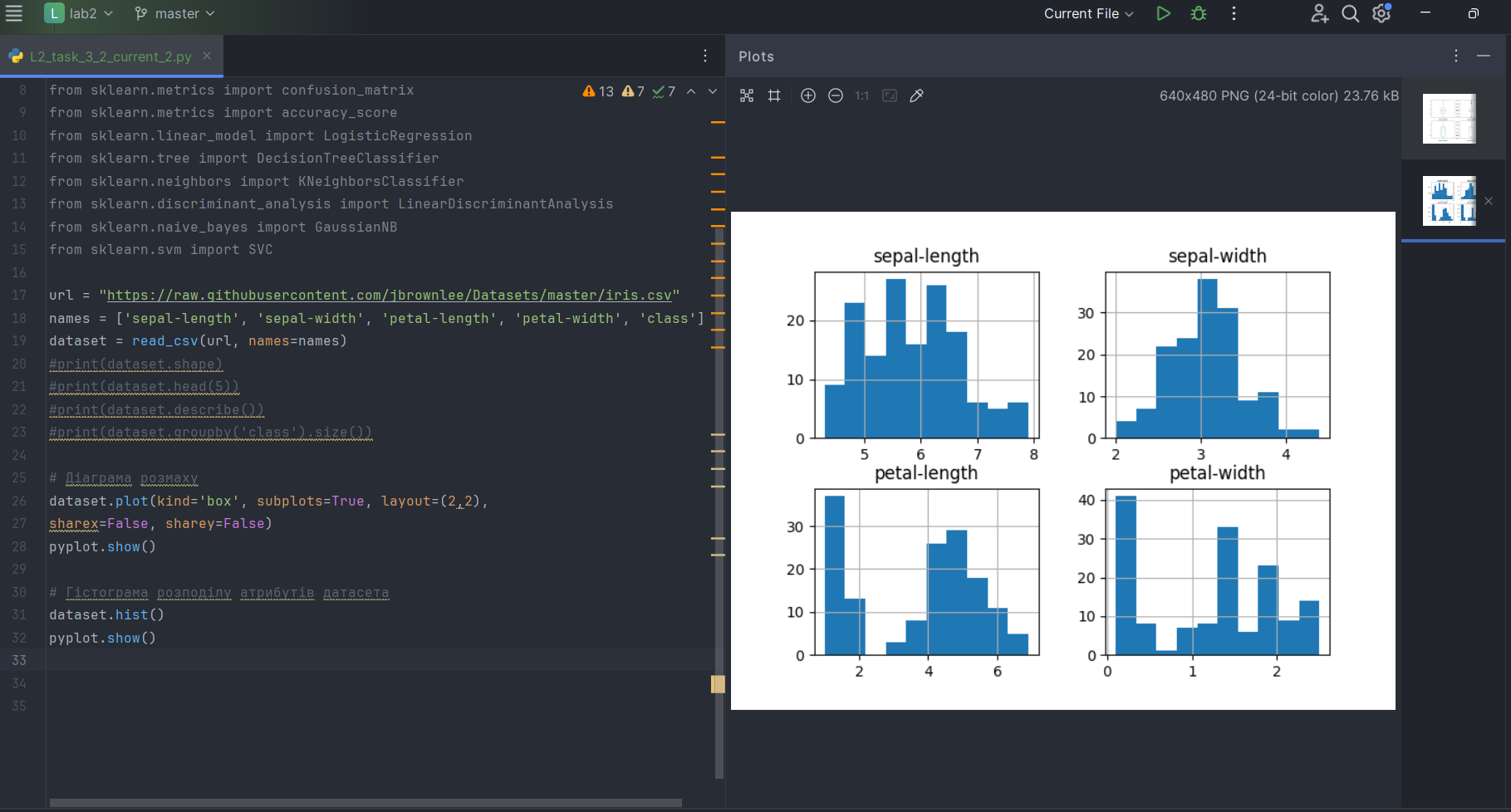
****

Рис 10. Гістограми

*Багатовимірні графіки*

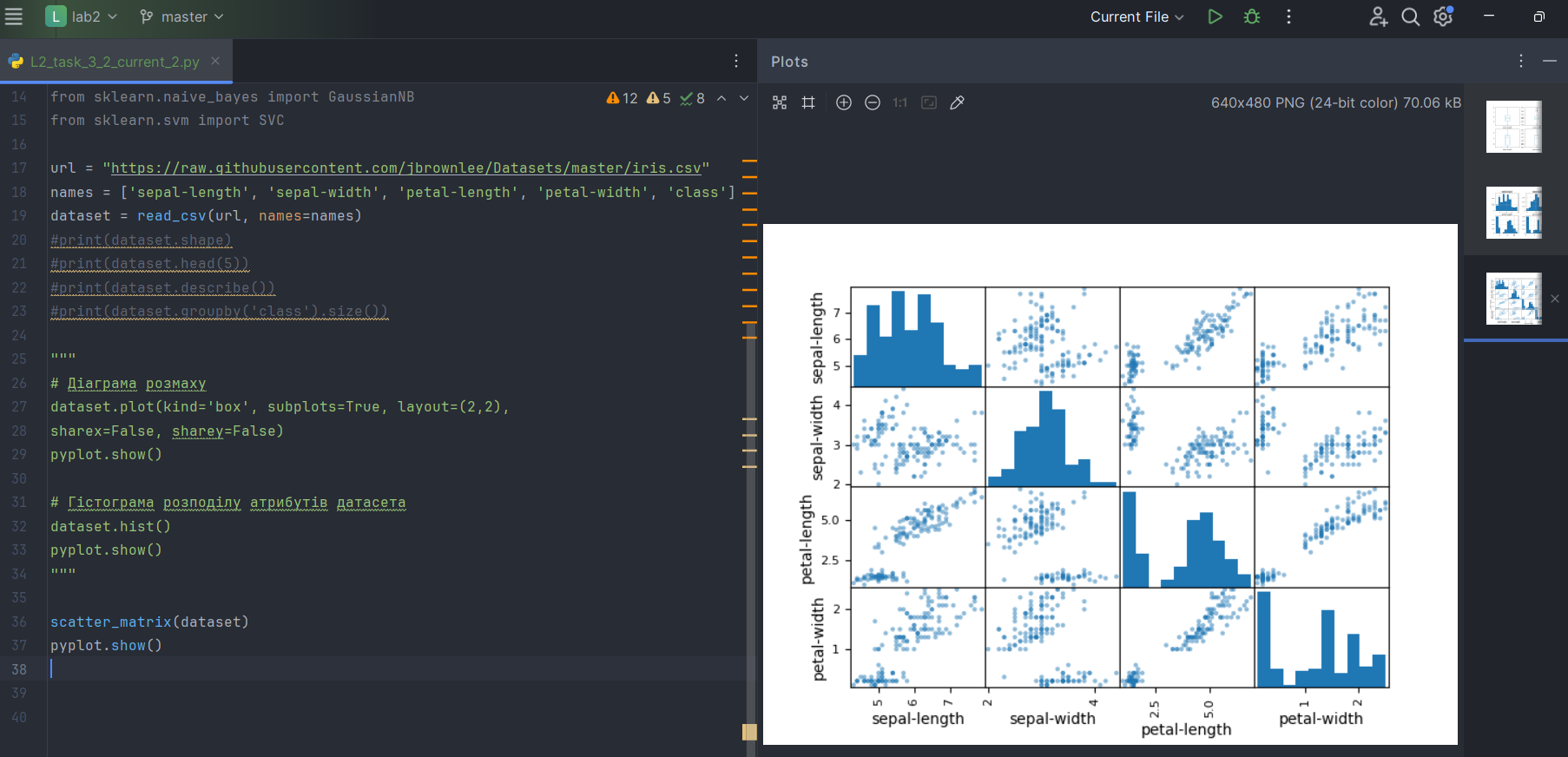
******

Рис 11. Scatterplot matrix

**Створення навчального та тестового наборів**

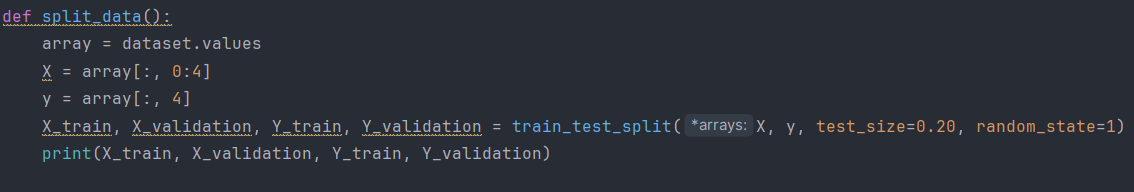
****

Рис 12. Створення навчального та тестового наборів

**Класифікація (побудова моделі)**

***Отримані графіки та результати занесіть у звіт Виберіть та напишіть чому обраний вами метод класифікації ви вважаєте найкращим***

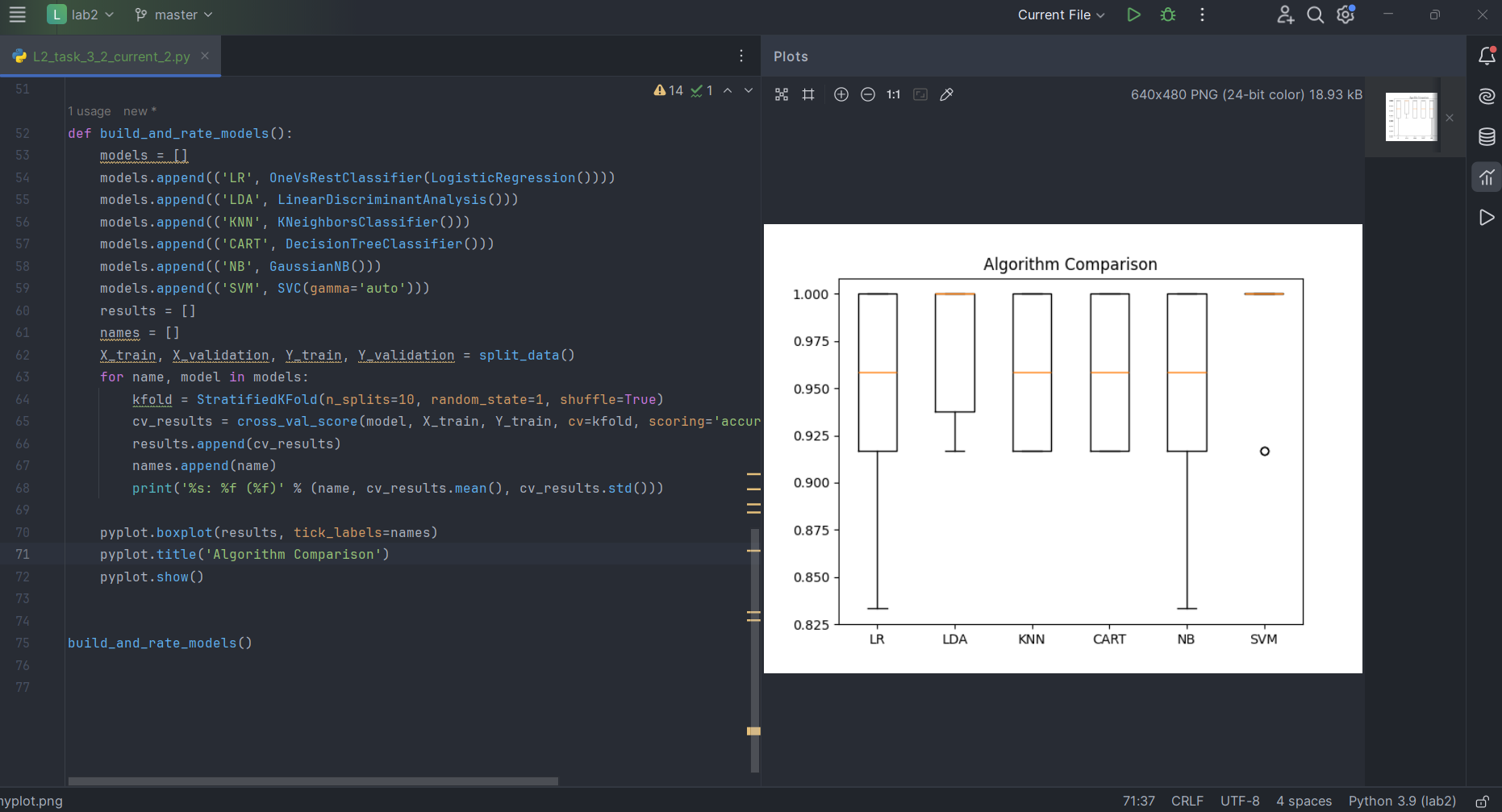
****

Рис 13. Box and whiskers diagram

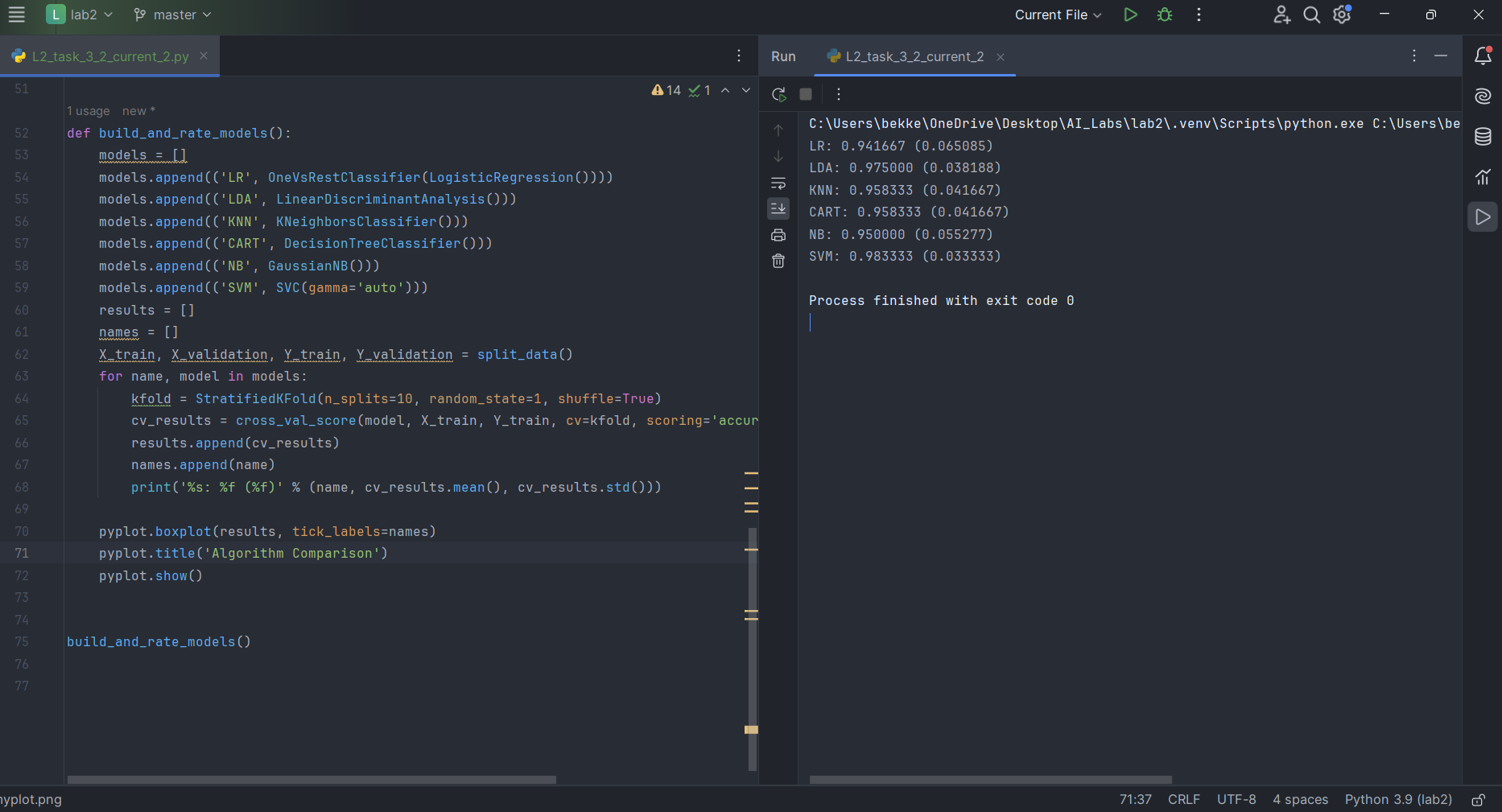
****

Рис 14. Розрахунки

**Висновок:** відповідно до розрахунків, найкращий метод – SVM, тому що він має найбільшу точність та найменше відхилення, але тут не враховані інші показники і значення інших моделей близькі за значеннями, тому важко обрати точно.

Відповідно до графіків, найкращим методом є також SVM, оскільки його boxpot виглядає як одна лінія, що означає, що метод є стабільним та забезпечує високу точність. Проте, є одне значення – викид, що свідчить про потенційні аномалії в деяких випадках.

**Крок 5. Оптимізація параметрів моделі**

**Крок 6. Отримання прогнозу (передбачення на тренувальному наборі)**

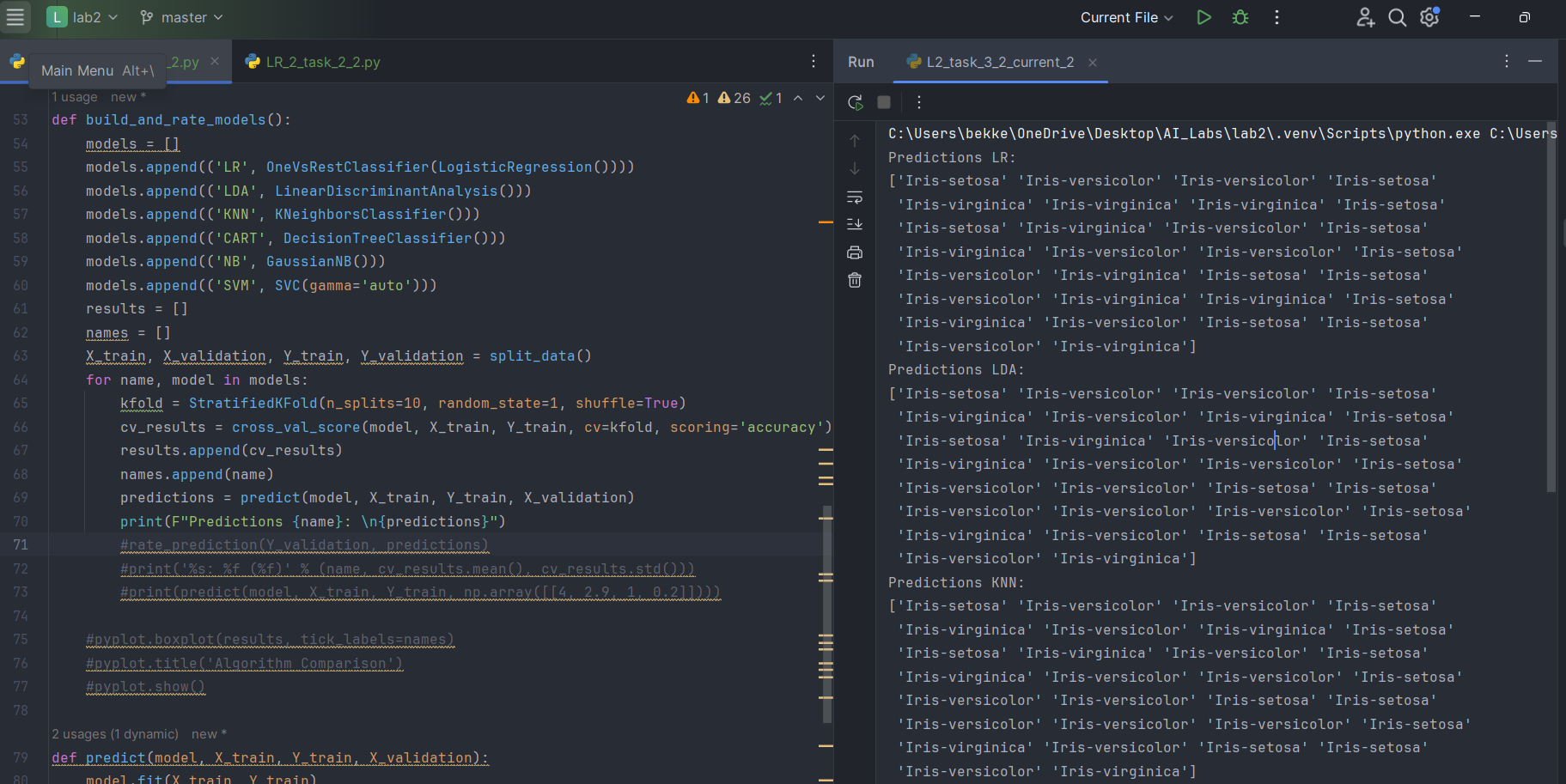
****

Рис 15. Отримання прогнозу

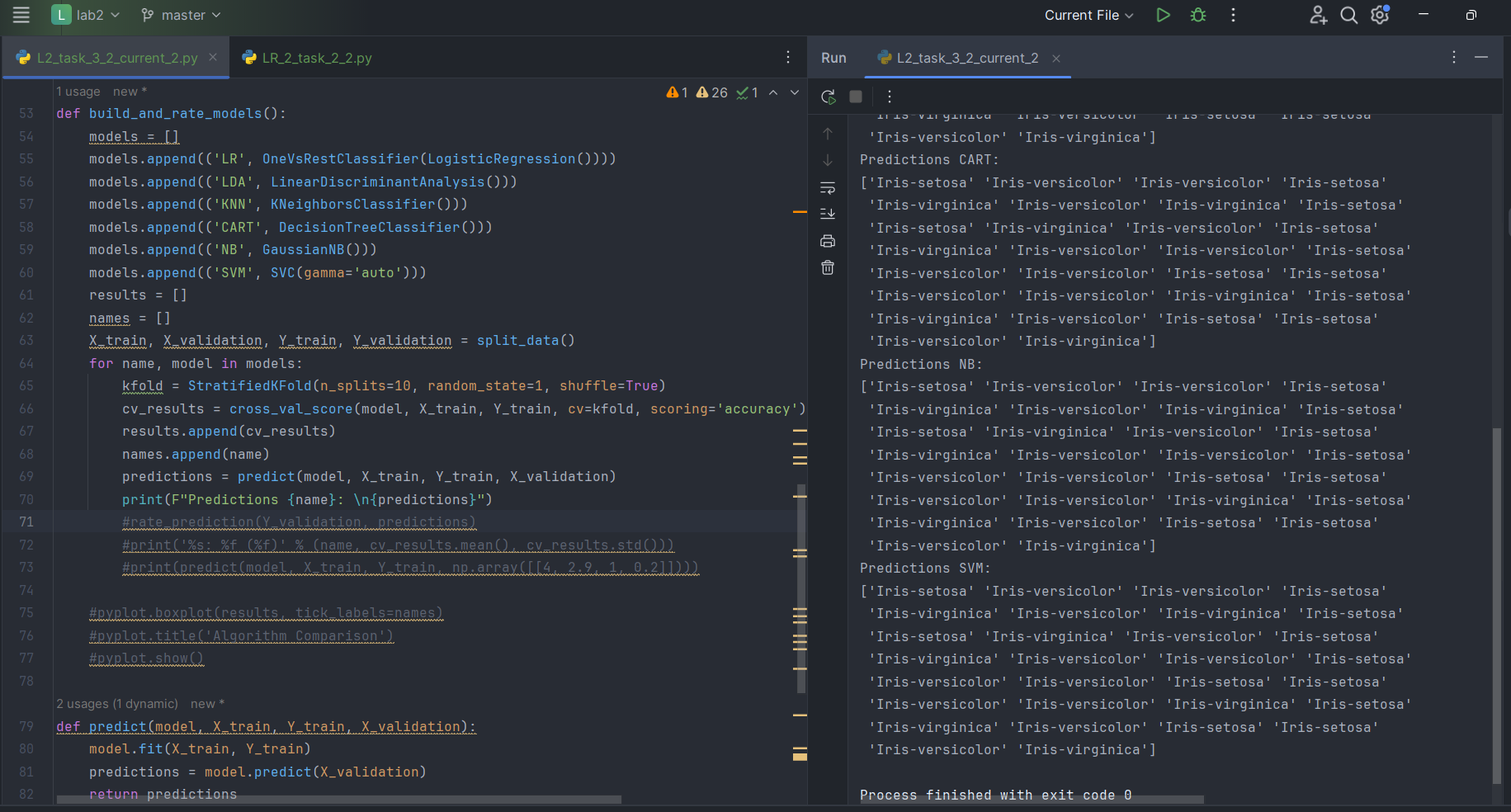
****

Рис 16. Отримання прогнозу

**Крок 7. Оцінка якості моделі**

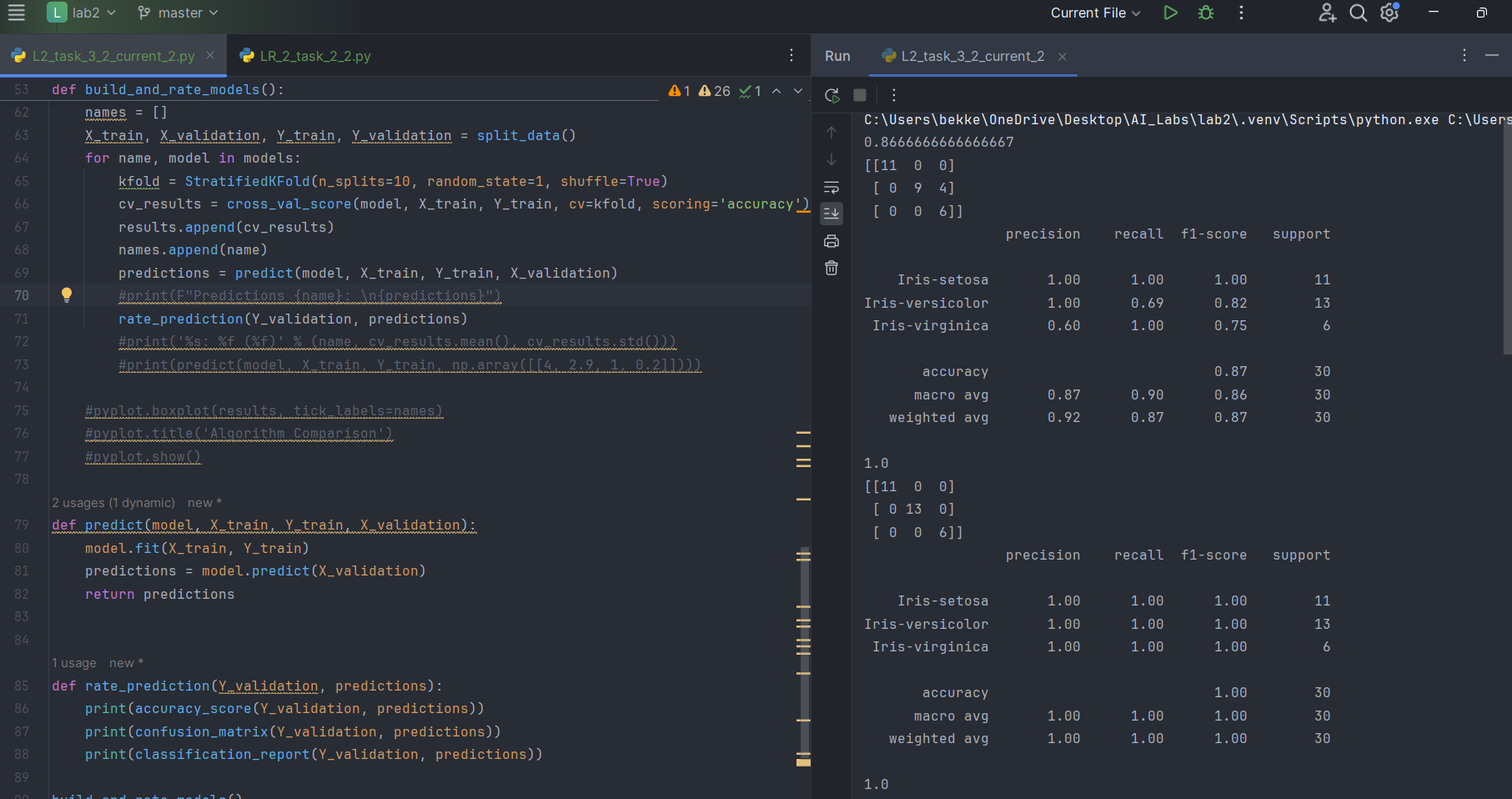
****

Рис 17. Оцінка якості моделей

**Крок 8. Отримання прогнозу (застосування моделі для передбачення)**

print("Спрогнозированная метка: {}".format( iris\_dataset['target\_names'][prediction])) – не зрозуміло, який результат очікується, тому на зображенні вивів назву класу.

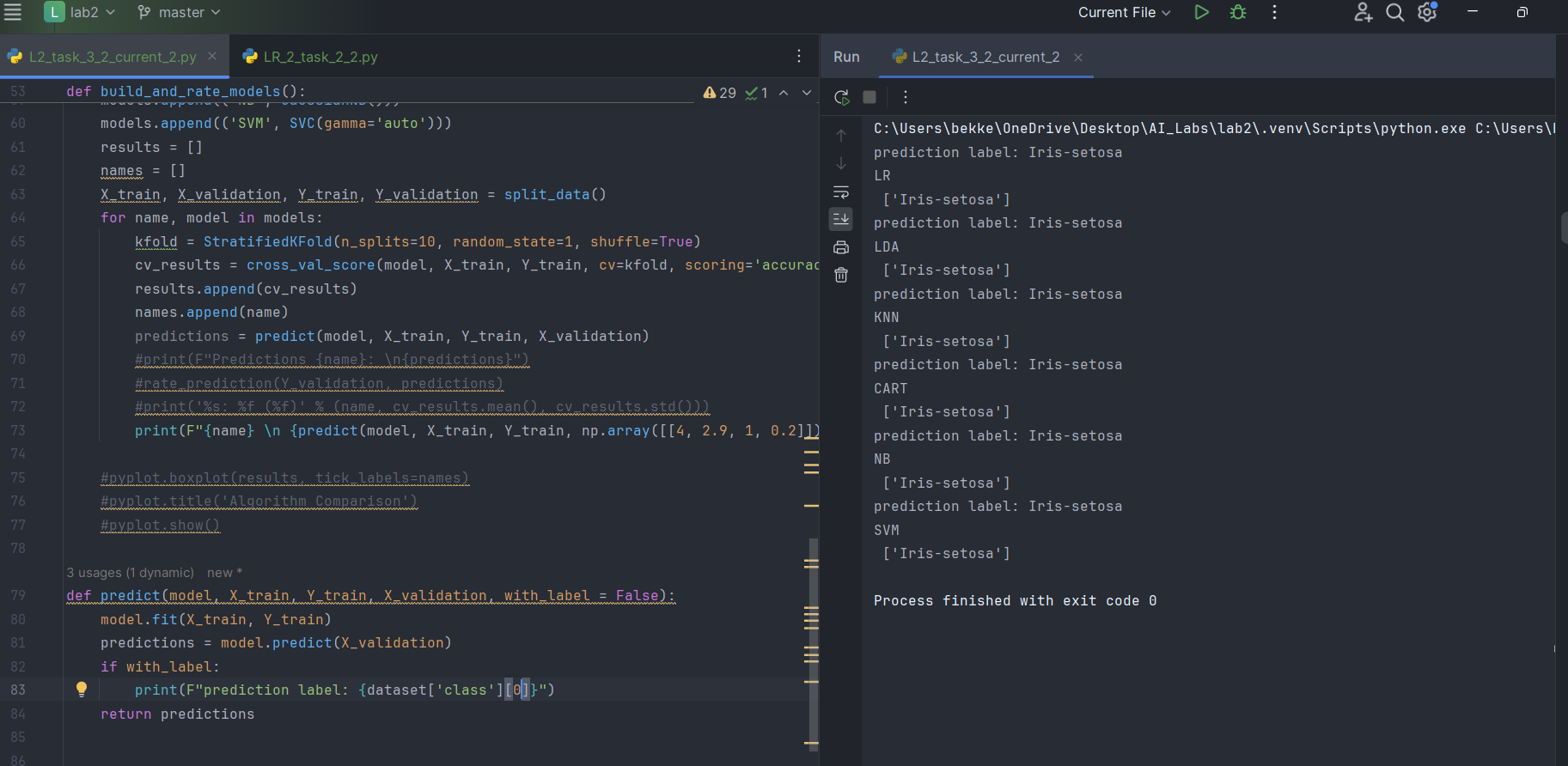
****

Рис 18. Прогнози моделей

**Висновок:** Квітка за кроку 8 належить до класу ‘Iris-setosa’. Якості класифікацій наведені в кроці 7.

**4. Порівняння якості класифікаторів для набору даних завдання 2.1**

***Коди та результати занесіть у звіт:***

from pandas import read\_csv  
from sklearn.linear\_model import LogisticRegression  
from sklearn.multiclass import OneVsRestClassifier  
from sklearn.tree import DecisionTreeClassifier  
from sklearn.neighbors import KNeighborsClassifier  
from sklearn.discriminant\_analysis import LinearDiscriminantAnalysis  
from sklearn.naive\_bayes import GaussianNB  
from sklearn.svm import SVC  
from sklearn.metrics import classification\_report  
from sklearn.model\_selection import train\_test\_split  
import numpy as np  
from sklearn import preprocessing  
  
names = ['age', 'work-class', 'final-weight',  
 'education', 'years-of-education', 'marital-status',  
 'occupation', 'relationship', 'race',  
 'sex', 'capital-gain', 'capital-loss',  
 'hours-per-week', 'native-country', 'income'  
 ]  
  
dataset = read\_csv('income\_data.txt', names=names)  
models = [('LR', OneVsRestClassifier(LogisticRegression())),  
 ('LDA', LinearDiscriminantAnalysis()),  
 ('KNN', KNeighborsClassifier()),  
 ('CART', DecisionTreeClassifier()),  
 ('NB', GaussianNB()),  
 ('SVM', SVC(gamma='auto'))  
 ]  
  
ignore = ['LDA', 'KNN', 'CART', 'NB', 'SVM']  
array = dataset.values  
  
  
def encode():  
 X\_encoded = np.empty(array.shape)  
 label\_encoders = []  
 for i in range(array.shape[1] - 1):  
 column = array[:, i]  
 if str(column).isdigit():  
 X\_encoded[:, i] = column  
 else:  
 le = preprocessing.LabelEncoder()  
 X\_encoded[:, i] = le.fit\_transform(column)  
 label\_encoders.append(le)  
  
 return (X\_encoded[:, :-1].astype(int), array[:, -1])  
  
  
def scale(X, y):  
 X\_train, X\_validation, Y\_train, Y\_validation = train\_test\_split(X, y, test\_size=0.20, random\_state=1)  
 scaler = preprocessing.StandardScaler()  
 X\_train = scaler.fit\_transform(X\_train)  
 X\_validation = scaler.transform(X\_validation)  
 return (X\_train, X\_validation, Y\_train, Y\_validation)  
  
  
X, y = encode()  
X\_train, X\_validation, Y\_train, Y\_validation = scale(X, y)  
reports = []  
for name, model in models:  
 model.fit(X\_train, Y\_train)  
 predictions = model.predict(X\_validation)  
 report = classification\_report(Y\_validation, predictions)  
 print(report)

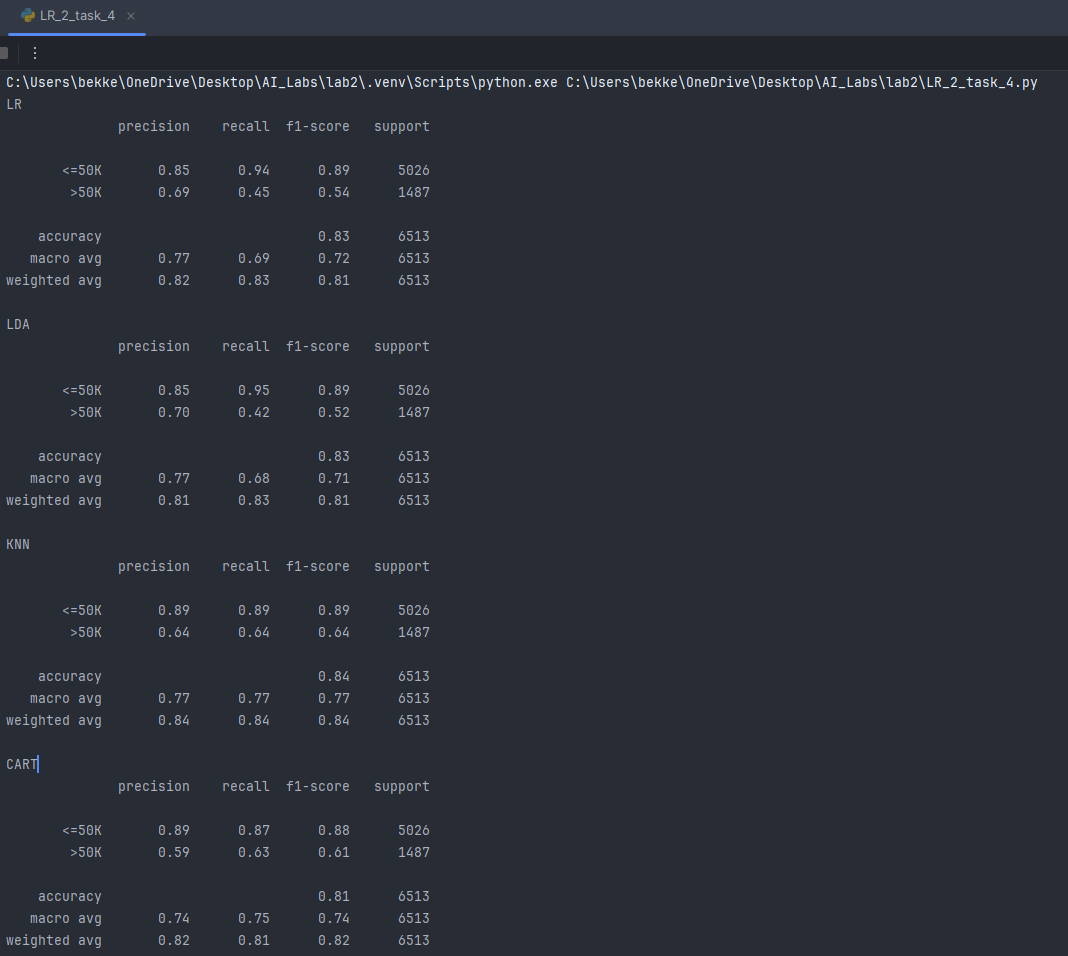


Рис 19. Оцінки якості моделей

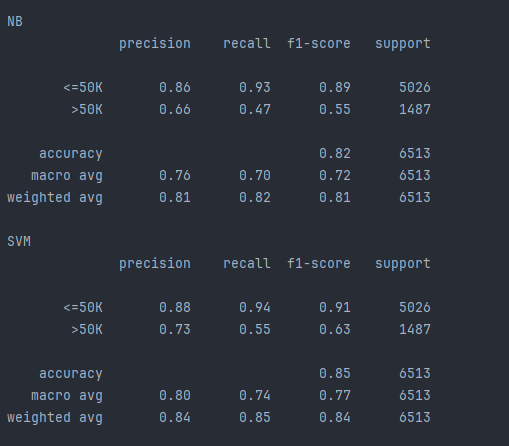
****

Рис 20. Оцінки якості моделей

**Висновок:** Мені важко визначити яку задачу ми вирішували, тому й важко обрати найкращу модель. Якщо задачею було знайти класифікатор, який краще визначає людей з <=50, то це буде один класифікатор, якщо >50, то другий, якщо потрібно в среденьому визначати, то третій. Можливо, існують інші зовнішні критериї, які також потрібно врахувати.  
Загалом, також важко порівнювати репорти, тому що їх багато, тому потрібно писати додаткові фунції, які оцінять найкращу модель, але цього не зазначено в завданні.

**Завдання 2.5. Класифікація даних лінійним класифікатором Ridge**

***Опишіть які налаштування класифікатора Ridge тут використані та що вони позначають***

[**Tol**](https://scikit-learn.org/dev/modules/generated/sklearn.linear_model.RidgeClassifier.html) **-** означає допуск на збіжність. Коли різниця в коефіцієнтах (вагах) між ітераціями стає меншою за це значення, розв'язувач завершує цикл. Це вказує на те, наскільки точною є процедура оптимізації.

[**Solver**](https://scikit-learn.org/dev/modules/generated/sklearn.linear_model.RidgeClassifier.html) **-** розв'язувач для використання в обчислювальних процедурах. Значення sagвикористовує метод стохастичного середнього градієнта, а «saga» - його неупереджену та гнучкішу версію під назвою SAGA. Обидва методи використовують ітераційну процедуру і часто є швидшими за інші розв'язувачі, коли n\_вибірок та n\_особливостей є великими.

***Опишіть які показники якості використовуються та їх отримані результати. Вставте у звіт та поясніть зображення Confusion.jpg***

|  |  |
| --- | --- |
| **Показник** | **Значення** |
| Accuracy | 0.7556 |
| Precision | 0.8333 |
| Recall | 0.7556 |
| F1 Score | 0.7503 |
| Cohen Kappa Score | 0.6431 |
| Matthews Corrcoef | 0.6831 |

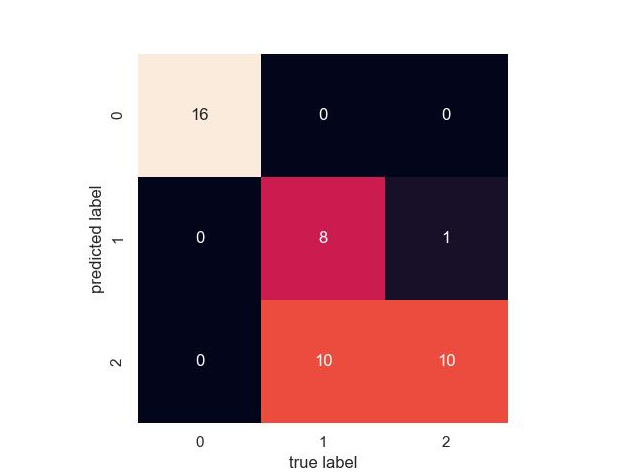


Рис 21. Confusion matrix

На даному зображенні по осі Х відображенні правильні відповіді. (правильні/вірні класи), а по осі У передбачені класифікатором чи моделлю класи (відповіді), тому:

Для «true label» 0, всі 16 екземлярів даних були правильно класифіковані як 0

Для «true label» 1, 10 екземлярів було неправильно класифіковано як 2 та 8 правильно класифікованих як 1

Для «true label» 2, 10 екземплярів даних було правильно класифіковано як 2 та 1 неправильно класифікований як 1

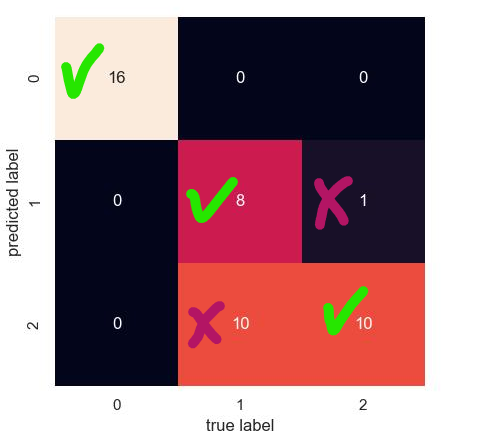


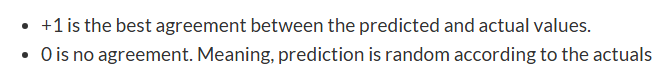
Рис 22. Візуалізація

Можна зробити висновки, що модель передбачує клас 0 ідеально, але плутається з 1 та 2 класами.

***Опишіть, що таке коефіцієнт Коена Каппа та коефіцієнт кореляції Метьюза. Що вони тут розраховують та що показують***

[Коефіцієнт Коена Каппа](https://datatab.net/tutorial/cohens-kappa)визначає наскільки збіг між двома оцінювачами перевищує те, що можна було б очікувати лише за умови випадковості, aле цей коефіцієнт не говорить чи ці оцінки є правдивими / вірними. Можливі значення [-1;1]. У нашому випадку, це значення = 0.6431, що вказує на помірну або значну згоду між двома оцінювачами.

[Коефіцієнт кореляції Метьюза](https://www.voxco.com/blog/matthewss-correlation-coefficient-definition-formula-and-advantages/) оцінює або виміряє різницю між прогнозованими даними і фактичними значеннями. Можливі значення [-1;1] У нашому випадку, ця різниця = 0.6831, що близько до 1, тому можна зробити висновок, що узгодження між передбачуваними даними та актуальними на хорошому рівні:

****

**Висновок:** У ході цієї лабораторної роботи я набув теоретичних і практичних навичок з дослідження різних методів класифікації даних, а також навчився порівнювати їх за допомогою мови програмування Python і спеціалізованих бібліотек.