МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ

імені Вадима Гетьмана

Кафедра інформатики та системології

ЛАБОРАТОРНА РОБОТА № 4

з дисципліни «Системи і методи штучного інтелекту»

Тема: «ДОСЛІДЖЕННЯ МЕТОДІВ РЕГРЕСІЇ»

Виконав: Виноградов Н.Р. ІН-405

Перевірив: Волошин А.П.

Київ – 2025

Мета роботи: використовуючи спеціалізовані бібліотеки та мову програмування Python дослідити методи регресії даних у машинному навчанні.

Хід роботи:

Завдання 2.1. Створення регресора однієї змінної Побудувати регресійну модель на основі однієї змінної. Використовувати файл вхідних даних: data_singlevar_regr.txt.

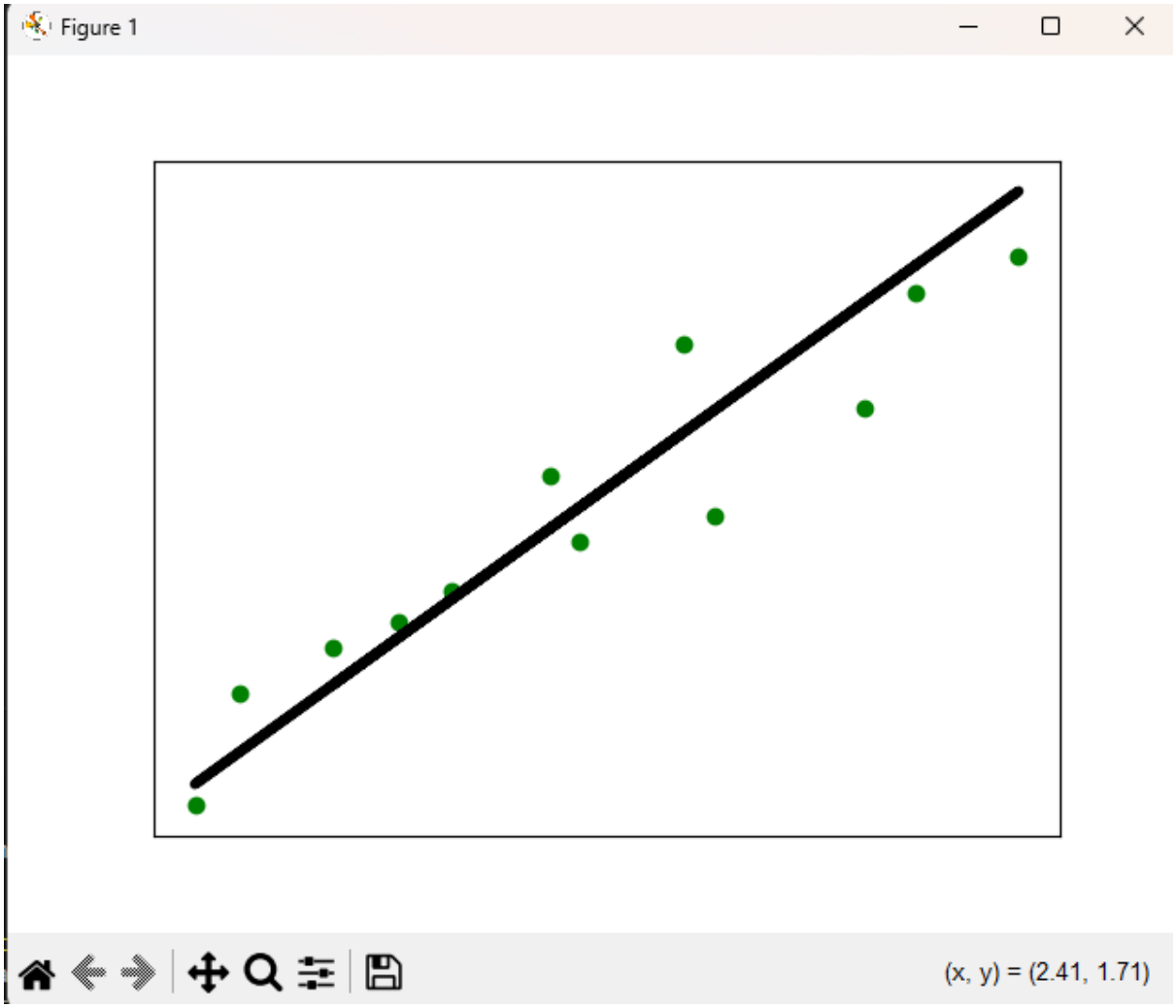

Рисунок 1.1 – Графік лінійної регресії

```
  input_file = 'C:\Codes\python co
Linear regressor performance:
Mean absolute error = 0.59
Mean squared error = 0.49
Median absolute error = 0.51
Explain variance score = 0.86
R2 score = 0.86

New mean absolute error = 0.59
```

Рисунок 1.2 – Результат роботи коду

Завдання 2.2. Передбачення за допомогою регресії однієї змінної Побудувати регресійну модель на основі однієї змінної. Використовувати вхідні дані відповідно свого варіанту, що визначається за списком групи у журналі.

Варіант 4, файл data_regr_4.txt

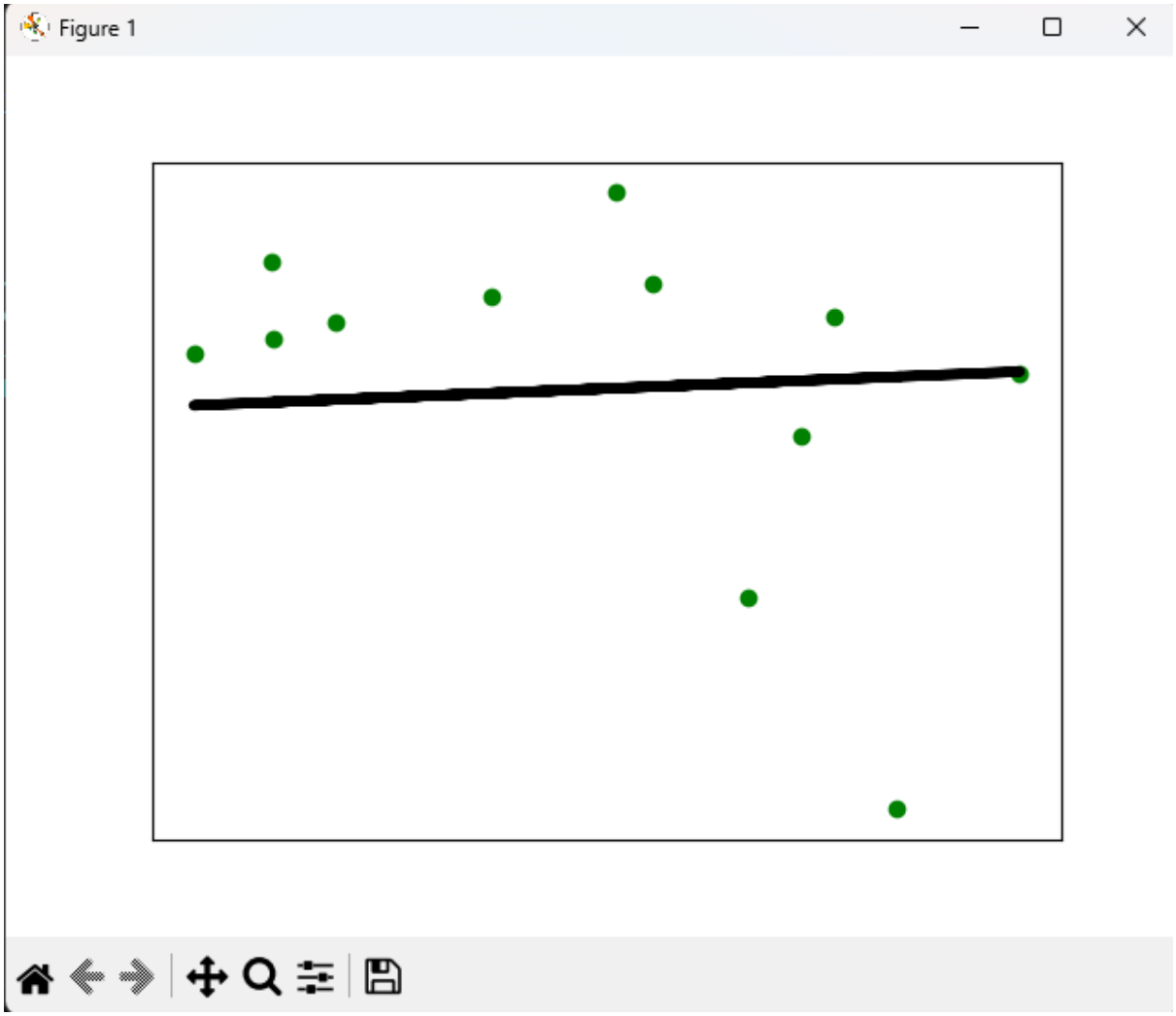

Рисунок 2.1 – Графік лінійної регресії

```
Linear regressor performance:
Mean absolute error = 2.72
Mean squared error = 13.16
Median absolute error = 1.9
Explain variance score = -0.07
R2 score = -0.07

New mean absolute error = 2.72
```

Рисунок 2.2 – Результат роботи коду

Завдання 2.3. Створення багатовимірного регресора. Використовувати файл вхідних даних: data_multivar_regr.txt, побудувати регресійну модель на основі багатьох змінних

```
Linear regression:
 [36.05286276]

Polynomial regression:
 [41.08289808]
```

Рисунок 3.1 – Результат роботи коду

Завдання 2.4. Регресія багатьох змінних Розробіть лінійний регресор, використовуючи набір даних по діабету, який існує в sklearn.datasets.

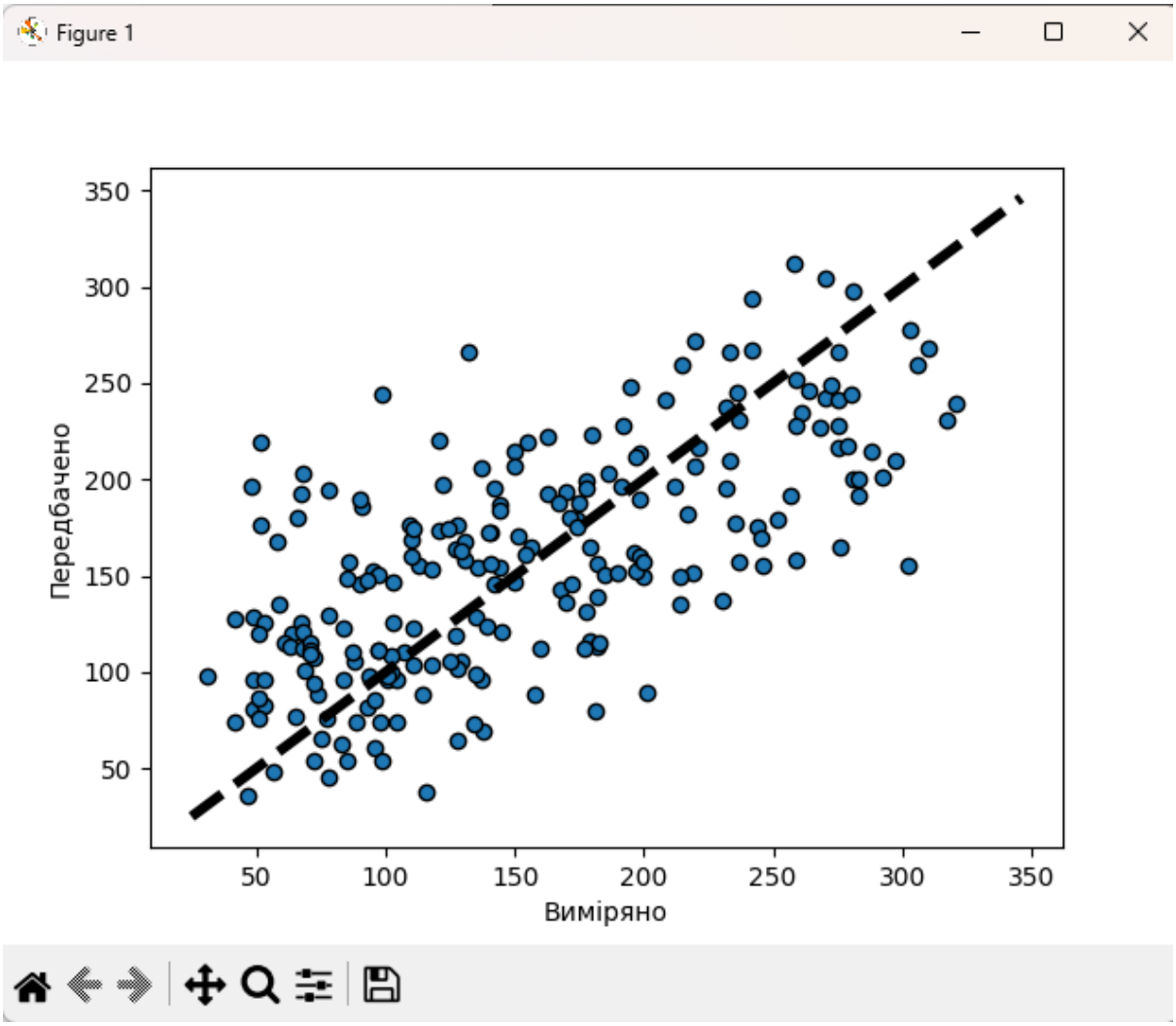

Рисунок 4.1 – Графік лінійної регресії багатьох змінних

```
Regr Coef = [ -20.4047621  -265.88518066  564.65086437  325.56226865 -692.16120333
  395.55720874   23.49659361  116.36402337  843.94613929   12.71856131]
Regr Intercept = 154.3589285280134
Mean absolute error = 44.8
Mean squared error = 3075.33
R2 score = 0.44
```

Рисунок 4.2 – Результат роботи коду

Завдання 2.5. Самостійна побудова регресії Згенеруйте свої випадкові дані обравши за списком відповідно свій варіант та виведіть їх на графік. Побудуйте по них модель лінійної регресії, виведіть на графік. Побудуйте по них модель поліноміальної регресії, виведіть на графік. Оцініть її якість.

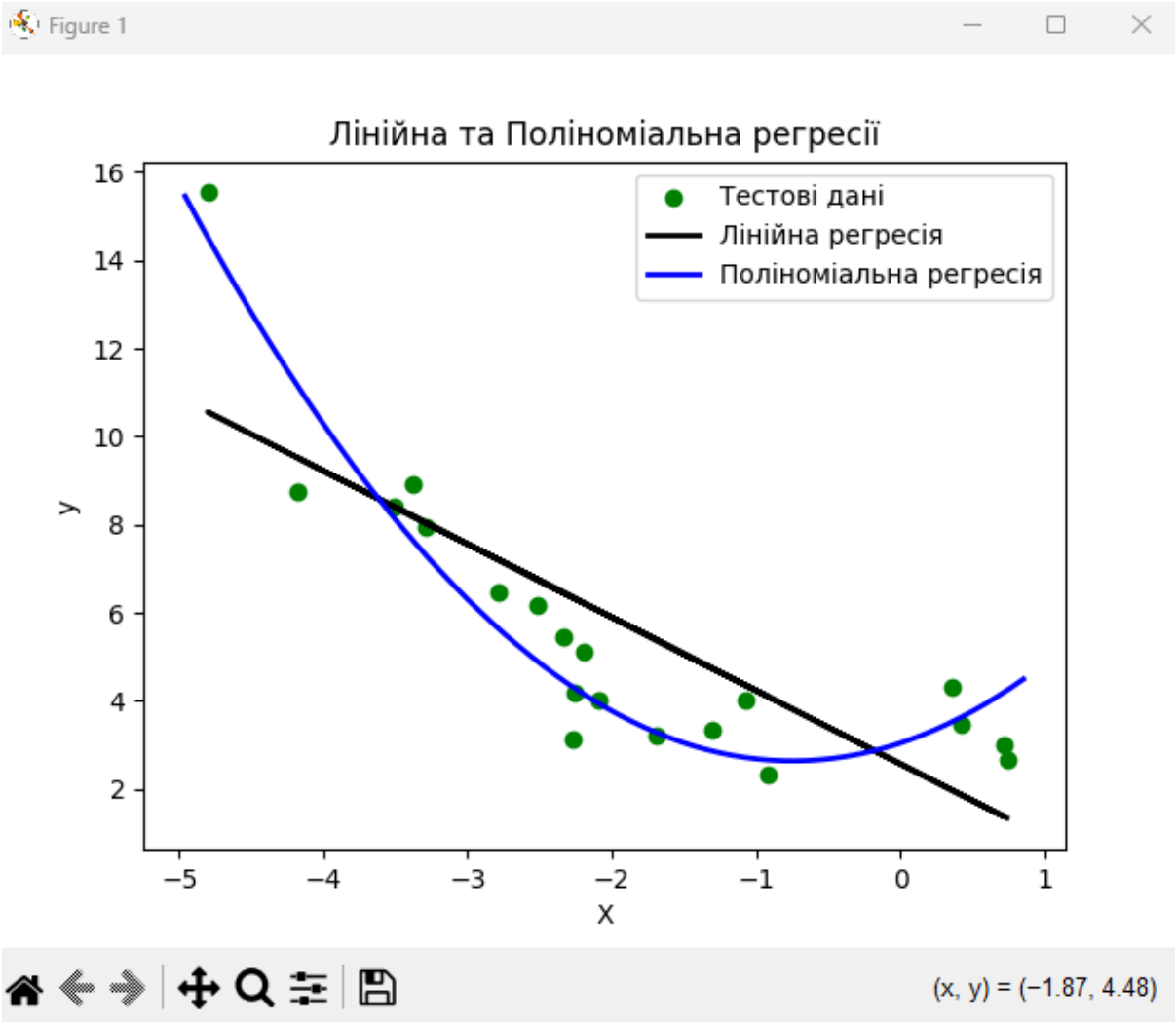

Рисунок 5.1 – Графік лінійної та поліноміальної регресії

Модель 4 варіанта у вигляді математичного рівняння:

$y\hat{} = 0.5x^2 + 1x + 3$ + шум гаусса

Отримана модель регресії з передбаченими коефіцієнтами:

$y\hat{}=0.72x^2+1.05x+3.14$

Завдання 2.6. Побудова кривих навчання. Побудуйте криві навчання для ваших даних у попередньому завданні.

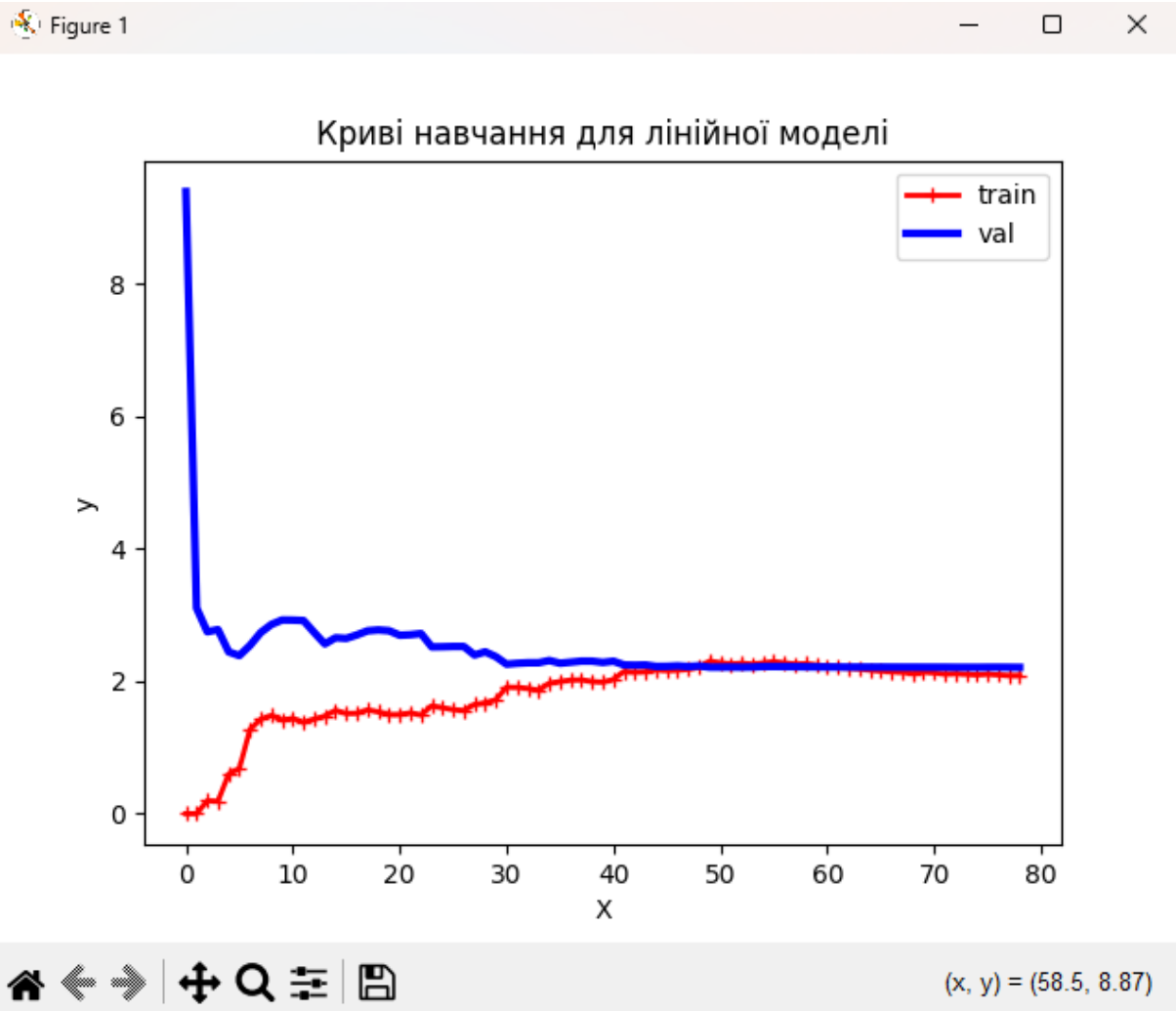

Рисунок 6.1 – Графік кривих навчань лінійного регресора

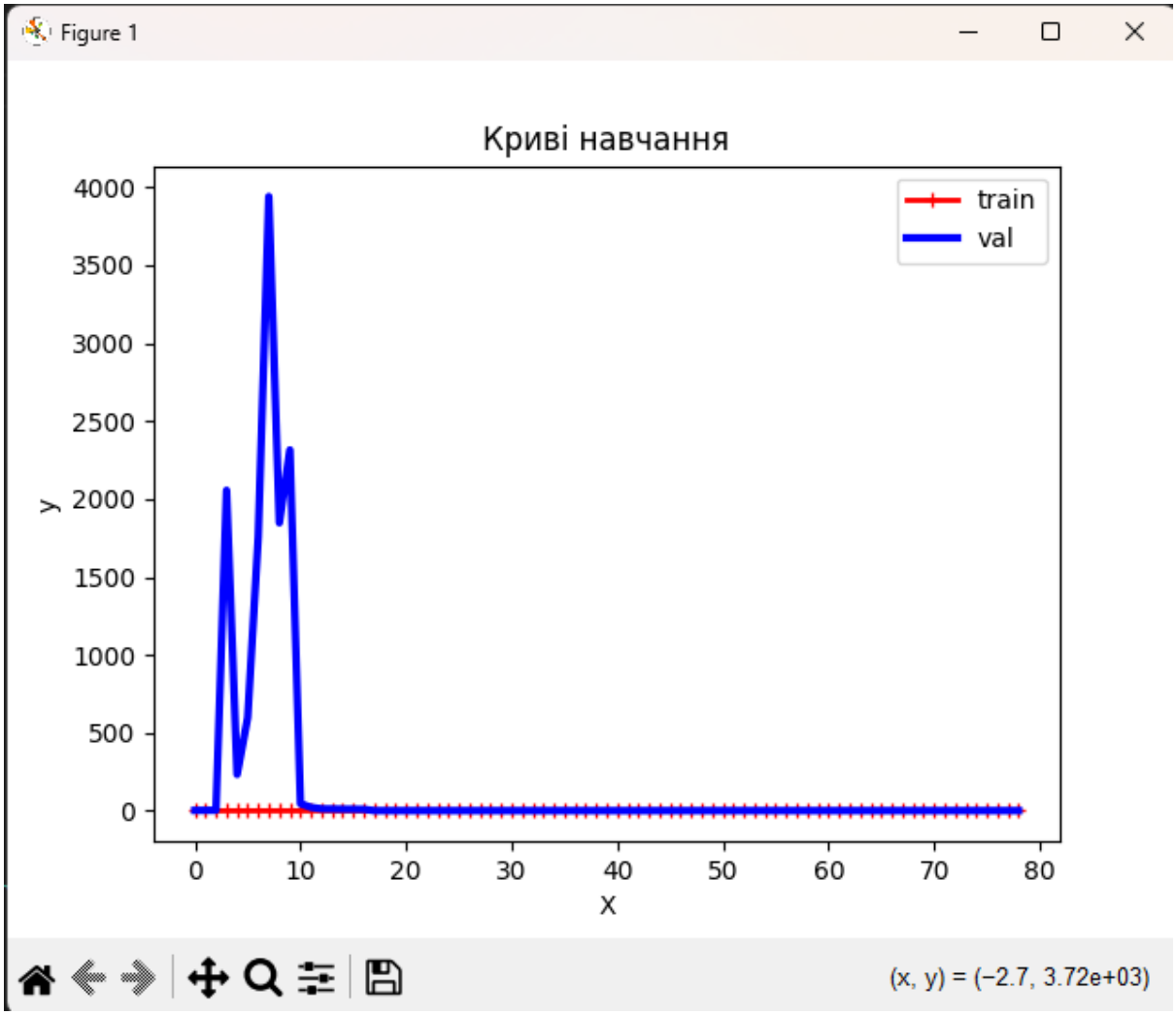

Рисунок 6.1 – Графік кривих навчань поліноміального регресора

Висновок

Під час виконання лабораторної роботи було розроблено декілька файлів на мові Python для лінійного та поліноміального регресора

Коди:

4.1

```
import pickle

import numpy as np

from sklearn import linear_model

import sklearn.metrics as sm

import matplotlib.pyplot as plt


# Вхідний файл, який містить дані
```

```
input_file = 'C:\Codes\python codes\AIUniver\AI_KNEU_2025\Lab4\data_singlevar_regr.txt'
#Завантаження даних
data = np.loadtxt(input_file, delimiter=',')
X, y = data[:, :-1], data[:, -1]
# Розбивка даних на навчальний та тестовий набори
num_training = int(0.8 * len(X))
num_test = len(X) - num_training
# Тренувальні дані
X_train, y_train = X[:num_training], y[:num_training]
# Тестові дані
X_test, y_test = X[num_training:], y[num_training:]
# Створення об'єкта лінійного регресора
regressor = linear_model.LinearRegression()
regressor.fit(X_train, y_train)
# Прогнозування результату
y_test_pred = regressor.predict(X_test)
# Побудова графіка
plt.scatter(X_test, y_test, color='green')
plt.plot(X_test, y_test_pred, color='black', linewidth=4)
plt.xticks(())
plt.yticks(())
plt.show()
print("Linear regressor performance:")
print("Mean absolute error =", round(sm.mean_absolute_error(y_test, y_test_pred), 2))
print("Mean squared error =", round(sm.mean_squared_error(y_test, y_test_pred), 2))
print("Median absolute error =", round(sm.median_absolute_error(y_test, y_test_pred), 2))
print("Explain variance score =", round(sm.explained_variance_score(y_test, y_test_pred), 2))
print("R2 score =", round(sm.r2_score(y_test, y_test_pred), 2))
# Файл для збереження моделі
output_model_file = 'model.pkl'
# Збереження моделі
with open(output_model_file, 'wb') as f:
```

```
    pickle.dump(regressor, f)
# Завантаження моделі
y_test_pred_new = regressor.predict(X_test)
print("\nNew mean absolute error =", round(sm.mean_absolute_error(y_test, y_test_pred_new), 2))
```

4.2

```
import pickle
import numpy as np
from sklearn import linear_model
import sklearn.metrics as sm
import matplotlib.pyplot as plt

# Вхідний файл, який містить дані
input_file = 'C:\Codes\python codes\AIUniver\AI_KNEU_2025\Lab4\data_regr_4.txt'
#Завантаження даних
data = np.loadtxt(input_file, delimiter=',')
X, y = data[:, :-1], data[:, -1]
# Розбивка даних на навчальний та тестовий набори
num_training = int(0.8 * len(X))
num_test = len(X) - num_training
# Тренувальні дані
X_train, y_train = X[:num_training], y[:num_training]
# Тестові дані
X_test, y_test = X[num_training:], y[num_training:]
# Створення об'єкта лінійного регресора
regressor = linear_model.LinearRegression()
regressor.fit(X_train, y_train)
# Прогнозування результату
y_test_pred = regressor.predict(X_test)
# Побудова графіка
plt.scatter(X_test, y_test, color='green')
plt.plot(X_test, y_test_pred, color='black', linewidth=4)
```

```
plt.xticks(())
plt.yticks(())
plt.show()
print("Linear regressor performance:")
print("Mean absolute error =", round(sm.mean_absolute_error(y_test, y_test_pred), 2))
print("Mean squared error =", round(sm.mean_squared_error(y_test, y_test_pred), 2))
print("Median absolute error =", round(sm.median_absolute_error(y_test, y_test_pred), 2))
print("Explain variance score =", round(sm.explained_variance_score(y_test, y_test_pred), 2))
print("R2 score =", round(sm.r2_score(y_test, y_test_pred), 2))
# Файл для збереження моделі
output_model_file = 'model2.pkl'
# Збереження моделі
with open(output_model_file, 'wb') as f:
    pickle.dump(regressor, f)
# Завантаження моделі
y_test_pred_new = regressor.predict(X_test)
print("\nNew mean absolute error =", round(sm.mean_absolute_error(y_test, y_test_pred_new), 2))
```

4.3

```
import pickle
import numpy as np
import matplotlib.pyplot as plt
from sklearn import linear_model
from sklearn.preprocessing import PolynomialFeatures

# Шлях до файлу з даними
input_file = 'C:/Codes/python codes/AIUniver/AI_KNEU_2025/Lab4/data_multivar_regr.txt'
data = np.loadtxt(input_file, delimiter=',')
# Розділення на X (всі колонки крім останньої) та y (остання колонка)
X = data[:, :-1]
y = data[:, -1]
```

```
# Розділення на тренувальні та тестові дані (80/20)
num_training = int(0.8 * len(X))
X_train, y_train = X[:num_training], y[:num_training]
X_test, y_test = X[num_training:], y[num_training:]
#Лінійна регресія
lin_regressor = linear_model.LinearRegression()
lin_regressor.fit(X_train, y_train)
y_pred_linear = lin_regressor.predict(X_test)

# Поліноміальна регресія
polynomial = PolynomialFeatures(degree=10)
X_train_transformed = polynomial.fit_transform(X_train)
datapoint = [[7.75, 6.35, 5.56]]
poly_datapoint = polynomial.fit_transform(datapoint)
poly_linear_model = linear_model.LinearRegression()
poly_linear_model.fit(X_train_transformed, y_train)
print("\nLinear regression:\n", lin_regressor.predict(datapoint))
print("\nPolynomial regression:\n", poly_linear_model.predict(poly_datapoint))
```

4.4

```
import matplotlib.pyplot as plt
import numpy as np
from sklearn import datasets, linear_model
from sklearn.metrics import mean_squared_error, r2_score, mean_absolute_error
from sklearn.model_selection import train_test_split
diabetes = datasets.load_diabetes()
X = diabetes.data
y = diabetes.target
Xtrain, Xtest, ytrain, ytest = train_test_split(X, y, test_size = 0.5, random_state = 0)
regr = linear_model.LinearRegression()
regr.fit(Xtrain, ytrain)
ypred = regr.predict(Xtest)
```

```
print("Regr Coef =", regr.coef_)
print("Regr Intercept =", regr.intercept_)
print("Mean absolute error =", round(mean_absolute_error(ytest, ypred), 2))
print("Mean squared error =", round(mean_squared_error(ytest, ypred), 2))
print("R2 score =", round(r2_score(ytest, ypred), 2))
fig, ax = plt.subplots()
ax.scatter(ytest, ypred, edgecolors = (0, 0, 0))
ax.plot([y.min(), y.max()], [y.min(), y.max()], 'k--', lw = 4)
ax.set_xlabel('Виміряно')
ax.set_ylabel('Передбачено')
plt.show()
```

4.5

```
import pickle
import numpy as np
import matplotlib.pyplot as plt
from sklearn import linear_model
from sklearn.preprocessing import PolynomialFeatures
# Генерація даних
m = 100
X = 6 * np.random.rand(m, 1) - 5
y = 0.7 * X ** 2 + X + 3 + np.random.randn(m, 1)
# Розділення на тренувальні та тестові дані
num_training = int(0.8 * len(X))
X_train, y_train = X[:num_training], y[:num_training]
X_test, y_test = X[num_training:], y[num_training:]
# Лінійна регресія
lin_regressor = linear_model.LinearRegression()
lin_regressor.fit(X_train, y_train)
y_pred_linear = lin_regressor.predict(X_test)
# Поліноміальна регресія
poly_features = PolynomialFeatures(degree=2, include_bias=False)
```

```
X_train_poly = poly_features.fit_transform(X_train)
X_test_poly = poly_features.transform(X_test)
poly_regressor = linear_model.LinearRegression()
poly_regressor.fit(X_train_poly, y_train)
y_pred_poly = poly_regressor.predict(X_test_poly)
# Сортування для гладкої кривої
X_range = np.linspace(X.min(), X.max(), 200).reshape(-1, 1)
X_range_poly = poly_features.transform(X_range)
y_range_poly = poly_regressor.predict(X_range_poly)
# Побудова графіка
plt.scatter(X_test, y_test, color='green', label='Тестові дані')
plt.plot(X_test, y_pred_linear, color='black', linewidth=2, label='Лінійна регресія')
plt.plot(X_range, y_range_poly, color='blue', linewidth=2, label='Поліноміальна регресія')
plt.legend()
plt.xlabel("X")
plt.ylabel("y")
plt.title("Лінійна та Поліноміальна регресії")
plt.show()
```

4.6

```
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
import numpy as np
from sklearn import linear_model
from sklearn.preprocessing import PolynomialFeatures
from sklearn.pipeline import Pipeline
m = 100
X = 6 * np.random.rand(m, 1) - 5
y = 0.7 * X ** 2 + X + 3 + np.random.randn(m, 1)
def plot_learning_curves (model, X, y):
    X_train, X_val, y_train, y_val = train_test_split (X, y, test_size=0.2)
```

```
    train_errors, val_errors = [], []
    for m in range (1, len (X_train)):
        model.fit (X_train [:m], y_train [:m])
        y_train_predict = model.predict (X_train[:m])
        y_val_predict = model.predict (X_val)
        train_errors.append (mean_squared_error (y_train_predict, y_train [:m]))
        val_errors.append (mean_squared_error (y_val_predict, y_val))
    plt.plot(np.sqrt(train_errors), "r-+", linewidth=2, label="train")
    plt.plot(np.sqrt(val_errors), "b-", linewidth=3, label="val")
    plt.legend()
    plt.xlabel("X")
    plt.ylabel("y")
    plt.title("Криві навчання")
    plt.show()
lin_reg = linear_model.LinearRegression()
plot_learning_curves (lin_reg, X, y)


polynomial_regression = Pipeline ([
    ("poly_features", PolynomialFeatures (degree=10, include_bias=False)), ("lin_reg",
linear_model.LinearRegression())])
plot_learning_curves (polynomial_regression, X, y)
```