**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.

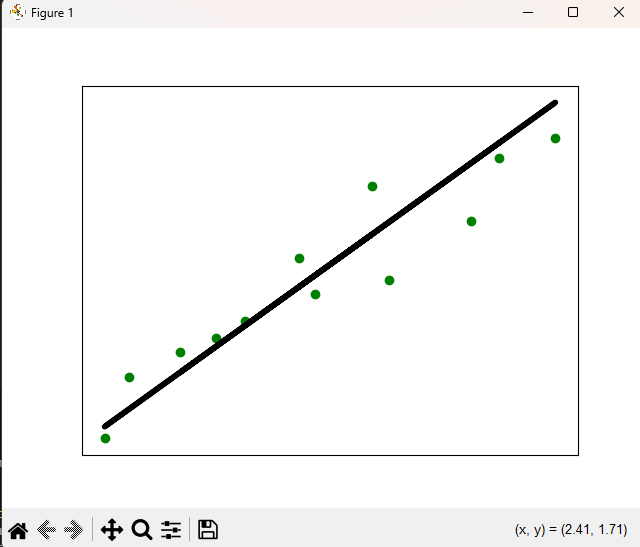


Рисунок 1.1 – Графік лінійної регресії

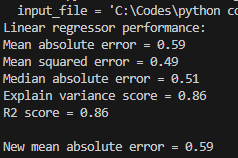


Рисунок 1.2 – Результат роботи коду

**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

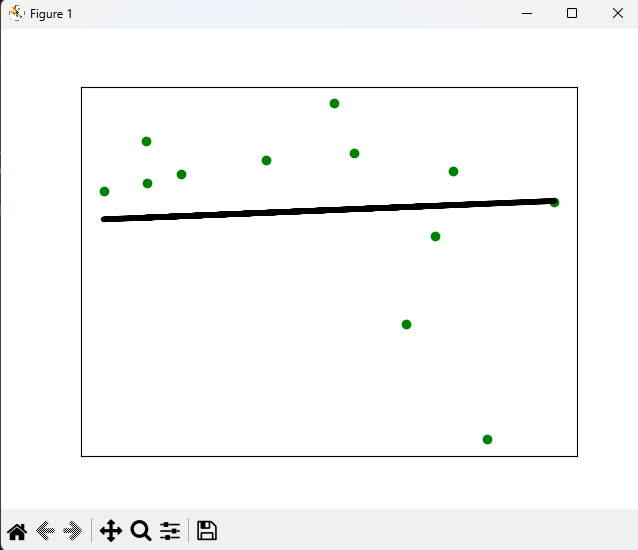


Рисунок 2.1 – Графік лінійної регресії

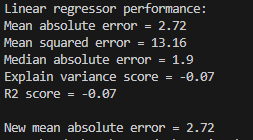


Рисунок 2.2 – Результат роботи коду

**Завдання 2.3.** Створення багатовимірного регресора. Використовувати файл вхідних даних: data\_multivar\_regr.txt, побудувати регресійну модель на основі багатьох змінних

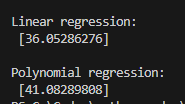


Рисунок 3.1 – Результат роботи коду

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

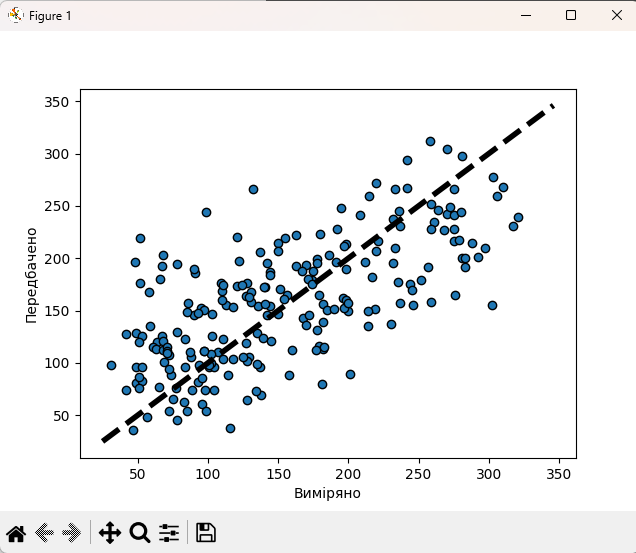


Рисунок 4.1 – Графік лінійної регресії багатьох змінних

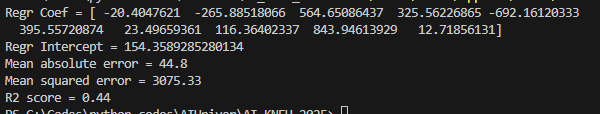


Рисунок 4.2 – Результат роботи коду

**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.

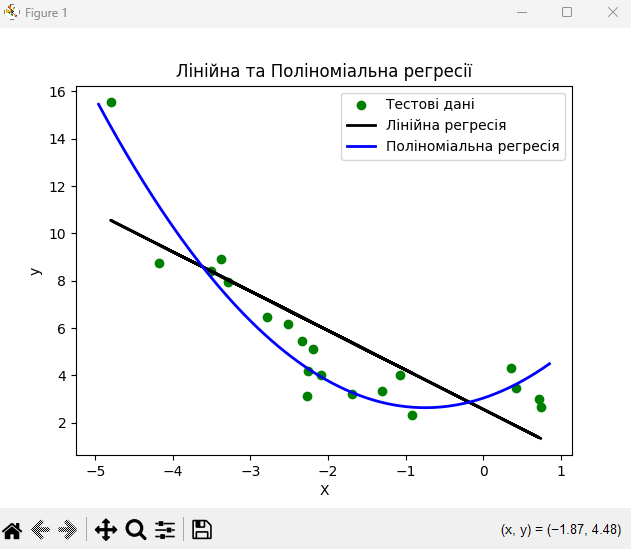


Рисунок 5.1 – Графік лінійної та поліноміальної регресії

Модель 4 варіанта у вигляді математичного рівняння:

y^​=0.5​x2​+1​x +3 + шум гаусса

Отримана модель регресії з передбаченими коефіцієнтами:

y^​=0.72x2+1.05x +3.14

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

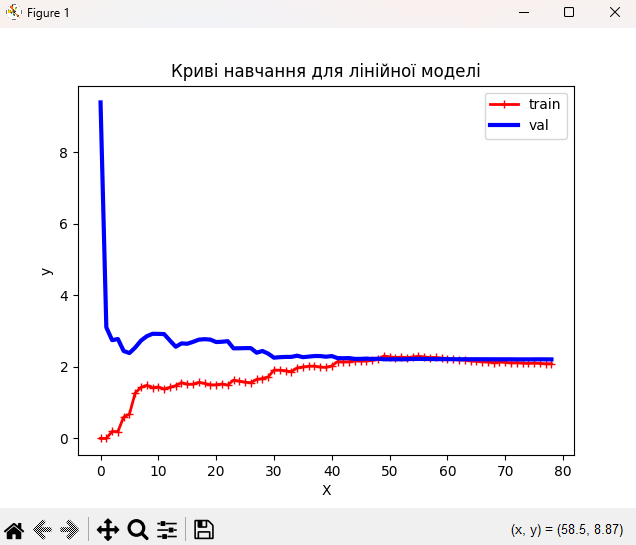


Рисунок 6.1 – Графік кривих навчань лінійного регресора

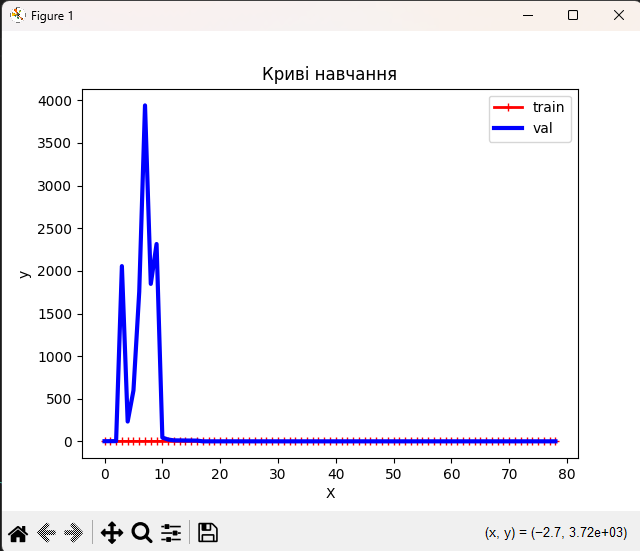


Рисунок 6.1 – Графік кривих навчань поліноміального регресора

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)