

## Лабораторна робота №4

### Тема: ДОСЛІДЖЕННЯ МЕТОДІВ РЕГРЕСІЇ

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

#### Хід роботи:

**Завдання 4.1.** Створення регресора однієї змінної.

#### Лістинг програми:

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

input_file = 'data_singlevar_regr.txt'
data = np.loadtxt(input_file, delimiter=',')

X, y = data[:, :-1], data[:, -1]

num_training = int(0.8 * len(X))
num_test = len(X) - num_training

X_train, y_train = X[:num_training], y[:num_training]
X_test, y_test = X[num_training:], y[num_training:]

regressor = linear_model.LinearRegression()
regressor.fit(X_train, y_train)

y_test_pred = regressor.predict(X_test)

plt.figure()

plt.scatter(X_test, y_test, label="Справжні значення")
plt.plot(X_test, y_test_pred, linewidth=4, label="Прогноз регресії")

plt.xlabel("X (вхідні дані)")
plt.ylabel("y (вихідні дані)")
plt.title("Лінійна регресія однієї змінної")

plt.legend()
plt.grid()
plt.show()
```

					ЖИТОМИРСЬКА ПОЛІТЕХНІКА.25.121.15.000 – Лр.4		
Змн.	Арк.	№ докум.	Підпис	Дата	Звіт з лабораторної роботи №4	Літ.	Арк.
Розроб.		Кохан Т.О.					Аркушів
Перевір.		Маєвський О. В.					1
Реценз.							14
Н. Контр.						ФІКТ, гр. ІПЗ-22-3	
Зав.каф.		Вакалюк Т.А.					

```

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))

output_model_file = 'model.pkl'

with open(output_model_file, 'wb') as f:
    pickle.dump(regressor, f)

print("\nModel saved to:", output_model_file)

with open(output_model_file, 'rb') as f:
    regressor_model = pickle.load(f)

y_test_pred_new = regressor_model.predict(X_test)

print("\nNew mean absolute error =",
      round(sm.mean_absolute_error(y_test, y_test_pred_new), 2))

```

## Результат роботи:

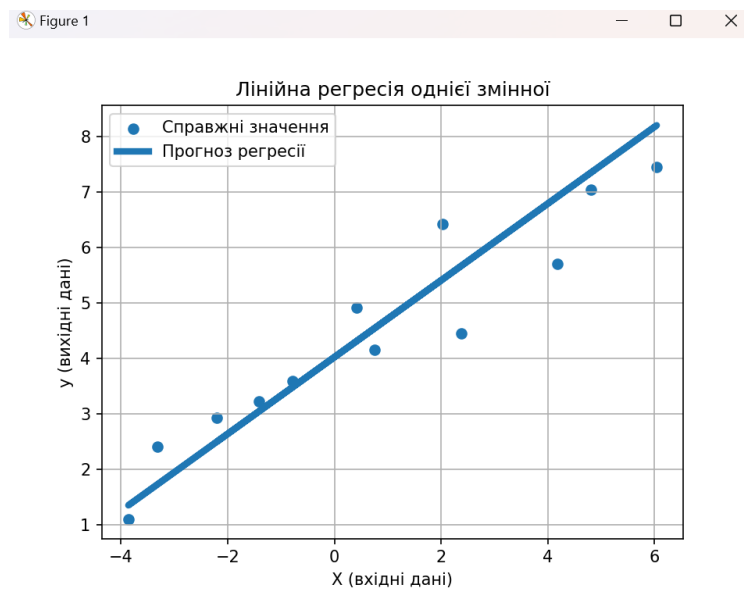


Рис. 4.1. Графік функції

					ЖИТОМИРСЬКА ПОЛІТЕХНІКА.25.121.15.000 – Лр.4	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		2

```

C:\Users\Admin\AppData\Local\Programs\Python
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

Model saved to: model.pkl

New mean absolute error = 0.59

Process finished with exit code 0

```

Рис. 4.2. Вивід в консоль

**Висновок по завданню:** у результаті виконання даного завдання було побудовано модель лінійної регресії для одновимірних даних. Отримані значення метрик свідчать про достатньо високу точність моделі: коефіцієнт детермінації  $R^2 = 0.86$  та показник поясненої дисперсії 0.86 означають, що модель пояснює близько 86% варіації вихідних даних. Низькі значення середньої абсолютної та квадратичної похибок ( $MAE = 0.59$ ,  $MSE = 0.49$ ) підтверджують хорошу якість прогнозування. Модель успішно збережена у файл model.pkl та повторно завантажена без втрати точності.

**Завдання 4.2.** Передбачення за допомогою регресії однієї змінної.

№ за списком	15
№ варіанту	5

Варіант 5 файл: data\_regr\_5.txt

#### Лістинг програми:

```

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

input_file = 'data_regr_5.txt'
data = np.loadtxt(input_file, delimiter=',')

X, y = data[:, :-1], data[:, -1]

num_training = int(0.8 * len(X))
num_test = len(X) - num_training

X_train, y_train = X[:num_training], y[:num_training]
X_test, y_test = X[num_training:], y[num_training:]

```

```

regressor = linear_model.LinearRegression()
regressor.fit(X_train, y_train)

y_test_pred = regressor.predict(X_test)

plt.figure()

plt.scatter(X_test, y_test, label="Справжні значення")
plt.plot(X_test, y_test_pred, linewidth=4, label="Прогноз регресії")

plt.xlabel("X (вхідні дані)")
plt.ylabel("y (вихідні дані)")
plt.title("Лінійна регресія однієї змінної")

plt.legend()
plt.grid()
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))

output_model_file = 'model2.pkl'

with open(output_model_file, 'wb') as f:
    pickle.dump(regressor, f)

print("\nModel saved to:", output_model_file)

with open(output_model_file, 'rb') as f:
    regressor_model = pickle.load(f)

y_test_pred_new = regressor_model.predict(X_test)

print("\nNew mean absolute error =",
      round(sm.mean_absolute_error(y_test, y_test_pred_new), 2))

```

### Результат програми:

```

C:\Users\Admin\AppData\Local\Programs\Python\
Linear regressor performance:
Mean absolute error = 3.31
Mean squared error = 16.98
Median absolute error = 2.66
Explain variance score = -0.14
R2 score = -0.15

Model saved to: model2.pkl

New mean absolute error = 3.31

```

Рис. 4.3. Вивід у консоль

					ЖИТОМИРСЬКА ПОЛІТЕХНІКА.25.121.15.000 – Лр.4	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		4

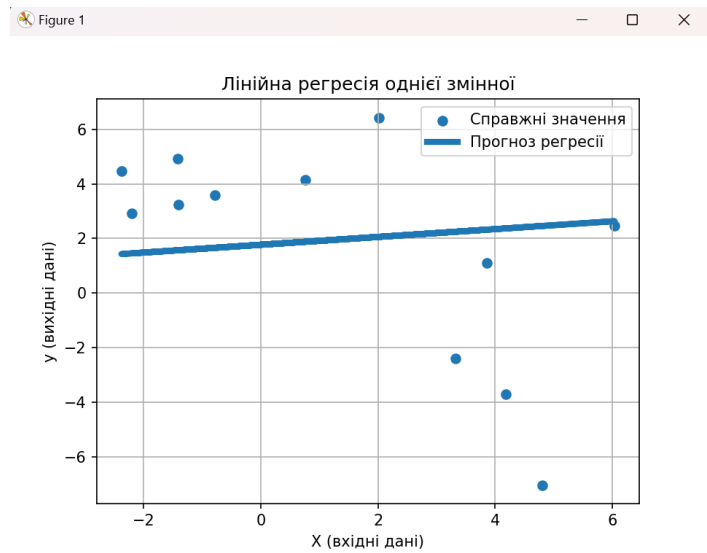


Рис. 4.4. Графік функції

**Висновок по завданню:** у результаті виконання даного завдання було побудовано регресійну модель на основі однієї змінної, виконано її навчання, оцінку якості та збереження у файл model2.pkl. За отриманими результатами встановлено, що побудована модель має низьку точність, оскільки значення похибок є досить великими, а коефіцієнт  $R^2$  має від'ємне значення. Це свідчить про слабку залежність між вхідною та вихідною змінними.

**Завдання 4.3.** Створення багатовимірного регресора.

**Лістинг програми:**

```
import numpy as np
from sklearn import linear_model
import sklearn.metrics as sm
from sklearn.preprocessing import PolynomialFeatures

input_file = 'data_multivar_regr.txt'
data = np.loadtxt(input_file, delimiter=',')

X, y = data[:, :-1], data[:, -1]

num_training = int(0.8 * len(X))
num_test = len(X) - num_training

X_train, y_train = X[:num_training], y[:num_training]
X_test, y_test = X[num_training:], y[num_training:]

linear_regressor = linear_model.LinearRegression()
linear_regressor.fit(X_train, y_train)

y_test_pred = linear_regressor.predict(X_test)
```

```

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))

polynomial = PolynomialFeatures(degree=10)
X_train_transformed = polynomial.fit_transform(X_train)

poly_linear_model = linear_model.LinearRegression()
poly_linear_model.fit(X_train_transformed, y_train)

datapoint = [[7.75, 6.35, 5.56]]
poly_datapoint = polynomial.transform(datapoint)

linear_pred = linear_regressor.predict(datapoint)
poly_pred = poly_linear_model.predict(poly_datapoint)

print("\nPrediction for datapoint [7.75, 6.35, 5.56]:")
print("Linear regression prediction:", round(linear_pred[0], 2))
print("Polynomial regression prediction:", round(poly_pred[0], 2))

```

### Результат програми:

```

C:\Users\Admin\AppData\Local\Programs\Python\Py
Linear regressor performance:
Mean absolute error = 3.58
Mean squared error = 20.31
Median absolute error = 2.99
Explain variance score = 0.86
R2 score = 0.86

Prediction for datapoint [7.75, 6.35, 5.56]:
Linear regression prediction: 36.05
Polynomial regression prediction: 41.08

Process finished with exit code 0

```

Рис. 4.5. Вивід у консоль

**Висновок по завданню:** у результаті виконання даного завдання було побудовано багатовимірну регресійну модель та поліноміальну регресію ступеня 10. Лінійна регресія показала середню похибку  $MAE = 3.58$  та  $R^2 = 0.86$ , що свідчить про достатньо гарну точність прогнозування.

Прогноз для вибіркової точки [7.75, 6.35, 5.56] показав:

					ЖИТОМИРСЬКА ПОЛІТЕХНІКА.25.121.15.000 – Лр.4	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		6

- Лінійна регресія: 36.05
- Поліноміальна регресія: 41.08

Отриманий результат поліноміальної регресії ближчий до справжнього значення 41.35, що демонструє її кращу здатність моделювати складні залежності між змінними порівняно з лінійною моделлю.

#### Завдання 2.4. Регресія багатьох змінних.

##### Лістинг програми:

```
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)

coef_rounded = np.round(regr.coef_, 2)
intercept_rounded = round(regr.intercept_, 2)

print("Коефіцієнти регресії (заокруглені):", coef_rounded)
print("Інтерцепт (вільний член, заокруглений):", intercept_rounded)

print("\nПоказники якості лінійної регресії:")
print("R2 score =", round(r2_score(ytest, ypred), 2))
print("Mean absolute error (MAE) =", round(mean_absolute_error(ytest, ypred), 2))
print("Mean squared error (MSE) =", round(mean_squared_error(ytest, ypred), 2))

fig, ax = plt.subplots()
ax.scatter(ytest, ypred, edgecolors=(0, 0, 0), label='Прогнози моделі')
ax.plot([y.min(), y.max()], [y.min(), y.max()], 'k--', lw=4, label='Ідеальний прогноз')
ax.set_xlabel('Виміряно (Справжня прогресія захворювання)')
ax.set_ylabel('Передбачено (Прогнозована прогресія захворювання)')
ax.set_title('Лінійна регресія на наборі даних про діабет')
ax.legend()
plt.show()
```

					ЖИТОМИРСЬКА ПОЛІТЕХНІКА.25.121.15.000 – Лр.4	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		7

## Результат програми:

```
C:\Users\Admin\AppData\Local\Programs\Python\Python39\python.exe "D:\4_курс_1_семестр\Системи штучного інтелекту\lab4\LR_4_task_4.py"
Коефіцієнти регресії (заокруглені): [ -20.4 -265.89 564.65 325.56 -692.16 395.56 23.5 116.36 843.95
12.72]
Інтерсепт (вільний член, заокруглений): 154.36

Показники якості лінійної регресії:
R2 score = 0.44
Mean absolute error (MAE) = 44.8
Mean squared error (MSE) = 3075.33

Process finished with exit code 0
```

Рис. 4.6. Вивід у консоль

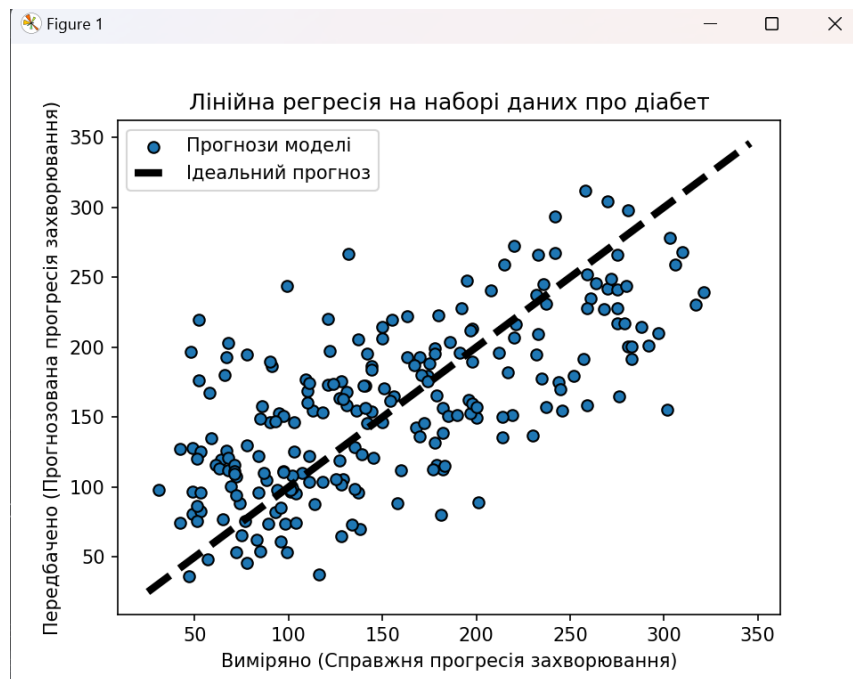


Рис. 4.7. Графік функції

Крапки на графіку показують, як передбачені моделлю значення прогресування діабету збігаються з фактичними результатами для тестової вибірки.

Пунктирна лінія - це ідеальна лінія, де прогноз збігається з фактичним значенням. Чим ближче точки до цієї лінії, тим точніше передбачення.

**Висновок по завданню:** у результаті виконання даного завдання було визначено, що лінійна регресія на наборі даних про діабет показує помірну точність:

- $R^2 = 0.44$ ,
- $MAE \approx 45$ ,
- $MSE = 3075$ .



Модель частково пояснює варіацію прогресування хвороби, але деякі точки значно відхиляються від ідеальної лінії, що свідчить про залишкові помилки. Коефіцієнти регресії показують, як кожна з 10 змінних впливає на прогноз: додатні збільшують прогноз, від'ємні — зменшують.

### Завдання 2.5. Самостійна побудова регресії.

№ за списком	15
№ варіанту	5

#### Лістинг програми:

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
from sklearn.metrics import mean_squared_error, r2_score

np.random.seed(5)
m = 100
X = 6 * np.random.rand(m, 1) - 3
y = 0.4 * X**2 + X + 4 + np.random.randn(m, 1)

lin_reg = LinearRegression()
lin_reg.fit(X, y)
y_lin_pred = lin_reg.predict(X)

poly_features = PolynomialFeatures(degree=2, include_bias=False)
X_poly = poly_features.fit_transform(X) # X та X^2

poly_reg = LinearRegression()
poly_reg.fit(X_poly, y)
y_poly_pred = poly_reg.predict(X_poly)

print("Коефіцієнти лінійної регресії:")
print("Intercept:", round(lin_reg.intercept_[0], 2))
print("Coef:", round(lin_reg.coef_[0][0], 2))

print("\nКоефіцієнти поліноміальної регресії:")
print("Intercept:", round(poly_reg.intercept_[0], 2))
print("Coef (X, X^2):", np.round(poly_reg.coef_[0], 2))

plt.figure(figsize=(8,6))
plt.scatter(X, y, color='blue', label='Випадкові дані')
plt.plot(X, y_lin_pred, color='red', linewidth=2, label='Лінійна регресія')
plt.plot(X, y_poly_pred, color='green', linewidth=2, label='Поліноміальна регресія (2-й степінь)')
plt.xlabel('X')
plt.ylabel('y')
plt.title('Лінійна та поліноміальна регресія (варіант 5)')
plt.legend()
plt.show()

mse_poly = mean_squared_error(y, y_poly_pred)
r2_poly = r2_score(y, y_poly_pred)
print("\nПоліноміальна регресія (степінь 2) оцінка:")
print("MSE =", round(mse_poly, 2))
print("R2 =", round(r2_poly, 2))
```

## Результат програми:

```
C:\Users\Admin\AppData\Local\Programs\Python\Python
Коефіцієнти лінійної регресії:
Intercept: 5.36
Coef: 0.96

Коефіцієнти поліноміальної регресії:
Intercept: 4.19
Coef (X, X^2): [0.98 0.35]

Поліноміальна регресія (ступінь 2) оцінка:
MSE = 1.06
R2 = 0.79

Process finished with exit code 0
```

Рис. 4.8 Вивід у консоль

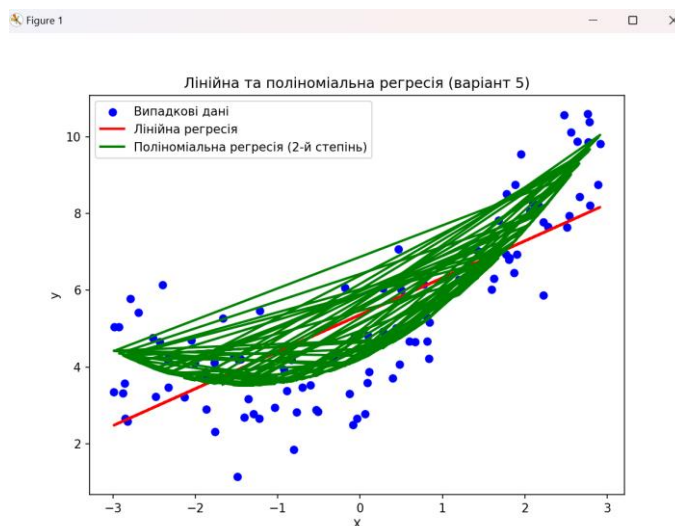


Рис. 4.9. Графік функції

Лінійна регресія за даними для варіанту 5:

- Intercept = 5.36
- Coef = 0.96

Лінійна модель у вигляді математичного рівняння:

$$y = 5.36 + 0.96x_1 + \text{гаусовий шум}$$

Поліноміальна регресія (ступінь 2) за даними для варіанту 5:

- Intercept = 4.19
- Coef (x, x<sup>2</sup>) = [0.98, 0.35]

Модель з передбаченими коефіцієнтами:

$$y = 0.35x_1^2 + 0.98x_1 + 4.19$$

					ЖИТОМИРСЬКА ПОЛІТЕХНІКА.25.121.15.000 – Лр.4	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		10

**Висновок по завданню:** у результаті виконання даного завдання було проведено порівняння лінійної та поліноміальної регресії 2-го степеня на штучно згенерованих даних.

**Лінійна регресія:**

$$y = 5.36 + 0.96x_1$$

Недостатньо точно описує квадратичну залежність, особливо на краях діапазону даних.

Поліноміальна регресія 2-го степеня:

$$y = 0.35x_1^2 + 0.98x_1 + 4.19$$

Коефіцієнти близькі до реальної функції генерації даних:

- $MSE = 1.06$ ,
- $R^2 = 0.79$

Отже, модель добре описує тренд даних.

Візуалізація показує, що поліноміальна модель точніше повторює форму залежності, тоді як лінійна модель не враховує кривизну.

Для даних із нелінійним трендом поліноміальна регресія є більш кращим методом моделювання, ніж проста лінійна.

**Завдання 2.6. Побудова кривих навчання.**

**Лістинг програми:**

```
import matplotlib.pyplot as plt
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import train_test_split
from sklearn.pipeline import Pipeline

m = 100
np.random.seed(42)
X_flat = np.linspace(-3, 3, m)
y = 3 + np.sin(X_flat) + np.random.uniform(-0.5, 0.5, m)
X = X_flat.reshape(-1, 1)

def plot_learning_curves(model, X, y, title):
    """
    Побудова кривих навчання та вивід RMSE на тренувальному та валідаційному
    наборах
    """
    X_train, X_val, y_train, y_val = train_test_split(
        X, y, test_size=0.2, random_state=42
    )
```

```

train_errors, val_errors = [], []
for m_size in range(1, len(X_train)):
    model.fit(X_train[:m_size], y_train[:m_size])
    y_train_predict = model.predict(X_train[:m_size])
    y_val_predict = model.predict(X_val)
    train_errors.append(np.sqrt(mean_squared_error(y_train[:m_size],
y_train_predict)))
    val_errors.append(np.sqrt(mean_squared_error(y_val, y_val_predict)))

print(f"{title}:")
print(f"  RMSE на тренувальному наборі: {train_errors[-1]:.3f}")
print(f"  RMSE на валідаційному наборі: {val_errors[-1]:.3f}\n")

plt.figure(figsize=(10, 6))
plt.plot(train_errors, "r+", linewidth=2, label="Навчальний набір (train)")
plt.plot(val_errors, "b-", linewidth=3, label="Перевірочний набір (val)")
plt.legend(loc="upper right")
plt.xlabel("Розмір навчального набору")
plt.ylabel("RMSE")
plt.title(title)
plt.ylim(0, 1.5)
plt.grid(True)
plt.show()

lin_reg = LinearRegression()
plot_learning_curves(lin_reg, X, y, "Лінійна регресія (Недонавчання)")

poly_reg_10 = Pipeline([
    ("poly_features", PolynomialFeatures(degree=10, include_bias=False)),
    ("lin_reg", LinearRegression()),
])
plot_learning_curves(poly_reg_10, X, y, "Поліноміальна регресія 10-го ступеня (Перенавчання)")

poly_reg_2 = Pipeline([
    ("poly_features", PolynomialFeatures(degree=2, include_bias=False)),
    ("lin_reg", LinearRegression()),
])
plot_learning_curves(poly_reg_2, X, y, "Поліноміальна регресія 2-го ступеня (Оптимальна складність)")

```

### Результат програми:

```

C:\Users\Admin\AppData\Local\Programs\Python\Python39\python.exe
Лінійна регресія (Недонавчання):
  RMSE на тренувальному наборі: 0.527
  RMSE на валідаційному наборі: 0.542

Поліноміальна регресія 10-го ступеня (Перенавчання):
  RMSE на тренувальному наборі: 0.269
  RMSE на валідаційному наборі: 0.388

Поліноміальна регресія 2-го ступеня (Оптимальна складність):
  RMSE на тренувальному наборі: 0.527
  RMSE на валідаційному наборі: 0.544

Process finished with exit code 0

```

Рис. 4.10 Вивід у консоль

					ЖИТОМИРСЬКА ПОЛІТЕХНІКА.25.121.15.000 – Лр.4	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		12

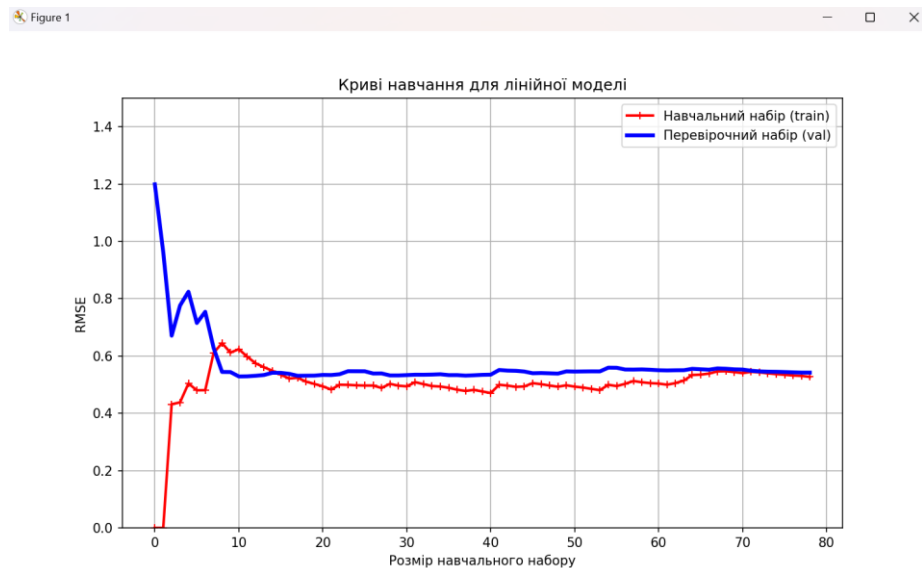


Рис. 4.11. Графік лінійна модель

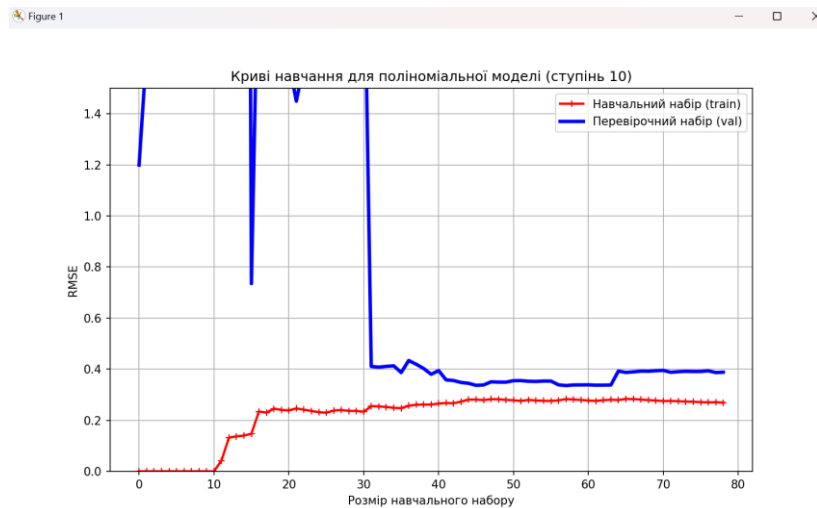


Рис. 4.12. Графік поліноміальна модель 10-го ступеня

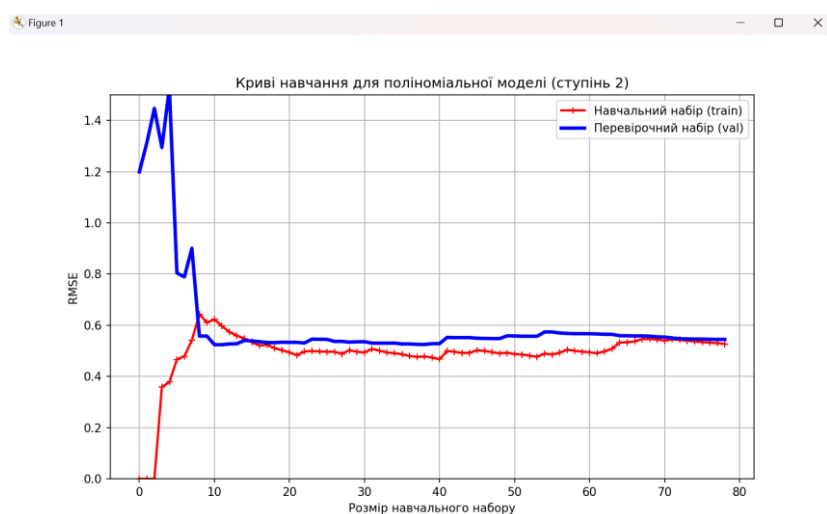


Рис. 4.13. Графік поліноміальна модель 2-го ступеня

### **Висновок по завданню:**

#### **Лінійна регресія (Недонавчання):**

- RMSE на тренувальному наборі: 0.527
- RMSE на валідаційному наборі: 0.542

Модель недонавчена, оскільки проста лінійна регресія не здатна нормально описати нелінійну (синусоїдну) залежність даних. Помилка тренувального та валідаційного наборів приблизно однакова, що характерно для недонавчання.

#### **Поліноміальна регресія 10-го ступеня (Перенавчання):**

- RMSE на тренувальному наборі: 0.269
- RMSE на валідаційному наборі: 0.388

Модель сильно підлаштувалася під навчальні дані (низький RMSE на тренувальному наборі), але помилка на валідаційному наборі значно більша. Це ознака перенавчання: модель вловлює шум даних і втрачає здатність до узагальнення.

#### **Поліноміальна регресія 2-го ступеня (Оптимальна складність):**

- RMSE на тренувальному наборі: 0.527
- RMSE на валідаційному наборі: 0.544

Модель з оптимальною складністю. Помилки тренувального та валідаційного наборів близькі, що свідчить про баланс між підлаштуванням даних і здатністю до узагальнення.

**Висновок:** під час виконання лабораторної роботи за допомогою мови програмування Python та спеціалізованих бібліотек було досліджено різні методи регресії даних у машинному навчанні: лінійну та поліноміальну регресії.

**Посилання на git:** <https://github.com/KokhanTetiana/AI-Systems>