

Лабораторна робота №4

Тема: дослідження методів регресії

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

Варіант 2

Хід роботи:

Посилання на GitHub :

https://github.com/Dubnitskyi/AI_all_labs/tree/master/Lab4

Завдання №1: Створення регресора однієї змінної

Код програми:

```
import pickle
import numpy as np
from sklearn import linear_model
import sklearn.metrics as sm
import matplotlib
import matplotlib.pyplot as plt
matplotlib.use('TkAgg')

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.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))

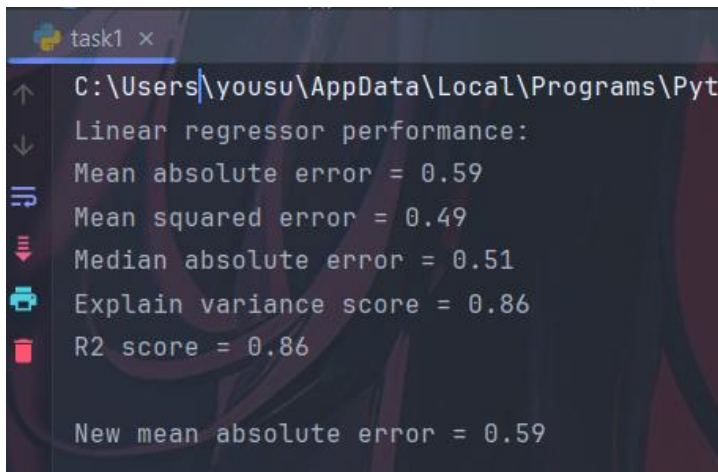
output_model_file = 'model.pkl'

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

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))

```

Результат :



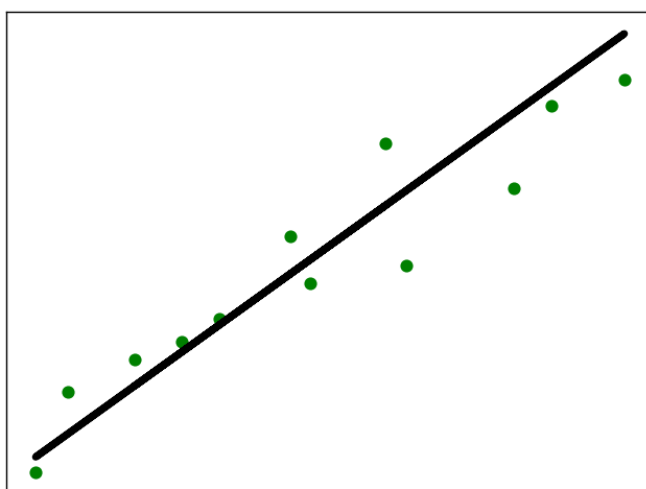
```

task1 x
C:\Users\yousu\AppData\Local\Programs\Pyt
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

```

Figure 1



Зробіть висновок

Ця регресивна модель має високу точність

Завдання №2: Передбачення за допомогою регресії однієї змінної

Код програми:

```
import pickle
import numpy as np
from sklearn import linear_model
import sklearn.metrics as sm
import matplotlib
import matplotlib.pyplot as plt
matplotlib.use('TkAgg')

input_file = 'data_regr_2.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.scatter(X_test, y_test, color='green')
plt.plot(X_test, y_test_pred, color='black', linewidth=4)
plt.xticks(())
plt.yticks(())
plt.show()

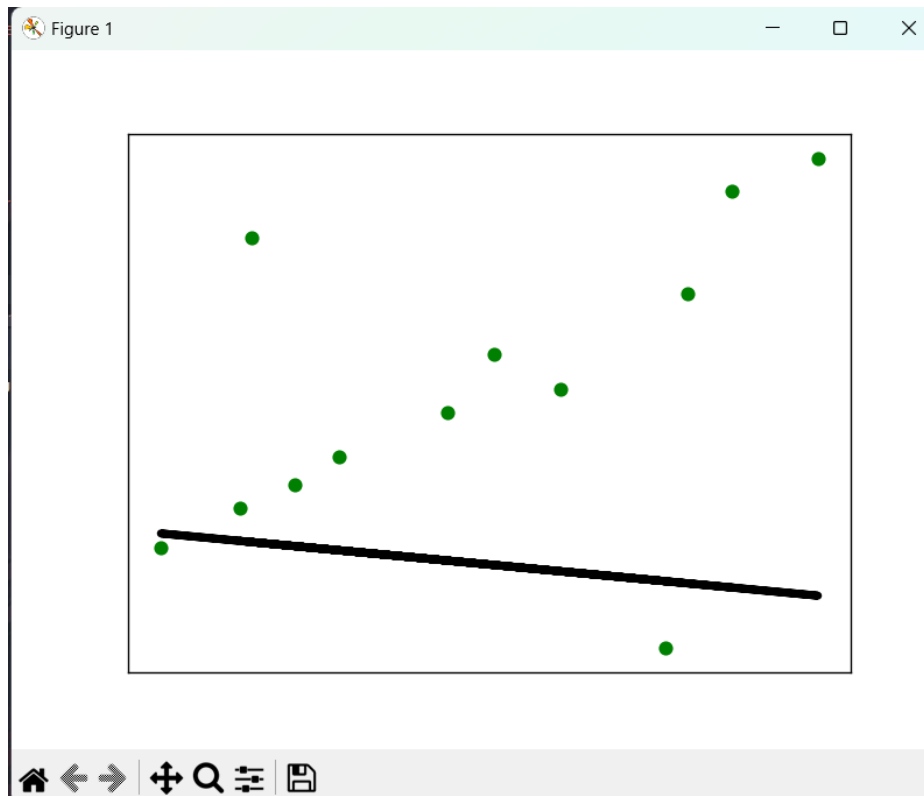
print("Linear regressor performance:")
print("Mean absolute error =", round(sm.mean_absolute_error(y_test, y_test_pred),
```

```

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))

```

Результат :



```

task2 x
C:\Users\yousu\AppData\Local\Programs\Python\Python
Linear regressor performance:
Mean absolute error = 2.42
Mean squared error = 9.02
Median absolute error = 2.14
Explain variance score = -0.15
R2 score = -1.61

```

Зробіть висновок

Ця модель має низьку точність.

Завдання 3 Створення багатовимірного регресора

Код 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]

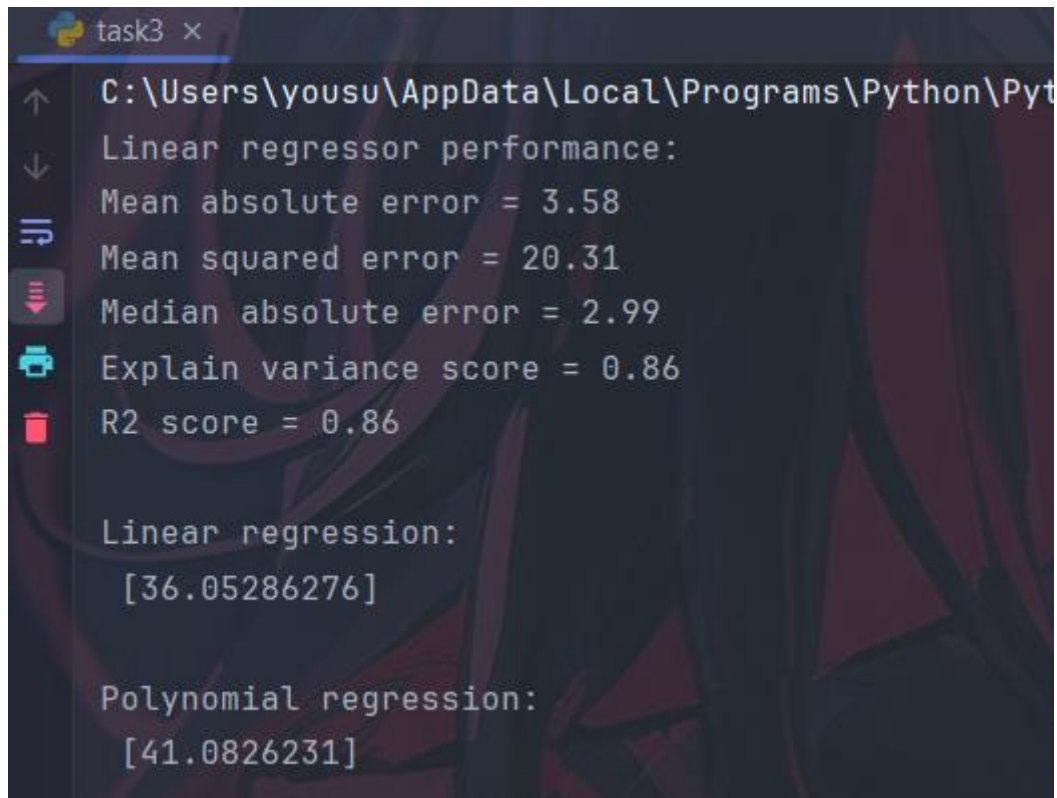
X_test, y_test = X[num_training:], y[num_training:]

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

y_test_pred = 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)
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", regressor.predict(datapoint))
print("\nPolynomial regression:\n", poly_linear_model.predict(poly_datapoint))
```

Результат :



```
task3 x
C:\Users\yousu\AppData\Local\Programs\Python\Pyt
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

Linear regression:
[36.05286276]

Polynomial regression:
[41.0826231]
```

Зробіть висновок

Пліноміальний регресор краще справляється за лінійний регресор

Завдання 4 Регресія багатьох змінних

Код програми:

```
import matplotlib
import matplotlib.pyplot as plt
matplotlib.use('TkAgg')

import numpy as np
import sklearn.metrics as sm
from sklearn import datasets, linear_model
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.metrics import mean_absolute_error
from sklearn.model_selection import train_test_split

diabetes = datasets.load_diabetes()
X = diabetes.data
y = diabetes.target
```

```

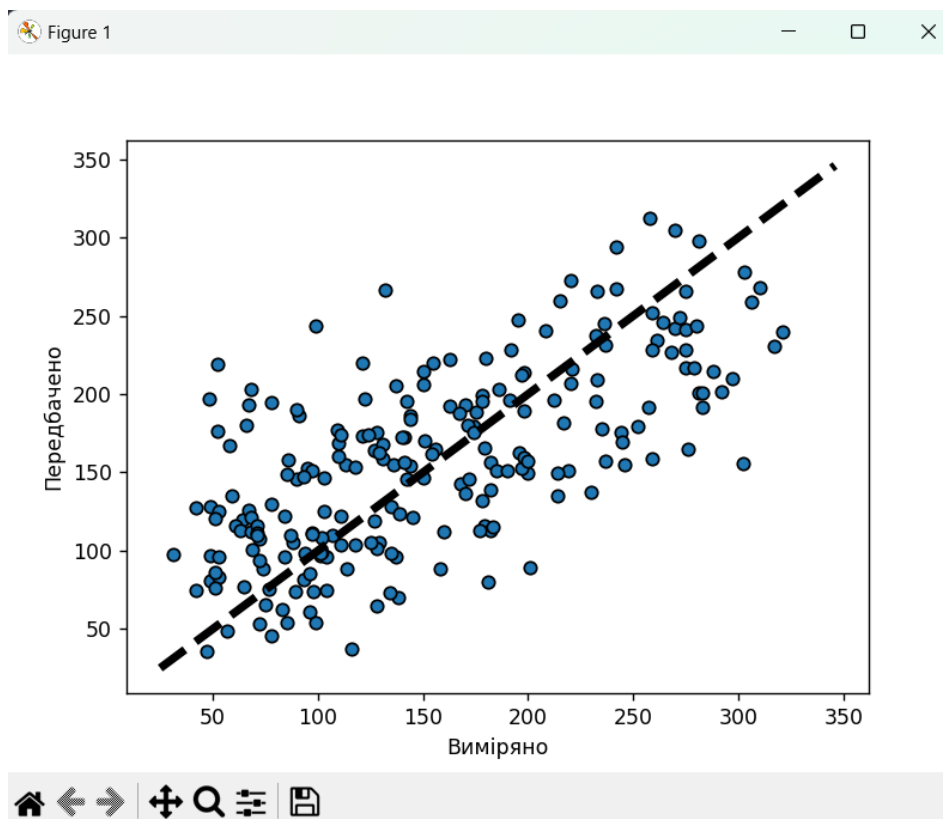
Xtrain, Xtest, ytrain, ytest = train_test_split(X, y, test_size=0.5, random_state=0)
regr = linear_model.LinearRegression()
regr.fit(Xtrain, ytrain)
ypred = regr.predict(Xtest)
fig, ax = plt.subplots()

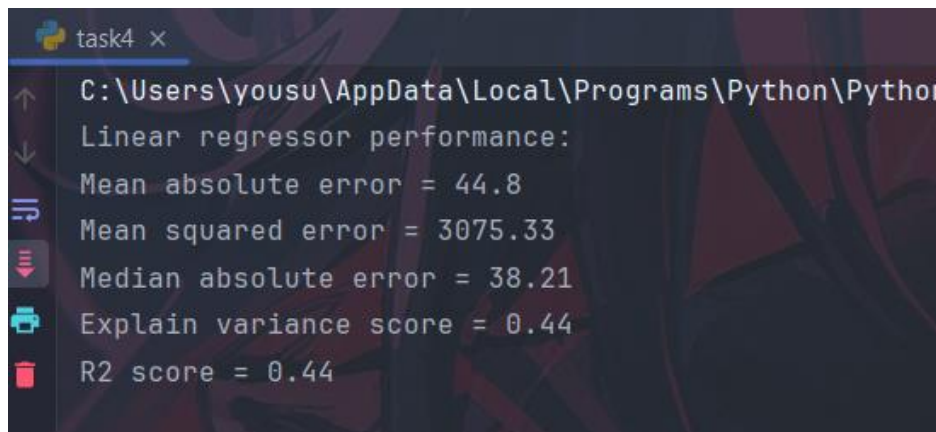
ax.scatter(ytest, ypred, edgecolors=(0, 0, 0))
ax.plot([y.min(), y.max()], [y.min(), y.max()], 'k--', lw=4)
ax.set_xlabel('Виміряно')
ax.set_ylabel('Передбачено')
plt.show()

print("Linear regressor performance:")
print("Mean absolute error =", round(sm.mean_absolute_error(ytest, ypred), 2))
print("Mean squared error =", round(sm.mean_squared_error(ytest, ypred), 2))
print("Median absolute error =", round(sm.median_absolute_error(ytest, ypred), 2))
print("Explain variance score =", round(sm.explained_variance_score(ytest, ypred), 2))
print("R2 score =", round(sm.r2_score(ytest, ypred), 2))

```

Результат :





```
task4 x
C:\Users\yousu\AppData\Local\Programs\Python\Python
Linear regressor performance:
Mean absolute error = 44.8
Mean squared error = 3075.33
Median absolute error = 38.21
Explain variance score = 0.44
R2 score = 0.44
```

Зробіть висновок

Дані занадто сильно розкидані, тому моделі складно точно обчислити велику кількість даних.

Завдання 5 Самостійна побудова регресії

Код програми:

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

import matplotlib
import matplotlib.pyplot as plt
matplotlib.use('TkAgg') \

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

poly = PolynomialFeatures(degree=2, include_bias=False)
X_poly = poly.fit_transform(X)

lin_reg = linear_model.LinearRegression()
lin_reg.fit(X_poly, y)

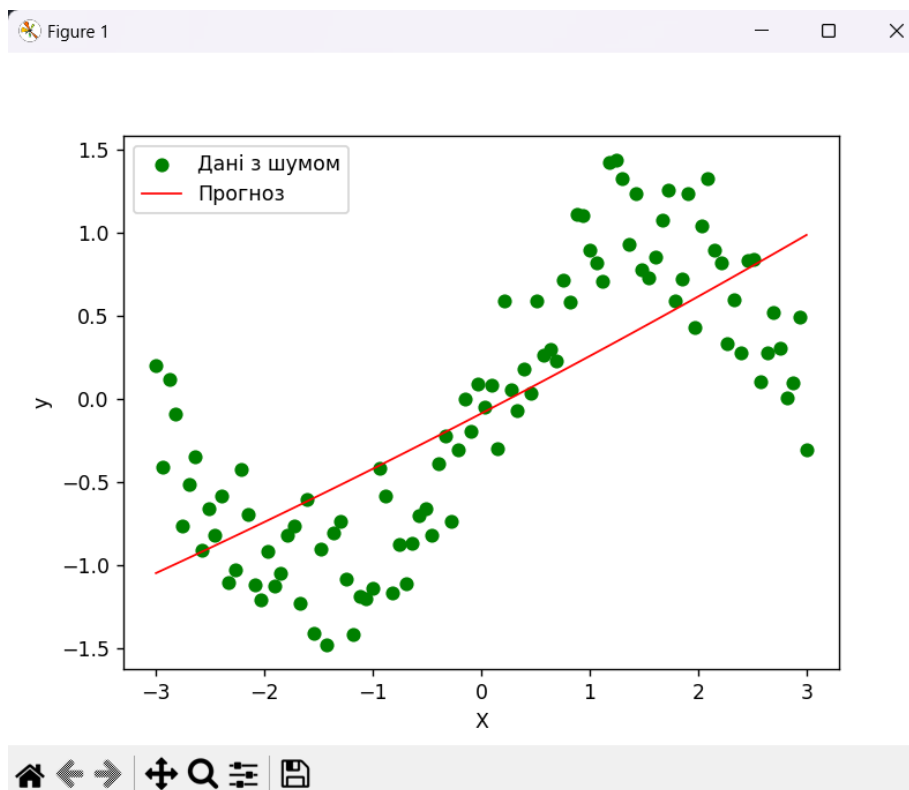
print("Intercept:", lin_reg.intercept_)
print("Coefficients:", lin_reg.coef_)

y_pred = lin_reg.predict(X_poly)
```



```
plt.scatter(X, y, color='green', label='Дані з шумом')
plt.plot(X, y_pred, color='red', linewidth=1, label='Прогноз')
plt.xlabel("X")
plt.ylabel("y")
plt.legend()
plt.show()
```

Результат :



```
task5 x
C:\Users\yousu\AppData\Local\Programs\Python\Python31
Intercept: -0.08763902478558057
Coefficients: [0.33961131 0.00633481]
Process finished with exit code 0
```

Зробіть висновок

Поліномна регресія дає змогу аналізувати не лінійні дані

Завдання 6 Побудова кривих навчання

Код програми:

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

import matplotlib
import matplotlib.pyplot as plt
matplotlib.use('TkAgg')

m = 100
X = 6 * np.random.rand(m, 1) - 3
y = 0.4 * X ** 2 + X + 4 + 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[:m], y_train_predict))
        val_errors.append(mean_squared_error(y_val, y_val_predict))

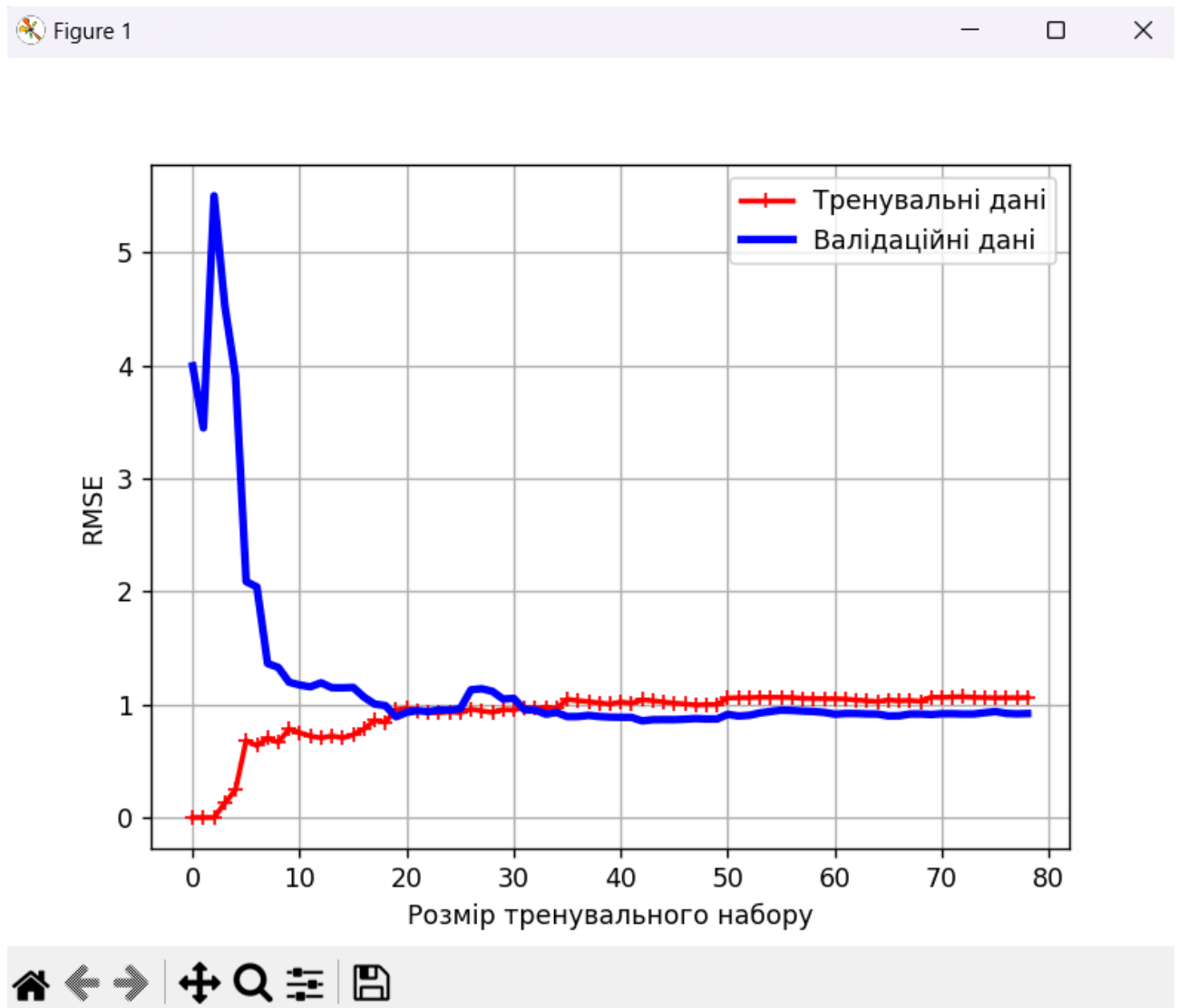
    plt.plot(np.sqrt(train_errors), "r-+", linewidth=2, label="Тренувальні дані")
    plt.plot(np.sqrt(val_errors), "b-", linewidth=3, label="Валідаційні дані")
    plt.xlabel("Розмір тренувального набору")
    plt.ylabel("RMSE")
    plt.legend()
    plt.grid()
    plt.show()

polynomial_regression = Pipeline([
    ("poly_features", PolynomialFeatures(degree=2, include_bias=False)),
    ("lin_reg", linear_model.LinearRegression()),
```

)

```
plot_learning_curves(polynomial_regression, X, y)
```

Результат :



Висновок: Під час лабораторної роботи я використав спеціалізовані бібліотеки та мову програмування Python та дослідив методи регресії даних у машинному навчанні.