

## ЛАБОРАТОРНА РОБОТА № 3

### ДОСЛІДЖЕННЯ МЕТОДІВ РЕГРЕСІЇ ТА НЕКОНТРОЛЬОВАНОГО НАВ- ЧАННЯ

**Мета:** Використовуючи спеціалізовані бібліотеки і мову програмування Python дослідити методи регресії та неконтрольованої класифікації даних у машинному навчанні.

#### Хід роботи:

**Завдання 2.1.** Створення регресора однієї змінної

Лістинг програми:

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

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

# Вхідний файл, який містить дані
input_file = '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)
```

					ДУ «Житомирська політехніка».23.121.05.000 – Лр3			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розроб.		Дубинченко Б.М.			Звіт з лабораторної роботи		Лім.	Арк.
Перевір.		Іванов Д.А.						1
Керівник								18
Н. контр.							ФІКТ Гр. ІПЗ-20-4[1]	
Зав. каф.								

```

# 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)

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

# Завантаження моделі
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 виконання завдання:

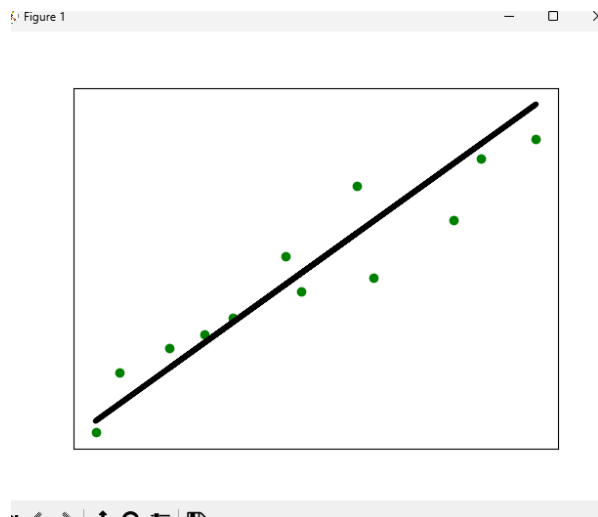


Рис. 3.1. Результат виконання програми

		Дубинченко Б.М.			ДУ «Житомирська політехніка».23.121.05.000 – Лр3	Арк.
		Іванов Д.А.				2
Змн.	Арк.	№ докум.	Підпис	Дата		

```

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

```

Рис. 3.2. Результат виконання програми

Отримані показники свідчать про недостатньо задовільні результати для поточної регресійної моделі. Для покращення ефективності необхідно розглянути використання поліноміального регресора

### Завдання 2.2. Передбачення за допомогою регресії однієї змінної

Таблиця 3.1

№ за списком	5
№ варіанту	5

Лістинг програми:

```

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

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

# Вхідний файл, який містить дані
input_file = 'data_regr_5.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

```

		Дубинченко Б.М.			ДУ «Житомирська політехніка».23.121.05.000 – Лр3	Арк.
		Іванов Д.А.				3
Змн.	Арк.	№ докум.	Підпис	Дата		

```

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)

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

# Завантаження моделі
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 виконання завдання:

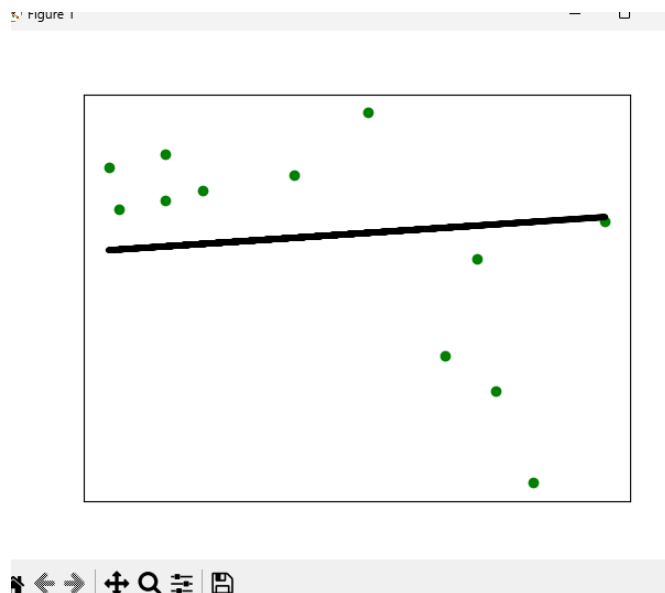


Рис. 3.3. Результат виконання програми

		Дубинченко Б.М.			ДУ «Житомирська політехніка».23.121.05.000 – ЛрЗ	Арк.
		Іванов Д.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		4

```

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

New mean absolute error = 3.31

```

Рис. 3.4. Результат виконання програми

За отриманими результатами можна зробити висновок, що вхідні дані, ймовірно, не відповідають вимогам. Для покращення ефективності регресійної моделі потрібно збирати більш якісні та докладні дані

### Завдання 2.3. Створення багатовимірного регресора

Лістинг програми:

```

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

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

# Вхідний файл, який містить дані
input_file = 'data_multivar_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)

```

		Дубинченко Б.М.			ДУ «Житомирська політехніка».23.121.05.000 – Лр3	Арк.
		Іванов Д.А.				5
Змн.	Арк.	№ докум.	Підпис	Дата		

```

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)

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

# Завантаження моделі
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))

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

```

Результат виконання завдання:

```

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

New mean absolute error = 3.58

Linear regression:
[36.05286276]

Polynomial regression:
[41.45976677]

Process finished with exit code 0

```

Рис. 3.5. Результат виконання програми

		Дубинченко Б.М.			ДУ «Житомирська політехніка».23.121.05.000 – ЛрЗ	Арк.
		Іванов Д.А.				6
Змн.	Арк.	№ докум.	Підпис	Дата		

На підставі отриманих даних, поліноміальна регресійна модель виявляється більш ефективною, ніж лінійна модель, для отримання кращих результатів

#### Завдання 2.4. Регресія багатьох змінних

Лістинг програми:

```
import matplotlib

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

print("Regr coef =", regr.coef_)
print("Regr intercept =", round(regr.intercept_, 2))
print("R2 score =", round(r2_score(ytest, ypred), 2))
print("Mean absolute error =", round(mean_absolute_error(ytest, ypred), 2))
print("Mean squared error =", round(mean_squared_error(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()
```

Результат виконання завдання:

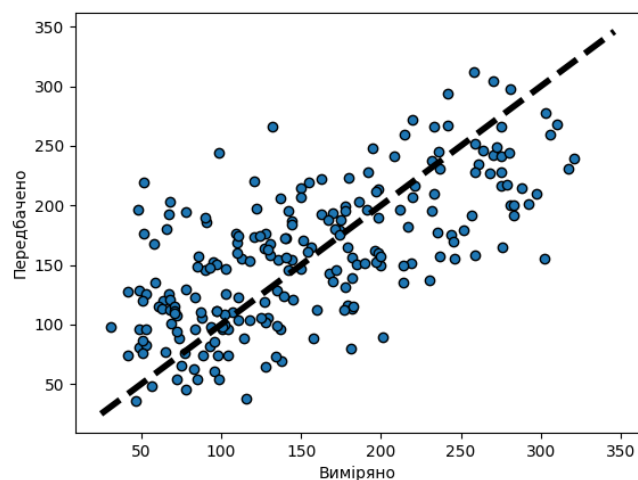


Рис. 3.6. Результат виконання програми

		Дубинченко Б.М.			ДУ «Житомирська політехніка».23.121.05.000 – Лр3	Арк.
		Іванов Д.А.				7
Змн.	Арк.	№ докум.	Підпис	Дата		

```

Regr coef = [ -20.4047621  -265.88518066  564.65086437  325.56226865 -692.16120333
 395.55720874  23.49659361  116.36402337  843.94613929  12.71856131]
Regr intercept = 154.36
R2 score = 0.44
Mean absolute error = 44.8
Mean squared error = 3075.33

```

Рис. 3.7. Результат виконання програми

Отримані показники свідчать про недостатньо задовільні результати для точної регресійної моделі. Для досягнення кращих результатів слід розглянути використання поліноміального регресора.

### Завдання 2.5. Самостійна побудова регресії

Таблиця 3.2

№ за списком	5
№ варіанту	5

Лістинг програми:

```

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

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

m = 100
X = 6 * np.random.rand(m, 1) - 5
y = 0.7 * X ** 2 + X + 3 + np.random.randn(m, 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')

```

		Дубинченко Б.М.			ДУ «Житомирська політехніка».23.121.05.000 – ПрЗ	Арк.
		Іванов Д.А.				8
Змн.	Арк.	№ докум.	Підпис	Дата		



```
plt.title("Лінійна регресія")
plt.plot(X_test, y_test_pred, color='black', linewidth=4)
plt.xticks(())
plt.yticks(())
plt.show()

poly = PolynomialFeatures(degree=2, include_bias=False)
poly_features = poly.fit_transform(X.reshape(-1, 1))
poly_reg_model = linear_model.LinearRegression()
poly_reg_model.fit(poly_features, y)
y_predicted = poly_reg_model.predict(poly_features)
plt.title("Поліноміальна регресія")
plt.scatter(X, y)
plt.plot(X, y_predicted, c="red")
plt.show()
print("Intercept = ", poly_reg_model.intercept_)
print("Coef = ", poly_reg_model.coef_)
```

Результат виконання завдання:

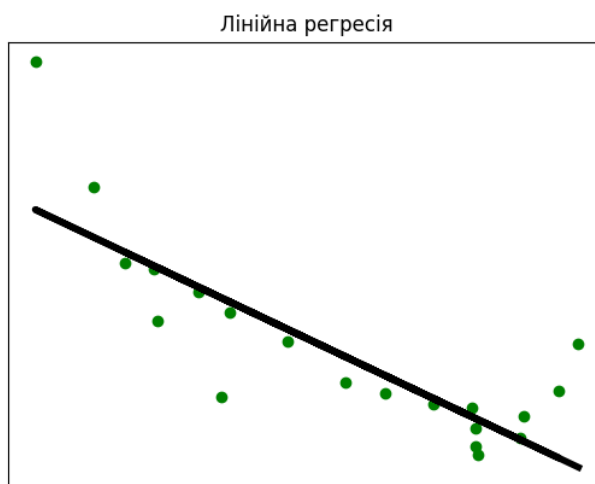


Рис. 3.8. Результат виконання програми

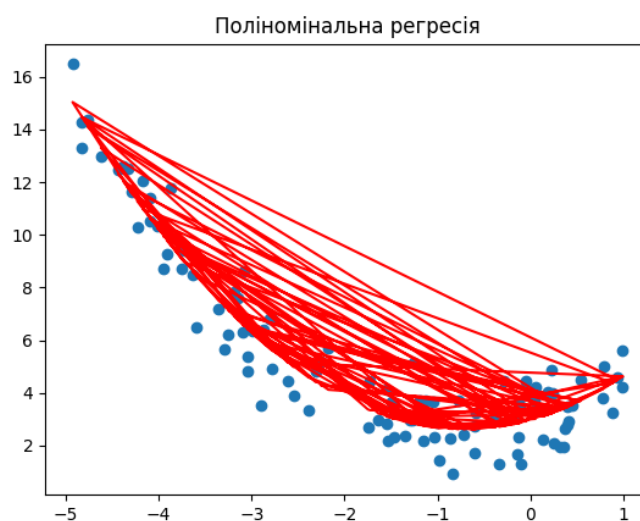


Рис. 3.9. Результат виконання програми

		Дубинченко Б.М.			ДУ «Житомирська політехніка».23.121.05.000 – ЛрЗ	Арк.
		Іванов Д.А.				9
Змн.	Арк.	№ докум.	Підпис	Дата		

```
Intercept = [2.96362222]
Coef = [[0.98076614 0.69683924]]
```

Рис. 3.10. Результат виконання програми

Початкова модель:  $y = 0,4x^2 + x + 4 + \text{Гауссів шум}$ .

Отримана модель регресії:  $y = 0,69x^2 + 0,98x + 2,96$ .

Отримані коефіцієнти близькі до модельних. І це буде означає що модель навчена правильно.

## Завдання 2.6. Побудова кривих навчання

Лістинг програми:

```
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import train_test_split
import matplotlib

matplotlib.use('TkAgg')
from matplotlib import pyplot as plt
import numpy as np
from sklearn import linear_model
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import PolynomialFeatures

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(['Навчальний набір', 'Перевіряючий набір'])
    plt.show()

m = 100
X = 6 * np.random.rand(m, 1) - 3
y = 0.6 * X ** 2 + X + 2 + np.random.randn(m, 1)
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)
```

Результат виконання завдання:

		Дубинченко Б.М.			ДУ «Житомирська політехніка».23.121.05.000 – Лр3	Арк.
		Іванов Д.А.				10
Змн.	Арк.	№ докум.	Підпис	Дата		

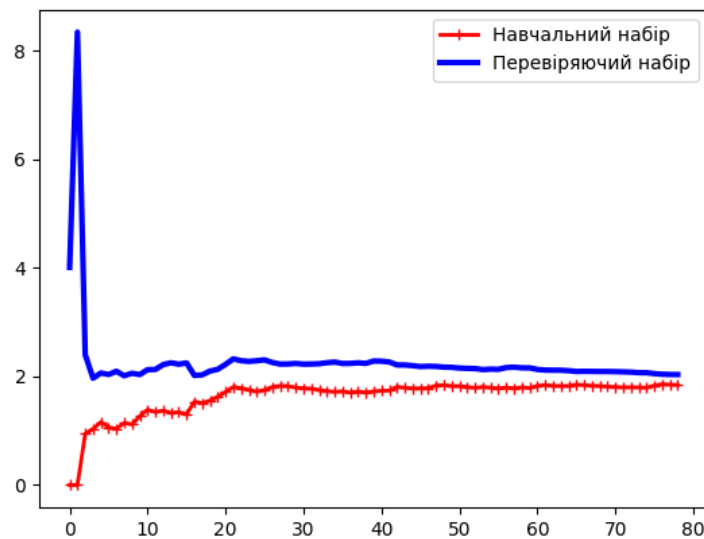


Рис. 3.11. Результат виконання програми

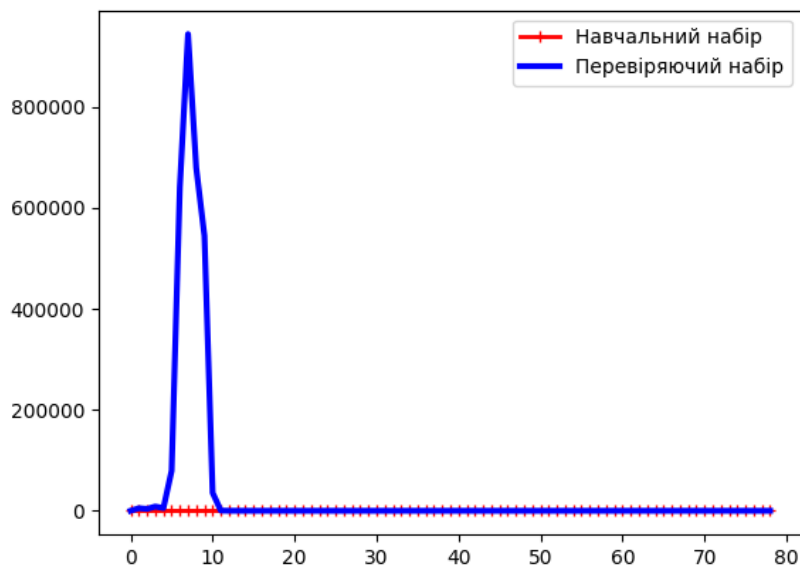


Рис. 3.12. Результат виконання програми

## Завдання 2.7. Кластеризація даних за допомогою методу k-середніх

Лістинг програми:

```
import numpy as np
import matplotlib

matplotlib.use('TkAgg')
from matplotlib import pyplot as plt
from sklearn.cluster import KMeans
from sklearn import metrics

# Завантаження вхідних даних
X = np.loadtxt('data_clustering.txt', delimiter=',')

num_clusters = 5
```

		Дубинченко Б.М.			ДУ «Житомирська політехніка».23.121.05.000 – Лр3	Арк.
		Іванов Д.А.				11
Змн.	Арк.	№ докум.	Підпис	Дата		

```

# Включення вхідних даних до графіка
plt.figure()
plt.scatter(X[:, 0], X[:, 1], marker='o', facecolors='none', edgecolors='black',
s=80)
x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
plt.title('Вхідні дані')
plt.xlim(x_min, x_max)
plt.ylim(y_min, y_max)
plt.xticks(())
plt.yticks(())
plt.show()

# Створення об'єкту KMeans
kmeans = KMeans(init='k-means++', n_clusters=num_clusters, n_init=10)

# Навчання моделі кластеризації KMeans
kmeans.fit(X)

# Визначення кроку сітки
step_size = 0.01

# Відображення точок сітки
x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
x_vals, y_vals = np.meshgrid(np.arange(x_min, x_max, step_size), np.arange(y_min,
y_max, step_size))

# Передбачення вихідних міток для всіх точок сітки
output = kmeans.predict(np.c_[x_vals.ravel(), y_vals.ravel()])

# Графічне відображення областей та виділення їх кольором
output = output.reshape(x_vals.shape)
plt.figure()
plt.clf()
plt.imshow(output, interpolation='nearest',
extent=(x_vals.min(), x_vals.max(),
y_vals.min(), y_vals.max()),
cmap=plt.cm.Paired,
aspect='auto',
origin='lower')

# Відображення вхідних точок
plt.scatter(X[:, 0], X[:, 1], marker='o', facecolors='none', edgecolors='black',
s=80)

# Відображення центрів кластерів
cluster_centers = kmeans.cluster_centers_
plt.scatter(cluster_centers[:, 0], cluster_centers[:, 1], marker='o', s=210,
linewidths=4, color='black', zorder=12,
facecolors='black')
x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
plt.title('Границя кластерів')
plt.xlim(x_min, x_max)
plt.ylim(y_min, y_max)
plt.xticks(())
plt.yticks(())
plt.show()

```

Результат виконання завдання:

		Дубинченко Б.М.			ДУ «Житомирська політехніка».23.121.05.000 – Лр3	Арк.
		Іванов Д.А.				12
Змн.	Арк.	№ докум.	Підпис	Дата		

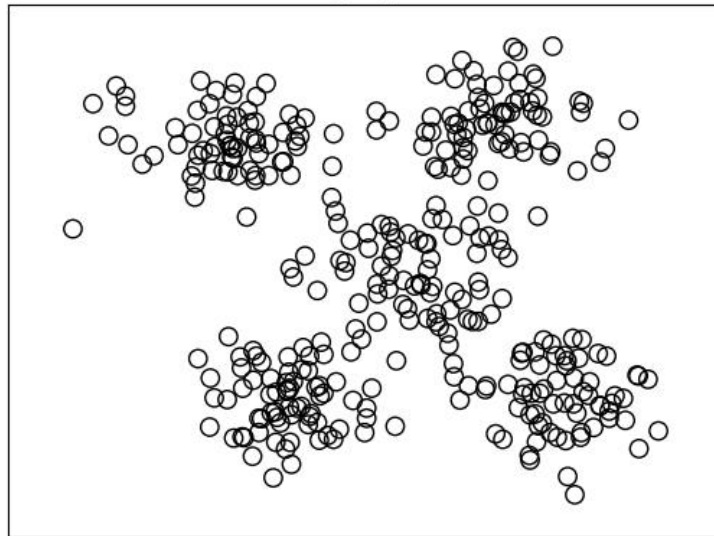


Рис. 3.13. Результат виконання програми

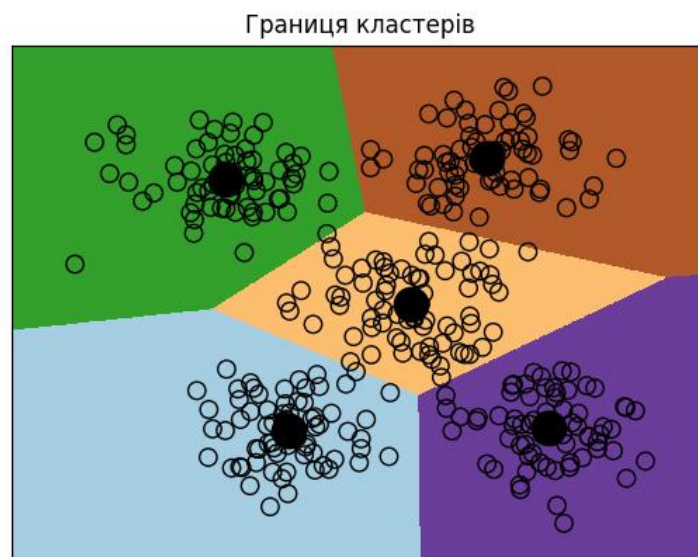


Рис. 3.14. Результат виконання програми

В результаті виконання програмного коду були отримані задовільні результати, де більшість точок знаходяться в межах визначеної області. Знаходження центроїдів відображає найбільшу концентрацію точок відповідного кластеру

## Завдання 2.8. Кластеризація К-середніх для набору даних Iris

Лістинг програми:

```
from sklearn.svm import SVC
from sklearn.metrics import pairwise_distances_argmin
import numpy as np
from sklearn.datasets import load_iris
from sklearn.cluster import KMeans
import matplotlib
```

		Дубинченко Б.М.			ДУ «Житомирська політехніка».23.121.05.000 – Лр3	Арк.
		Іванов Д.А.				13
Змн.	Арк.	№ докум.	Підпис	Дата		

```

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

iris = load_iris()
X = iris['data']
y = iris['target']

# Створення об'єкту KMeans
kmeans = KMeans(n_clusters=3, init='k-means++', n_init=10)

# Навчання моделі кластеризації KMeans
kmeans.fit(X)

# Передбачення вихідних міток
y_kmeans = kmeans.predict(X)

# Відображення вхідних точок
plt.scatter(X[:, 0], X[:, 1], c=y_kmeans, s=50, cmap='viridis')

# Відображення центрів кластерів
centers = kmeans.cluster_centers_
plt.scatter(centers[:, 0], centers[:, 1], c='black', s=200, alpha=0.5)

def find_clusters(X, n_clusters, rseed=2):
    # Довільне обрання кластерів
    rng = np.random.RandomState(rseed)
    i = rng.permutation(X.shape[0])[:n_clusters]
    centers = X[i]

    while True:
        # Призначення міток на основі найближчого центру
        labels = pairwise_distances_argmin(X, centers)

        # Знаходження нових центрів за середніми точками
        new_centers = np.array([X[labels == i].mean(0)
                                for i in range(n_clusters)])

        # Перевірка на збіжність
        if np.all(centers == new_centers):
            break
        centers = new_centers

    return centers, labels

# Відображення точок
centers, labels = find_clusters(X, 3)
plt.scatter(X[:, 0], X[:, 1], c=labels, s=50, cmap='viridis')
labels = KMeans(3, random_state=0).fit_predict(X)
plt.scatter(X[:, 0], X[:, 1], c=labels, s=50, cmap='viridis')
plt.show()

```

Результат виконання завдання:

		Дубинченко Б.М.			ДУ «Житомирська політехніка».23.121.05.000 – Лр3	Арк.
		Іванов Д.А.				14
Змн.	Арк.	№ докум.	Підпис	Дата		

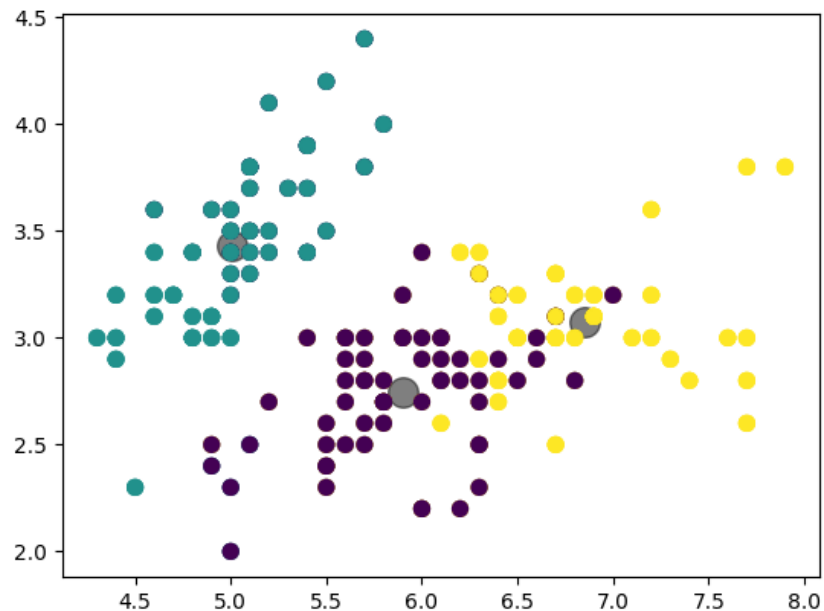


Рис. 3.15. Результат виконання програми

Після виконання програмного коду були отримані середні результати. Проте визначення центроїдів відображає найбільшу концентрацію точок у відповідному кластері.

**Завдання 2.9.** Оцінка кількості кластерів з використанням методу зсуву середнього

Лістинг програми:

```
import numpy as np
import matplotlib

matplotlib.use('TkAgg')
from matplotlib import pyplot as plt
from sklearn.cluster import MeanShift, estimate_bandwidth
from itertools import cycle

# Завантаження даних
X = np.loadtxt('data_clustering.txt', delimiter=',')

# Оцінка ширини вікна для X
bandwidth_X = estimate_bandwidth(X, quantile=0.1, n_samples=len(X))

# Кластеризація даних методом зсуву середнього
meanshift_model = MeanShift(bandwidth=bandwidth_X, bin_seeding=True)
meanshift_model.fit(X)

# Витягування центрів кластерів
cluster_centers = meanshift_model.cluster_centers_
print('\nCenter of clusters:\n', cluster_centers)
```

		Дубинченко Б.М.			ДУ «Житомирська політехніка».23.121.05.000 – Лр3	Арк.
		Іванов Д.А.				15
Змн.	Арк.	№ докум.	Підпис	Дата		

```
# Оцінка кількості кластерів
labels = meanshift_model.labels_
num_clusters = len(np.unique(labels))
print('\nCenter of clusters in input data =', num_clusters)

# Відображення на графіку точок та центрів кластерів
plt.figure()
markers = 'o*xvs'
for i, marker in zip(range(num_clusters), markers):
    plt.scatter(X[labels == i, 0], X[labels == i, 1], marker=marker, color='blue')
    cluster_center = cluster_centers[i]
    plt.plot(cluster_center[0], cluster_center[1], marker='o',
             markerfacecolor='black', markeredgecolor='black',
             markersize=15)
plt.title('Кластери')
plt.show()
```

Результат виконання завдання:

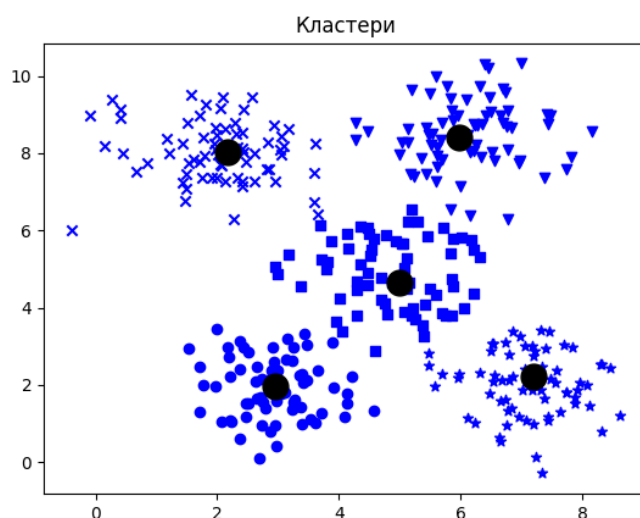


Рис. 3.16. Результат виконання програми

```
Center of clusters:
[[2.95568966 1.95775862]
 [7.20690909 2.20836364]
 [2.17603774 8.03283019]
 [5.97960784 8.39078431]
 [4.99466667 4.65844444]]

Center of clusters in input data = 5
```

Рис. 3.17. Результат виконання програми

Результати, отримані за допомогою методу кластеризації зсуву середнього, свідчать про успішну роботу. Було виявлено 5 кластерів, що відповідає кількості, визначеній вручну в попередніх завданнях

		Дубинченко Б.М.			ДУ «Житомирська політехніка».23.121.05.000 – Лр3	Арк.
		Іванов Д.А.				16
Змн.	Арк.	№ докум.	Підпис	Дата		



## Завдання 2.10. Знаходження підгруп на фондовому ринку з використанням моделі поширення подібності

Лістинг програми:

```
import datetime
import json
import numpy as np
from sklearn import covariance, cluster
import matplotlib

matplotlib.use('TkAgg')
from matplotlib import pyplot as plt
from matplotlib.finance import quotes_historical_yahoo_ochl as quotes_yahoo

# Вхідний файл із символічними позначеннями компаній
input_file = 'company_symbol_mapping.json'

# Завантаження прив'язок символів компаній до їх повних назв
with open(input_file, 'r') as f:
    company_symbols_map = json.loads(f.read())

symbols, names = np.array(list(company_symbols_map.items())).T

# Завантаження архівних даних котирувань
start_date = datetime.datetime(2003, 7, 3)
end_date = datetime.datetime(2007, 5, 4)
quotes = [quotes_yahoo(symbol, start_date, end_date, asobject=True) for symbol in symbols]

# Вилучення котирувань, що відповідають відкриттю та закриттю біржі
opening_quotes = np.array([quote.open for quote in quotes]).astype(np.float)
closing_quotes = np.array([quote.close for quote in quotes]).astype(np.float)

# Обчислення різниці між двома видами котирувань
quotes_diff = closing_quotes - opening_quotes
X = quotes_diff.copy().T
X /= X.std(axis=0)

# Створення моделі графа
edge_model = covariance.GraphicalLassoCV()

# Навчання моделі
with np.errstate(invalid='ignore'):
    edge_model.fit(X)

# Створення моделі кластеризації на основі поширення подібності
_, labels = cluster.affinity_propagation(edge_model.covariance_)
num_labels = labels.max()

for i in range(num_labels + 1):
    print("Cluster", i + 1, "==>", ','.join(names[labels == i]))
```

Результат виконання завдання:

```
Traceback (most recent call last):
  File "C:\Users\dubin\OneDrive\Робочий стол\4 course\AI\lab3\LR_3_task_10.py", line 9, in <module>
    from matplotlib.finance import quotes_historical_yahoo_ochl as quotes_yahoo
ModuleNotFoundError: No module named 'matplotlib.finance'
```

Рис. 3.18. Результат виконання програми

		Дубинченко Б.М.			ДУ «Житомирська політехніка».23.121.05.000 – ЛрЗ	Арк.
		Іванов Д.А.				17
Змн.	Арк.	№ докум.	Підпис	Дата		

Посилання на GitHub: [https://github.com/BogdanStelmah/Basics-of-AI\\_labs](https://github.com/BogdanStelmah/Basics-of-AI_labs)

**Висновок:** На даній лабораторній роботі ми використовуємо спеціалізовані бібліотеки і мову програмування Python дослідили методи регресії та неконтрольованої класифікації даних у машинному навчанні.

		Дубинченко Б.М.			ДУ «Житомирська політехніка».23.121.05.000 – Лр3	Арк.
		Іванов Д.А.				18
Змн.	Арк.	№ докум.	Підпис	Дата		