

ЛАБОРАТОРНА РОБОТА № 3

ДОСЛІДЖЕННЯ МЕТОДІВ РЕГРЕСІЇ ТА НЕКОНТРОЛЬОВАНОГО НАВЧАННЯ

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

Хід роботи:

ЧАСТИНА 1. ДОСЛІДЖЕННЯ МЕТОДІВ РЕГРЕСІЇ

Завдання 3.1: 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

# 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)

# 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()
```

					ДУ «Житомирська політехніка».22.121.14.000 – ПрЗ			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розроб.		Сірач А.С.			Звіт з лабораторної роботи		Літ.	Арк.
Перевір.		Філіпов В.О.						1
Керівник							ФІКТ Гр. ІПЗ-19-3[2]	
Н. контр.								
Зав. каф.								

```

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))

```

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. 3.1 Результати оцінки якості

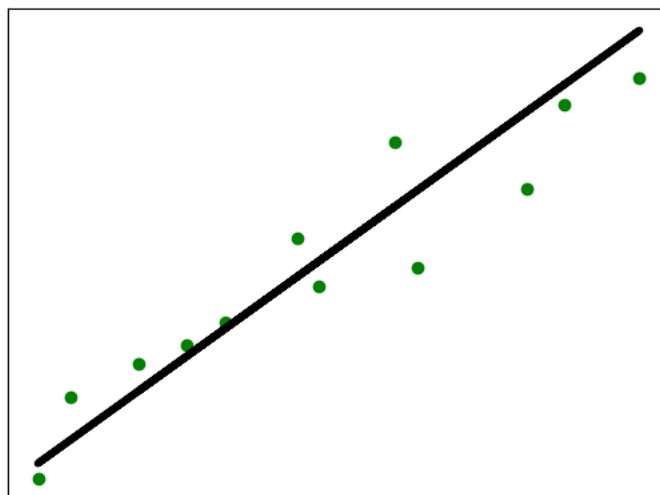


Рис. 3.2 Графік функції

Висновок: ми можемо використовувати цей спосіб для статистичного аналізу, який намагається показати зв'язок між двома змінними, наприклад відстежувати

		Сірач А.С.			ДУ «Житомирська політехніка».22.121.14.000 – ЛрЗ	Арк.
		Філіпов В.О.				2
Змн.	Арк.	№ докум.	Підпис	Дата		

тенденцію цін. Для отримання кращого результату необхідно використовувати поліноміальний регресор.

Завдання 3.2: Передбачення за допомогою регресії однієї змінної

Варіант 4 файл: data_regr_4.txt

Лістинг програми:

```
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 = '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 = 'model.pkl'
```

		Сірач А.С.			ДУ «Житомирська політехніка».22.121.14.000 – Лр3	Арк.
		Філіпов В.О.				3
Змн.	Арк.	№ докум.	Підпис	Дата		

```
# 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))
```

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

Рис. 3.3 Результат програми та оцінки якості з використанням даних свого варіанту

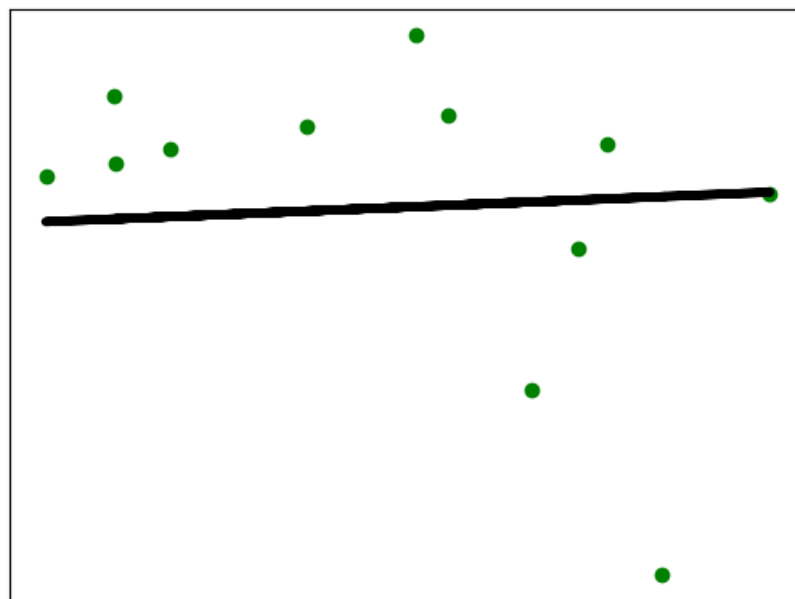


Рис. 3.4 Графік функції

Висновок: з графіка видно, що залишки розподілені не рівномірно щодо горизонтальної осі. Виходячи з R^2 оцінки можна зробити висновок, що продуктивність цієї моделі машинного навчання на основі регресії є поганою. А для покращення роботи регресійної моделі необхідно зібрати більш якісні вхідні дані.

		Сірач А.С.			ДУ «Житомирська політехніка».22.121.14.000 – ЛрЗ	Арк.
		Філіпов В.О.				4
Змн.	Арк.	№ докум.	Підпис	Дата		

Завдання 3.3: Створення багатовимірного регресора

Лістинг програми:

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

# 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)

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]]
```

		Сірач А.С.			ДУ «Житомирська політехніка».22.121.14.000 – ЛрЗ	Арк.
		Філіпов В.О.				5
Змн.	Арк.	№ докум.	Підпис	Дата		

```
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))
```

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:

[41.46072631]

Рис. 3.5 Результат виконання програми

Висновок: згідно отриманих результатів поліноміальний регресор забезпечує отримання кращого результату, порівняно з лінійним регресором. Адже коефіцієнт детермінації є наближеним до 1.

Завдання 3.4. 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

# Load the diabetes dataset
diabetes_X, diabetes_y = datasets.load_diabetes(return_X_y=True)

# Use only one feature
diabetes_X = diabetes_X[:, np.newaxis, 2]
```

		Сірач А.С.			ДУ «Житомирська політехніка».22.121.14.000 – ЛрЗ	Арк.
		Філіпов В.О.				6
Змн.	Арк.	№ докум.	Підпис	Дата		

```

# Split the data into training/testing sets
diabetes_X_train = diabetes_X[:-20]
diabetes_X_test = diabetes_X[-20:]

# Split the targets into training/testing sets
diabetes_y_train = diabetes_y[:-20]
diabetes_y_test = diabetes_y[-20:]

# Create linear regression object
regr = linear_model.LinearRegression()

# Train the model using the training sets
regr.fit(diabetes_X_train, diabetes_y_train)

# Make predictions using the testing set
diabetes_y_pred = regr.predict(diabetes_X_test)

# The coefficients
print("Regression coef: \n", regr.coef_)
print("Regression intercept: \n", regr.intercept_)
# Середня абсолютна похибка
print("Mean absolute error :",
      round(mean_absolute_error(diabetes_y_test, diabetes_y_pred), 2))
# The mean squared error
print("Mean squared error: %.2f" % mean_squared_error(diabetes_y_test,
diabetes_y_pred))
# The coefficient of determination: 1 is perfect prediction
print("R2 score: %.2f" % r2_score(diabetes_y_test, diabetes_y_pred))

fig, ax = plt.subplots()
ax.scatter(diabetes_y_test, diabetes_y_pred, edgecolors=(0, 0, 0))
ax.plot([diabetes_y.min(), diabetes_y.max()], [diabetes_y.min(),
diabetes_y.max()], 'k--', lw=4)
ax.set_xlabel('Виміряно')
ax.set_ylabel('Передбачено')
plt.show()

```

Regression coef:
 [938.23786125]
 Regression intercept:
 152.91886182616113
 Mean absolute error : 41.23
 Mean squared error: 2548.07
 R2 score: 0.47

Рис. 3.6 Результат роботи додатку

		Сірач А.С.			ДУ «Житомирська політехніка».22.121.14.000 – ЛрЗ	Арк.
		Філіпов В.О.				7
Змн.	Арк.	№ докум.	Підпис	Дата		

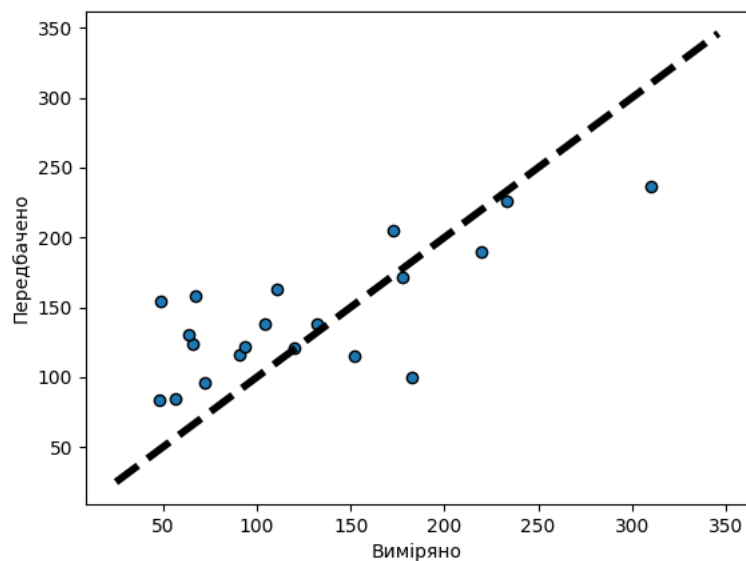


Рис. 3.7 Графік функції

Завдання 3.5: Самостійна побудова регресії

№ за списком: 14, № варіанту: 4

Лістинг програми:

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

		Сірач А.С.			ДУ «Житомирська політехніка».22.121.14.000 – ЛрЗ	Арк.
		Філіпов В.О.				8
Змн.	Арк.	№ докум.	Підпис	Дата		


```
plt.scatter(X_test, y_test, color='green')
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_)
```

Intercept = [3.14891847]

Coef = [[1.30064404 0.76242576]]

Рис. 3.8 Результат виконання програми

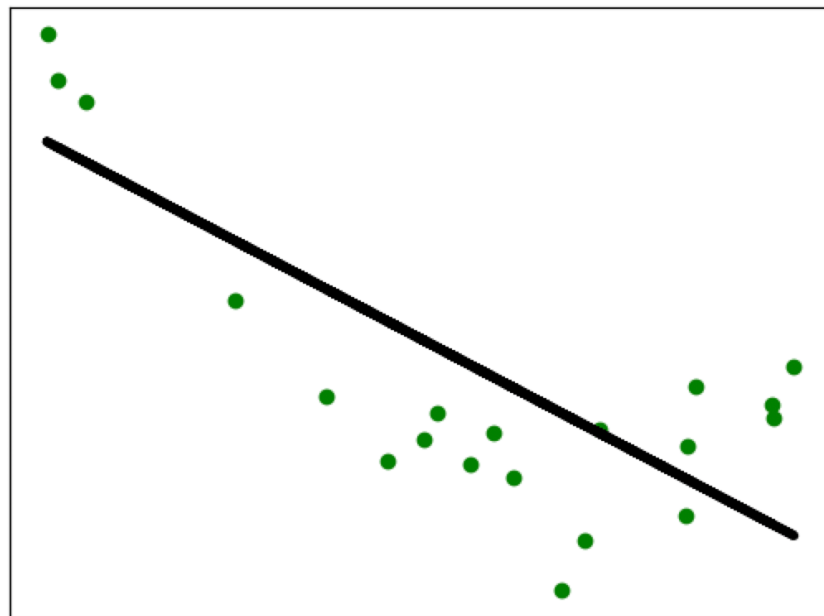


Рис. 3.9 Лінійна регресія

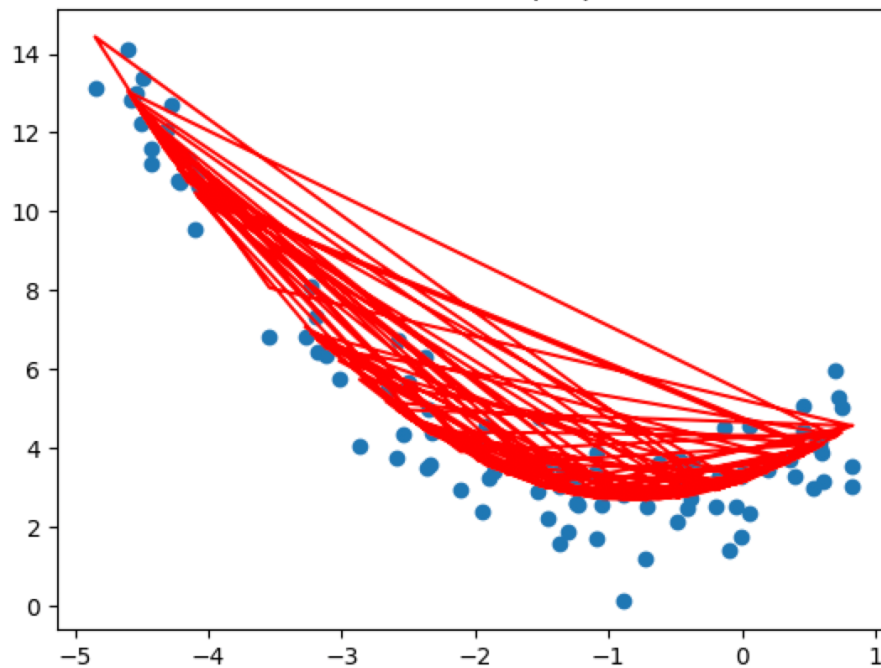


Рис. 3.10 Поліноміальна регресія

Початкова модель: $y = 0,7x^2 + x + 3 + \text{гауссів шум}$.

Отримана модель регресії: $y = 0,76x^2 + 1,3x + 3,1$.

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

Завдання 3.6: Побудова кривих навчання

Лістинг програми:

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

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 = [], []
    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) - 5
y = 0.7 * X ** 2 + X + 3 + 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)

```

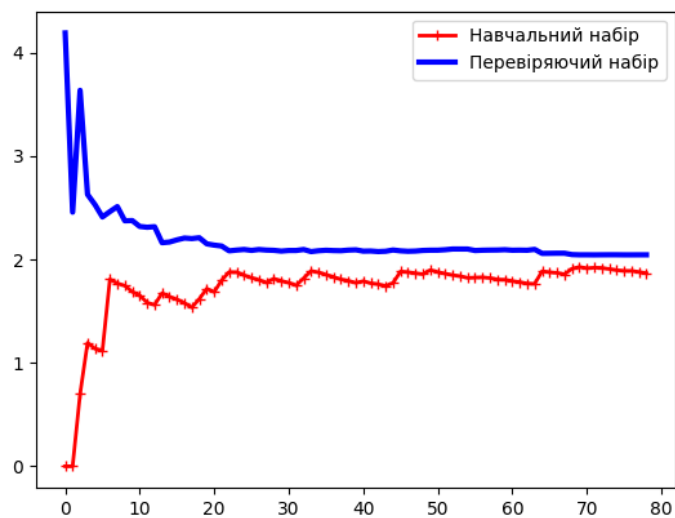


Рис. 3.11 Результат виконання програми

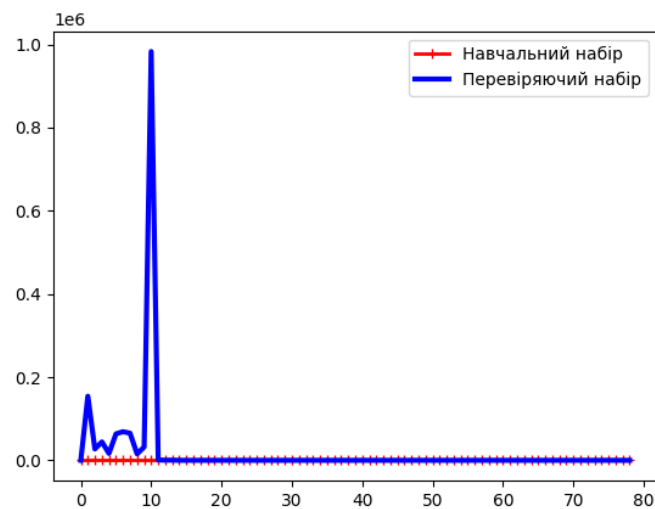


Рис. 3.12 Результат виконання програми

		Сірач А.С.			ДУ «Житомирська політехніка».22.121.14.000 – ЛрЗ	Арк.
		Філіпов В.О.				11
Змн.	Арк.	№ докум.	Підпис	Дата		

ЧАСТИНА 2. ДОСЛІДЖЕННЯ МЕТОДІВ НЕКОНТРОЛЬОВАНОГО НАВЧАННЯ

Завдання 3.7: Кластеризація даних за допомогою методу k-середніх

Лістинг програми:

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans

X = np.loadtxt('data_clustering.txt', delimiter=',')
num_clusters = 5

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('Input Data')
plt.xlim(x_min, x_max)
plt.ylim(y_min, y_max)
plt.xticks(())
plt.yticks(())

kmeans = KMeans(init='k-means++', n_clusters=num_clusters, n_init=10)
kmeans.fit(X)

step_size = 0.01

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))

output = kmeans.predict(np.c_[x_vals.ravel(), y_vals.ravel()])
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')

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

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("Cluster Edges")
plt.xlim(x_min, x_max)
plt.ylim(y_min, y_max)
```

		Сірач А.С.			ДУ «Житомирська політехніка».22.121.14.000 – Лр3	Арк.
		Філіпов В.О.				12
Змн.	Арк.	№ докум.	Підпис	Дата		

```
plt.xticks(())
plt.yticks(())
plt.show()
```

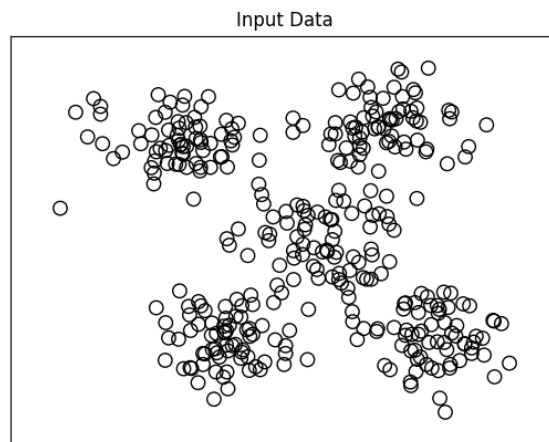


Рис. 3.13 Графіки К-середніх

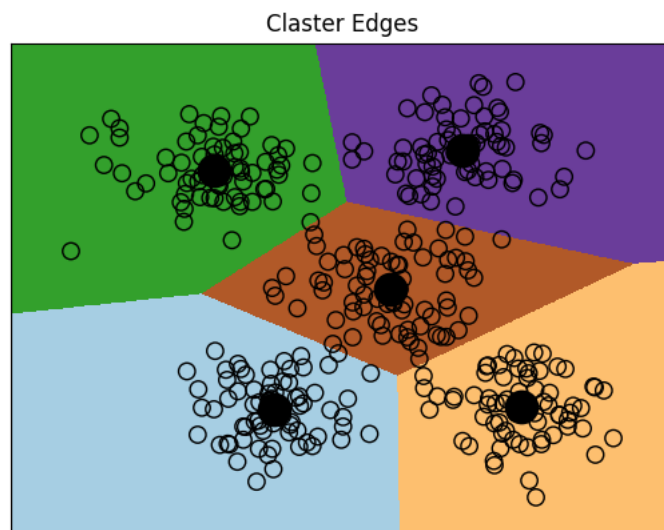


Рис. 3.14 Графіки К-середніх

Висновок: В результаті виконання програми ми отримали два графіки. Завдяки ним, можна побачити, що більшість точок повністю перебувають у визначеній області. А знаходження центроїдів зображає найбільше скупчення точок відповідного кластеру.

Завдання 3.8: Кластеризація К-середніх для набору даних Iris

Лістинг програми:

```
from sklearn.metrics import pairwise_distances_argmin
import numpy as np
from sklearn.datasets import load_iris
```

		Сірач А.С.			ДУ «Житомирська політехніка».22.121.14.000 – Пр3	Арк.
		Філіпов В.О.				13
Змн.	Арк.	№ докум.	Підпис	Дата		

```

from sklearn.cluster import KMeans
import matplotlib

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()

```

		Сірач А.С.			ДУ «Житомирська політехніка».22.121.14.000 – Лр3	Арк.
		Філіпов В.О.				14
Змн.	Арк.	№ докум.	Підпис	Дата		

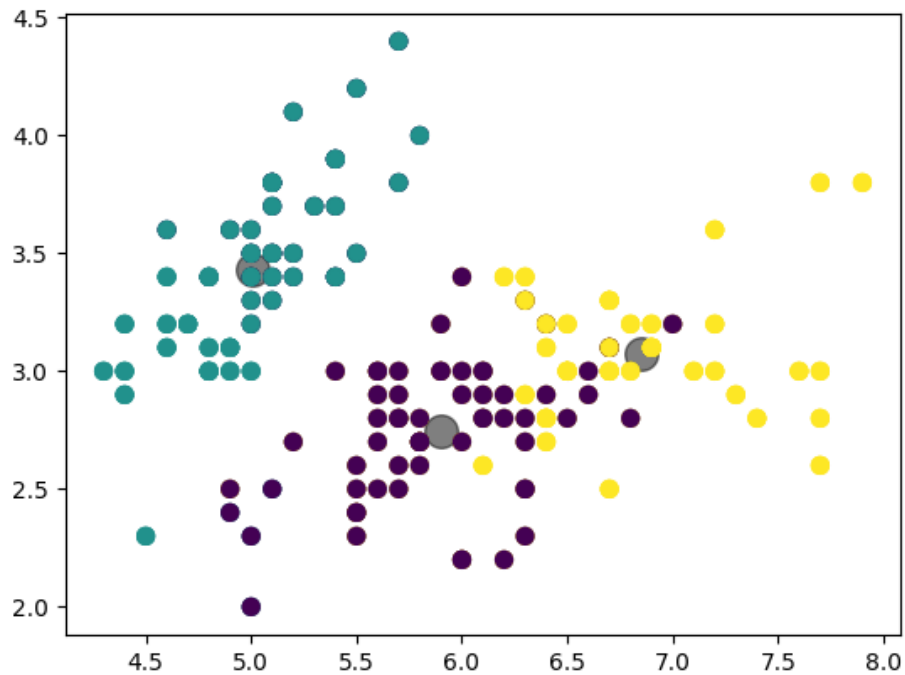


Рис. 3.15 Результат виконання програми

Висновки: В результаті виконання програми було отримано посередні результати, проте знаходження центроїдів зображає найбільше скупчення точок відповідного кластеру.

Завдання 3.9: Оцінка кількості кластерів з використанням методу зсуву середнього

Лістинг програми:

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import MeanShift, estimate_bandwidth

# 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))

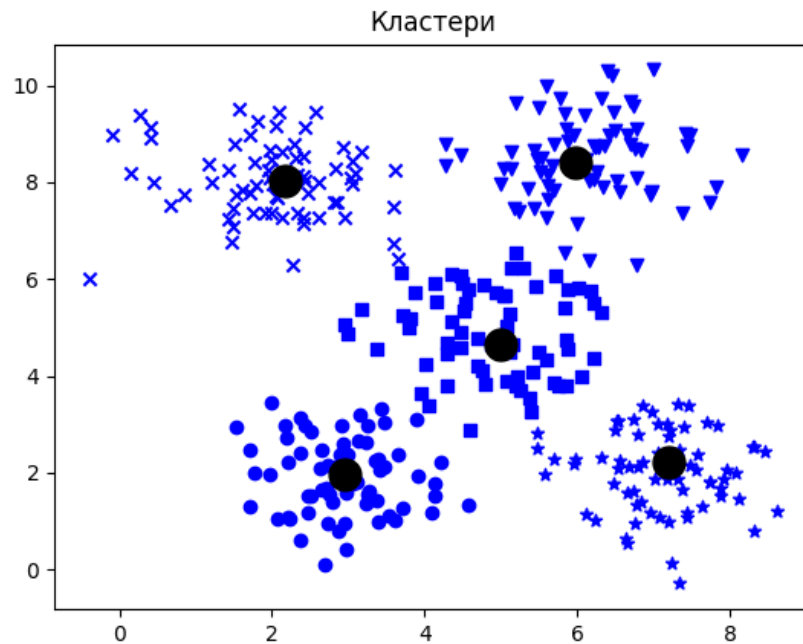
# Кластеризація даних методом зсуву середнього
meanshift_model = MeanShift(bandwidth=bandwidth_X, bin_seeding=True)
meanshift_model.fit(X)

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

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

		Сірач А.С.			ДУ «Житомирська політехніка».22.121.14.000 – Лр3	Арк.
		Філіпов В.О.				15
Змн.	Арк.	№ докум.	Підпис	Дата		

```
# 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()
```



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. 3.16 Результат виконання програми

Висновки: отримані результати вказують на гарні результати для методу кластеризації зсуву середнього. Було отримано 5 кластерів, стільки ж, як було вказано вручну в попередніх завданнях.

		Сірач А.С.			ДУ «Житомирська політехніка».22.121.14.000 – ЛрЗ	Арк.
		Філіпов В.О.				16
Змн.	Арк.	№ докум.	Підпис	Дата		

Завдання 3.10: 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.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)

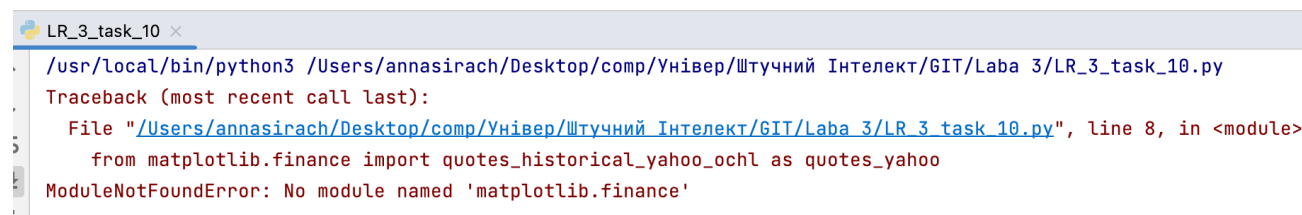
# Обчислення різниці між двома видами котирувань
quotes_diff = closing_quotes - opening_quotes
X = quotes_diff.copy().T
X /= X.std(axis=0)

# 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]))
```



```
LR_3_task_10 x
/usr/local/bin/python3 /Users/annasirach/Desktop/comp/Універ/Штучний Інтелект/GIT/Laba 3/LR_3_task_10.py
Traceback (most recent call last):
  File "/Users/annasirach/Desktop/comp/Універ/Штучний Інтелект/GIT/Laba 3/LR_3_task_10.py", line 8, in <module>
    from matplotlib.finance import quotes_historical_yahoo_ochl as quotes_yahoo
ModuleNotFoundError: No module named 'matplotlib.finance'
```

Рис. 3.17 Результат виконання програми

		Сірач А.С.			ДУ «Житомирська політехніка».22.121.14.000 – Лр3	Арк.
		Філіпов В.О.				17
Змн.	Арк.	№ докум.	Підпис	Дата		

Завдання №10 не вдалося виконати, адже пакет, який використовується для отримання початкових даних є неактуальним, через що отримати дані неможливо.

Висновки: в ході виконання лабораторної роботи було досліджено методи регресії та неконтрольованої класифікації даних у машинному навчанні, використовуючи спеціалізовані бібліотеки і мову програмування Python.

Посилання на GitHub: https://github.com/annasirach/AI_IPZ193_Sirach

		Сірач А.С.			ДУ «Житомирська політехніка».22.121.14.000 – Лр3	Арк.
		Філіпов В.О.				18
Змн.	Арк.	№ докум.	Підпис	Дата		