

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.

Завдання 2.1 Створення регресора однієї змінної.

					ДУ «Житомирська політехніка».20.121.18.			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розроб.		Соболевський Д.А						
Перевір.		Філіпов В.О.						
Керівник								
Н. контр.								
Зав. каф.								
						Літ.	Арк.	Аркушів
							1	17
						ФІКТ Гр. ІПЗк-20-1		

```

1 import pickle
2 import numpy as np
3 from sklearn import linear_model
4 import sklearn.metrics as sm
5 import matplotlib.pyplot as plt
6
7 input_file = 'data_singlevar_regr.txt'
8
9 # Завантаження даних
10 data = np.loadtxt(input_file, delimiter=',')
11 X, y = data[:, :-1], data[:, -1]
12
13 # Розбивка даних на навчальний та тестовий набори
14 num_training = int(0.8 * len(X))
15 num_test = len(X) - num_training
16
17 # Тренувальні дані
18 X_train, y_train = X[:num_training], y[:num_training]
19
20 # Тестові дані
21 X_test, y_test = X[num_training:], y[num_training:]
22
23 # Створення об'єкта лінійного регресора
24 regressor = linear_model.LinearRegression()
25
26 regressor.fit(X_train, y_train)
27
28 # Прогнозування результату
29 y_test_pred = regressor.predict(X_test)
30
31 # Побудова графіка
32 plt.scatter(X_test, y_test, color = 'green')
33 plt.plot(X_test, y_test_pred, color = 'black', linewidth = 4)
34 plt.xticks(())
35 plt.yticks(())
36 plt.show()
37
38 # Обрахування метрик
39 print("Linear regressor performance:")
40 print("Mean absolute error =", round(sm.mean_absolute_error(y_test, y_test_pred), 2))
41 print("Mean squared error =", round(sm.mean_squared_error(y_test, y_test_pred), 2))
42 print("Median absolute error =", round(sm.median_absolute_error(y_test, y_test_pred), 2))
43 print("Explain variance score =", round(sm.explained_variance_score(y_test, y_test_pred), 2))
44 print("R2 score =", round(sm.r2_score(y_test, y_test_pred), 2))
45
46 # Файл для збереження моделі
47 output_model_file = 'model.pkl'
48
49 # Збереження моделі
50 with open(output_model_file, 'wb') as f:
51     pickle.dump(regressor, f)
52
53 # Завантаження моделі
54 with open(output_model_file, 'rb') as f:
55     regressor_model = pickle.load(f)
56
57 y_test_pred_new = regressor_model.predict(X_test)
58
59 print("\nNew mean absolute error =", round(sm.mean_absolute_error(y_test, y_test_pred_new), 2))
60

```

Рис 1. Код програми файлу LR_3_task_1m

		Соболевський Д.А.			ДУ «Житомирська політехніка».20.121.18 – Лр3	Арк.
		Філіпов В.О.				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
```

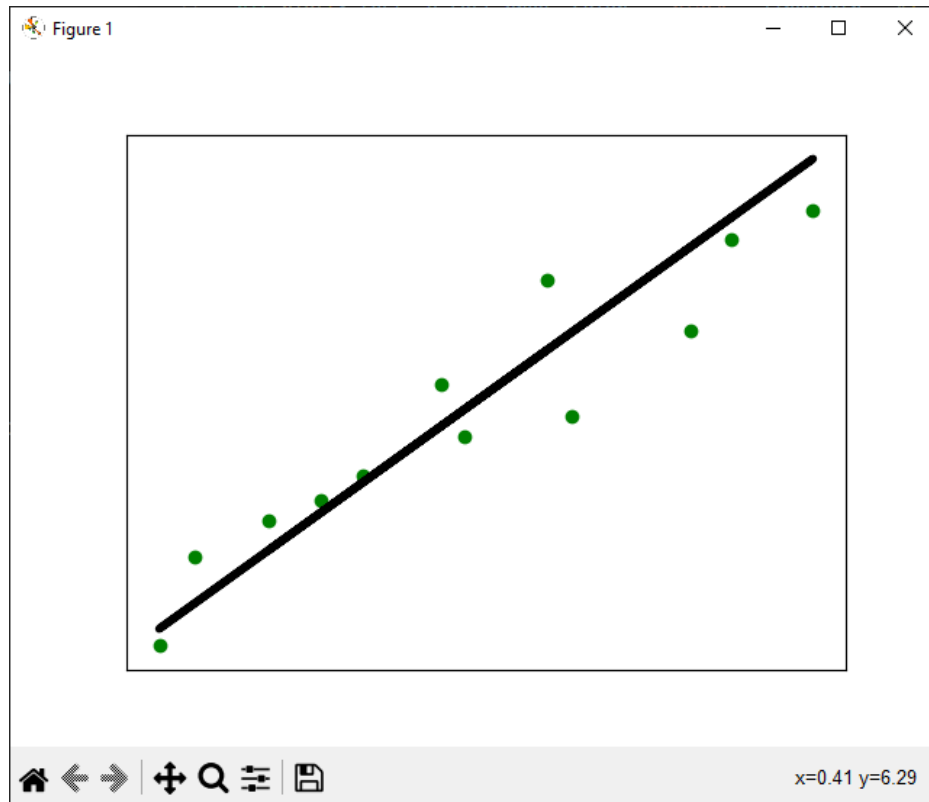


Рис 2. Результат виконання коду файлу LR_3_task_1

Висновок: модель для вихідних даних побудована валідно. MAE, MSE – середня якість. Показник R2 – добре.

Завдання 2.2. Передбачення за допомогою регресії однієї змінної.

Номер – 18

Варіант – 3

		Соболевський Д.А			ДУ «Житомирська політехніка».20.121.18 – Лр3	Арк.
		Філіпов В.О.				3
Змн.	Арк.	№ докум.	Підпис	Дата		

№ за списком	11	12	13	14	15	16	17	18	19	20
№ варіанту	1	2	3	4	5	1	2	3	4	5

Варіант 3 файл: data_regr_3.txt

Варіант 4 файл: data_regr_4.txt

		Соболевський Д.А.			ДУ «Житомирська політехніка».20.121.18 – Лр3	Арк.
		Філіпов В.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		4

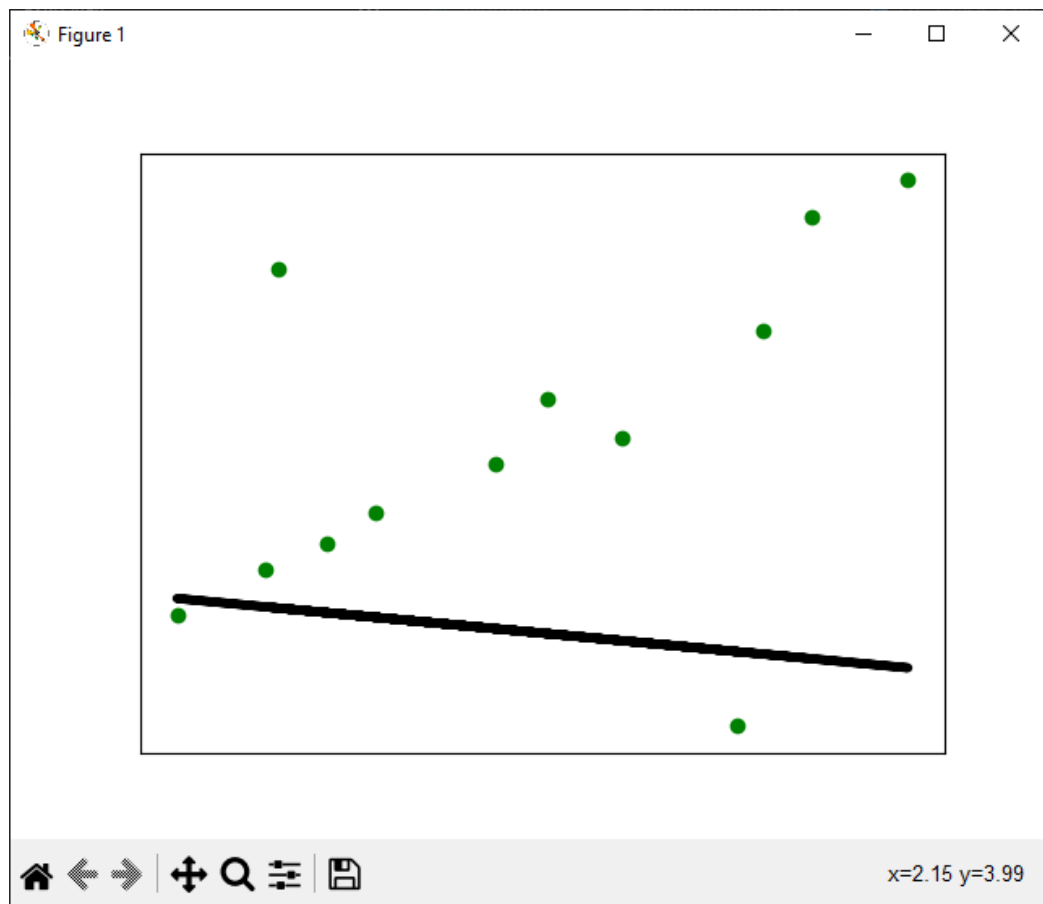
```

1 import pickle
2 import numpy as np
3 from sklearn import linear_model
4 import sklearn.metrics as sm
5 import matplotlib.pyplot as plt
6
7 # Вхідний файл, який містить дані
8 input_file = 'data_regr_3.txt'
9
10 # Завантаження даних
11 data = np.loadtxt(input_file, delimiter=',')
12 X, y = data[:, :-1], data[:, -1]
13
14 # Розбивка даних на навчальний та тестовий набори
15 num_training = int(0.8 * len(X))
16 num_test = len(X) - num_training
17
18 # Тренувальні дані
19 X_train, y_train = X[:num_training], y[:num_training]
20
21 # Тестові дані
22 X_test, y_test = X[num_training:], y[num_training:]
23
24 # Створення об'єкта лінійного регресора
25 regressor = linear_model.LinearRegression()
26
27 # Тренування моделі
28 regressor.fit(X_train, y_train)
29
30 # Прогнозування результату
31 y_test_pred = regressor.predict(X_test)
32
33 # Побудова графіка
34 plt.scatter(X_test, y_test, color = 'green')
35 plt.plot(X_test, y_test_pred, color='black', linewidth = 4)
36 plt.xticks(())
37 plt.yticks(())
38 plt.show()
39
40 # Обрахування метрик
41 print("Linear regressor performance:")
42 print("Mean absolute error =", round(sm.mean_absolute_error(y_test, y_test_pred), 2))
43 print("Mean squared error =", round(sm.mean_squared_error(y_test, y_test_pred), 2))
44 print("Median absolute error =", round(sm.median_absolute_error(y_test, y_test_pred), 2))
45 print("Explain variance score =", round(sm.explained_variance_score(y_test, y_test_pred), 2))
46 print("R2 score =", round(sm.r2_score(y_test, y_test_pred), 2))
47
48 # Файл для збереження моделі
49 output_model_file = 'model.pkl'
50
51 # Збереження моделі
52 with open(output_model_file, 'wb') as f:
53     pickle.dump(regressor, f)
54
55 # Завантаження моделі
56 with open(output_model_file, 'rb') as f:
57     regressor_model = pickle.load(f)
58
59 y_test_pred_new = regressor_model.predict(X_test)
60
61 print("\nNew mean absolute error =", round(sm.mean_absolute_error(y_test, y_test_pred_new), 2))
62

```

Рис 3. Код програми файлу LR_3_task_2

		Соболевський Д.А.			ДУ «Житомирська політехніка».20.121.18 – Лр3	Арк.
		Філіпов В.О.				5
Змн.	Арк.	№ докум.	Підпис	Дата		



```

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

New mean absolute error = 2.42

```

Рис 4. Результат виконання коду файлу LR_3_task_2

Завдання 2.3. Створення багатовимірного регресора.

```

1 import numpy as np
2 from sklearn import linear_model
3 import sklearn.metrics as sm
4 from sklearn.preprocessing import PolynomialFeatures
5
6 # Вхідний файл, який містить дані
7 input_file = 'data_multivar_regr.txt'
8
9 # Завантаження даних
10 data = np.loadtxt(input_file, delimiter=',')
11 X, y = data[:, :-1], data[:, -1]
12
13 # Розбивка даних на навчальний та тестовий набори
14 num_training = int(0.8 * len(X))
15 num_test = len(X) - num_training
16
17 # Тренувальні дані
18 X_train, y_train = X[:num_training], y[:num_training]
19
20 # Тестові дані
21 X_test, y_test = X[num_training:], y[num_training:]
22
23 # Створення об'єкта лінійного регресора
24 linear_regressor = linear_model.LinearRegression()
25
26 # Тренування моделі
27 linear_regressor.fit(X_train, y_train)
28
29 # Прогнозування результату
30 y_test_pred = linear_regressor.predict(X_test)
31
32 # Обрахування метрик
33 print("Linear Regressor performance:")
34 print("Mean absolute error =", round(sm.mean_absolute_error(y_test, y_test_pred), 2))
35 print("Mean squared error =", round(sm.mean_squared_error(y_test, y_test_pred), 2))
36 print("Median absolute error =", round(sm.median_absolute_error(y_test, y_test_pred), 2))
37 print("Explained variance score =", round(sm.explained_variance_score(y_test, y_test_pred), 2))
38 print("R2 score =", round(sm.r2_score(y_test, y_test_pred), 2))
39
40 # Поліноміальна регресія
41 polynomial = PolynomialFeatures(degree = 10)
42 X_train_transformed = polynomial.fit_transform(X_train)
43 datapoint = [[7.75, 6.35, 5.56]]
44 poly_datapoint = polynomial.fit_transform(datapoint)
45 poly_linear_model = linear_model.LinearRegression()
46
47 poly_linear_model.fit(X_train_transformed, y_train)
48
49 print("\nLinear regression:\n", linear_regressor.predict(datapoint))
50 print("\nPolynomial regression:\n", poly_linear_model.predict(poly_datapoint))
51

```

Рис 5. Код програми файлу LR_3_task_3

		Соболевський Д.А.			ДУ «Житомирська політехніка».20.121.18 – Лр3	Арк.
		Філіпов В.О.				7
Змн.	Арк.	№ докум.	Підпис	Дата		

```

Linear Regressor performance:
Mean absolute error = 3.58
Mean squared error = 20.31
Median absolute error = 2.99
Explained variance score = 0.86
R2 score = 0.86

Linear regression:
[36.05286276]

Polynomial regression:
[41.46678412]

```

Рис 6. Результат виконання коду файлу LR_3_task_3

Висновок: Порівнюючи з лінійним регресором, поліноміальний регресор більш кращий, тобто дозволяє показувати кращі результати.

Завдання 2.4. Регресія багатьох змінних.

		Соболевський Д.А			ДУ «Житомирська політехніка».20.121.18 – Лр3	Арк.
		Філіпов В.О.				8
Змн.	Арк.	№ докум.	Підпис	Дата		


```

1 import matplotlib.pyplot as plt
2 import numpy as np
3 from sklearn import datasets, linear_model
4 from sklearn.metrics import mean_squared_error, r2_score
5 from sklearn.metrics import mean_absolute_error
6 from sklearn.model_selection import train_test_split
7
8 diabetes = datasets.load_diabetes()
9 X = diabetes.data
10 y = diabetes.target
11 Xtrain, Xtest, ytrain, ytest = train_test_split(X, y, test_size = 0.5, random_state = 0)
12 regr = linear_model.LinearRegression()
13
14 regr.fit(Xtrain, ytrain)
15
16 ypred = regr.predict(Xtest)
17
18 # Обрахування метрик
19 print("regr.coef =", np.round(regr.coef_, 2))
20 print("regr.intercept =", round(regr.intercept_, 2))
21 print("R2 score =", round(r2_score(ytest, ypred), 2))
22 print("Mean absolute error =", round(mean_absolute_error(ytest, ypred), 2))
23 print("Mean squared error =", round(mean_squared_error(ytest, ypred), 2))
24
25 fig, ax = plt.subplots()
26
27 ax.scatter(ytest, ypred, edgecolors = (0, 0, 0))
28 ax.plot([y.min(), y.max()], [y.min(), y.max()], 'k--', lw = 4)
29 ax.set_xlabel('Виміряно')
30 ax.set_ylabel('Передбачено')
31 plt.show()
32

```

Рис 7. Код програми файлу LR_3_task_4

```

PS C:\ztu\штучний інтелект\lab3> python LR_3_task_4.py
regr.coef = [ -20.4  -265.89  564.65  325.56 -692.16  395.56   23.5   116.36  843.95
  12.72]
regr.intercept = 154.36
R2 score = 0.44
Mean absolute error = 44.8
Mean squared error = 3075.33
PS C:\ztu\штучний інтелект\lab3>

```

		Соболевський Д.А.			ДУ «Житомирська політехніка».20.121.18 – Лр3	Арк.
		Філіпов В.О.				9
Змн.	Арк.	№ докум.	Підпис	Дата		

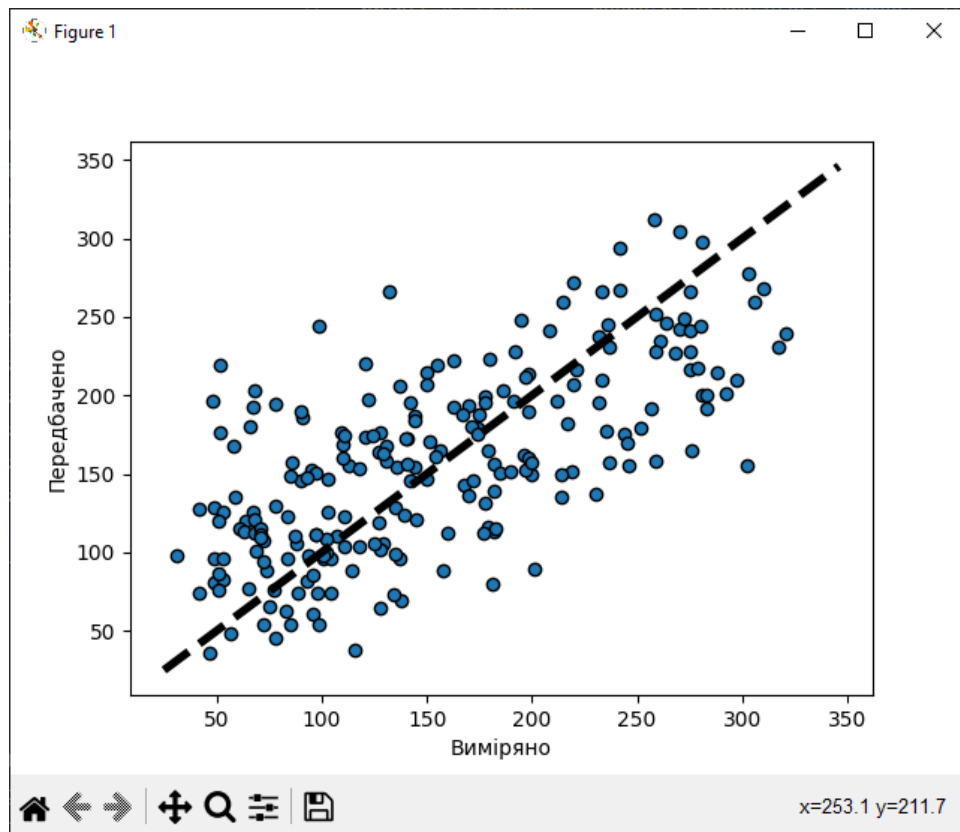


Рис 8. Результат виконання коду файлу LR_3_task_4

Завдання 2.5. Самостійна побудова регресії.

№ за списком	11	12	13	14	15	16	17	18	19	20
№ варіанту	1	2	3	4	5	6	7	8	9	10

```

1 import numpy as np
2 from matplotlib import pyplot as plt
3 from sklearn import linear_model
4 import sklearn.metrics as sm
5 from sklearn.preprocessing import PolynomialFeatures
6 # Генерація даних
7
8
9 m = 100
10 X = np.linspace(-3, 3, m)
11 y = 2 * np.sin(X) + np.random.uniform(-0.5, 0.5, m)
12 X = X.reshape(-1, 1)
13 y = y.reshape(-1, 1)
14
15
16 # Лінійна регресія
17 linear_regressor = linear_model.LinearRegression()
18 linear_regressor.fit(X, y)
19
20 # Поліноміальна регресія
21 polynomial = PolynomialFeatures(degree = 2, include_bias = False)
22 X_poly = polynomial.fit_transform(X)
23
24 polynomial.fit(X_poly, y)
25
26 poly_linear_model = linear_model.LinearRegression()
27
28 poly_linear_model.fit(X_poly, y)
29
30 y_pred = poly_linear_model.predict(X_poly)
31
32 print("\nr2: ", sm.r2_score(y, y_pred))
33
34 # Лінійна регресія
35 plt.scatter(X, y, color = 'red')
36 plt.plot(X, linear_regressor.predict(X), color = 'blue', linewidth = 1)
37 plt.title("Лінійна регресія")
38 plt.show()
39
40 # Поліноміальна регресія
41 plt.scatter(X, y, color='red')
42 plt.plot(X, y_pred, "+", color = 'blue', linewidth = 2)
43 plt.title("Поліноміальна регресія")
44 plt.show()
45

```

Рис 9. Код програми файлу LR_3_task_5

		Соболевський Д.А			ДУ «Житомирська політехніка».20.121.18 – Лр3	Арк.
		Філіпов В.О.				11
Змн.	Арк.	№ докум.	Підпис	Дата		

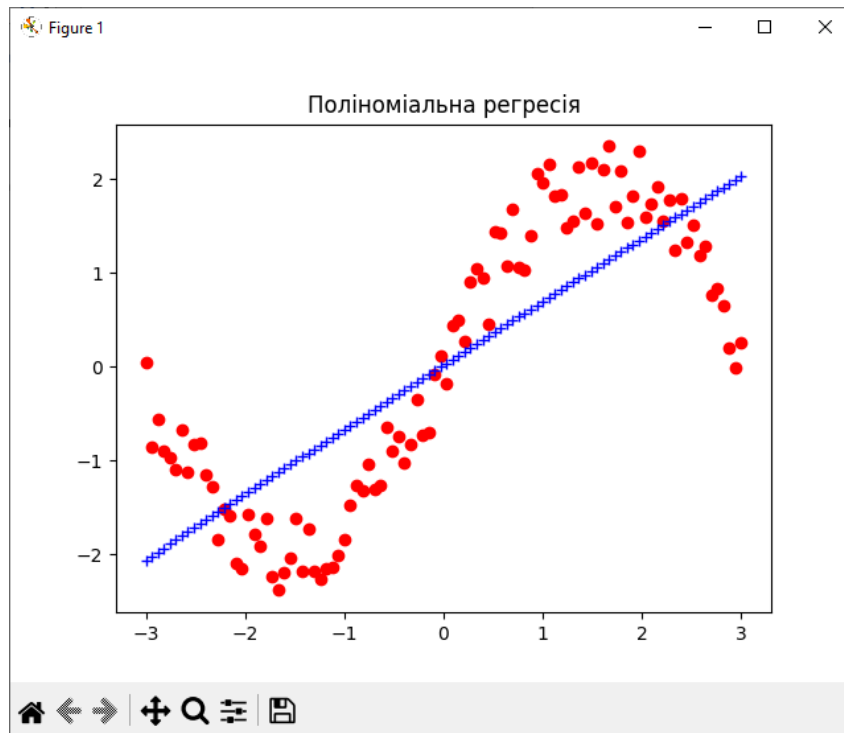
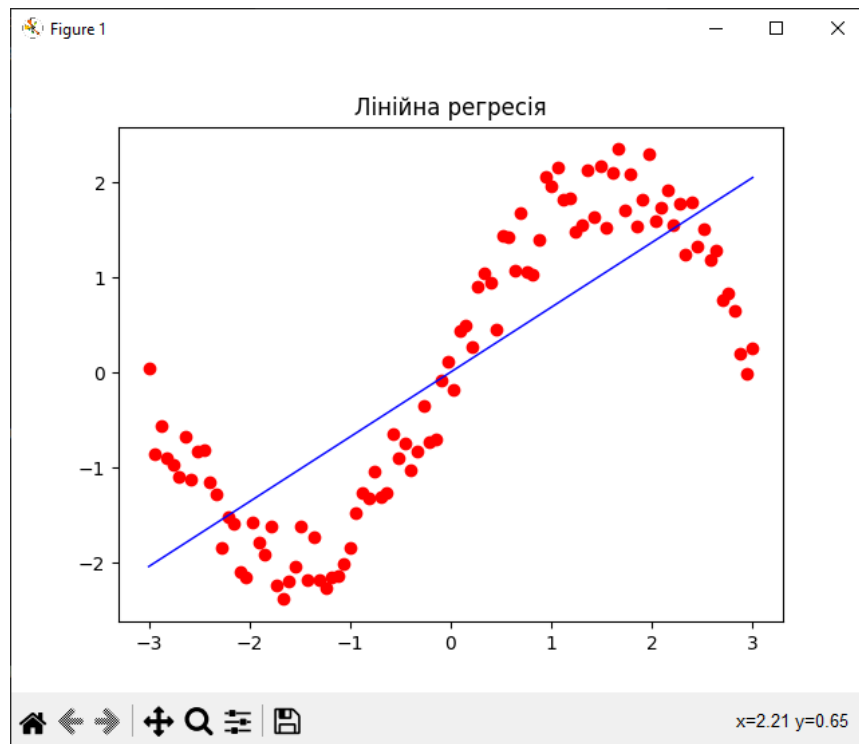


Рис 10. Результат виконання коду файлу LR_3_task_5

		Соболевський Д.А.			ДУ «Житомирська політехніка».20.121.18 – Лр3	Арк.
		Філіпов В.О.				12
Змн.	Арк.	№ докум.	Підпис	Дата		

Завдання 2.6. Побудова кривих навчання.

№ за списком	11	12	13	14	15	16	17	18	19	20
№ варіанту	1	2	3	4	5	6	7	8	9	10

```

1  import matplotlib.pyplot as plt
2  import numpy as np
3  from sklearn import linear_model
4  from sklearn.metrics import mean_squared_error
5  from sklearn.model_selection import train_test_split
6  from sklearn.preprocessing import PolynomialFeatures
7  from sklearn.pipeline import Pipeline
8
9  # Генерація даних
10 m = 100
11 X = np.linspace(-3, 3, m)
12 y = 2 * np.sin(X) + np.random.uniform(-0.5, 0.5, m)
13 X = X.reshape(-1, 1)
14 y = y.reshape(-1, 1)
15
16 def plot_learning_curves(model, X, y):
17     X_train, X_val, y_train, y_val = train_test_split(X, y, test_size = 0.2)
18     train_errors, val_errors = [], []
19
20     for m in range(1, len(X_train)):
21         model.fit(X_train[:m], y_train[:m])
22         y_train_predict = model.predict(X_train[:m])
23         y_val_predict = model.predict(X_val)
24         train_errors.append(mean_squared_error(y_train_predict, y_train[:m]))
25         val_errors.append(mean_squared_error(y_val_predict, y_val))
26
27     plt.plot(np.sqrt(train_errors), "r-+", linewidth = 2, label = 'train')
28     plt.plot(np.sqrt(val_errors), "b-", linewidth = 3, label = 'val')
29     plt.legend()
30     plt.show()
31
32 lin_reg = linear_model.LinearRegression()
33
34 plot_learning_curves(lin_reg, X, y)
35
36 polynomial_regression = Pipeline([
37     ("poly_features", PolynomialFeatures(degree = 10, include_bias = False)),
38     ("lin_reg", linear_model.LinearRegression())
39 ])
40
41 plot_learning_curves(polynomial_regression, X, y)
42
43

```

Рис 11. Код програми файлу LR_3_task_6

		Соболевський Д.А.			ДУ «Житомирська політехніка».20.121.18 – Лр3	Арк.
		Філіпов В.О.				13
Змн.	Арк.	№ докум.	Підпис	Дата		

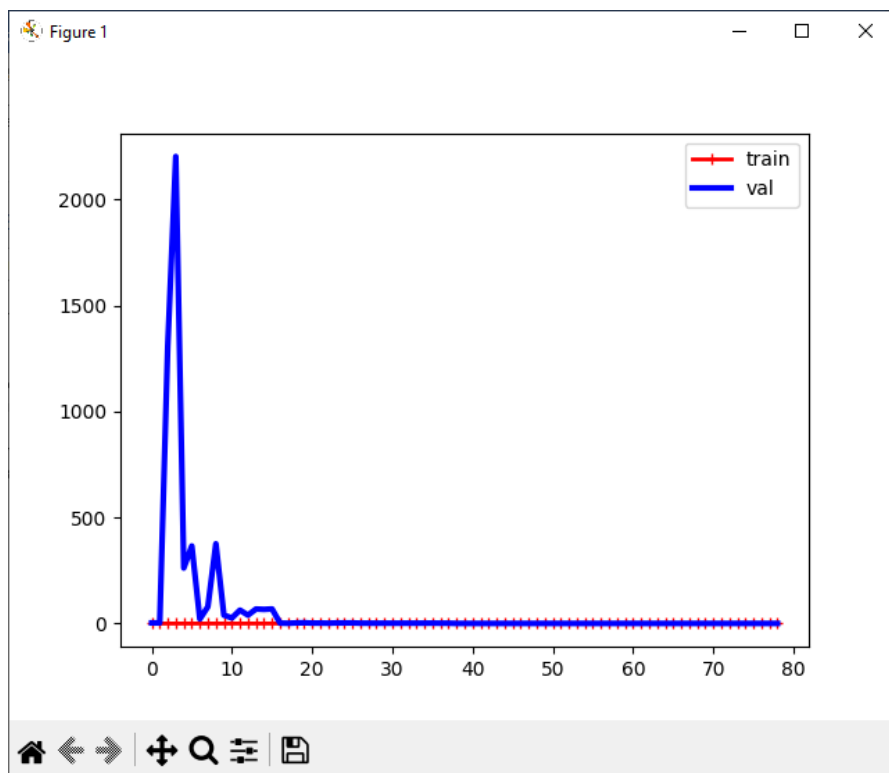
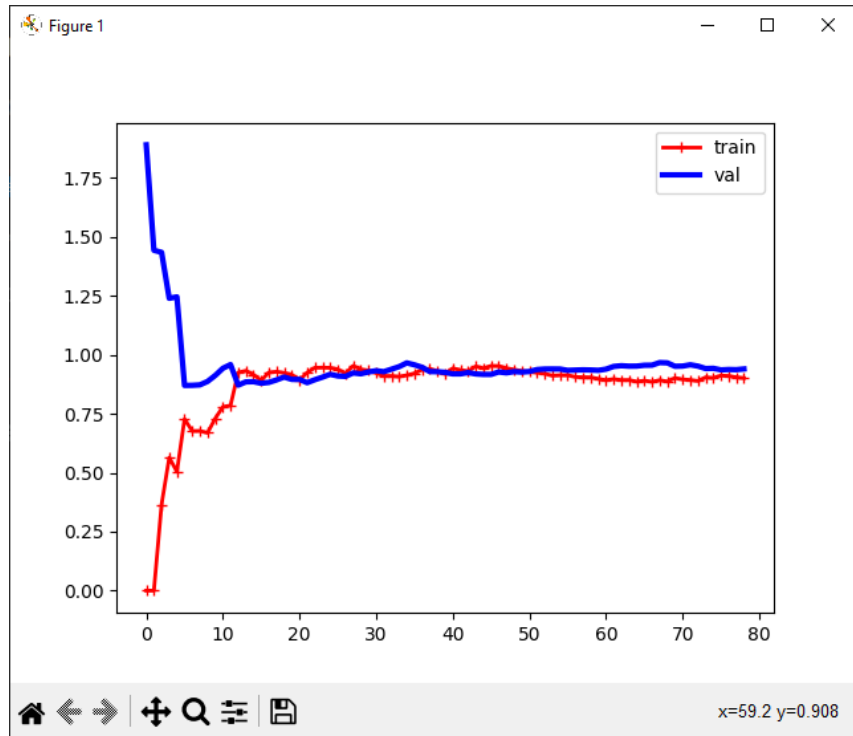


Рис 12. Результат виконання коду файлу LR_3_task_6

Висновок: для з'ясування ступеня складності необхідної моделі ми можемо використати криві навчання. Для досягнення успіху необхідно досягти компромісу

		Соболевський Д.А.			ДУ «Житомирська політехніка».20.121.18 – Лр3	Арк.
		Філіпов В.О.				14
Змн.	Арк.	№ докум.	Підпис	Дата		

між зміщенням та дисперсією. В нашому випадку найкращий результат показала модель 2 ступеня.

```

1 import numpy as np
2 import matplotlib.pyplot as plt
3 from sklearn.cluster import KMeans
4 from sklearn import metrics
5
6 # Завантаження вхідних даних
7 X = np.loadtxt('data_clustering.txt', delimiter=',')
8 num_clusters = 5
9
10 # Включення вхідних даних до графіка
11 plt.figure()
12 plt.scatter(X[:, 0], X[:, 1], marker='o', facecolors='none', edgecolors='black', s=80)
13
14 x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
15 y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
16
17 plt.title('Вхідні дані')
18 plt.xlim(x_min, x_max)
19 plt.ylim(y_min, y_max)
20 plt.xticks(())
21 plt.yticks(())
22
23 # Створення об'єкту KMeans
24 kmeans = KMeans(init='k-means++', n_clusters = num_clusters, n_init = 10)
25
26 # Навчання моделі кластеризації KMeans
27 kmeans.fit(X)
28
29 # Визначення кроку сітки
30 step_size = 0.01
31
32 # Відображення точок сітки
33 x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
34 y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
35 x_vals, y_vals = np.meshgrid(np.arange(x_min, x_max, step_size), np.arange(y_min, y_max, step_size))
36
37 # Передбачення вихідних міток для всіх точок сітки
38 output = kmeans.predict(np.c_[x_vals.ravel(), y_vals.ravel()])
39
40 # Графічне відображення областей та виділення їх кольором
41 output = output.reshape(x_vals.shape)
42
43 plt.figure()
44 plt.clf()

```

		Соболевський Д.А.			ДУ «Житомирська політехніка».20.121.18 – Лр3	Арк.
		Філіпов В.О.				15
Змн.	Арк.	№ докум.	Підпис	Дата		

```

1 plt.imshow(
2     output,
3     interpolation = 'nearest',
4     extent = (x_vals.min(), x_vals.max(), y_vals.min(), y_vals.max()),
5     cmap = plt.cm.Paired,
6     aspect = 'auto',
7     origin = 'lower'
8 )
9
10
11 # Відображення вхідних точок
12 plt.scatter(X[:, 0], X[:, 1], marker = 'o', facecolors = 'none', edgecolors = 'black', s = 80)
13
14 # Відображення центрів кластерів
15 cluster_centers = kmeans.cluster_centers_
16 plt.scatter(
17     cluster_centers[:, 0],
18     cluster_centers[:, 1],
19     marker = 'o',
20     s = 210,
21     linewidths = 4,
22     color = 'black',
23     zorder = 12,
24     facecolors = 'black'
25 )
26
27 x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
28 y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
29
30 plt.title('Межі кластерів')
31 plt.xlim(x_min, x_max)
32 plt.ylim(y_min, y_max)
33 plt.xticks(())
34 plt.yticks(())
35 plt.show()

```

Завдання 2.7. Кластеризація даних за допомогою методу к-середніх.

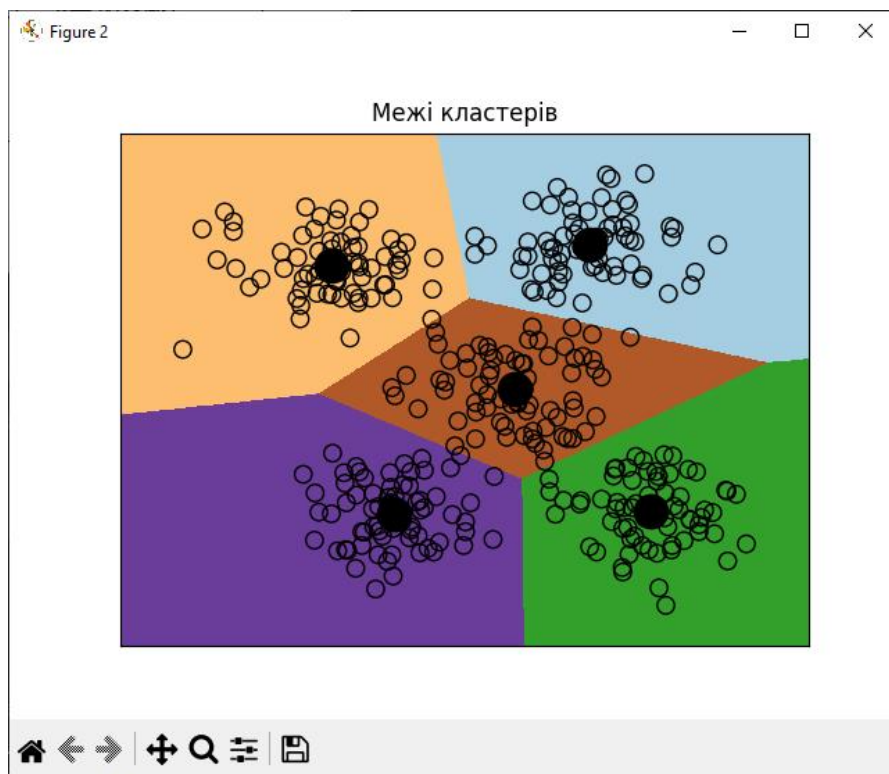


Рис 13. Код програми файлу LR_3_task_7

		Соболевський Д.А.		
		Філіпов В.О.		
Змн.	Арк.	№ докум.	Підпис	Дата

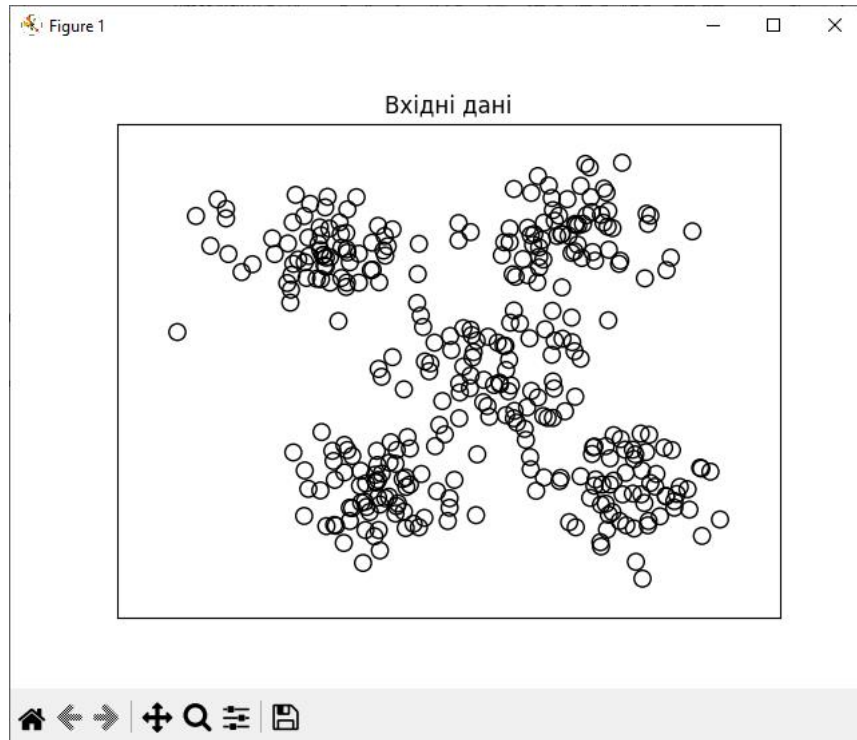


Рис 14. Результат виконання коду файлу LR_3_task_7

Висновок: метод k-середніх валідно працює, але за умови, відомої кількості кластерів.

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

		Соболевський Д.А			ДУ «Житомирська політехніка».20.121.18 – Лр3	Арк.
		Філіпов В.О.				17
Змн.	Арк.	№ докум.	Підпис	Дата		

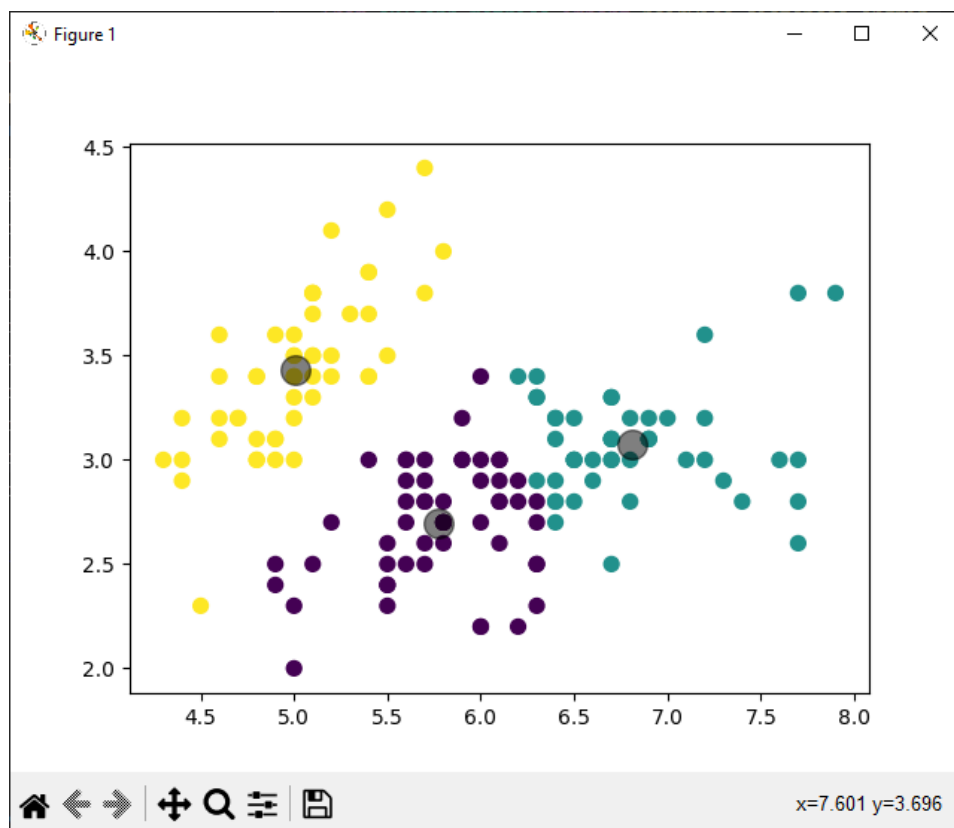
```

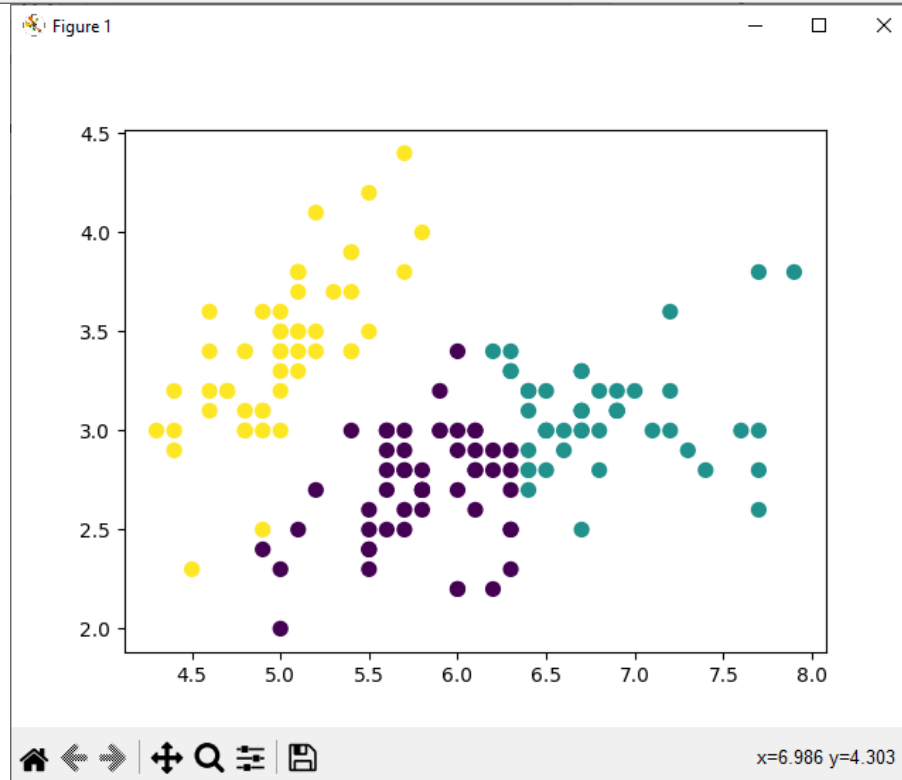
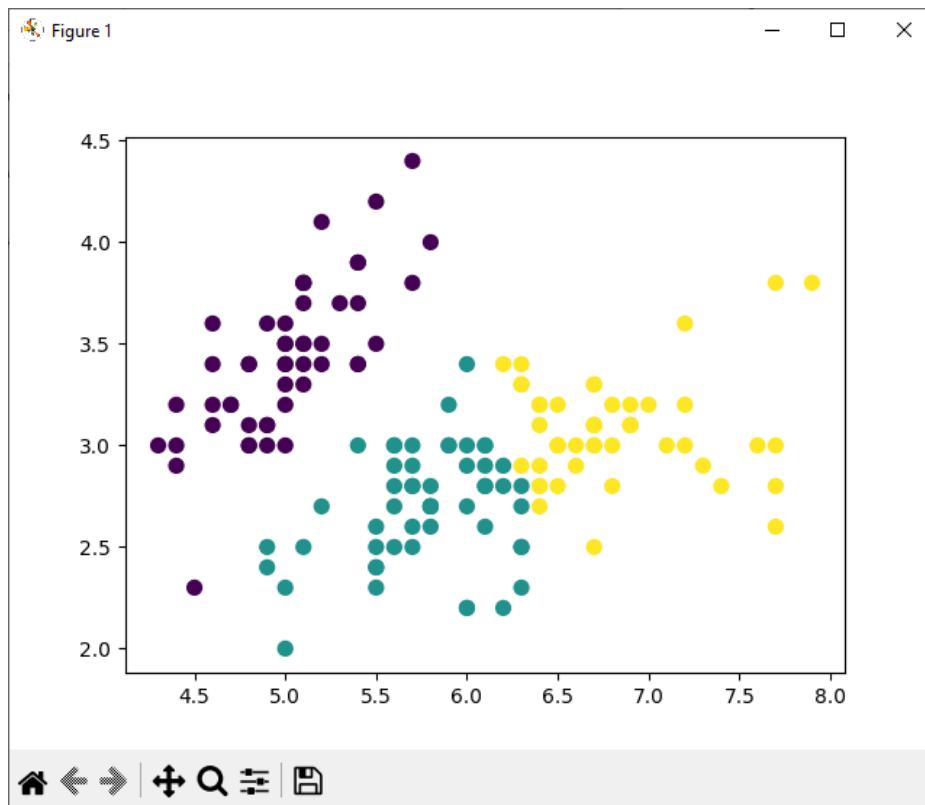
1 import matplotlib.pyplot as plt
2 from sklearn import datasets
3 from sklearn.cluster import KMeans
4 from sklearn.metrics import pairwise_distances_argmin
5 import numpy as np
6
7 # Отримуємо дані
8 iris = datasets.load_iris()
9 X = iris.data[:, :2]
10 Y = iris.target
11
12 # Визначаємо початкові кластери
13 kmeans = KMeans(
14     n_clusters = Y.max() + 1,
15     init = 'k-means++',
16     n_init = 10,
17     max_iter = 300,
18     tol = 0.0001,
19     verbose = 0,
20     random_state = None,
21     copy_X = True
22 )
23
24 kmeans.fit(X)
25
26 y_pred = kmeans.predict(X)
27
28 print("n_clusters: 3, n_init: 10, max_iter: 300, tol: 0.0001, verbose: 0, random_state: None, copy_X: True")
29 print(y_pred)
30
31 plt.figure()
32 plt.scatter(X[:, 0], X[:, 1], c = y_pred, s = 50, cmap = 'viridis')
33
34 centers = kmeans.cluster_centers_
35
36 plt.scatter(centers[:, 0], centers[:, 1], c = 'black', s = 200, alpha = 0.5)
37 plt.show()
38
39 def find_clusters(X, n_clusters, rseed = 2):
40     # Випадково обираємо кластери
41     rng = np.random.RandomState(rseed)
42     i = rng.permutation(X.shape[0])[:n_clusters]
43     centers = X[i]
44
45     while True:
46         # Оголошуємо label базуючись на найближчому центрі
47         labels = pairwise_distances_argmin(X, centers)
48
49         # Знаходимо нові центри з середини точок
50         new_centers = np.array([X[labels == i].mean(0) for i in range(n_clusters)])
51
52         # Перевірка збіжності
53         if np.all(centers == new_centers):
54             break
55
56         centers = new_centers
57
58     return centers, labels
59
60 print("using find_clusters():")
61 centers, labels = find_clusters(X, 3)
62
63 print("n_clusters: 3, rseed: 2")
64
65 plt.scatter(X[:, 0], X[:, 1], c = labels, s = 50, cmap = 'viridis')
66 plt.show()
67
68 centers, labels = find_clusters(X, 3, rseed = 0)
69
70 print("n_clusters: 3, rseed: 0")
71
72 plt.scatter(X[:, 0], X[:, 1], c = labels, s = 50, cmap = 'viridis')
73 plt.show()
74
75 labels = KMeans(3, random_state = 0).fit_predict(X)
76
77 print("n_clusters: 3, rseed: 0")
78
79 plt.scatter(X[:, 0], X[:, 1], c = labels, s = 50, cmap = 'viridis')
80 plt.show()
81
82

```

Рис 15. Код програми файлу LR_3_task_8

		Соболевський Д.А.			ДУ «Житомирська політехніка».20.121.18 – Лр3	Арк.
		Філіпов В.О.				18
Змн.	Арк.	№ докум.	Підпис	Дата		





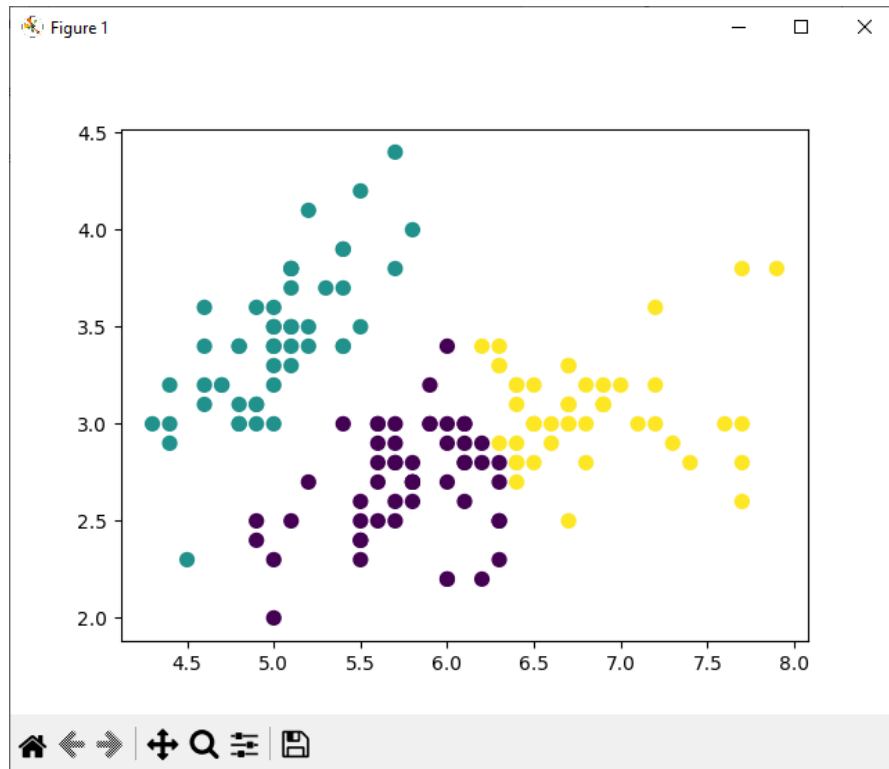


Рис 16. Результат виконання коду файлу LR_3_task_8

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

		Соболевський Д.А			ДУ «Житомирська політехніка».20.121.18 – Лр3	Арк.
		Філіпов В.О.				21
Змн.	Арк.	№ докум.	Підпис	Дата		

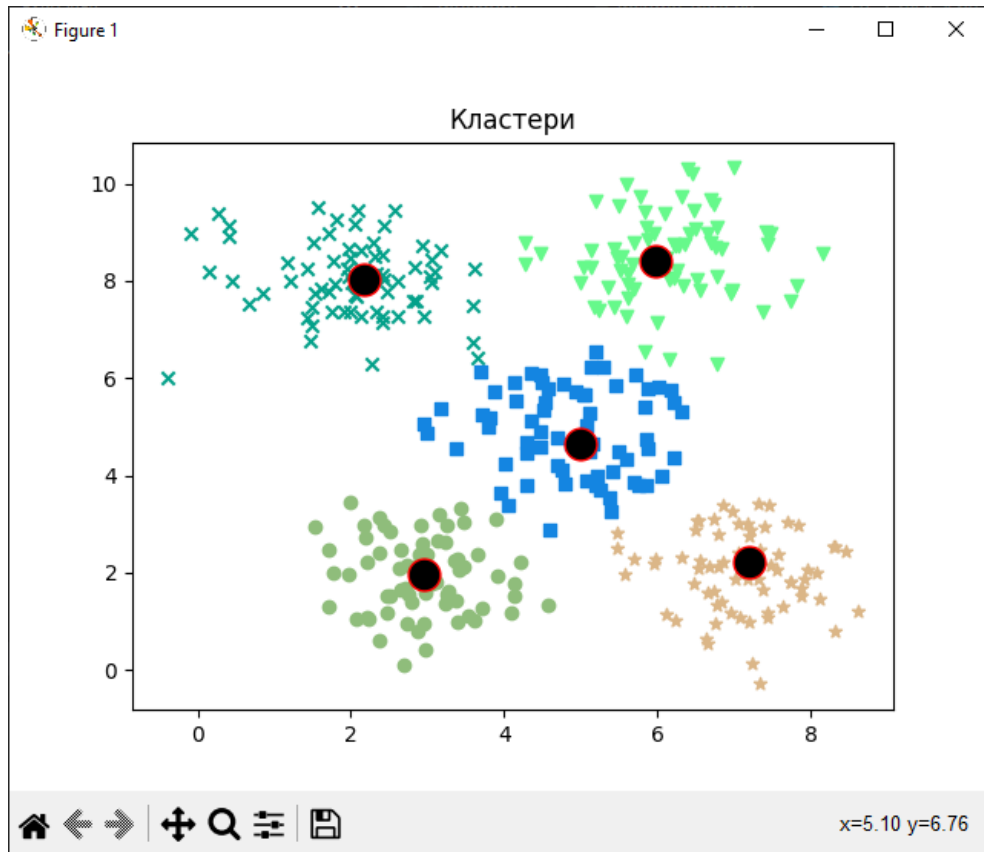
```

1 import numpy as np
2 import matplotlib.pyplot as plt
3 from sklearn.cluster import MeanShift, estimate_bandwidth
4 from itertools import cycle
5
6 # Завантаження даних
7 X = np.loadtxt('data_clustering.txt', delimiter = ',')
8
9 # Оцінка ширини вікна для X
10 bandwidth_X = estimate_bandwidth(X, quantile = 0.1, n_samples = len(X))
11
12 # Кластеризація даних методом зсуву середнього
13 meanshift_model = MeanShift(bandwidth=bandwidth_X, bin_seeding = True)
14 meanshift_model.fit(X)
15
16 # Витягування центрів кластерів
17 cluster_centers = meanshift_model.cluster_centers_
18
19 print('\nCenters of clusters:\n', cluster_centers)
20
21 # Оцінка кількості кластерів
22 labels = meanshift_model.labels_
23 num_clusters = len(np.unique(labels))
24
25 print("\nNumber of clusters in input data =", num_clusters)
26
27 # Відображення на графіку точок та центрів кластерів
28 plt.figure()
29
30 markers = 'o*xvs'
31
32 for i, marker in zip(range(num_clusters), markers):
33     # Відображення на графіку точок, що належать поточному кластеру
34     plt.scatter(X[labels == i, 0], X[labels == i, 1], marker = marker,
35               color = np.random.rand(3,))
36
37     # Відображення на графіку центру кластера
38     cluster_center = cluster_centers[i]
39
40     plt.plot(
41         cluster_center[0],
42         cluster_center[1],
43         marker = 'o',
44         markerfacecolor = 'black',
45         markeredgcolor = 'red',
46         markersize = 15
47     )
48
49 plt.title('Кластери')
50 plt.show()
51

```

Рис 17. Код програми файлу LR_3_task_9

		Соболевський Д.А			ДУ «Житомирська політехніка».20.121.18 – Лр3	Арк.
		Філіпов В.О.				22
Змн.	Арк.	№ докум.	Підпис	Дата		



```
PS C:\ztu\штучний інтелект\lab3> python LR_3_task_9.py

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

Number of clusters in input data = 5
```

Рис 18. Результат виконання коду файлу LR_3_task_9

Метод зсуву середнього – доволі валідний алгоритм, головною перевагою якого є непотрібність жодних припущень щодо базового розподілу даних, має змогу обробляти довільні простори функцій, проте важливу роль відіграє обрана ширина вікна (bandwidth).

Висновок: під час виконання лабораторної роботи я навчився: використовувати спеціалізовані бібліотеки і мову програмування Python, досліджувати методи регресії та неконтрольованої класифікації даних у машинному навчанні.

		Соболевський Д.А.			ДУ «Житомирська політехніка».20.121.18 – Лр3	Арк.
		Філіпов В.О.				23
Змн.	Арк.	№ докум.	Підпис	Дата		