

**ЛАБОРАТОРНА РОБОТА №2**  
**ПОРІВНЯННЯ МЕТОДІВ КЛАСИФІКАЦІЇ ДАНИХ**

**Мета роботи:** використовуючи спеціалізовані бібліотеки та мову програмування Python дослідити різні методи класифікації даних та навчитися їх порівнювати.

Хід роботи:

Завдання 2.1.

Класифікація за допомогою машин опорних векторів (SVM) Код програми:

					ДУ «Житомирська політехніка».20.121.18						
Змн.	Арк.	№ докум.	Підпис	Дата							
Розроб.		Соболевський Д.А						Літ.	Арк.	Аркушів	
Перевір.		Філіпов В.О.								1	19
Керівник								ФІКТ Гр. ІПЗк-20-1			
Н. контр.											
Зав. каф.											



```
1 import numpy as np
2 from sklearn import preprocessing
3 from sklearn.svm import LinearSVC
4 from sklearn.multiclass import OneVsOneClassifier
5 from sklearn.model_selection import train_test_split
6 from sklearn.model_selection import cross_val_score
7
8 input_file = "income_data.txt"
9 count_class1 = 0
10 count_class2 = 0
11 max_datapoints = 25000
12 X = []
13 Y = []
14
15
16 with open(input_file, "r") as f:
17     for line in f.readlines():
18         if count_class1 >= max_datapoints and count_class2 >= max_datapoints:
19             break
20         if '?' in line:
21             continue
22         data = line[:-1].split(',')
23         if data[-1] == '<=50K' and count_class1 < max_datapoints:
24             X.append(data)
25             count_class1 += 1
26         if data[-1] == '>50K' and count_class2 < max_datapoints:
27             X.append(data)
28             count_class2 += 1
29
30 X = np.array(X)
31 label_encoder = []
32 X_encoded = np.empty(X.shape)
33
34 for i, item in enumerate(X[0]):
35     if item.isdigit():
36         X_encoded[:, i] = X[:, i]
37     else:
38         label_encoder.append(preprocessing.LabelEncoder())
39         X_encoded[:, i] = label_encoder[-1].fit_transform(X[:, i])
40
41 X = X_encoded[:, :-1].astype(int)
42 Y = X_encoded[:, -1].astype(int)
43 scaller = preprocessing.MinMaxScaler(feature_range=(0, 1))
44 X = scaller.fit_transform(X)
```

		Соболевський Д.А.			ДУ «Житомирська політехніка».20.121.18 – Лр2	Арк.
		Філіпов В.О.				2
Змн.	Арк.	№ докум.	Підпис	Дата		

```

1 classifier = OneVsOneClassifier(LinearSVC(random_state = 0))
2
3 classifier.fit(X = X, y = Y)
4
5 X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size = 0.2, random_state = 5)
6 scaller = preprocessing.MinMaxScaler(feature_range = (0, 1))
7 X_train = scaller.fit_transform(X_train)
8 classifier.fit(X = X_train, y = y_train)
9 y_test_pred = classifier.predict(X_test)
10 f1 = cross_val_score(classifier, X, Y, scoring='f1_weighted', cv = 3)
11 accuracy_values = cross_val_score(classifier, X, Y, scoring='accuracy', cv = 3)
12
13 print("Accuracy: " + str(round(100 * accuracy_values.mean(), 2)) + "%")
14
15 precision_values = cross_val_score(classifier, X, Y, scoring='precision_weighted', cv = 3)
16
17 print("Precision: " + str(round(100 * precision_values.mean(), 2)) + "%")
18
19 recall_values = cross_val_score(classifier, X, Y, scoring='recall_weighted', cv = 3)
20
21 print("Recall: " + str(round(100 * recall_values.mean(), 2)) + "%")
22
23 f1_values = cross_val_score(classifier, X, Y, scoring='f1_weighted', cv = 3)
24
25 print("F1: " + str(round(100 * f1_values.mean(), 2)) + "%")
26 print("F1 score: " + str(round(100 * f1.mean(), 2)) + "%")
27
28 input_data = ['37', 'Private', '215646', 'HS-grad', '9', 'Never-married', 'Handlers-cleaners', 'Not-in-family', 'White', 'Male', '0', '0', '40', 'United-States']
29
30 input_data_encoded = np.array([-1] * len(input_data))
31 count = 0
32
33 for i, item in enumerate(input_data):
34     if item.isdigit():
35         input_data_encoded[i] = item
36     else:
37         input_data_encoded[i] = int(label_encoder[count].transform([item]))
38         count += 1
39
40 input_data_encoded = input_data_encoded.astype(int)
41 input_data_encoded = [input_data_encoded]
42
43 predicate_class = classifier.predict(input_data_encoded)
44 print(label_encoder[-1].inverse_transform(predicate_class)[0])

```

Рис 2.1 Код файлу LR\_2\_task\_1.py

Результат:

```

PS C:\ztu\штучний інтелект\lab2> python .\LR_2_task_1.py
Accuracy: 81.95%
Precision: 80.94%
Recall: 81.95%
F1: 80.13%
F1 score: 80.13%
>50K
PS C:\ztu\штучний інтелект\lab2>

```

Рис 2.2 Результат програми файлу LR\_2\_task\_1.py

## Завдання 2.2.

Порівняння якості класифікаторів SVM з нелінійними ядрами

Код програми:

		Соболевський Д.А.			ДУ «Житомирська політехніка».20.121.18 – Лр2	Арк.
		Філіпов В.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		3



```
1 import numpy as np
2 from sklearn import preprocessing
3 from sklearn.svm import SVC
4 from sklearn.multiclass import OneVsOneClassifier
5 from sklearn.model_selection import train_test_split
6 from sklearn.model_selection import cross_val_score
7
8 input_file = "income_data.txt"
9 count_class1 = 0
10 count_class2 = 0
11 max_datapoints = 25000
12 X = []
13 Y = []
14
15 with open(input_file, "r") as f:
16     for line in f.readlines():
17         if count_class1 >= max_datapoints and count_class2 >= max_datapoints:
18             break
19         if '?' in line:
20             continue
21         data = line[:-1].split(',')
22         if data[-1] == '<=50K' and count_class1 < max_datapoints:
23             X.append(data)
24             count_class1 += 1
25         if data[-1] == '>50K' and count_class2 < max_datapoints:
26             X.append(data)
27             count_class2 += 1
28
29 X = np.array(X)
30 label_encoder = []
31 X_encoded = np.empty(X.shape)
32
33 for i, item in enumerate(X[0]):
34     if item.isdigit():
35         X_encoded[:, i] = X[:, i]
36     else:
37         label_encoder.append(preprocessing.LabelEncoder())
38         X_encoded[:, i] = label_encoder[-1].fit_transform(X[:, i])
39
40 X = X_encoded[:, :-1].astype(int)
41 Y = X_encoded[:, -1].astype(int)
42 scaller = preprocessing.MinMaxScaler(feature_range=(0, 1))
43 X = scaller.fit_transform(X)
44 classifier = SVC(kernel='poly', degree = 8)
45
46 classifier.fit(X = X, y = Y)
47 X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size = 0.2, random_state = 5)
48 scaller = preprocessing.MinMaxScaler(feature_range = (0, 1))
49 X_train = scaller.fit_transform(X_train)
```

		Соболевський Д.А.			ДУ «Житомирська політехніка».20.121.18 – Лр2	Арк.
		Філіпов В.О.				4
Змн.	Арк.	№ докум.	Підпис	Дата		

```

1 classifier.fit(X = X_train, y = y_train)
2
3 y_test_pred = classifier.predict(X_test)
4 f1 = cross_val_score(classifier, X, Y, scoring = "f1_weighted", cv = 3)
5 accuracy_values = cross_val_score(classifier, X, Y, scoring = 'accuracy', cv = 3)
6
7 print("Accuracy: " + str(round(100 * accuracy_values.mean(), 2)) + "%")
8
9 precision_values = cross_val_score(classifier, X, Y, scoring='precision_weighted', cv = 3)
10
11 print("Precision: " + str(round(100 * precision_values.mean(), 2)) + "%")
12
13 recall_values = cross_val_score(classifier, X, Y, scoring = 'recall_weighted', cv = 3)
14
15 print("Recall: " + str(round(100 * recall_values.mean(), 2)) + "%")
16
17 f1_values = cross_val_score(classifier, X, Y, scoring = 'f1_weighted', cv = 3)
18
19 print("F1: " + str(round(100 * f1_values.mean(), 2)) + "%")
20 print("F1 score: " + str(round(100 * f1.mean(), 2)) + "%")
21
22 input_data = ['37', 'Private', '215646', 'HS-grad', '9', 'Never-married', 'Handlers-cleaners', 'Not-in-family', 'White', 'Male', '0', '0', '40', 'United-States']
23 input_data_encoded = np.array([-1] * len(input_data))
24 count = 0
25
26 for i, item in enumerate(input_data):
27     if item.isdigit():
28         input_data_encoded[i] = item
29     else:
30         input_data_encoded[i] = int(label_encoder[count].transform([item]))
31         count += 1
32
33 input_data_encoded = input_data_encoded.astype(int)
34 input_data_encoded = [input_data_encoded]
35 predicate_class = classifier.predict(input_data_encoded)
36
37 print(label_encoder[-1].inverse_transform(predicate_class)[0])

```

Рис 2.3 Код файлу LR\_2\_task\_2\_1.py

		Соболевський Д.А			ДУ «Житомирська політехніка».20.121.18 – Лр2	Арк.
		Філіпов В.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		5



```
1 import numpy as np
2 from sklearn import preprocessing
3 from sklearn.svm import SVC
4 from sklearn.multiclass import OneVsOneClassifier
5 from sklearn.model_selection import train_test_split
6 from sklearn.model_selection import cross_val_score
7
8 input_file = "income_data.txt"
9 count_class1 = 0
10 count_class2 = 0
11 max_datapoints = 25000
12 X = []
13 Y = []
14
15 with open(input_file, "r") as f:
16     for line in f.readlines():
17         if count_class1 >= max_datapoints and count_class2 >= max_datapoints:
18             break
19         if '?' in line:
20             continue
21         data = line[:-1].split(',')
22         if data[-1] == '<=50K' and count_class1 < max_datapoints:
23             X.append(data)
24             count_class1 += 1
25         if data[-1] == '>50K' and count_class2 < max_datapoints:
26             X.append(data)
27             count_class2 += 1
28
29 X = np.array(X)
30 label_encoder = []
31 X_encoded = np.empty(X.shape)
32
33 for i, item in enumerate(X[0]):
34     if item.isdigit():
35         X_encoded[:, i] = X[:, i]
36     else:
37         label_encoder.append(preprocessing.LabelEncoder())
38         X_encoded[:, i] = label_encoder[-1].fit_transform(X[:, i])
39
40 X = X_encoded[:, :-1].astype(int)
41 Y = X_encoded[:, -1].astype(int)
42 scaller = preprocessing.MinMaxScaler(feature_range = (0, 1))
43 X = scaller.fit_transform(X)
44 classifier = SVC(kernel = 'rbf')
45
46 classifier.fit(X = X, y = Y)
```

		Соболевський Д.А.			ДУ «Житомирська політехніка».20.121.18 – Лр2	Арк.
		Філіпов В.О.				6
Змн.	Арк.	№ докум.	Підпис	Дата		

```

1 X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size = 0.2, random_state = 5)
2 scaler = preprocessing.MinMaxScaler(feature_range = (0, 1))
3 X_train = scaler.fit_transform(X_train)
4
5 classifier.fit(X = X_train, y = y_train)
6
7 y_test_pred = classifier.predict(X_test)
8 f1 = cross_val_score(classifier, X, Y, scoring = "f1_weighted", cv = 3)
9 accuracy_values = cross_val_score(classifier, X, Y, scoring = 'accuracy', cv = 3)
10
11 print("Accuracy: " + str(round(100 * accuracy_values.mean(), 2)) + "%")
12
13 precision_values = cross_val_score(classifier, X, Y, scoring='precision_weighted', cv=3)
14
15 print("Precision: " + str(round(100 * precision_values.mean(), 2)) + "%")
16
17 recall_values = cross_val_score(classifier, X, Y, scoring='recall_weighted', cv=3)
18
19 print("Recall: " + str(round(100 * recall_values.mean(), 2)) + "%")
20
21 f1_values = cross_val_score(classifier, X, Y, scoring='f1_weighted', cv=3)
22
23 print("F1: " + str(round(100 * f1_values.mean(), 2)) + "%")
24 print("F1 score: " + str(round(100 * f1.mean(), 2)) + "%")
25
26 input_data = ['37', 'Private', '215646', 'HS-grad', '9', 'Never-married', 'Handlers-cleaners', 'Not-in-family', 'White', 'Male', '0', '0', '40', 'United-States']
27 input_data_encoded = np.array([-1] * len(input_data))
28 count = 0
29
30 for i, item in enumerate(input_data):
31     if item.isdigit():
32         input_data_encoded[i] = item
33     else:
34         input_data_encoded[i] = int(label_encoder[count].transform([item]))
35         count += 1
36
37 input_data_encoded = input_data_encoded.astype(int)
38 input_data_encoded = [input_data_encoded]
39 predicate_class = classifier.predict(input_data_encoded)
40 print(label_encoder[-1].inverse_transform(predicate_class)[0])

```

Рис 2.4 Код файлу LR\_2\_task\_2\_2.py

		Соболевський Д.А.			ДУ «Житомирська політехніка».20.121.18 – Лр2	Арк.
		Філіпов В.О.				7
Змн.	Арк.	№ докум.	Підпис	Дата		

```

1 import numpy as np
2 from sklearn import preprocessing
3 from sklearn.svm import SVC
4 from sklearn.multiclass import OneVsOneClassifier
5 from sklearn.model_selection import train_test_split
6 from sklearn.model_selection import cross_val_score
7
8 input_file = "income_data.txt"
9 count_class1 = 0
10 count_class2 = 0
11 max_datapoints = 25000
12 X = []
13 Y = []
14
15 with open(input_file, "r") as f:
16     for line in f.readlines():
17         if count_class1 >= max_datapoints and count_class2 >= max_datapoints:
18             break
19         if '?' in line:
20             continue
21         data = line[:-1].split(',')
22         if data[-1] == '<=50K' and count_class1 < max_datapoints:
23             X.append(data)
24             count_class1 += 1
25         if data[-1] == '>50K' and count_class2 < max_datapoints:
26             X.append(data)
27             count_class2 += 1
28
29 X = np.array(X)
30 label_encoder = []
31 X_encoded = np.empty(X.shape)
32
33 for i, item in enumerate(X[0]):
34     if item.isdigit():
35         X_encoded[:, i] = X[:, i]
36     else:
37         label_encoder.append(preprocessing.LabelEncoder())
38         X_encoded[:, i] = label_encoder[-1].fit_transform(X[:, i])
39
40 X = X_encoded[:, :-1].astype(int)
41 Y = X_encoded[:, -1].astype(int)
42 scaller = preprocessing.MinMaxScaler(feature_range = (0, 1))
43 X = scaller.fit_transform(X)
44 classifier = SVC(kernel = 'sigmoid')
45
46 classifier.fit(X = X, y = Y)

```



```

1 X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size = 0.2, random_state = 5)
2 scaller = preprocessing.MinMaxScaler(feature_range = (0, 1))
3 X_train = scaller.fit_transform(X_train)
4 classifier.fit(X = X_train, y = y_train)
5 y_test_pred = classifier.predict(X_test)
6 f1 = cross_val_score(classifier, X, Y, scoring = "f1_weighted", cv = 3)
7 accuracy_values = cross_val_score(classifier, X, Y, scoring='accuracy', cv = 3)
8
9 print("Accuracy: " + str(round(100 * accuracy_values.mean(), 2)) + "%")
10
11 precision_values = cross_val_score(classifier, X, Y, scoring='precision_weighted', cv = 3)
12
13 print("Precision: " + str(round(100 * precision_values.mean(), 2)) + "%")
14
15 recall_values = cross_val_score(classifier, X, Y, scoring='recall_weighted', cv = 3)
16
17 print("Recall: " + str(round(100 * recall_values.mean(), 2)) + "%")
18
19 f1_values = cross_val_score(classifier, X, Y, scoring='f1_weighted', cv = 3)
20
21 print("F1: " + str(round(100 * f1_values.mean(), 2)) + "%")
22 print("F1 score: " + str(round(100 * f1.mean(), 2)) + "%")
23
24 input_data = ['37', 'Private', '215646', 'HS-grad', '9', 'Never-married', 'Handlers-cleaners', 'Not-in-family', 'White', 'Male', '0', '0', '40', 'United-States']
25 input_data_encoded = np.array([-1] * len(input_data))
26 count = 0
27
28 for i, item in enumerate(input_data):
29     if item.isdigit():
30         input_data_encoded[i] = item
31     else:
32         input_data_encoded[i] = int(label_encoder[count].transform([item]))
33         count += 1
34
35 input_data_encoded = input_data_encoded.astype(int)
36 input_data_encoded = [input_data_encoded]
37 predicate_class = classifier.predict(input_data_encoded)
38
39 print(label_encoder[-1].inverse_transform(predicate_class)[0])

```

Рис 2.5 Код файлу LR\_2\_task\_2\_3.py

Результат:

```

Accuracy: 73.15%
Precision: 70.37%
Recall: 73.15%
F1: 71.28%
F1 score: 71.28%
<=50K

```

Рис 2.6 Результат Поліноміального ядра

		Соболевський Д.А.			ДУ «Житомирська політехніка».20.121.18 – Лр2	Арк.
		Філіпов В.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		9

```

Accuracy: 78.97%
Precision: 82.07%
Recall: 78.97%
F1: 72.92%
F1 score: 72.92%
<=50K

```

Рис 2.7 Результат гаусового ядра

```

Accuracy: 73.11%
Precision: 59.02%
Recall: 73.11%
F1: 64.06%
F1 score: 64.06%
<=50K

```

Рис 2.8 Результат сигмоїдального ядра

Висновок: в даній ситуації краще за всього справляється RBF, має кращу точність та швидкість. Сигмоїдне ядро не так добре, так як відстає по швидкості.

### Завдання 2.3

Порівняння якості класифікаторів на прикладі класифікації сортів ірисів

Код програми:

		Соболевський Д.А			ДУ «Житомирська політехніка».20.121.18 – Лр2	Арк.
		Філіпов В.О.				10
Змн.	Арк.	№ докум.	Підпис	Дата		

```

1 from sklearn.datasets import load_iris
2 import numpy as np
3 from pandas import read_csv
4 from pandas.plotting import scatter_matrix
5 from matplotlib import pyplot
6 from sklearn.model_selection import train_test_split
7 from sklearn.model_selection import cross_val_score
8 from sklearn.model_selection import StratifiedKFold
9 from sklearn.metrics import classification_report
10 from sklearn.metrics import confusion_matrix
11 from sklearn.metrics import accuracy_score
12 from sklearn.linear_model import LogisticRegression
13 from sklearn.tree import DecisionTreeClassifier
14 from sklearn.neighbors import KNeighborsClassifier
15 from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
16 from sklearn.naive_bayes import GaussianNB
17 from sklearn.svm import SVC
18
19 iris_dataset = load_iris()
20
21 print("Ключі iris dataset : \n{}".format(iris_dataset.keys()))
22 print(iris_dataset["DESCR"][:193] + "\n...")
23 print("Назви відповідей: {}".format(iris_dataset["target_names"]))
24 print("Назви ознак: \n{}".format(iris_dataset["feature_names"]))
25 print("Тип масиву data: {}".format(type(iris_dataset["data"])))
26 print("Форма масиву data: {}".format(iris_dataset["data"].shape))
27 print("Тип масиву target: {}".format(type(iris_dataset['target'])))
28 print("Відповіді:\n{}".format(iris_dataset['target']))
29
30 url = "https://raw.githubusercontent.com/jbrownlee/Datasets/master/iris.csv"
31 names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'class']
32 dataset = read_csv(url, names = names)
33
34 # shape
35 print(dataset.shape)
36
37 # Зріз даних head
38 print(dataset.head(20))
39
40 # Статистичні зведення методом describe
41 print(dataset.describe())
42
43 # Розподіл за атрибутом class
44 print(dataset.groupby('class').size())
45
46 # Діаграма розмаху
47 dataset.plot(kind = 'box', subplots = True, layout = (2, 2), sharex = False, sharey = False)
48 pyplot.show()
49
50 # Гістограма розподілу атрибутів датасета
51 dataset.hist()
52 pyplot.show()

```

		Соболевський Д.А.			ДУ «Житомирська політехніка».20.121.18 – Лр2	Арк.
		Філіпов В.О.				11
Змн.	Арк.	№ докум.	Підпис	Дата		

```

1  # Матриця діаграм розсіювання
2  scatter_matrix(dataset)
3  pyplot.show()
4
5  # Розділення датасету на навчальну та контрольну вибірки
6  array = dataset.values
7
8  # Вибір перших 4-х стовпців
9  X = array[:, 0:4]
10
11 # Вибір 5-го стовпця
12 y = array[:, 4]
13
14 # Разделение X и y на обучающую и контрольную выборки
15 X_train, X_validation, Y_train, Y_validation = train_test_split(X, y, test_size = 0.20, random_state = 1)
16 models = []
17 models.append(('LR', LogisticRegression(solver = 'liblinear', multi_class = 'ovr')))
18 models.append(('LDA', LinearDiscriminantAnalysis()))
19 models.append(('KNN', KNeighborsClassifier()))
20 models.append(('CART', DecisionTreeClassifier()))
21 models.append(('NB', GaussianNB()))
22 models.append(('SVM', SVC(gamma = 'auto')))
23 results = []
24 names = []
25
26 for name, model in models:
27     kfold = StratifiedKFold(n_splits = 10, random_state = 1, shuffle = True)
28     cv_results = cross_val_score(model, X_train, Y_train, cv = kfold, scoring = 'accuracy')
29
30     results.append(cv_results)
31     names.append(name)
32     print('%s: %f (%f)' % (name, cv_results.mean(), cv_results.std()))
33
34 # Порівняння алгоритмів
35 pyplot.boxplot(results, labels = names)
36 pyplot.title('Algorithm Comparison')
37 pyplot.show()
38
39 # Створюємо прогноз на контрольній вибірці
40 model = SVC(gamma = 'auto')
41
42 model.fit(X_train, Y_train)
43
44 predictions = model.predict(X_validation)
45
46 # Оцінюємо прогноз
47 print(accuracy_score(Y_validation, predictions))
48 print(confusion_matrix(Y_validation, predictions))
49 print(classification_report(Y_validation, predictions))
50
51 X_new = np.array([[5, 2.9, 1, 0.2]])
52
53 for name, model in models:
54     model.fit(X_train, Y_train)
55
56     prediction = model.predict(X_new)
57     print("Прогноз: {}".format(prediction))
58     print(accuracy_score(Y_validation, predictions))
59     print(confusion_matrix(Y_validation, predictions))
60     print(classification_report(Y_validation, predictions))

```

Рис 2.9 Код файлу LR\_2\_task\_3.py

Результат:

		Соболевський Д.А.			ДУ «Житомирська політехніка».20.121.18 – Лр2	Арк.
		Філіпов В.О.				12
Змн.	Арк.	№ докум.	Підпис	Дата		

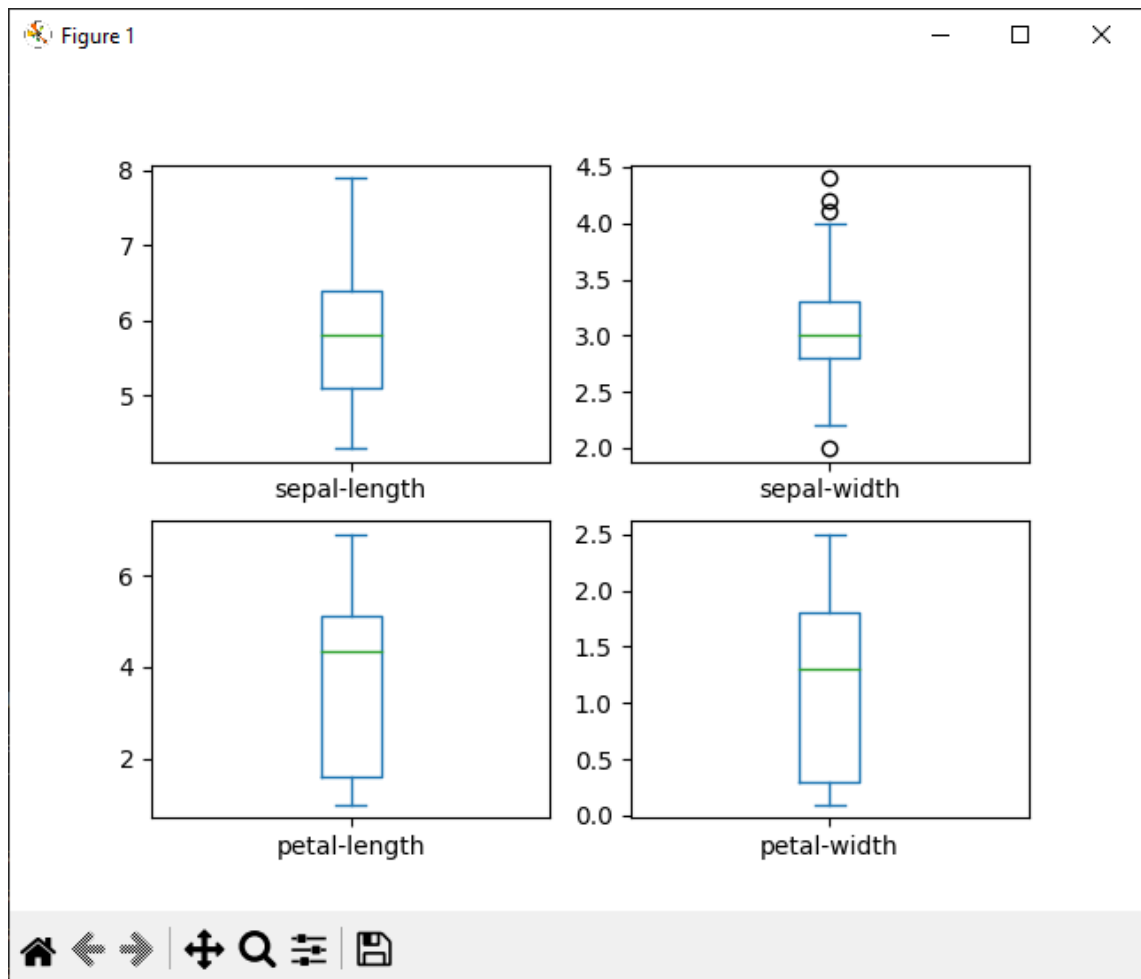


Рис. 2.10 Результат діаграми розмаху

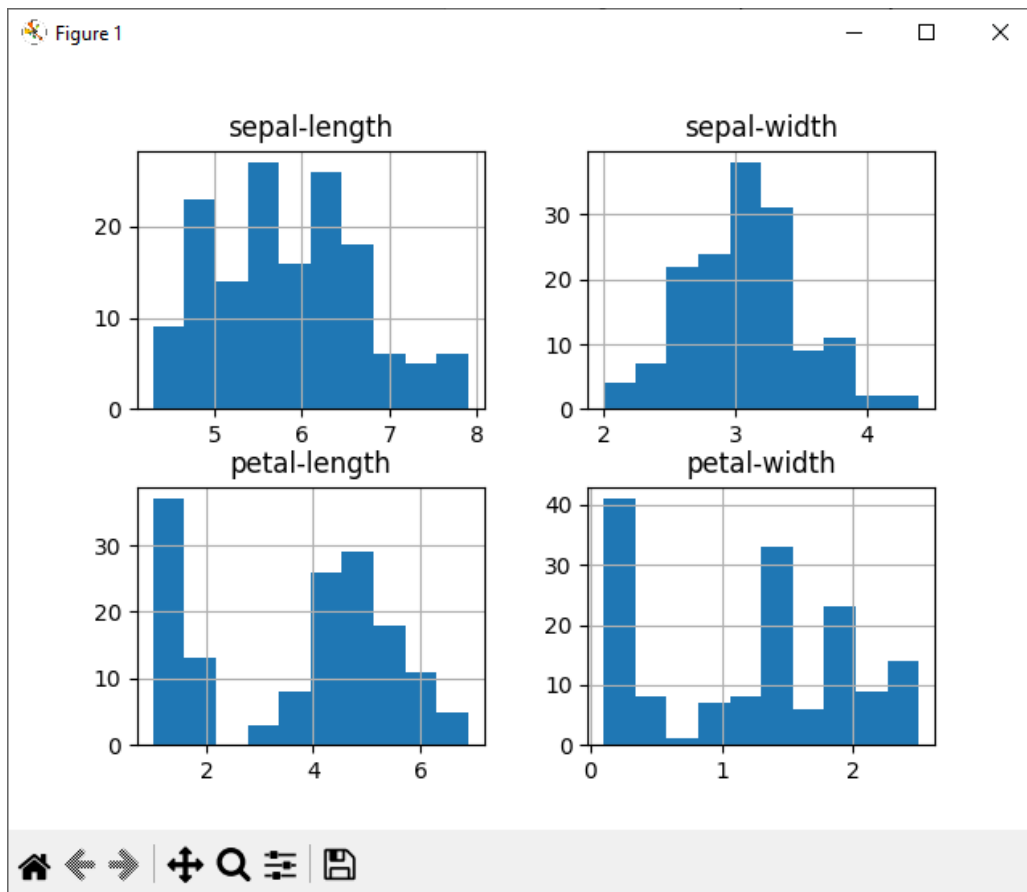


Рис. 2.11 Гістограма розподілу атрибутів

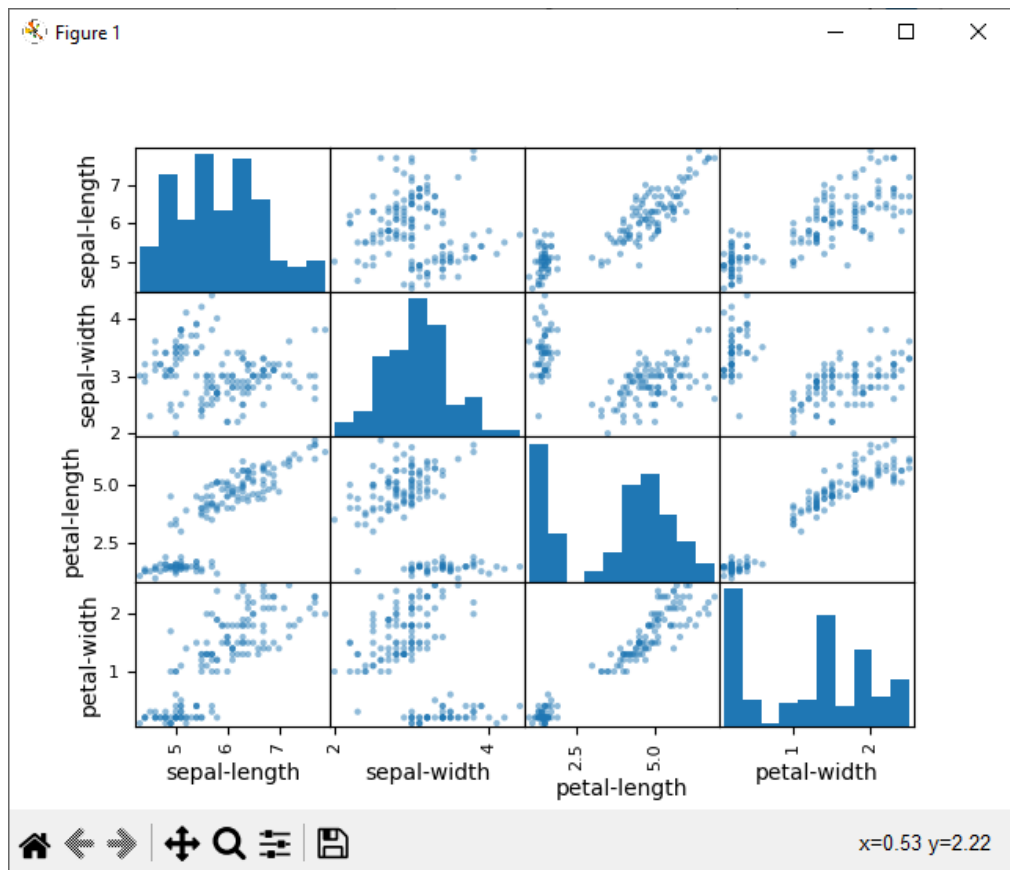


Рис. 2.12 Матриця діаграми розсіювання

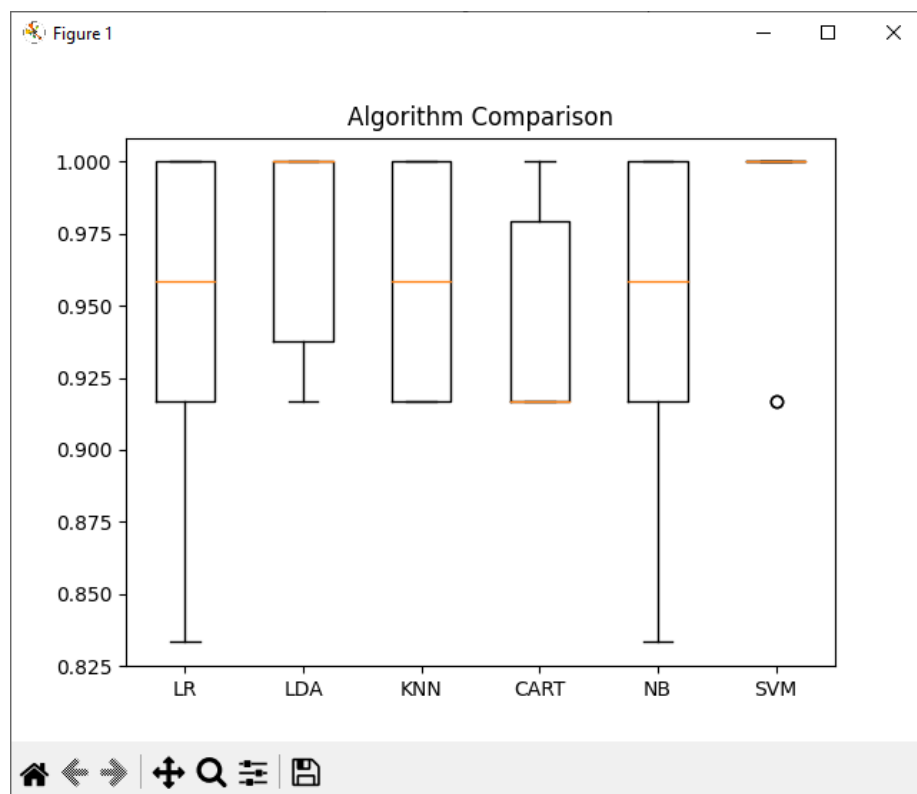


Рис. 2.13 Рисунок порівняння алгоритмів





## Порівняння якості класифікаторів для набору даних завдання 2.1

Код програми:

```
1 import numpy as np
2 from sklearn import preprocessing
3 from sklearn.svm import SVC
4 from sklearn.model_selection import train_test_split
5 from sklearn.model_selection import cross_val_score
6 from sklearn.linear_model import LogisticRegression
7 from sklearn.tree import DecisionTreeClassifier
8 from sklearn.neighbors import KNeighborsClassifier
9 from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
10 from sklearn.naive_bayes import GaussianNB
11
12 input_file = "income_data.txt"
13 count_class1 = 0
14 count_class2 = 0
15 max_datapoints = 25000
16 X = []
17 Y = []
18
19 with open(input_file, "r") as f:
20     for line in f.readlines():
21         if count_class1 >= max_datapoints and count_class2 >= max_datapoints:
22             break
23         if '?' in line:
24             continue
25         data = line[:-1].split(',')
26         if data[-1] == '<=50K' and count_class1 < max_datapoints:
27             X.append(data)
28             count_class1 += 1
29         if data[-1] == '>50K' and count_class2 < max_datapoints:
30             X.append(data)
31             count_class2 += 1
32
33 X = np.array(X)
34 label_encoder = []
35 X_encoded = np.empty(X.shape)
36
37 for i, item in enumerate(X[0]):
38     if item.isdigit():
39         X_encoded[:, i] = X[:, i]
40     else:
41         label_encoder.append(preprocessing.LabelEncoder())
42         X_encoded[:, i] = label_encoder[-1].fit_transform(X[:, i])
43
44 X = X_encoded[:, :-1].astype(int)
45 Y = X_encoded[:, -1].astype(int)
46 scaller = preprocessing.MinMaxScaler(feature_range=(0, 1))
47 X = scaller.fit_transform(X)
48
49 classifier = SVC(gamma = 'auto')
```

		Соболевський Д.А.			ДУ «Житомирська політехніка».20.121.18 – Лр2	Арк.
		Філіпов В.О.				17
Змн.	Арк.	№ докум.	Підпис	Дата		

```

1 classifier.fit(X = X, y = Y)
2 X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size = 0.2, random_state = 5)
3 scaler = preprocessing.MinMaxScaler(feature_range = (0, 1))
4 X_train = scaler.fit_transform(X_train)
5
6 classifier.fit(X = X_train, y = y_train)
7
8 y_test_pred = classifier.predict(X_test)
9 f1 = cross_val_score(classifier, X, Y, scoring = 'f1_weighted', cv = 3)
10 accuracy_values = cross_val_score(classifier, X, Y, scoring = 'accuracy', cv = 3)
11
12 print("Accuracy: " + str(round(100 * accuracy_values.mean(), 2)) + "%")
13
14 precision_values = cross_val_score(classifier, X, Y, scoring = 'precision_weighted', cv = 3)
15
16 print("Precision: " + str(round(100 * precision_values.mean(), 2)) + "%")
17
18 recall_values = cross_val_score(classifier, X, Y, scoring = 'recall_weighted', cv = 3)
19
20 print("Recall: " + str(round(100 * recall_values.mean(), 2)) + "%")
21
22 f1_values = cross_val_score(classifier, X, Y, scoring = 'f1_weighted', cv = 3)
23
24 print("F1: " + str(round(100 * f1_values.mean(), 2)) + "%")
25 print("F1 score: " + str(round(100 * f1.mean(), 2)) + "%")
26
27 input_data = ['37', 'Private', '215646', 'HS-grad', '9', 'Never-married', 'Handlers-cleaners', 'Not-in-family', 'White', 'Male', '0', '0', '40', 'United-States']
28 input_data_encoded = np.array([-1] * len(input_data))
29 count = 0
30
31 for i, item in enumerate(input_data):
32     if item.isdigit():
33         input_data_encoded[i] = item
34     else:
35         input_data_encoded[i] = int(label_encoder[count].transform([item]))
36         count += 1
37
38 input_data_encoded = input_data_encoded.astype(int)
39 input_data_encoded = [input_data_encoded]
40 predicate_class = classifier.predict(input_data_encoded)
41
42 print(label_encoder[-1].inverse_transform(predicate_class)[0])

```

Рис 2.15 Код файлу LR\_2\_task\_4.py

Результат:

```

Accuracy: 78.62%
Precision: 78.57%
Recall: 78.62%
F1: 73.37%
F1 score: 73.37%
>50K

```

Рис.2.16 Точність класифікатора LR

```

Accuracy: 80.34%
Precision: 78.95%
Recall: 80.34%
F1: 78.27%
F1 score: 78.27%
>50K

```

Рис. 2.17 Точність класифікатора LDA

		Соболевський Д.А			ДУ «Житомирська політехніка».20.121.18 – Лр2	Арк.
		Філіпов В.О.				18
Змн.	Арк.	№ докум.	Підпис	Дата		

```
Accuracy: 74.15%
Precision: 71.36%
Recall: 74.15%
F1: 71.69%
F1 score: 71.69%
<=50K
```

Рис. 2.18 Точність класифікатора KNN

```
Accuracy: 76.88%
Precision: 79.51%
Recall: 78.97%
F1: 81.24%
F1 score: 80.53%
>50K
```

Рис. 2.19 Точність класифікатора CART

```
Accuracy: 80.35%
Precision: 79.08%
Recall: 80.35%
F1: 79.27%
F1 score: 79.27%
>50K
```

Рис. 2.20 Точність класифікатора NB

```
Accuracy: 75.17%
C:\Users\Діма\AppData\Local\Temp\
ed and being set to 0.0 in la
_warn_prf(average, modifier
C:\Users\Діма\AppData\Local\Temp\
ed and being set to 0.0 in la
_warn_prf(average, modifier
C:\Users\Діма\AppData\Local\Temp\
ed and being set to 0.0 in la
_warn_prf(average, modifier
Precision: 56.51%
Recall: 75.17%
F1: 64.52%
F1 score: 64.52%
<=50K
```

Рис. 2.21 Точність класифікатора SVM

Завдання 2.5.

Класифікація даних лінійним класифікатором Ridge

Код програми:

		Соболевський Д.А.			ДУ «Житомирська політехніка».20.121.18 – Лр2	Арк.
		Філіпов В.О.				20
Змн.	Арк.	№ докум.	Підпис	Дата		

```

1 import numpy as np
2 import seaborn as sns
3 from sklearn.datasets import load_iris
4 from sklearn.linear_model import RidgeClassifier
5 from sklearn.model_selection import train_test_split
6 from sklearn.metrics import confusion_matrix
7 from io import BytesIO
8 import matplotlib.pyplot as plt
9 from sklearn import metrics
10
11 sns.set()
12
13 iris = load_iris()
14 X, y = iris.data, iris.target
15
16 Xtrain, Xtest, ytrain, ytest = train_test_split(X, y, test_size = 0.3, random_state = 0)
17 clf = RidgeClassifier(tol=1e-2, solver="sag")
18
19 clf.fit(Xtrain, ytrain)
20
21 ypred = clf.predict(Xtest)
22
23 print('Accuracy:', np.round(metrics.accuracy_score(ytest, ypred), 4))
24 print('Precision:', np.round(metrics.precision_score(ytest, ypred, average = 'weighted'), 4))
25 print('Recall:', np.round(metrics.recall_score(ytest, ypred, average = 'weighted'), 4))
26 print('F1 Score:', np.round(metrics.f1_score(ytest, ypred, average = 'weighted'), 4))
27 print('Cohen Kappa Score:', np.round(metrics.cohen_kappa_score(ytest, ypred), 4))
28 print('Matthews Corrcoef:', np.round(metrics.matthews_corrcoef(ytest, ypred), 4))
29 print('\t\tClassification Report:\n', metrics.classification_report(ytest, ypred))
30
31 mat = confusion_matrix(ytest, ypred)
32
33 sns.heatmap(mat.T, square = True, annot = True, fmt = 'd', cbar = False)
34 plt.xlabel('true label')
35 plt.ylabel('predicted label')
36 plt.savefig("Confusion.jpg")
37
38 f = BytesIO()
39
40 plt.savefig(f, format = "svg")

```

Рис 2.22 Код файлу LR\_2\_task\_5.py

Результат:

		Соболевський Д.А			ДУ «Житомирська політехніка».20.121.18 – Лр2	Арк.
		Філіпов В.О.				21
Змн.	Арк.	№ докум.	Підпис	Дата		

```

Accuracy: 0.7556
Precision: 0.8333
Recall: 0.7556
F1 Score: 0.7503
Cohen Kappa Score: 0.6431
Matthews Corrcoef: 0.6831

Classification Report:

```

	precision	recall	f1-score	support
0	1.00	1.00	1.00	16
1	0.44	0.89	0.59	9
2	0.91	0.50	0.65	20
accuracy			0.76	45
macro avg	0.78	0.80	0.75	45
weighted avg	0.85	0.76	0.76	45

Рис. 2.23 Результат програми

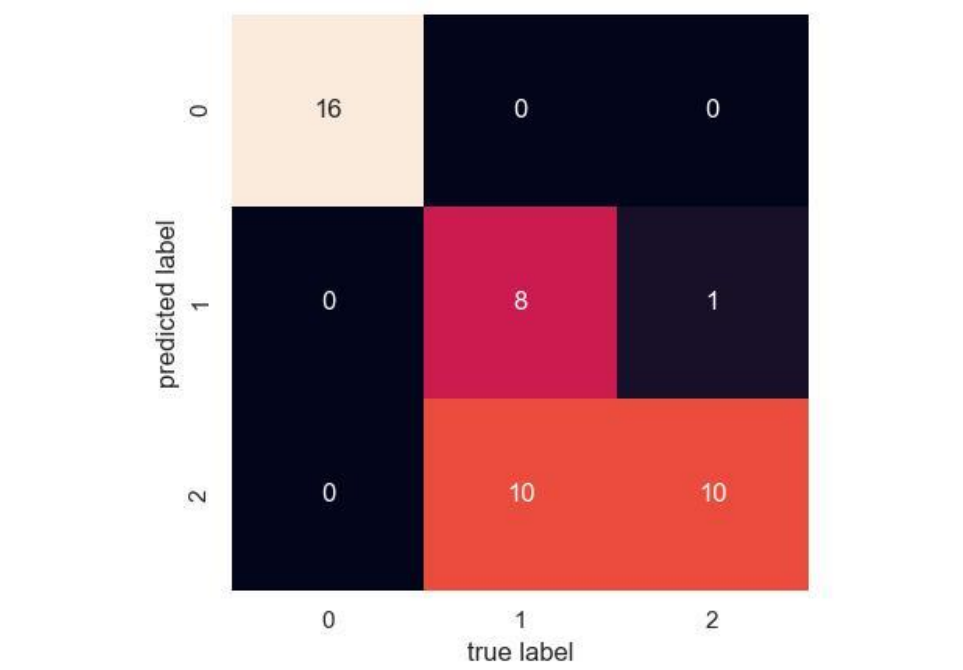


Рис. 2.24 Матриця невідповідності

Висновок: Матриця невідповідності – це таблиця особливого компонування, що дає можливість унаочнювати продуктивність алгоритму. Кожен з рядків цієї матриці представляє зразки прогнозованого класу, тоді як кожен зі стовпців представляє зразки справжнього класу (або навпаки).

Коефіцієнт каппа Коена статистика, яка використовується для вимірювання надійності між експертами для якісних пунктів.

Кореляції Метьюза – використовується в машинному навчанні, як міра якості бінарних мультикласних класифікацій.

Висновок: в ході виконання лабораторної роботи використовуючи спеціалізовані бібліотеки та мову програмування Python дослідив різні методи класифікації даних та навчився їх порівнювати.

		Соболевський Д.А			ДУ «Житомирська політехніка».20.121.18 – Лр2	Арк.
		Філіпов В.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		23