Липецкий государственный технический университет

Факультет автоматизации и информатики

Кафедра автоматизированных систем управления

ЛАБОРАТОРНАЯ РАБОТА №2

по дисциплине «Прикладные интеллектуальные системы и экспертные системы»

Бинарная классификация

Студент Крутских А.Ю.

Группа М-ИАП-22

Руководитель Кургасов В.В.

Липецк 2022 г.

Цель работы

Получить практические навыки решения задачи бинарной классификации данных в среде Jupiter Notebook. Научиться загружать данные, обучать классификаторы и проводить классификацию. Научиться оценивать точность полученных моделей.

Задание кафедры

1) В среде Jupiter Notebook создать новый ноутбук (Notebook);

2) Импортировать необходимые для работы библиотеки и модули;

3) Загрузить данные в соответствие с вариантом;

4) Вывести первые 15 элементов выборки (координаты точек и метки класса);

5) Отобразить на графике сгенерированную выборку. Объекты разных классов должны иметь разные цвета;

6) Разбить данные на обучающую (train) и тестовую (test) выборки в пропорции 75% - 25% соответственно;

7) Отобразить на графике обучающую и тестовую выборки. Объекты разных классов должны иметь разные цвета;

8) Реализовать модели классификаторов, обучить их на обучающем множестве. Применить модели на тестовой выборке, вывести результаты классификации:

- Истинные и предсказанные метки классов
- Матрицу ошибок (confusion matrix)
- Значения полноты, точности, f1-меры и аккуратности
- Значение площади под кривой ошибок (AUC ROC)
- Отобразить на графике область принятия решений по каждому классу В качестве методов классификации использовать:

a) Метод к-ближайших соседей (n_neighbors = {1, 3, 5, 9})

b) Наивный байесовский метод

c) Случайный лес (n_estimators = {5, 10, 15, 20, 50})

9) По каждому пункту работы занести в отчет программный код и результат вывода;

10) По результатам п.8 занести в отчет таблицу с результатами классификации всеми методами и выводы о наиболее подходящем методе классификации ваших данных.

Ход работы

Вариант 5.

Вариант	**5**
Вид классов	moons
Random_state	41
cluster_std	-
noise	0.25
Centers	-

Для всех вариантов, использующих для генерации make_classification, дополнительные параметры: n_features=2, n_redundant=0, n_informative=1, n_clusters_per_class=1.

1) в среде Jupiter Notebook создать новый ноутбук (Notebook);

2) импортировать необходимые для работы библиотеки и модули;

3) загрузить данные в соответствие с вариантом;

4) вывести первые 15 элементов выборки (координаты точек и метки класса);

5) отобразить на графике сгенерированную выборку. Объекты разных классов должны иметь разные цвета;

6) разбить данные на обучающую (train) и тестовую (test) выборки в пропорции 75% - 25% соответственно;

7) отобразить на графике обучающую и тестовую выборки. Объекты разных классов должны иметь разные цвета;

8) реализовать модели классификаторов, обучить их на обучающем множестве. Применить модели на тестовой выборке, вывести результаты классификации:

- Истинные и предсказанные метки классов
- Матрицу ошибок (confusion matrix)
- Значения полноты, точности, f1-меры и аккуратности

- Значение площади под кривой ошибок (AUC ROC)
- Отобразить на графике область принятия решений по каждому классу В качестве методов классификации использовать:

a) Метод к-ближайших соседей (n_neighbors = {1, 3, 5, 9})

b) Наивный байесовский метод

c) Случайный лес (n_estimators = {5, 10, 15, 20, 50})

9) по каждому пункту работы занести в отчет программный код и результат вывода;

10) по результатам п.8 занести в отчет таблицу с результатами классификации всеми методами и выводы о наиболее подходящем методе классификации ваших данных.

Ход работы

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import make_moons
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
from sklearn.metrics import accuracy_score
from sklearn.metrics import roc_auc_score
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import GaussianNB
```

Рисунок 1 - Импортируем библиотеки

Вид класса	random_state	cluster_std	noise	centers
blobs	41	3	-	2

```
x, y = make_moons(noise=0.25, random_state=41)
```

```
print("Координаты x y:\n", x[:15])
print("Метки класса: ", y[:15])
```

```
Координаты x y:
 [[-1.53415216 -0.08217918]
 [ 2.16440543 -0.2734159 ]
 [-0.58527459  0.85376957]
 [ 1.35195628  0.38893433]
 [ 1.46479345 -0.25172895]
 [-0.64362199  1.06018572]
 [ 0.40847955  0.87854153]
 [ 1.16692891 -0.03600687]
 [-0.70389555  0.90891989]
 [ 1.69466082 -0.36203457]
 [ 0.60290958 -0.69829682]
 [ 0.16150261 -0.15701266]
 [-0.07098503 -0.50788692]
 [ 0.6133523   0.04294663]
 [ 0.43329615  0.04721913]]
Метки класса:  [0 1 0 0 1 0 0 0 0 1 1 1 1 0 1]
```

Рисунок 2 – Генерируем выборку

```
plt.scatter(x[:,0], x[:,1], c=y)
```

```
<matplotlib.collections.PathCollection at 0x1e6b6091030>
```

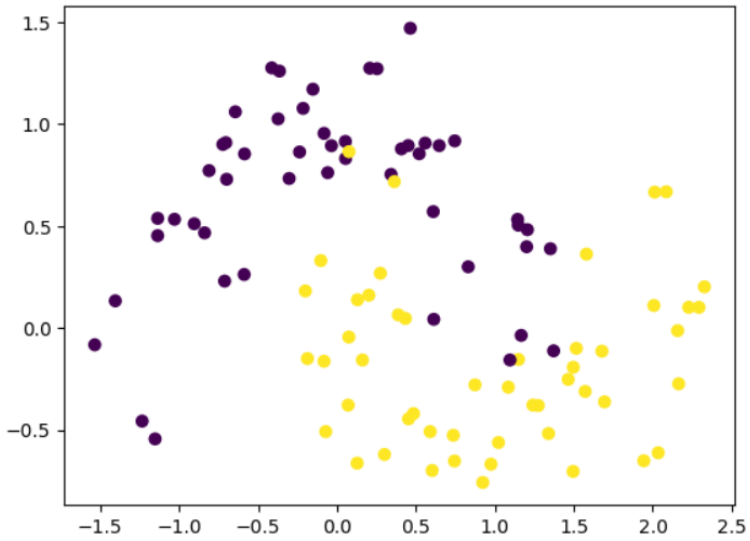

Рисунок 3 – График сгенерированной выборки

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.25, random_state=1)
```

Рисунок 4 - Разбиваем данные на обучающую (train) и тестовую (test) выборки в пропорции 75% - 25% соответственно

```
plt.scatter(x_train[:,0], x_train[:,1], c=y_train)
```

```
<matplotlib.collections.PathCollection at 0x1e6b8416980>
```

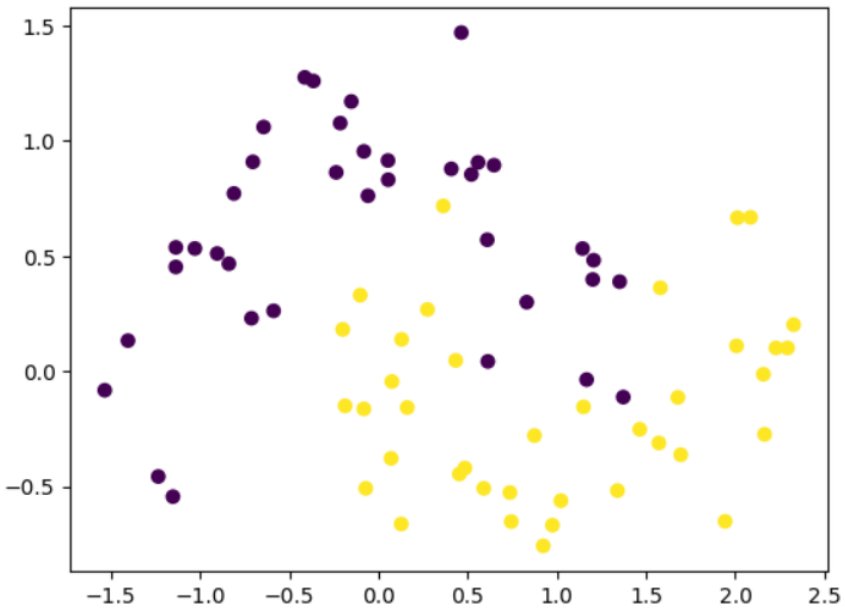

Рисунок 5 - График обучающей выборки

```
In [8]: plt.scatter(x_test[:,0], x_test[:,1], c=y_test)
```

Рисунок 6 - График тестовой выборки

```python
for i in [1, 3, 5, 9]:
    knn = KNeighborsClassifier(n_neighbors=i, metric='euclidean')
    knn.fit(x_train, y_train)
    prediction = knn.predict(x_test)
    print("n_neightbors = ", i)
    print_classification_metrics(knn, x, y, prediction, y_test)
```

```
n_neightbors =  1
Предсказанные и истинные значения
[1 1 1 0 0 0 1 1 1 0 0 1 0 1 1 0 1 0 0 1 0 0 0 1 0]
[1 1 1 1 0 0 1 1 0 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]
Матрица ошибок
[[11  2]
 [ 2 10]]
Точность классификации:  0.84
Значения полноты, точности, f1-меры и аккуратности
              precision    recall  f1-score   support

           0       0.85      0.85      0.85        13
           1       0.83      0.83      0.83        12

    accuracy                           0.84        25
   macro avg       0.84      0.84      0.84        25
weighted avg       0.84      0.84      0.84        25

Значение площади под кривой ошибок (AUC ROC)
```

```
Значение площади под кривой ошибок (AUC ROC)
0.8397435897435898
Область принятия решений
```

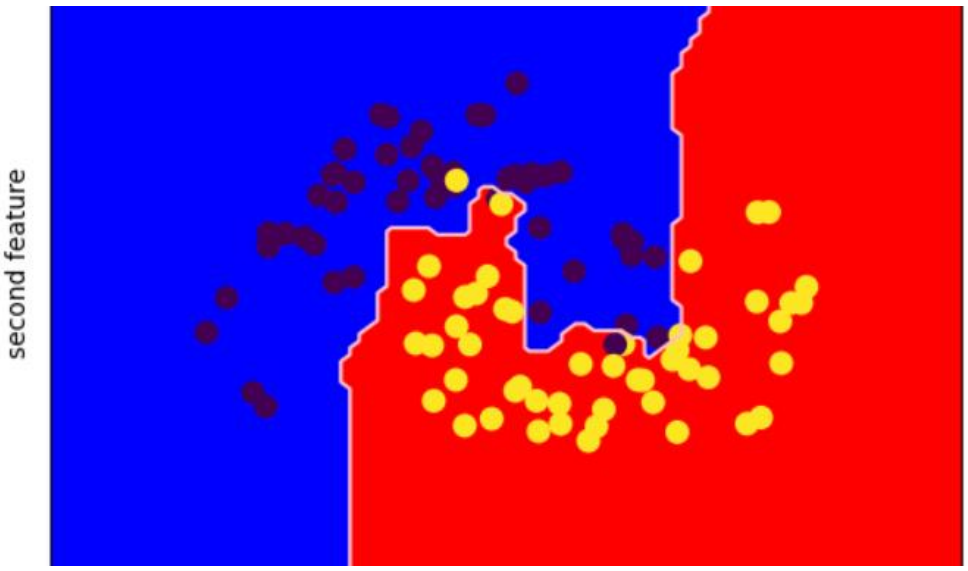

Рисунок 7 – Метод k-ближайших соседей (n=1)

```
n_neightbors =  3
Предсказанные и истинные значения
[1 1 1 0 0 0 1 1 1 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]
[1 1 1 1 0 0 1 1 0 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]
Матрица ошибок
[[12  1]
 [ 1 11]]
Точность классификации:  0.92
Значения полноты, точности, f1-меры и аккуратности
              precision    recall  f1-score   support

           0       0.92      0.92      0.92        13
           1       0.92      0.92      0.92        12

    accuracy                           0.92        25
   macro avg       0.92      0.92      0.92        25
weighted avg       0.92      0.92      0.92        25

Значение площади под кривой ошибок (AUC ROC)
0.9198717948717948
Область принятия решений
```

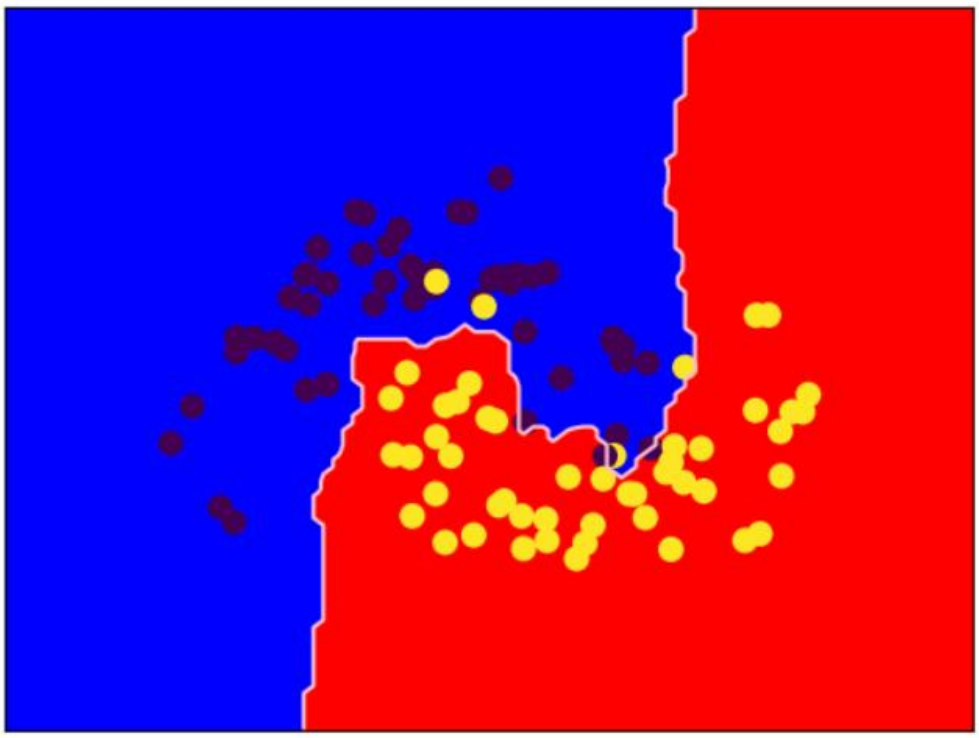

Рисунок 8 – Метод k-ближайших соседей (n=3)

```
n_neightbors =  5
Предсказанные и истинные значения
[1 1 1 0 0 0 1 1 1 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]
[1 1 1 1 0 0 1 1 0 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]
Матрица ошибок
[[12  1]
 [ 1 11]]
Точность классификации:  0.92
Значения полноты, точности, f1-меры и аккуратности
              precision    recall  f1-score   support

           0       0.92      0.92      0.92        13
           1       0.92      0.92      0.92        12

    accuracy                           0.92        25
   macro avg       0.92      0.92      0.92        25
weighted avg       0.92      0.92      0.92        25

Значение площади под кривой ошибок (AUC ROC)
0.9198717948717948
Область принятия решений
```

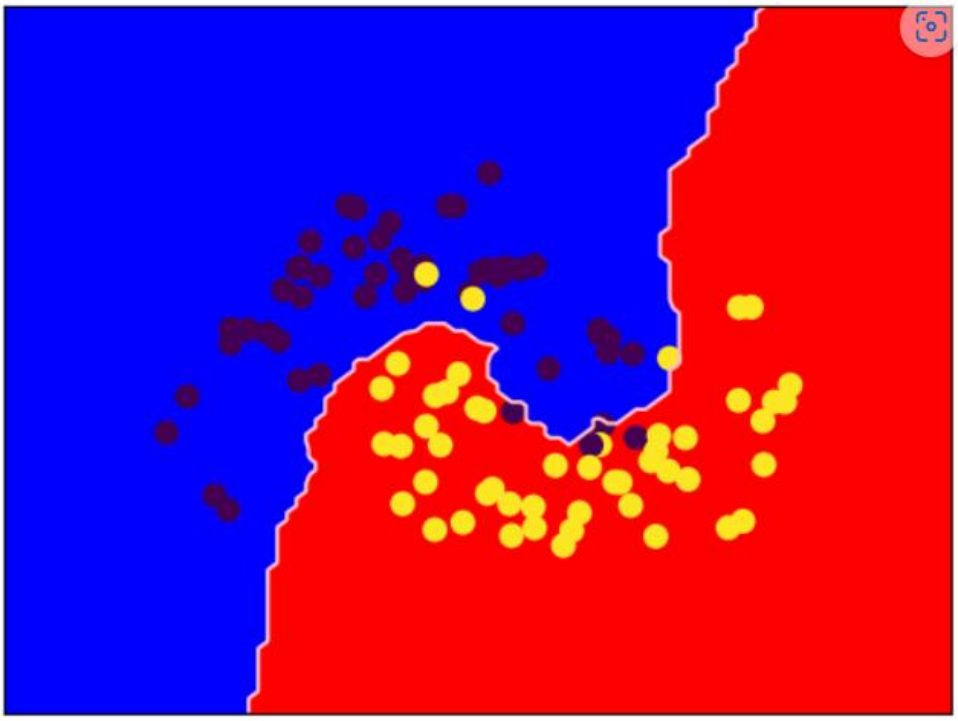

Рисунок 9 – Метод k-ближайших соседей (n=5)

```
n_neightbors =  9
Предсказанные и истинные значения
[1 1 1 0 0 0 1 1 1 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]
[1 1 1 1 0 0 1 1 0 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]
Матрица ошибок
[[12  1]
 [ 1 11]]
Точность классификации:  0.92
Значения полноты, точности, f1-меры и аккуратности
              precision    recall  f1-score   support

           0       0.92      0.92      0.92        13
           1       0.92      0.92      0.92        12

    accuracy                           0.92        25
   macro avg       0.92      0.92      0.92        25
weighted avg       0.92      0.92      0.92        25

Значение площади под кривой ошибок (AUC ROC)
0.9198717948717948
Область принятия решений
```

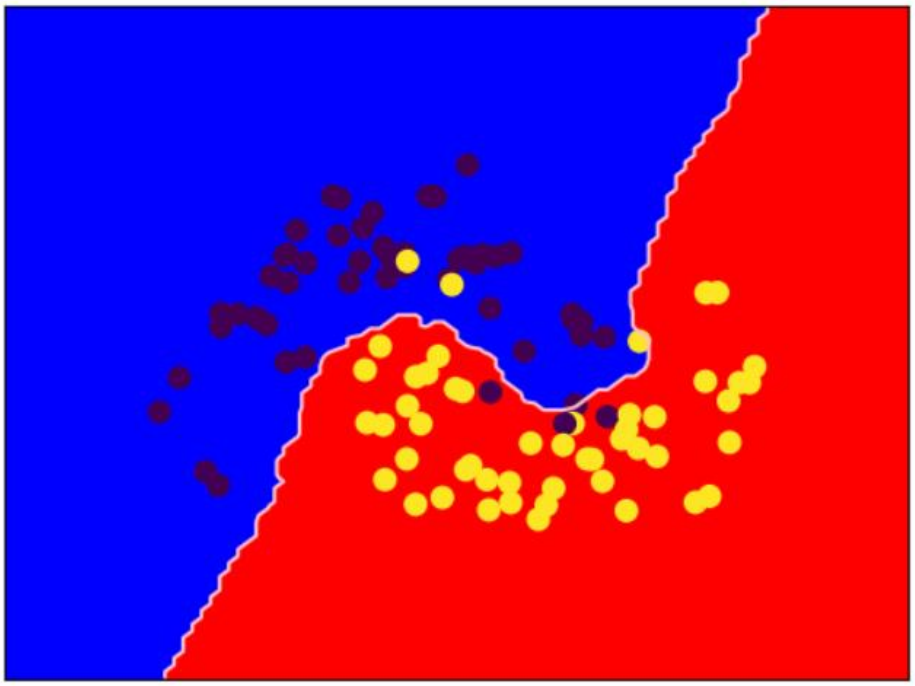

Рисунок 10 – Метод k-ближайших соседей (n=9)

```
naive = GaussianNB()
naive.fit(x_train, y_train)
predict = naive.predict(x_test)
print_classification_metrics(naive, x, y, predict, y_test)
```

```
Предсказанные и истинные значения
[0 1 1 0 0 0 1 1 1 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]
[1 1 1 1 0 0 1 1 0 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]
Матрица ошибок
[[12  1]
 [ 2 10]]
Точность классификации:  0.88
Значения полноты, точности, f1-меры и аккуратности
              precision    recall  f1-score   support

           0       0.86      0.92      0.89        13
           1       0.91      0.83      0.87        12

    accuracy                           0.88        25
   macro avg       0.88      0.88      0.88        25
weighted avg       0.88      0.88      0.88        25

Значение площади под кривой ошибок (AUC ROC)
0.8782051282051283
Область принятия решений
```

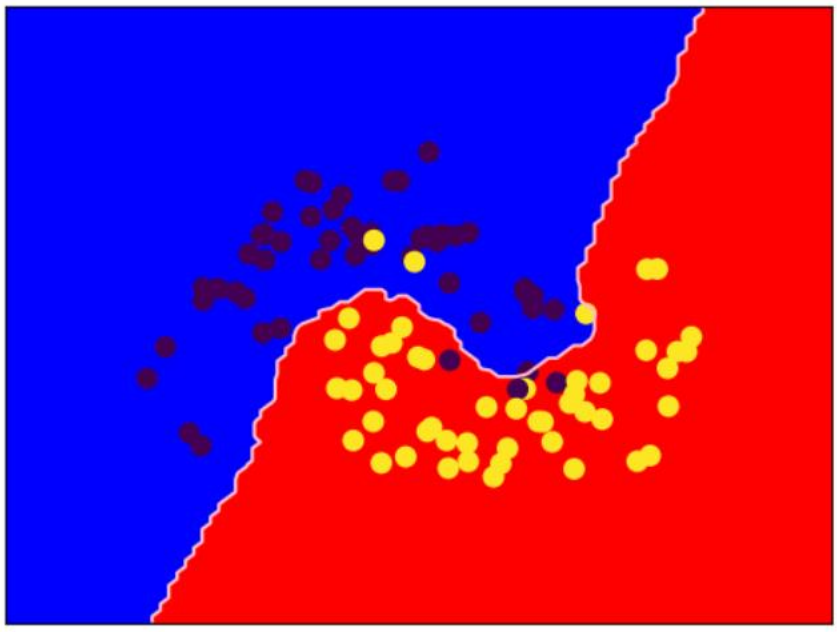

Рисунок 11 – Наивный байесовский метод

```python
for i in [5, 10, 15, 20, 50]:
    rand_forest = RandomForestClassifier(n_estimators=i)
    rand_forest.fit(x_train, y_train)
    prediction = rand_forest.predict(x_test)
    print("n_estimators = ", i)
    print_classification_metrics(knn, x, y, prediction, y_test)
```

```
n_estimators =  5
Предсказанные и истинные значения
[1 1 1 0 0 0 1 1 1 0 1 1 0 1 1 0 1 0 0 1 0 0 0 1 0]
[1 1 1 1 0 0 1 1 0 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]
Матрица ошибок
[[11  2]
 [ 1 11]]
Точность классификации:  0.88
Значения полноты, точности, f1-меры и аккуратности
              precision    recall  f1-score   support

           0       0.92      0.85      0.88        13
           1       0.85      0.92      0.88        12

    accuracy                           0.88        25
   macro avg       0.88      0.88      0.88        25
weighted avg       0.88      0.88      0.88        25

Значение площади под кривой ошибок (AUC ROC)
0.8814102564102563
Область принятия решений
```

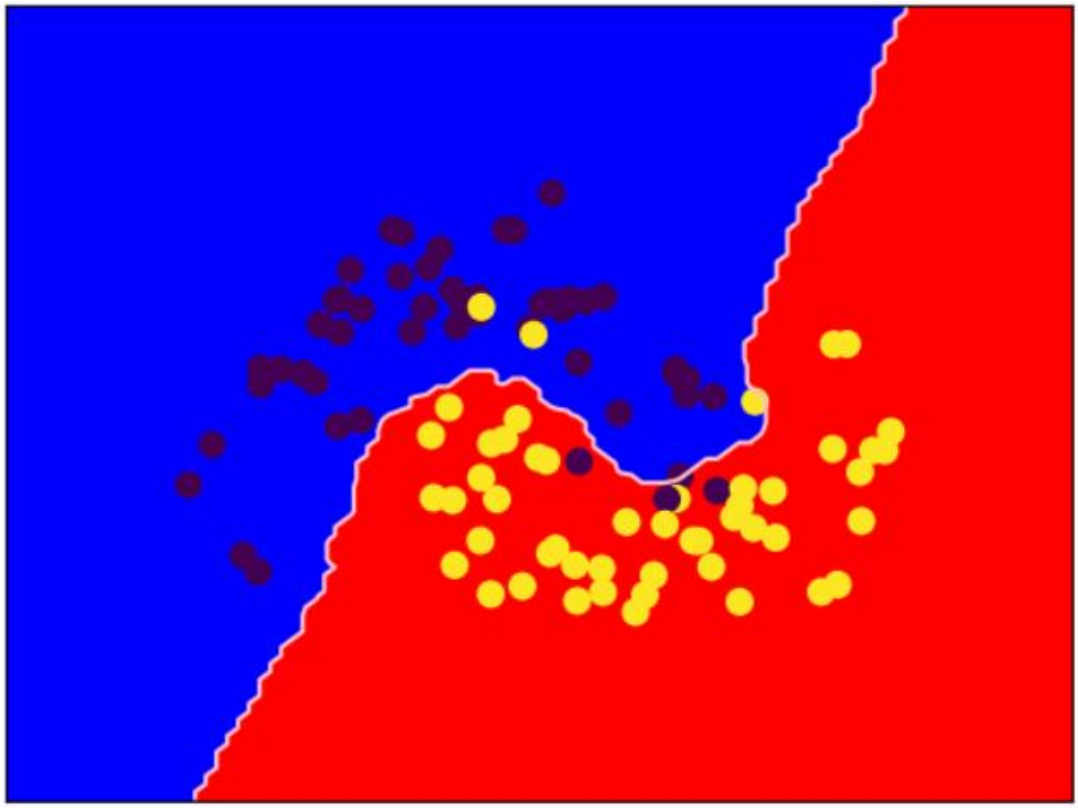

Рисунок 12 – Случайный лес n=5

```
n_estimators =  10
Предсказанные и истинные значения
[1 1 1 0 0 0 1 1 1 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]
[1 1 1 1 0 0 1 1 0 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]
Матрица ошибок
[[12  1]
 [ 1 11]]
Точность классификации:  0.92
Значения полноты, точности, f1-меры и аккуратности
              precision    recall  f1-score   support

           0       0.92      0.92      0.92        13
           1       0.92      0.92      0.92        12

    accuracy                           0.92        25
   macro avg       0.92      0.92      0.92        25
weighted avg       0.92      0.92      0.92        25

Значение площади под кривой ошибок (AUC ROC)
0.9198717948717948
Область принятия решений
```

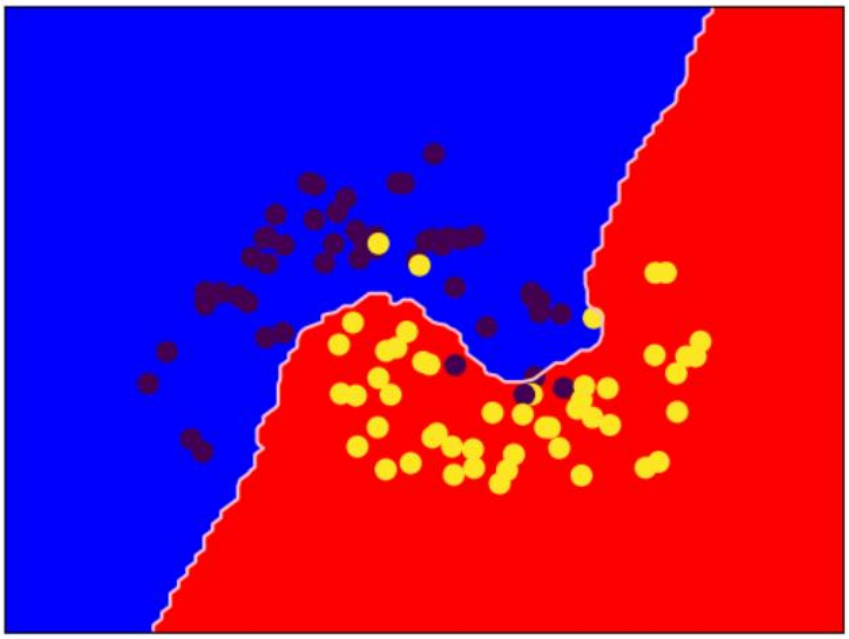

Рисунок 13 – Случайный лес n=10

```
n_estimators =  15
Предсказанные и истинные значения
[1 1 1 0 0 0 1 1 1 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]
[1 1 1 1 0 0 1 1 0 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]
Матрица ошибок
[[12  1]
 [ 1 11]]
Точность классификации:  0.92
Значения полноты, точности, f1-меры и аккуратности
              precision    recall  f1-score   support

           0       0.92      0.92      0.92        13
           1       0.92      0.92      0.92        12

    accuracy                           0.92        25
   macro avg       0.92      0.92      0.92        25
weighted avg       0.92      0.92      0.92        25

Значение площади под кривой ошибок (AUC ROC)
0.9198717948717948
Область принятия решений
```

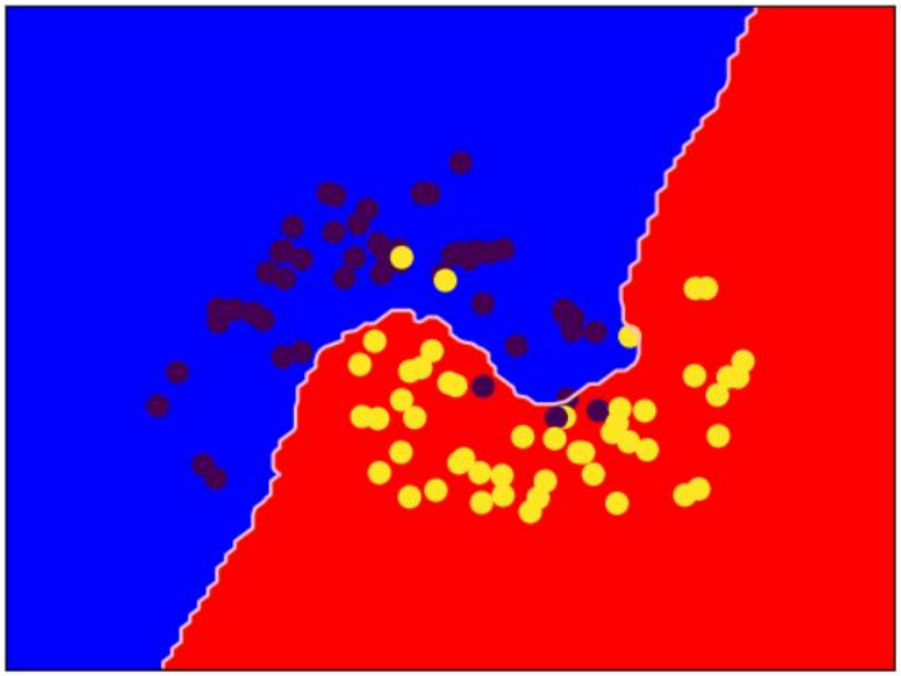

Рисунок 14 – Случайный лес n=15

```
n_estimators =  20
Предсказанные и истинные значения
[1 1 1 0 0 0 1 1 1 0 1 1 0 1 1 0 1 0 0 1 0 0 0 1 0]
[1 1 1 1 0 0 1 1 0 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]
Матрица ошибок
[[11  2]
 [ 1 11]]
Точность классификации:  0.88
Значения полноты, точности, f1-меры и аккуратности
              precision    recall  f1-score   support

           0       0.92      0.85      0.88        13
           1       0.85      0.92      0.88        12

    accuracy                           0.88        25
   macro avg       0.88      0.88      0.88        25
weighted avg       0.88      0.88      0.88        25

Значение площади под кривой ошибок (AUC ROC)
0.8814102564102563
Область принятия решений
```

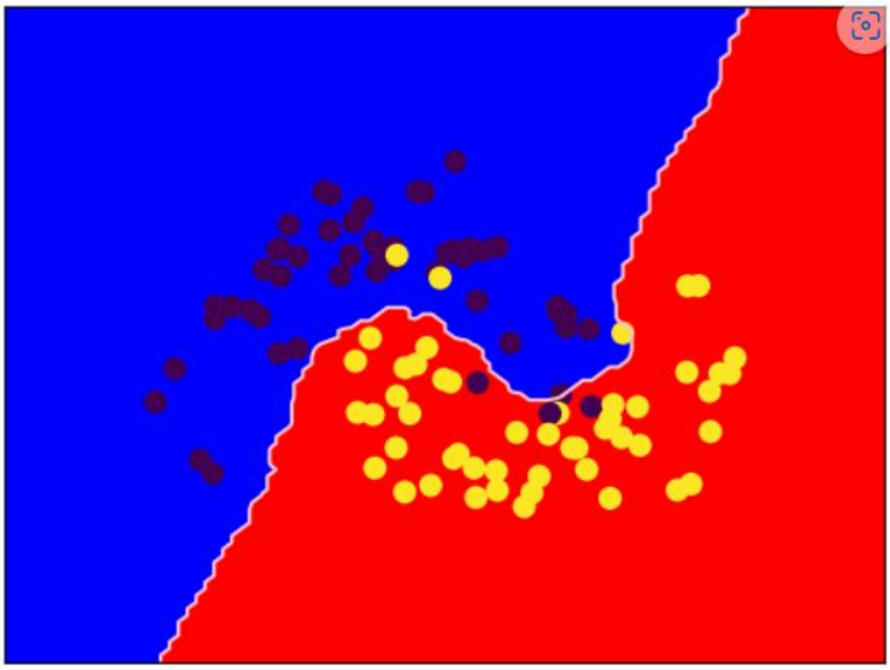

Рисунок 14 – Случайный лес n=20

```
n_estimators =  50
Предсказанные и истинные значения
[1 1 1 0 0 0 1 1 1 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]
[1 1 1 1 0 0 1 1 0 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]
Матрица ошибок
[[12  1]
 [ 1 11]]
Точность классификации:  0.92
Значения полноты, точности, f1-меры и аккуратности
              precision    recall  f1-score   support

           0       0.92      0.92      0.92        13
           1       0.92      0.92      0.92        12

    accuracy                           0.92        25
   macro avg       0.92      0.92      0.92        25
weighted avg       0.92      0.92      0.92        25

Значение площади под кривой ошибок (AUC ROC)
0.9198717948717948
Область принятия решений
```

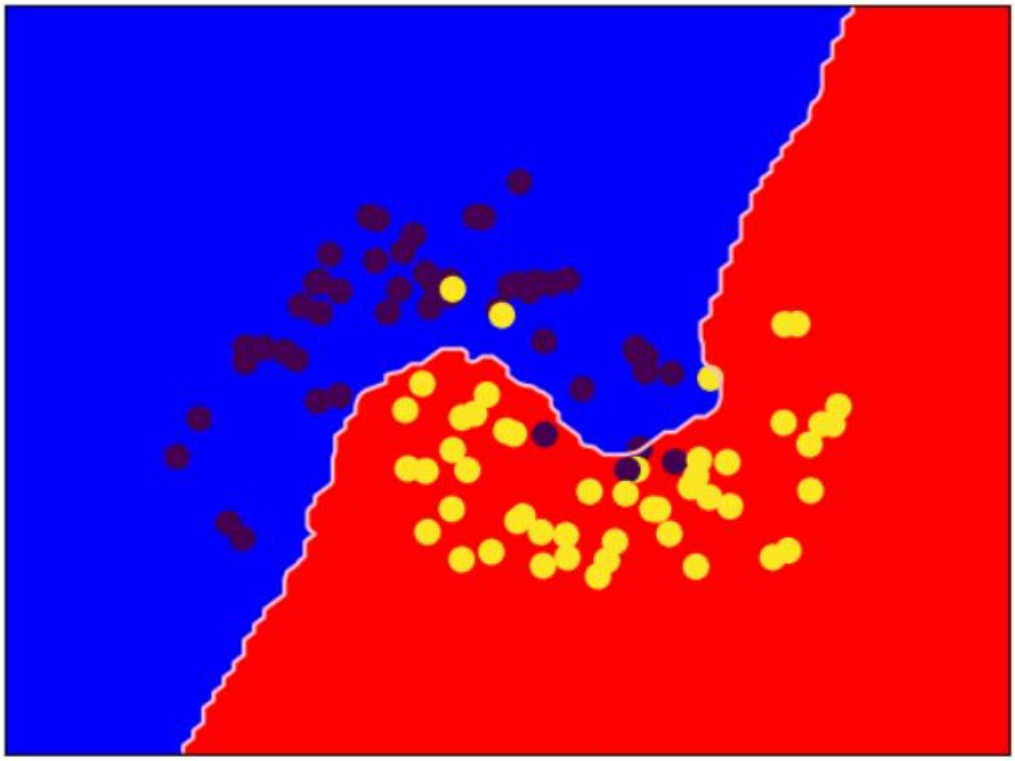

Рисунок 15 – Случайный лес n=50

Таблица 1 – Результаты работы программы

Метод	Истинные и предсказанные метки классов	Матрица ошибок	Значения полноты, точности, f1- меры и аккуратности	Значение площади под кривой ошибок
k-ближайших соседей n=1	[1 1 1 0 0 0 1 1 1 0 0 1 0 1 1 0 1 0 0 1 0 0 0 1 0] [1 1 1 1 1 0 0 1 1 0 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]	[[11 2] [2 10]]	precision recall f1-score support 0 0.85 0.85 0.85 13 1 0.83 0.83 0.83 12 accuracy 0.84 25 macro avg 0.84 0.84 0.84 25 weighted avg 0.84 0.84 0.84 25	0.8397435897435898
k-ближайших соседей n=3	[1 1 1 0 0 0 1 1 1 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0] [1 1 1 1 1 0 0 1 1 0 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]	[[12 1] [1 11]]	Значения полноты, точности, f1-меры и аккуратности precision recall f1-score support 0 0.92 0.92 0.92 13 1 0.92 0.92 0.92 12 accuracy 0.92 25 macro avg 0.92 0.92 0.92 25 weighted avg 0.92 0.92 0.92 25	0.9198717948717948
k-ближайших соседей n=5	[1 1 1 0 0 0 1 1 1 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0] [1 1 1 1 1 0 0 1 1 0 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]	[[12 1] [1 11]]	Значения полноты, точности, f1-меры и аккуратности precision recall f1-score support 0 0.92 0.92 0.92 13 1 0.92 0.92 0.92 12 accuracy 0.92 25 macro avg 0.92 0.92 0.92 25 weighted avg 0.92 0.92 0.92 25	0.9198717948717948
k-ближайших соседей n=9	[1 1 1 0 0 0 1 1 1 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0] [1 1 1 1 1 0 0 1 1 0 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]	[[12 1] [1 11]]	Значения полноты, точности, f1-меры и аккуратности precision recall f1-score support 0 0.92 0.92 0.92 13 1 0.92 0.92 0.92 12 accuracy 0.92 25 macro avg 0.92 0.92 0.92 25 weighted avg 0.92 0.92 0.92 25	0.9198717948717948

Наивный байесовский метод

```
[0 1 1 0 0 0 1 1 1 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]
[1 1 1 1 0 0 1 1 0 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]
```

```
[[12  1]
 [ 2 10]]
```

Значения полноты, точности, f1-меры и аккуратности

	precision	recall	f1-score	support
0	0.86	0.92	0.89	13
1	0.91	0.83	0.87	12
accuracy			0.88	25
macro avg	0.88	0.88	0.88	25
weighted avg	0.88	0.88	0.88	25

0.8782051282051283

Случайный лес n=5

```
[1 1 1 0 0 0 1 1 1 0 1 1 0 1 1 0 1 0 0 1 0 0 0 1 0]
[1 1 1 1 0 0 1 1 0 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]
```

```
[[11  2]
 [ 1 11]]
```

Значения полноты, точности, f1-меры и аккуратности

	precision	recall	f1-score	support
0	0.92	0.85	0.88	13
1	0.85	0.92	0.88	12
accuracy			0.88	25
macro avg	0.88	0.88	0.88	25
weighted avg	0.88	0.88	0.88	25

0.8814102564102563

Случайный лес n=10

```
[1 1 1 0 0 0 1 1 1 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]
[1 1 1 1 0 0 1 1 0 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]
```

```
[[12  1]
 [ 1 11]]
```

Значения полноты, точности, f1-меры и аккуратности

	precision	recall	f1-score	support
0	0.92	0.92	0.92	13
1	0.92	0.92	0.92	12
accuracy			0.92	25
macro avg	0.92	0.92	0.92	25
weighted avg	0.92	0.92	0.92	25

0.9198717948717948

Случайный лес n=15

```
[1 1 1 0 0 0 1 1 1 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]
[1 1 1 1 0 0 1 1 0 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]
```

```
[[12  1]
 [ 1 11]]
```

Значения полноты, точности, f1-меры и аккуратности

	precision	recall	f1-score	support
0	0.92	0.92	0.92	13
1	0.92	0.92	0.92	12
accuracy			0.92	25
macro avg	0.92	0.92	0.92	25
weighted avg	0.92	0.92	0.92	25

0.9198717948717948

Случайный лес n=20

```
[1 1 1 0 0 0 1 1 1 0 1 1 0 1 1 0 1 0 0 1 0 0 0 1 0]
[1 1 1 1 0 0 1 1 0 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]
```

```
[[11  2]
 [ 1 11]]
```

Значения полноты, точности, f1-меры и аккуратности

	precision	recall	f1-score	support
0	0.92	0.85	0.88	13
1	0.85	0.92	0.88	12
accuracy			0.88	25
macro avg	0.88	0.88	0.88	25
weighted avg	0.88	0.88	0.88	25

0.8814102564102563

```
[1 1 1 0 0 0 1 1 1 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]  [[12  1]
[1 1 1 1 0 0 1 1 0 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0]   [ 1 11]]
```

Значения полноты, точности, f1-меры и аккуратности

	precision	recall	f1-score	support	
0	0.92	0.92	0.92	13	
1	0.92	0.92	0.92	12	
accuracy			0.92	25	0.9198717948717948
macro avg	0.92	0.92	0.92	25	
weighted avg	0.92	0.92	0.92	25	

Случайный лес
n=50

Аккуратность при данном разбиении больше всего у метода случайного леса, следовательно он подходит для классификации данных.

Вывод

В ходе выполнения данной лабораторной работы мы получили базовые навыки работы с языком python и набором функций для анализа и обработки данных. Получили практические навыки решения задачи бинарной классификации данных в среде Jupiter Notebook. Научились загружать данные, обучать классификаторы и проводить классификацию. Научились оценивать точность полученных моделей.