

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МОЭВМ

ОТЧЕТ

по лабораторной работе № 7

по дисциплине «Машинное обучение»

Тема: Классификация (Байесовские методы, деревья)

Студент гр. 8304

Сергеев А.Д.

Преподаватель

Жангиров Т.Р.

Санкт-Петербург

2021

Цель работы

Ознакомиться с методами классификации модуля *Sklearn*.

Загрузка данных

1. Данные загружены в датафрейм.

```
data = pd.read_csv('iris.data', header=None)
data
```

	0	1	2	3	4
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa
...
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

Рисунок 1 – Исходные данные

2. Выделены данные и их метки, тексты меток преобразованы к числам.

```
X = data.iloc[:, :4].to_numpy()
labels = data.iloc[:, 4].to_numpy()
X
```

```
array([[5.1, 3.5, 1.4, 0.2],
       [4.9, 3. , 1.4, 0.2],
       [4.7, 3.2, 1.3, 0.2],
       [4.6, 3.1, 1.5, 0.2],
       [5. , 3.6, 1.4, 0.2],
       [5.4, 3.9, 1.7, 0.4],
       [4.6, 3.4, 1.4, 0.3],
       [5. , 3.4, 1.5, 0.2],
       [4.4, 2.9, 1.4, 0.2],
       [4.9, 3.1, 1.5, 0.1],
       [5.4, 3.7, 1.5, 0.2],
       [4.8, 3.4, 1.6, 0.2],
       [4.8, 3. , 1.4, 0.1],
       [4.3, 3. , 1.1, 0.1],
       [5.8, 4. , 1.2, 0.2],
       [5.7, 4.4, 1.5, 0.4],
       [5.4, 3.9, 1.3, 0.4],
       [5.1, 3.5, 1.4, 0.3],
       [5.7, 3.8, 1.7, 0.3],
       [5.1, 3.8, 1.5, 0.3],
       [5.4, 3.4, 1.7, 0.2],
       [5.1, 3.7, 1.5, 0.4],
       [4.6, 3.6, 1. , 0.2],
       [5.1, 3.3, 1.7, 0.5],
       [4.8, 3.4, 1.9, 0.2],
```

Рисунок 2 – Выделенные данные

Рисунок 3 — Метки, преобразованные к числам

3. Выборка разбита на обучающую и тестовую *train_test_split*.

Байесовские методы

1. Проведена классификация наблюдений наивным байесовским методом.

```
gnb = GaussianNB()
y_pred = gnb.fit(X_train, y_train).predict(X_test)
print((y_test != y_pred).sum()) #количество наблюдений, который были неправильно определены
```

Рисунок 4 — Классификация наивным байесовским методом

Выявлено 5 неправильно классифицированных наблюдения.

Таблица 1 – Атрибуты *GaussianNB*

Атрибут	Описание
class_count_	Количество выборок каждого класса, участвующих в обучении
class_prior_	Вероятность каждого класса в данных для обучения
classes_	Метки классов
n_features_in_	Количество признаков в данных обучения
theta_ndarray of shape (n_classes, n_features)	Среднее значение каждого признака в классах

2. Точность классификации получена с помощью функции `score()` и составляет 93.3%.

```
print(gnb.fit(X_train, y_train).score(X_test, y_test))
0.9333333333333333
```

Рисунок 5 — Точность классификации

- Построен график зависимости неправильно классифицированных наблюдений и точности классификации от размера тестовой выборки (0.05 до 0.95 с шагом 0.05), `random_state = 830434`.

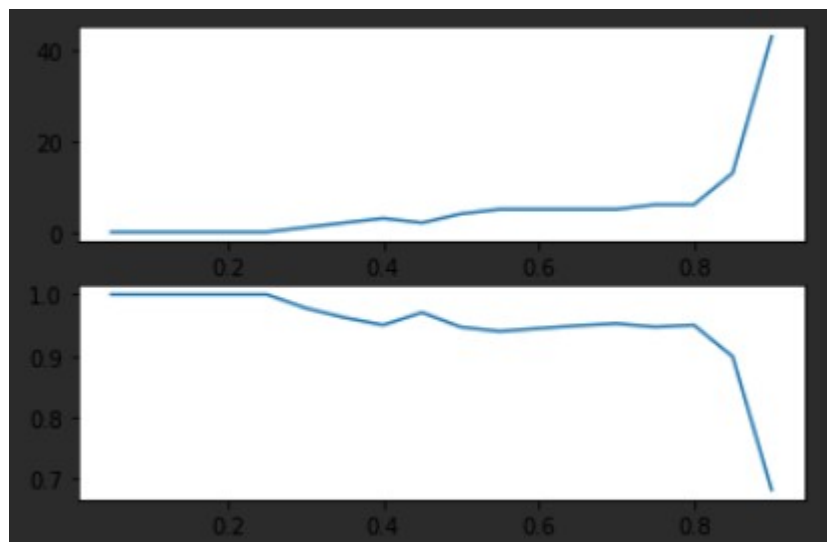


Рисунок 6 – График зависимости точности классификации и неправильно классифицированных наблюдений от размера тестовой выборки

Такой метод используется для данных, которые имеют непрерывную описательную функцию.

- Классификация проведена с помощью *GaussianNB*, *MultinomialNB*, *ComplementNB*, *BernoulliNB*.

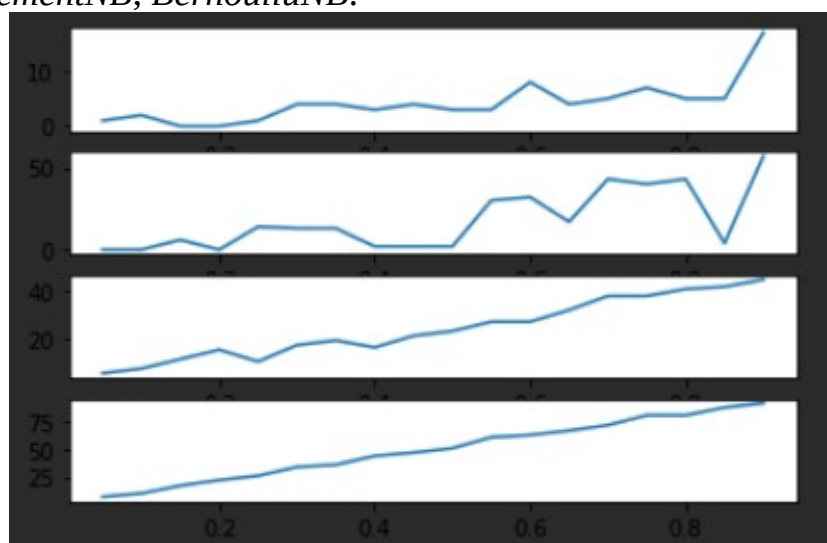


Рисунок 7 – Точность классификации различными методами

MultinomialNB – полиномиальный наивный байесовский классификатор, подходит для классификации с дискретными признаками (например, подсчет слов для классификации текста). *MultinomialNB* реализует наивный алгоритм Байеса для полиномиально распределенных данных. Распределение для каждого класса параметризуется векторами, содержащими вероятности вхождения признаков в элемент выборки, соответствующий данному классу.

ComplementNB – адаптация *MultinomialNB*, подходит для несбалансированных наборов данных. В частности, CNB использует статистику из дополнения каждого класса для вычисления весов модели. *ComplementNB* часто превосходит *MultinomialNB* в задачах классификации текста.

BernoulliNB – как и *MultinomialNB*, этот классификатор подходит для дискретных данных. Разница в том, что в то время, как *MultinomialNB* работает с подсчетом вхождений, *BernoulliNB* предназначен для двоичных/логических признаков.

Классифицирующие деревья

1. Проведена классификация наблюдений с помощью деревьев решений на тех же данных.

```
clf = tree.DecisionTreeClassifier()
y_pred = clf.fit(X_train, y_train).predict(X_test)
print((y_test != y_pred).sum())
7
```

Рисунок 8 — Классификация с помощью деревьев решений

Выявлено 7 неправильно классифицированных наблюдений.

2. Точность классификации получена с помощью функции `score()` и составляет 95.6%.

```
print(clf.fit(X_train, y_train).score(X_test, y_test))
0.9555555555555556
```

Рисунок 9 — Точность классификации

3. Получившееся дерево имеет глубину 2, и 3 листа.

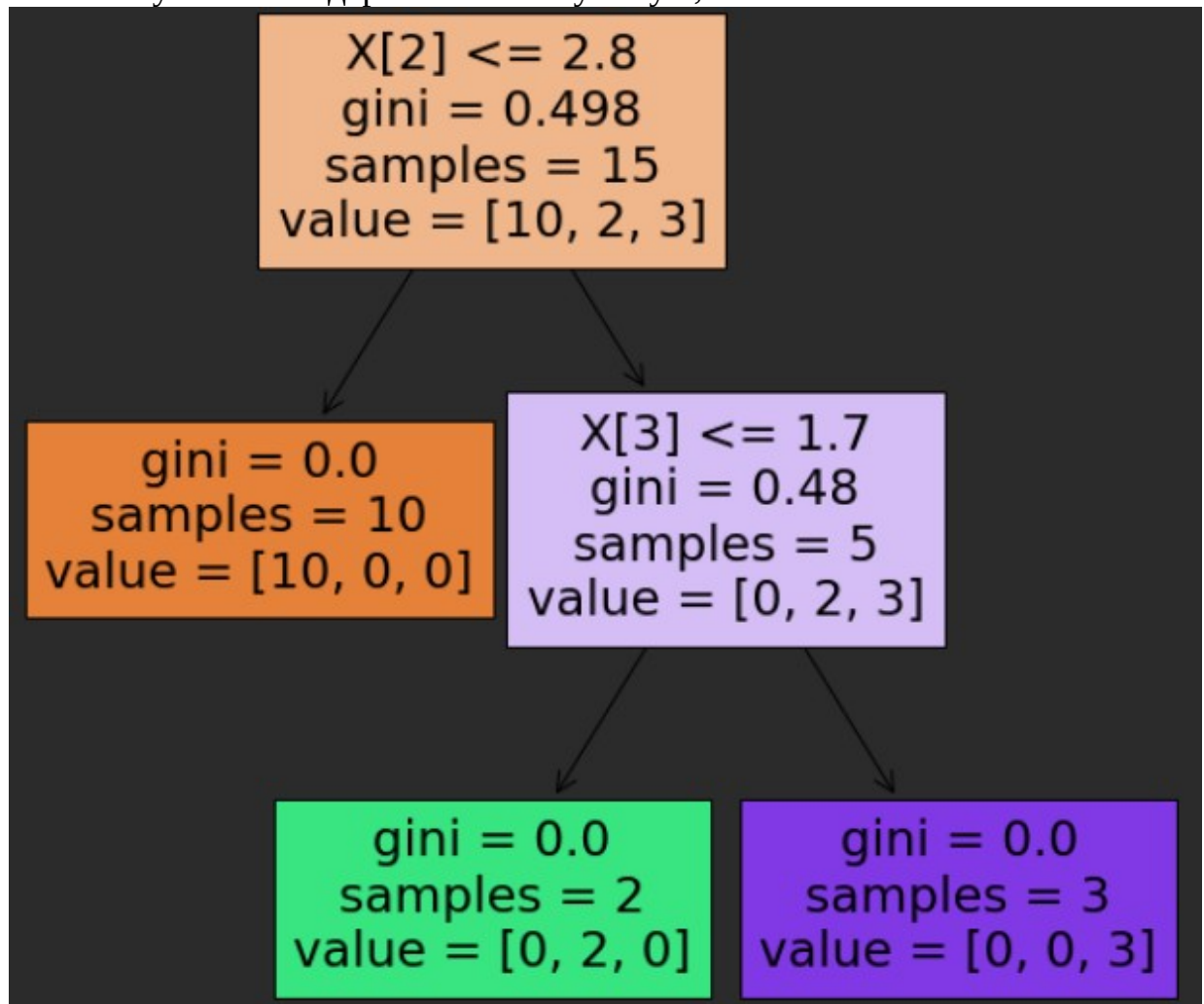


Рисунок 10 – Дерево решений для классификации

Узлом дерева является критерий, разделяющий данные на два подмножества. В узле содержится:

- 1) вопрос, на основании которого происходит разделение
- 2) коэффициент Джини – показатель, определяющий чистоту узла
- 3) количество элементов в разделяемом множестве
- 4) список, показывающий, сколько образцов на данном узле попадают в каждую категорию

В листьях не содержится вопрос. В листьях содержатся экземпляры только одного класса.

4. Построен график зависимости неправильно классифицированных наблюдений и точности классификации от размера тестовой выборки. Размер тестовой выборки изменяется от 0.05 до 0.95 с шагом 0.05. Параметр `random_state` выбран равным номеру зачетной книжки (830434).

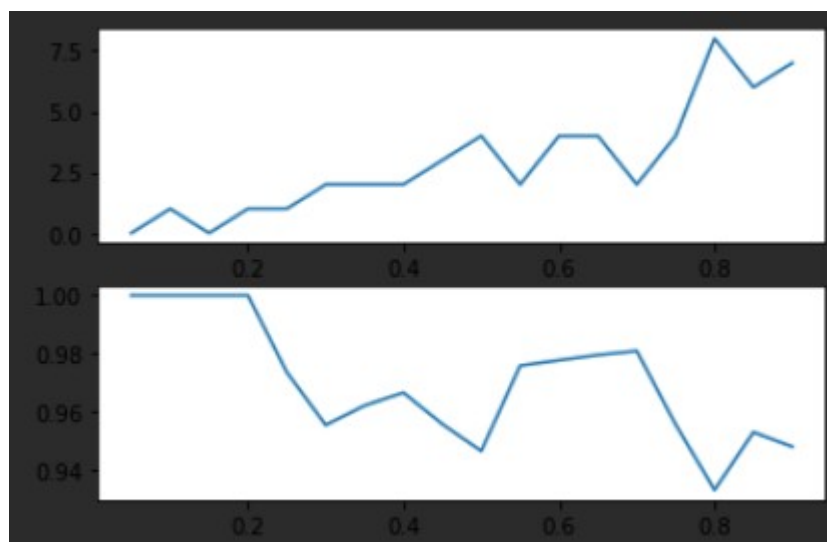
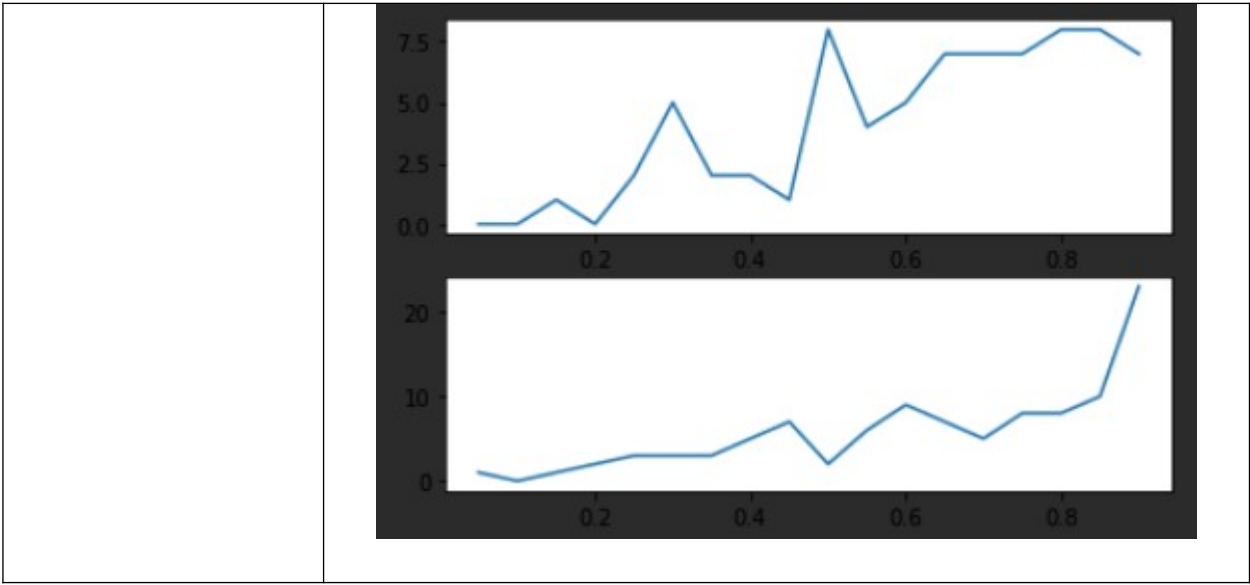
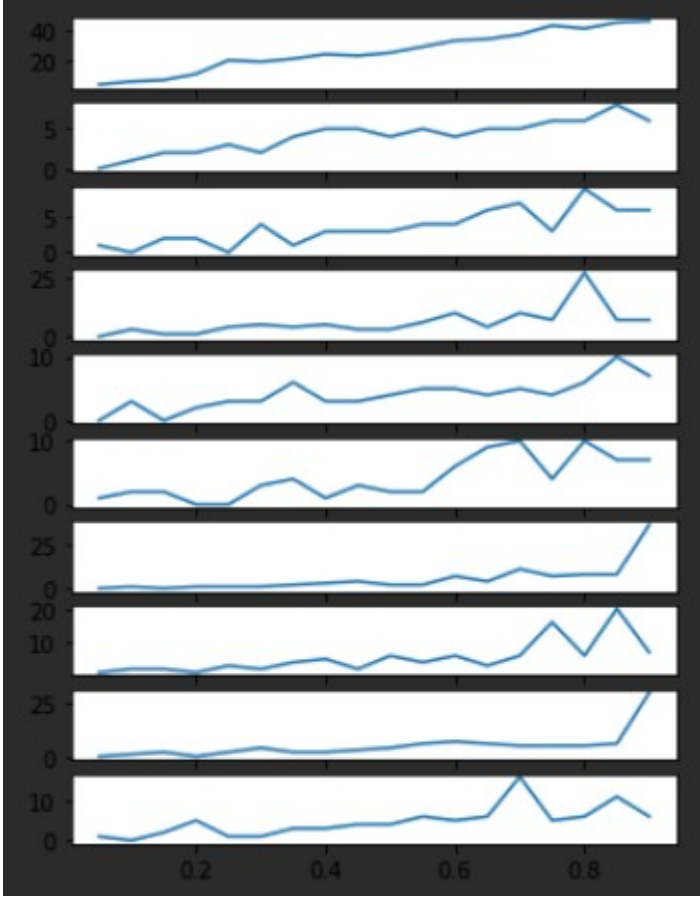


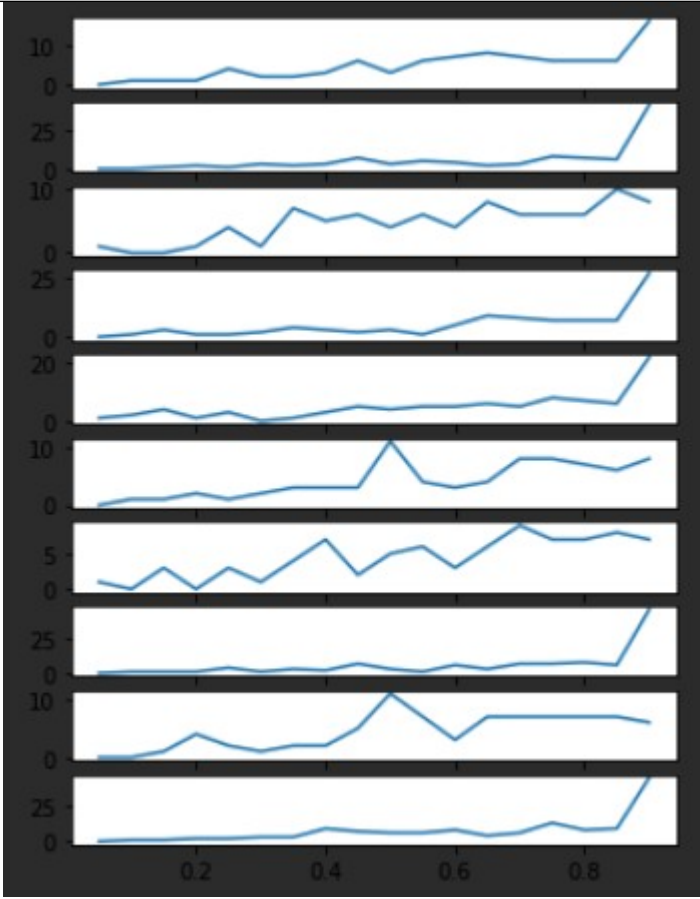
Рисунок 11 – График зависимости точности классификации и неправильно классифицированных наблюдений от размера тестовой выборки

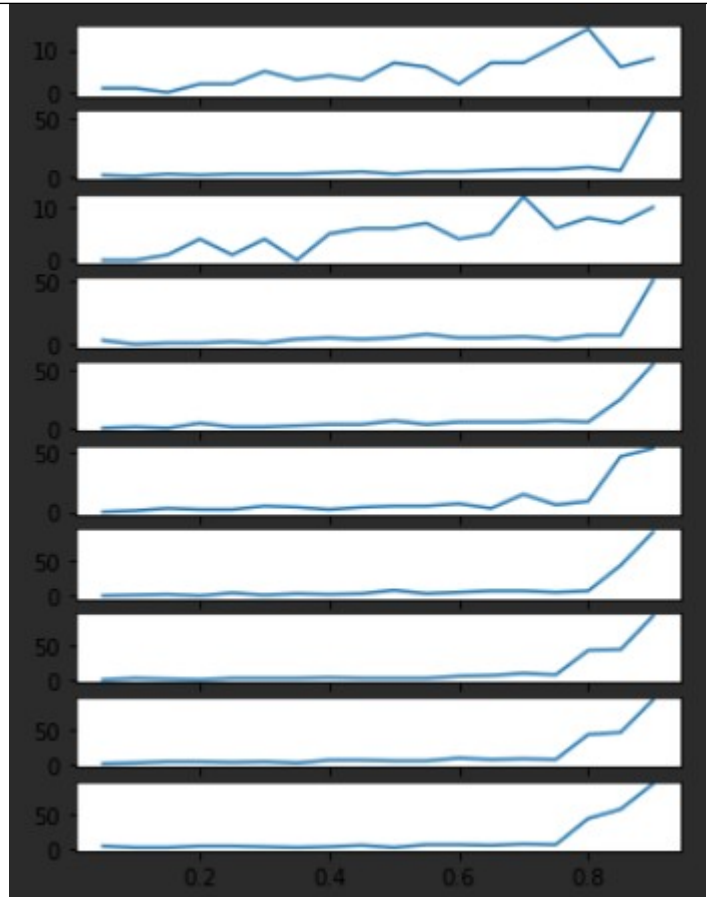
5. Исследованы параметры *DecisionTreeClassifier*.

Параметр	Описание
criterion	<p>Функция измерения качества разбиения.</p> <p>Поддерживается индекс Джини и энтропия.</p> <p>На исходных данных для обоих параметров алгоритм даёт похожие результаты.</p>  <p>The figure consists of two vertically stacked line graphs sharing a common x-axis representing sample size, with major ticks at 0.2, 0.4, 0.6, and 0.8. The top graph's y-axis ranges from 0.0 to 10.0. The data points are approximately: (0.1, 0.0), (0.15, 0.0), (0.2, 0.0), (0.25, 1.0), (0.3, 6.0), (0.35, 2.0), (0.4, 2.0), (0.45, 4.0), (0.5, 3.0), (0.55, 4.0), (0.6, 2.0), (0.65, 5.0), (0.7, 5.0), (0.75, 9.0), (0.8, 9.0), (0.85, 7.0), (0.9, 10.0). The bottom graph's y-axis ranges from 0.0 to 15.0. The data points are approximately: (0.1, 3.0), (0.15, 2.0), (0.2, 2.0), (0.25, 1.0), (0.3, 2.0), (0.35, 2.0), (0.4, 4.0), (0.45, 3.0), (0.5, 5.0), (0.55, 5.0), (0.6, 7.0), (0.65, 7.0), (0.7, 6.0), (0.75, 4.0), (0.8, 7.0), (0.85, 10.0), (0.9, 15.0).</p>
splitter	<p>Стратегия, используемая для выбора разбиения на каждом узле.</p> <p>Поддерживается выбор наилучшего разбиения и случайный выбор.</p> <p>На исходных данных случайная стратегия деления дает больший результат ошибочных измерений.</p>



<p>max_depth</p>	<p>Максимальная глубина дерева. Если None, то узлы расширяются до тех пор, пока все листья не станут чистыми или пока все листья не будут содержать менее min_samples_split выборок.</p> <p>При увеличении максимально возможной глубины число ошибочно классифицированных значений уменьшается.</p> 
<p>min_samples_split</p>	<p>Минимальное количество выборок, необходимых для разделения внутреннего узла.</p> <p>При увеличении минимального числа наблюдений для разбиения узла значительно возрастает количество неправильно классифицированных наблюдений.</p>

	
min_samples_leaf	<p>Минимальное количество выборок, которое требуется для конечного узла. Точка разделения на любой глубине будет учитываться только в том случае, если она оставляет не менее min_samples_leaf обучающих выборок в каждой из левой и правой ветвей.</p> <p>При увеличении минимально возможного числа наблюдений в конечном узле увеличивается число неправильно классифицированных наблюдений.</p>



Выводы

В ходе лабораторной работы рассмотрены такие методы классификации модуля *Sklearn*, как *GaussianNB*, *MultinomialNB*, *ComplementNB*, *BernoulliNB* и *DecisionTreeClassifier*.