Рубежный контроль №1

Номер варианта	Номер задачи	Номер набора данных, указанного в задаче
17	3	1

Для студентов группы ИУ5-65Б для набора данных построить "парные диаграммы".

Условие задачи

Задача №3. Для заданного набора данных произведите масштабирование данных (для одного признака) и преобразование категориальных признаков в количественные двумя способами (label encoding, one hot encoding) для одного признака. Какие методы Вы использовали для решения задачи и почему?

Набор данных №1:
https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_iris.html#sklearn.datasets.load_iris

```
# импорт основных библиотек
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# загрузка датасета
from sklearn.datasets import load_iris

iris_dataset = load_iris()
```

Анализ датасета

```
# наименования признаков
iris_dataset.feature_names
```

```
['sepal length (cm)',
 'sepal width (cm)',
 'petal length (cm)',
 'petal width (cm)']
```

```
# значения целевого признака
iris_dataset.target
```

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
       0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1,
```

```
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2,
2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2])
```

```
iris_dataset.target_names
```

```
array(['setosa', 'versicolor', 'virginica'], dtype='<U10')
```

```
target_dict = dict(zip(np.unique(iris_dataset.target),
iris_dataset.target_names))
target_dict
```

```
{0: 'setosa', 1: 'versicolor', 2: 'virginica'}
```

Поскольку стоит задача преобразования **категориальных** признаков в количественные, а в data нет столбцов категориальных признаков, при преобразовании sklearn dataset в pandas dataframe вставим 2 столбца целевого значения и из target_names, и из target. Так категориальным столбцом для преобразования будет **target_name**.

```
# dataset to pd.DataFrame
iris_df = pd.DataFrame(data=iris_dataset['data'],
columns=iris_dataset['feature_names'])
iris_df['target'] = iris_dataset['target']
iris_df['target_name'] = [target_dict[i] for i in
iris_dataset['target']]

# первые 5 строк
iris_df.head(5)
```

```
   sepal length (cm)  sepal width (cm)  petal length (cm)  petal width
(cm)  \
0                5.1               3.5                1.4
0.2
1                4.9               3.0                1.4
0.2
2                4.7               3.2                1.3
0.2
3                4.6               3.1                1.5
0.2
4                5.0               3.6                1.4
0.2

   target target_name
0       0      setosa
1       0      setosa
2       0      setosa
```

```
3       0      setosa
4       0      setosa
```

```
# типы данных
iris_df.dtypes
```

```
sepal length (cm)    float64
sepal width (cm)     float64
petal length (cm)    float64
petal width (cm)     float64
target                 int32
target_name           object
dtype: object
```

```
print(f"Строк: {iris_df.shape[0]}\nСтолбцов: {iris_df.shape[1]}")
```

```
Строк: 150
Столбцов: 6
```

```
# проверка наличия пропусков
iris_df.isnull().sum()
```

```
sepal length (cm)    0
sepal width (cm)     0
petal length (cm)    0
petal width (cm)     0
target               0
target_name          0
dtype: int64
```

(пропусков нет)

```
# описание количественных признаков
iris_df.describe()
```

```
       sepal length (cm)  sepal width (cm)  petal length (cm)  \
count         150.000000        150.000000         150.000000
mean            5.843333          3.057333           3.758000
std             0.828066          0.435866           1.765298
min             4.300000          2.000000           1.000000
25%             5.100000          2.800000           1.600000
50%             5.800000          3.000000           4.350000
75%             6.400000          3.300000           5.100000
max             7.900000          4.400000           6.900000

       petal width (cm)      target
count        150.000000  150.000000
mean           1.199333    1.000000
std            0.762238    0.819232
min            0.100000    0.000000
25%            0.300000    0.000000
```

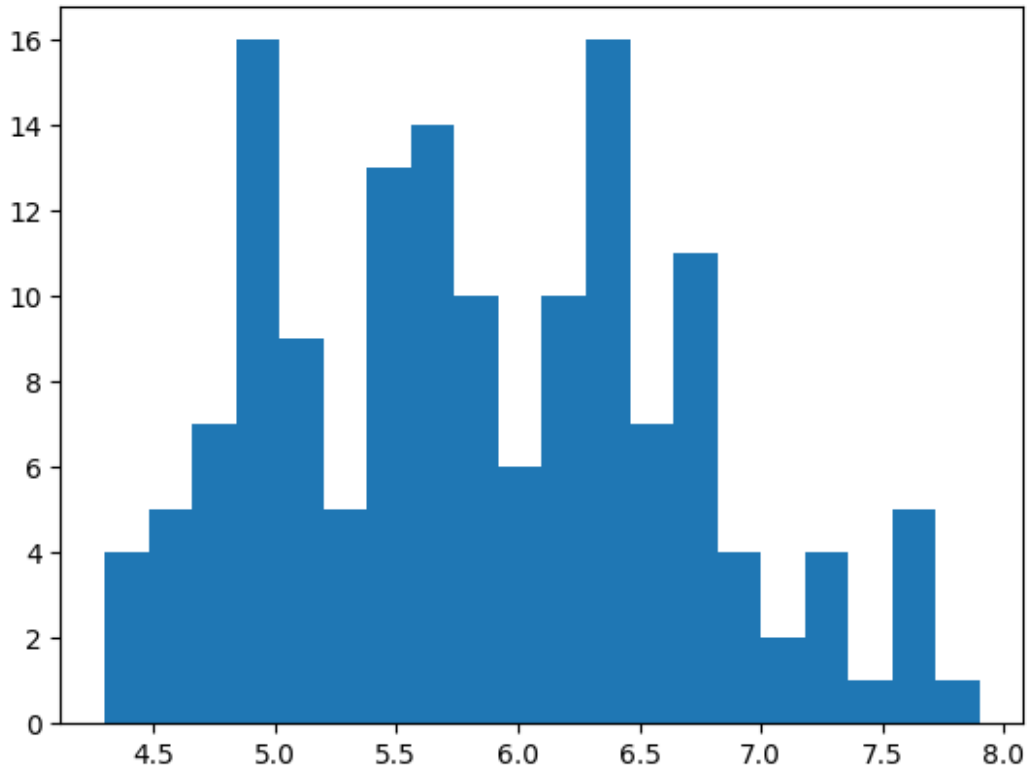
```
50%             1.300000    1.000000
75%             1.800000    2.000000
max             2.500000    2.000000
```

Масштабирование данных

Для масштабирования выберем признак *sepal length (cm)*.

Для выбора метода масштабирования рассмотрим распределение признака.

```
# изначальные значения в выбранном столбце
data = iris_df[['sepal length (cm)']]
# гистограмма данных
plt.hist(data, 20)
plt.show()
```

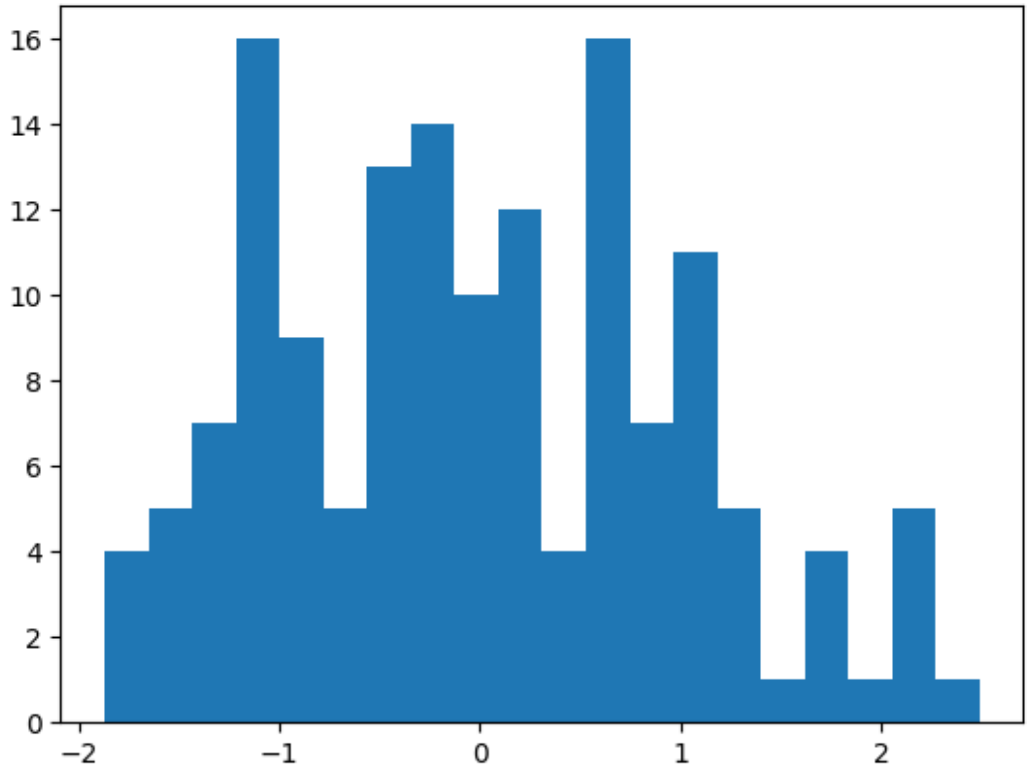


С выбранным признаком лучше применять метод нормализации на основе z-оценки (среднее значение всех значений равно 0, а стандартное отклонение равно 1), т.к.:

- распределение близко к нормальному, выбросов данных нет, все значения находятся в определенном диапазоне;
- многие модели машинного обучения, такие как линейные модели или метод ближайших соседей, функционируют лучше на данных, масштабированных этим способом.

Такое масштабирование реализовано в методе класса **StandardScaler** из библиотеки scikit-learn в Python:

```
# на основе Z-оценки
from sklearn.preprocessing import StandardScaler
stand = StandardScaler()
scaled_df2 = stand.fit_transform(data)
plt.hist(scaled_df2, 20)
plt.show()
```



Преобразование признаков

Для преобразования выберем единственный категориальный признак target_name.

```
# значения выбранного признака
print(*iris_df['target_name'].unique(), sep=", ")
```

```
setosa, versicolor, virginica
```

Многие методы преобразования категориальных признаков в количественные реализованы в модуле **sklearn.preprocessing**, поэтому будем использовать его для преобразования выбранного признака.

Достоинство методов из sklearn в том, что результат преобразования сохраняется и затем может быть применен к новым наборам данных, которые используют те же

категориальные переменные, с согласованными результатами. Поэтому для задач машинного обучения лучше применять именно их.

1. Label Encoding

Label Encoding присваивает каждому категориальному значению целое значение, основанное на алфавитном порядке.

```
# импорт класса для label encoding из sklearn
from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()
enc_target1 = le.fit_transform(iris_df['target_name'])
# срез значений преобразованного признака
enc_target1[0:-1:10]
```

```
array([0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2])
```

2. One Hot Encoding

One Hot Encoding преобразует категориальное значение в бинарный вектор с длиной, равной количеству значений, которое может принимать категориальный признак, в котором 1 ставиться на месте, соответствующему значению признака, а остальные значение вектора равны 0.

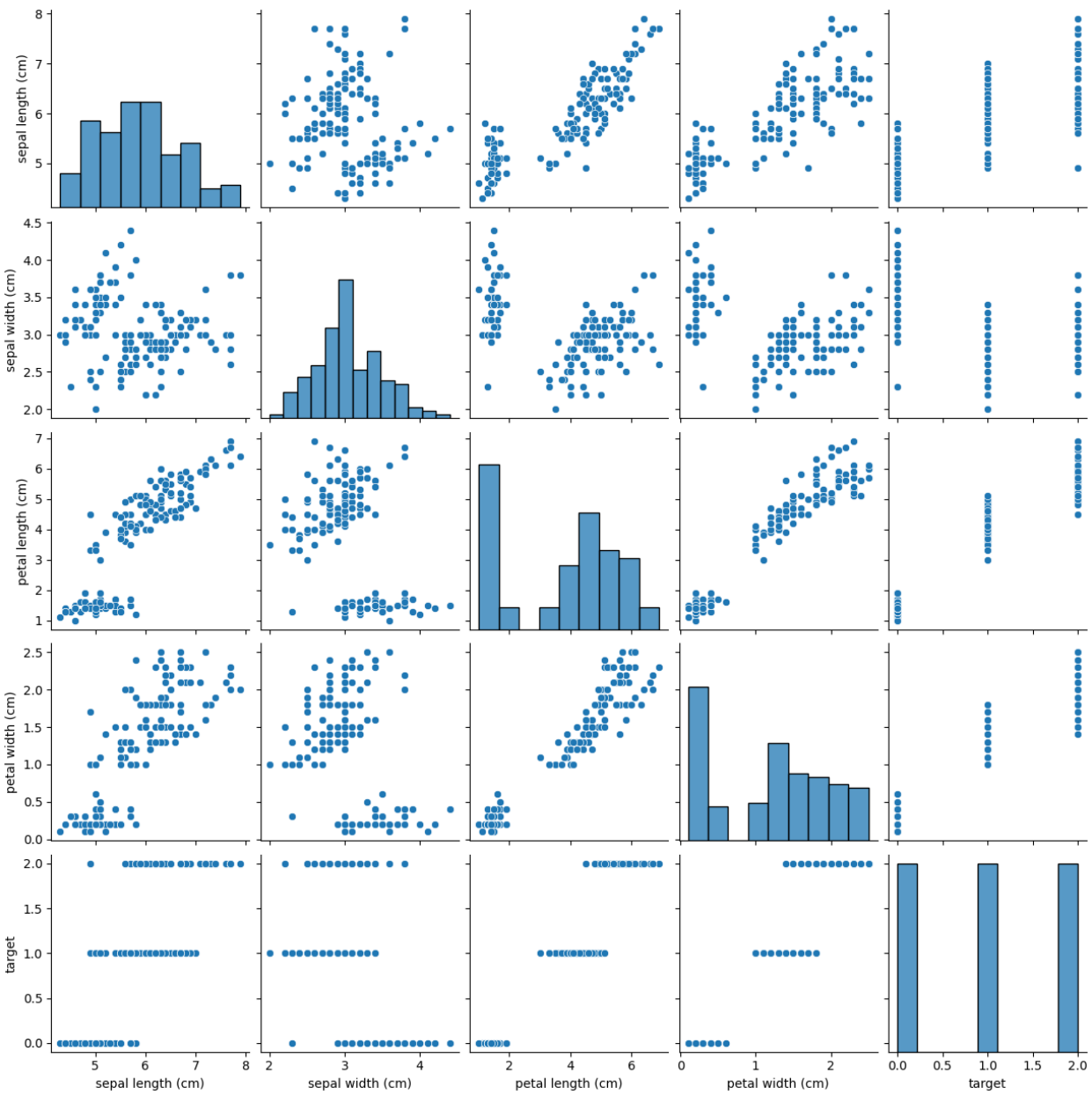
```
# импорт класса для one hot encoding из sklearn
from sklearn.preprocessing import OneHotEncoder

ohe = OneHotEncoder()
enc_target2 = ohe.fit_transform(iris_df[['target_name']])
# срез значений преобразованного признака
enc_target2.todense()[0:-1:10]
```

```
matrix([[1., 0., 0.],
        [1., 0., 0.],
        [1., 0., 0.],
        [1., 0., 0.],
        [1., 0., 0.],
        [0., 1., 0.],
        [0., 1., 0.],
        [0., 1., 0.],
        [0., 1., 0.],
        [0., 1., 0.],
        [0., 0., 1.],
        [0., 0., 1.],
        [0., 0., 1.],
        [0., 0., 1.],
        [0., 0., 1.]])
```

Парные диаграммы

```
sns.pairplot(iris_df)
```

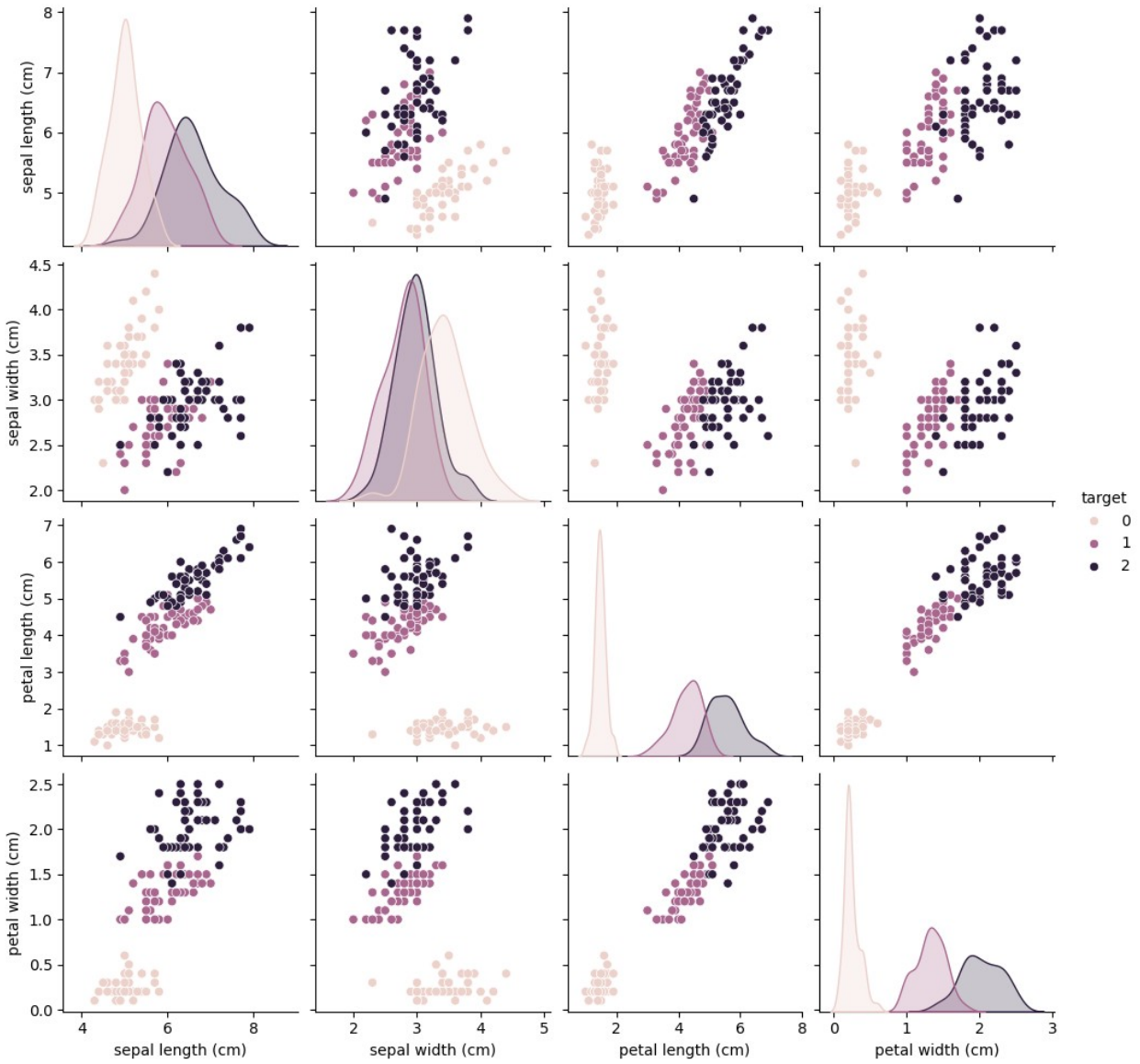
```
<seaborn.axisgrid.PairGrid at 0x28320b4ab10>
```



С группировкой по целевому признаку:

```
sns.pairplot(iris_df, hue="target")
```

```
<seaborn.axisgrid.PairGrid at 0x28322a26a90>
```



На основе парных диаграмм можем сделать следующие выводы:

- Все признаки для каждого сорта ириса распределены нормально.
- Сорта 1 и 2 (versicolor и virginica) схожи между собой больше, чем ирисы сорта 0 (setosa).
- Для сорта 0 разброс значений признаков меньше по длине чашелистика и длине и ширине лепестка, чем для остальных.
- Существует прямая зависимость между длиной и шириной лепестка, длиной чашелистика и длиной лепестка, между остальными признаками зависимость выражена слабее.