Защищено:
Гапанюк Ю.Е.

"__"_____2022 г.

Демонстрация:
Гапанюк Ю.Е.

"__"_____2022 г.

# Отчет по лабораторной работе № 1 по курсу Технологии машинного обучения ГУИМЦ

## Тема работы: " Разведочный анализ данных. Исследование и визуализация данных. "

10
(количество листов)
Вариант № **4**

ИСПОЛНИТЕЛЬ:

студент группы ИУ5Ц-84Б _____
(подпись)

Шанаурина Е.Г.

"__"_____2022 г.

Москва, МГТУ - 2022

# Цель лабораторной работы

Изучить различные методы визуализации данных.

# Задание

- Выбрать набор данных (датасет). Вы можете найти список свободно распространяемых датасетов здесь.
- Для первой лабораторной работы рекомендуется использовать датасет без пропусков в данных, например из Scikit-learn.
- Пример преобразования датасетов Scikit-learn в Pandas Dataframe можно посмотреть здесь.

Для лабораторных работ не рекомендуется выбирать датасеты большого размера.

- Создать ноутбук, который содержит следующие разделы:

1. Текстовое описание выбранного Вами набора данных.
2. Основные характеристики датасета.
3. Визуальное исследование датасета.
4. Информация о корреляции признаков.

- Сформировать отчет и разместить его в своем репозитории на github.

Средства и способы визуализации данных можно посмотреть здесь.

В качестве опорного примера для выполнения лабораторной работы можно использовать пример.

Дополнительно примеры решения задач, содержащие визуализацию, можно посмотреть в репозитории курса mlcourse.ai - https://github.com/Yorko/mlcourse.ai/wiki/Individual-projects-and-tutorials-(in-Russian)

# Ход выполнения работы

## Текстовое описание набора данных

В качестве набора данных используется toy dataset iris из библиотеки scikit-learn. Этот dataset содержит ирисы Фишера.

Этот набор данных состоит из одного файла со 150-ю записями. Данный файл содержит следующие колонки:

- `sepal length (cm)` — длина чашелистика в сантиметрах
- `sepal width (cm)` — ширина чашелистика в сантиметрах
- `petal length` — длинна лепестка
- `petal width (cm)` — ширина липестка
- `target` — вид ириса (0 = setosa; 1 = versicolor; 2 = virginica)

## Основные характеристики набора данных

Подключим все необходимые библиотеки:

In [2]:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib
```

```
import matplotlib_inline
import matplotlib.pyplot as plt
from sklearn.datasets import *
```

Преобразуем данные

In [4]:

```
iris = load_iris(as_frame=True)
df = pd.DataFrame(data= np.c_[iris['data'], iris['target']], columns= iris['feature_name
s'] + ['target'])
#df = iris.data
df
```

Out[4]:

| | sepal length (cm) | sepal width (cm) | petal length (cm) | petal width (cm) | target |
|---|---|---|---|---|---|
| 0 | 5.1 | 3.5 | 1.4 | 0.2 | 0.0 |
| 1 | 4.9 | 3.0 | 1.4 | 0.2 | 0.0 |
| 2 | 4.7 | 3.2 | 1.3 | 0.2 | 0.0 |
| 3 | 4.6 | 3.1 | 1.5 | 0.2 | 0.0 |
| 4 | 5.0 | 3.6 | 1.4 | 0.2 | 0.0 |
| ... | ... | ... | ... | ... | ... |
| 145 | 6.7 | 3.0 | 5.2 | 2.3 | 2.0 |
| 146 | 6.3 | 2.5 | 5.0 | 1.9 | 2.0 |
| 147 | 6.5 | 3.0 | 5.2 | 2.0 | 2.0 |
| 148 | 6.2 | 3.4 | 5.4 | 2.3 | 2.0 |
| 149 | 5.9 | 3.0 | 5.1 | 1.8 | 2.0 |

150 rows × 5 columns

In [21]:

```
# Список колонок с типами данных
df.dtypes
```

Out[21]:

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

In [22]:

```
# Проверим наличие пустых значений
# Цикл по колонкам датасета
for col in df.columns:
    # Количество пустых значений - все значения заполнены
    temp_null_count = df[df[col].isnull()].shape[0]
    print('{} - {}'.format(col, temp_null_count))
```

```
sepal length (cm) - 0
sepal width (cm) - 0
petal length (cm) - 0
petal width (cm) - 0
target - 0
```

In [23]:

```
# Основные статистические характеристки набора данных
df.describe()
```

Out[23]:

| | sepal length (cm) | sepal width (cm) | petal length (cm) | petal width (cm) | target |
|---|---|---|---|---|---|
| count | 150.000000 | 150.000000 | 150.000000 | 150.000000 | 150.000000 |
| mean | 5.843333 | 3.057333 | 3.758000 | 1.199333 | 1.000000 |
| std | 0.828066 | 0.435866 | 1.765298 | 0.762238 | 0.819232 |
| min | 4.300000 | 2.000000 | 1.000000 | 0.100000 | 0.000000 |
| 25% | 5.100000 | 2.800000 | 1.600000 | 0.300000 | 0.000000 |
| 50% | 5.800000 | 3.000000 | 4.350000 | 1.300000 | 1.000000 |
| 75% | 6.400000 | 3.300000 | 5.100000 | 1.800000 | 2.000000 |
| max | 7.900000 | 4.400000 | 6.900000 | 2.500000 | 2.000000 |

Настройка отображения графиков

In [24]:

```
# Enable inline plots
%matplotlib inline

# Задание стиля графиков
sns.set(style="ticks")

# Задание формата графиков для сохранения высокого качества PNG
from IPython.display import set_matplotlib_formats
matplotlib_inline.backend_inline.set_matplotlib_formats("retina")
```

Зададим ширину текста, чтобы он влезал на А4

In [25]:

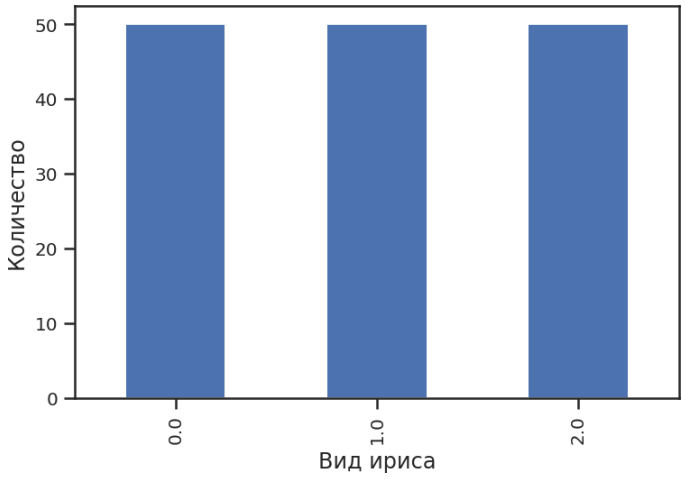
```
pd.set_option("display.width", 70)
```

# Визуальное исследование датасета

Оценим наиболее распространённый вид ириса

In [26]:

```
count_full = df.groupby("target")["target"].count().sort_values()
count_full.plot(x="Вид ириса", y="Количество", kind="bar", fontsize=10)
plt.xlabel("Вид ириса")
plt.ylabel("Количество")
plt.show()
```



Видно, что все виды ирисов одинаково распространены. Каждого вида — 50 штук.

## Диаграммы рассеяния

Диаграмма рассеяния для размеров чашелистика

In [27]:

```
fig, ax = plt.subplots(figsize=(10,10))
sns.scatterplot(ax=ax, x='sepal length (cm)', y='sepal width (cm)', data=df)
```

Out[27]:

```
<AxesSubplot:xlabel='sepal length (cm)', ylabel='sepal width (cm)'>
```

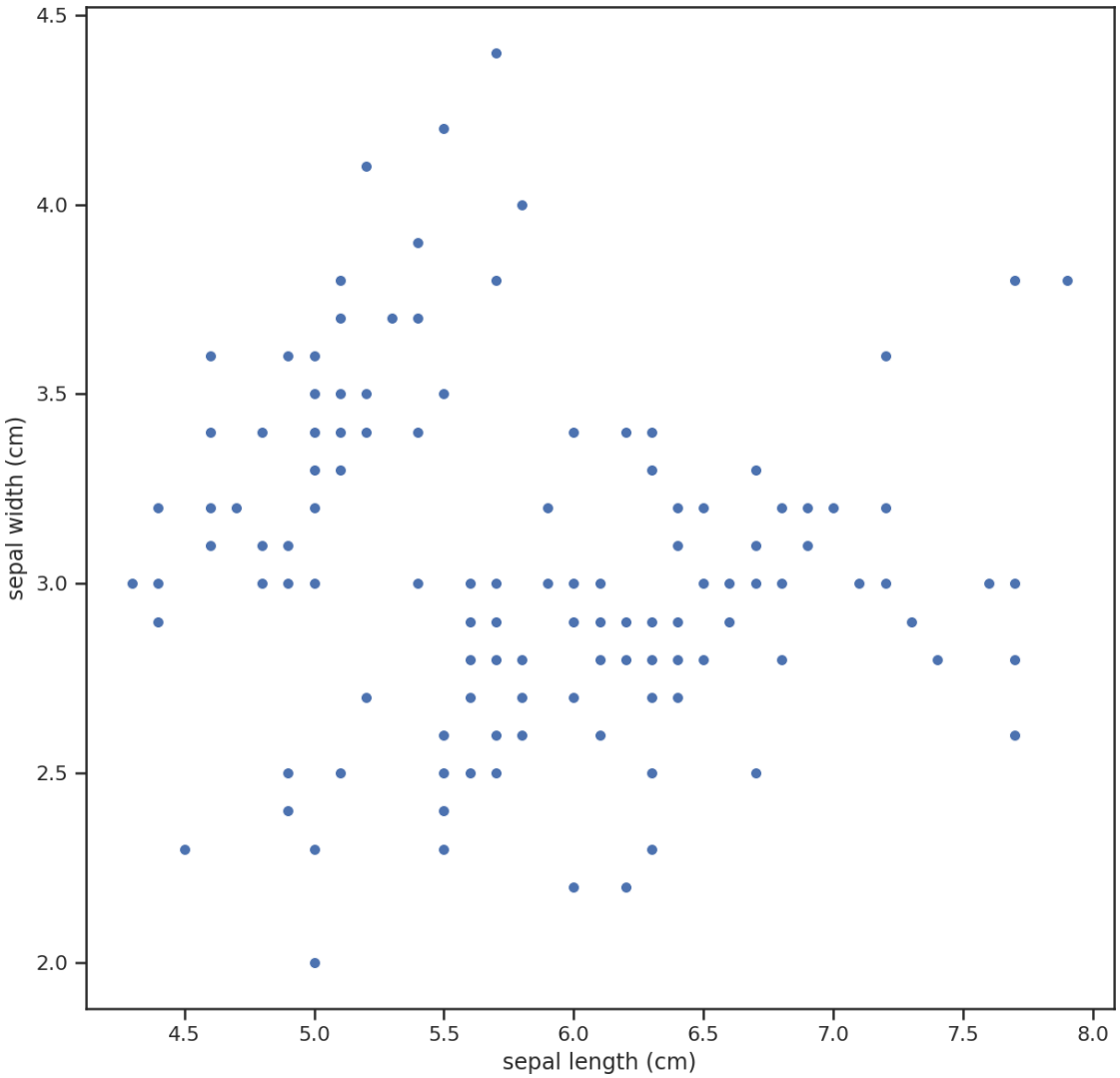


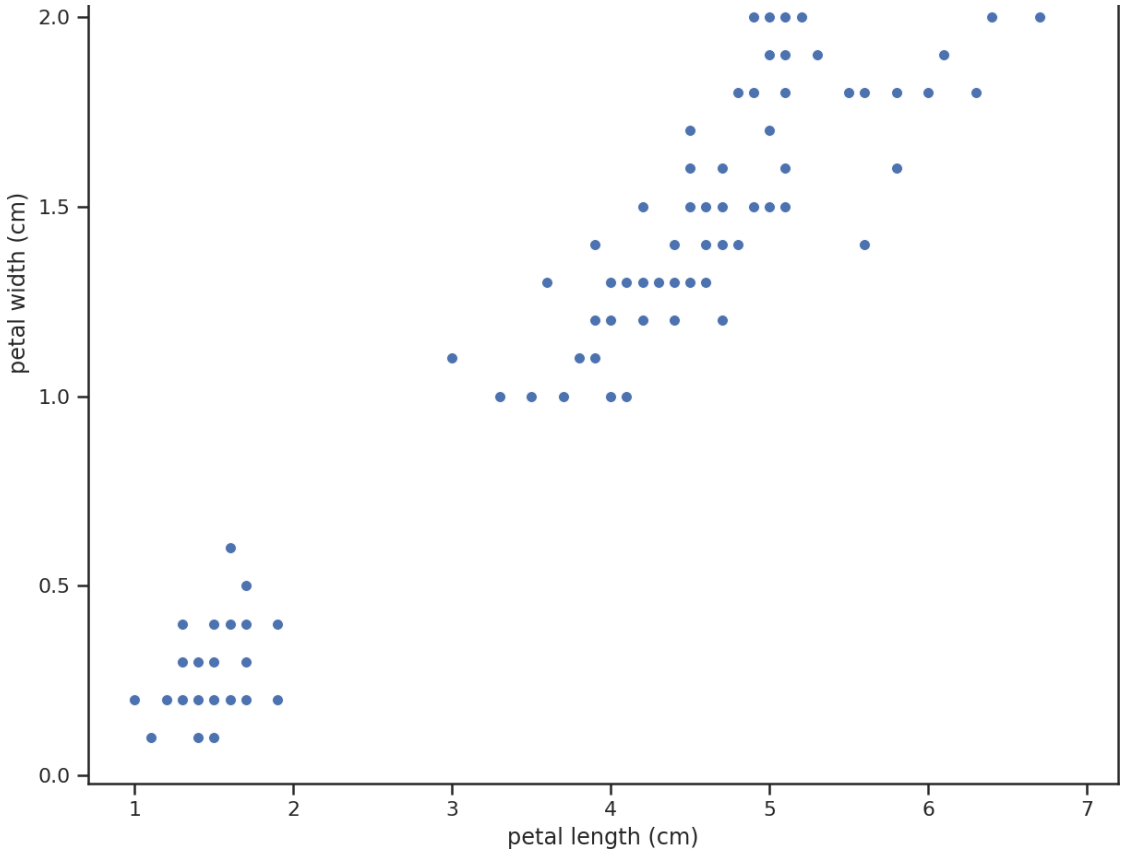

Диаграмма рассеяния для размеров лепестка

In [28]:

```
fig, ax = plt.subplots(figsize=(10,10))
sns.scatterplot(ax=ax, x='petal length (cm)', y='petal width (cm)', data=df)
```

Out[28]:

```
<AxesSubplot:xlabel='petal length (cm)', ylabel='petal width (cm)'>
```




## Гистограммы

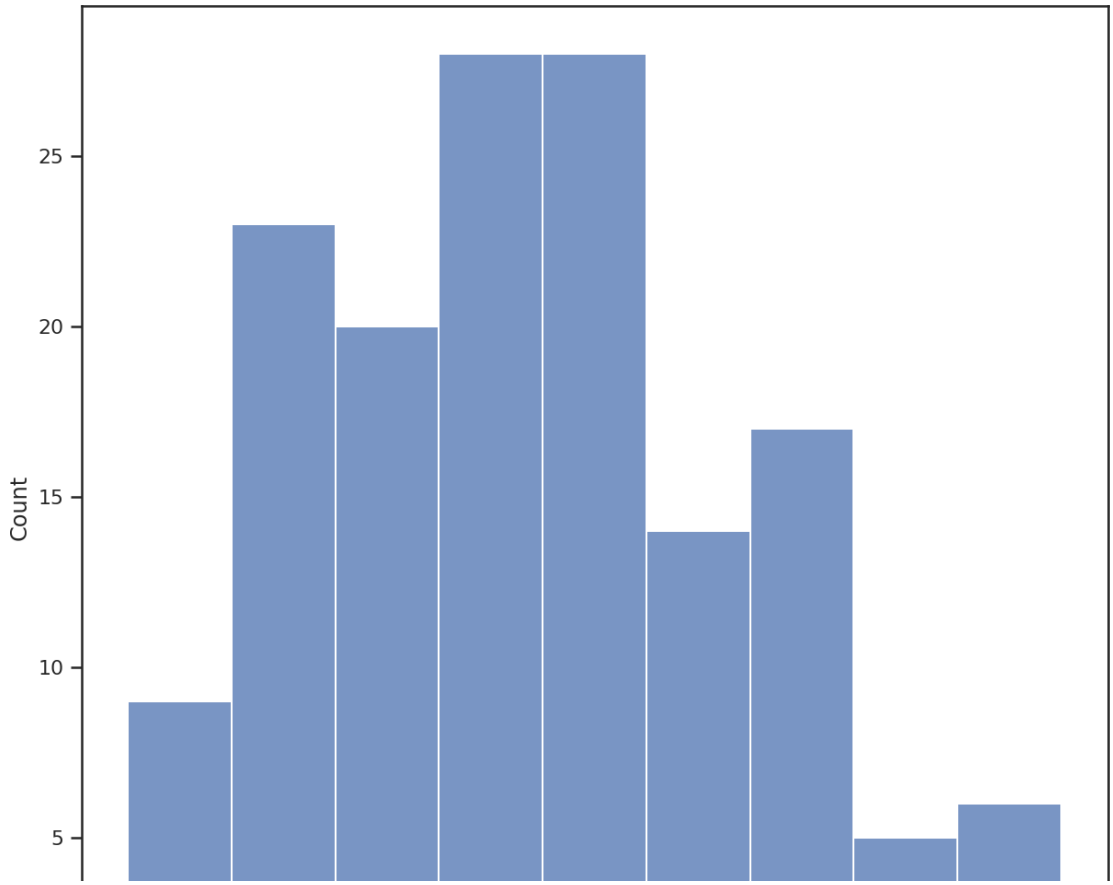
Гистограмма распределения длинны чашелистика

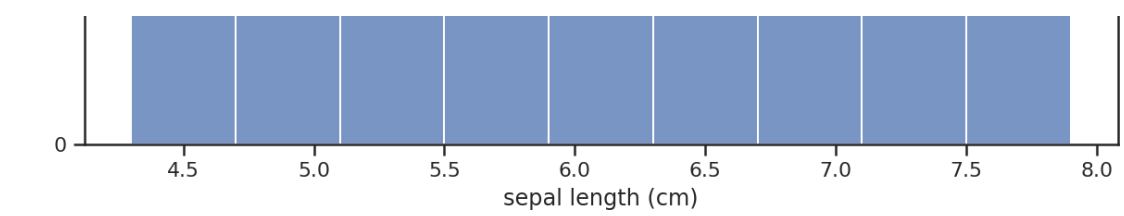
In [29]:

```
fig, ax = plt.subplots(figsize=(10,10))
sns.histplot(df['sepal length (cm)'])
```

Out[29]:

```
<AxesSubplot:xlabel='sepal length (cm)', ylabel='Count'>
```





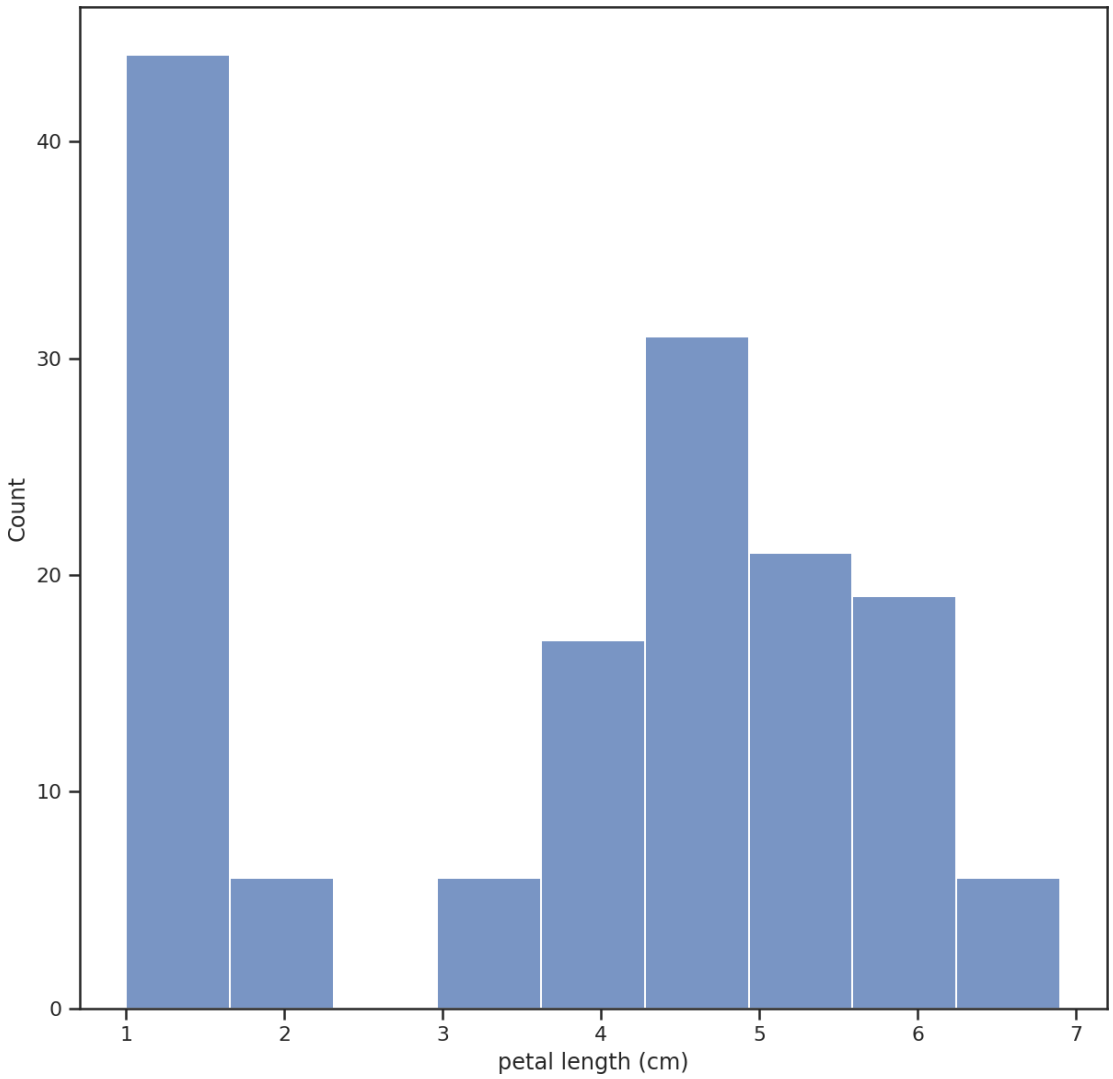

Гистограмма распределения длинны лепестка

In [30]:

```
fig, ax = plt.subplots(figsize=(10,10))
sns.histplot(df['petal length (cm)'])
```

Out[30]:

```
<AxesSubplot:xlabel='petal length (cm)', ylabel='Count'>
```



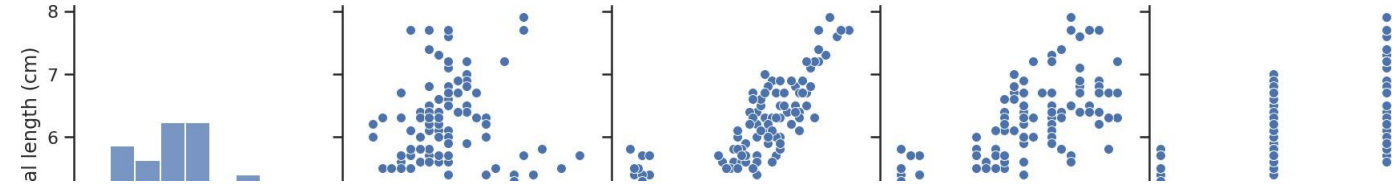

## Парные диаграммы

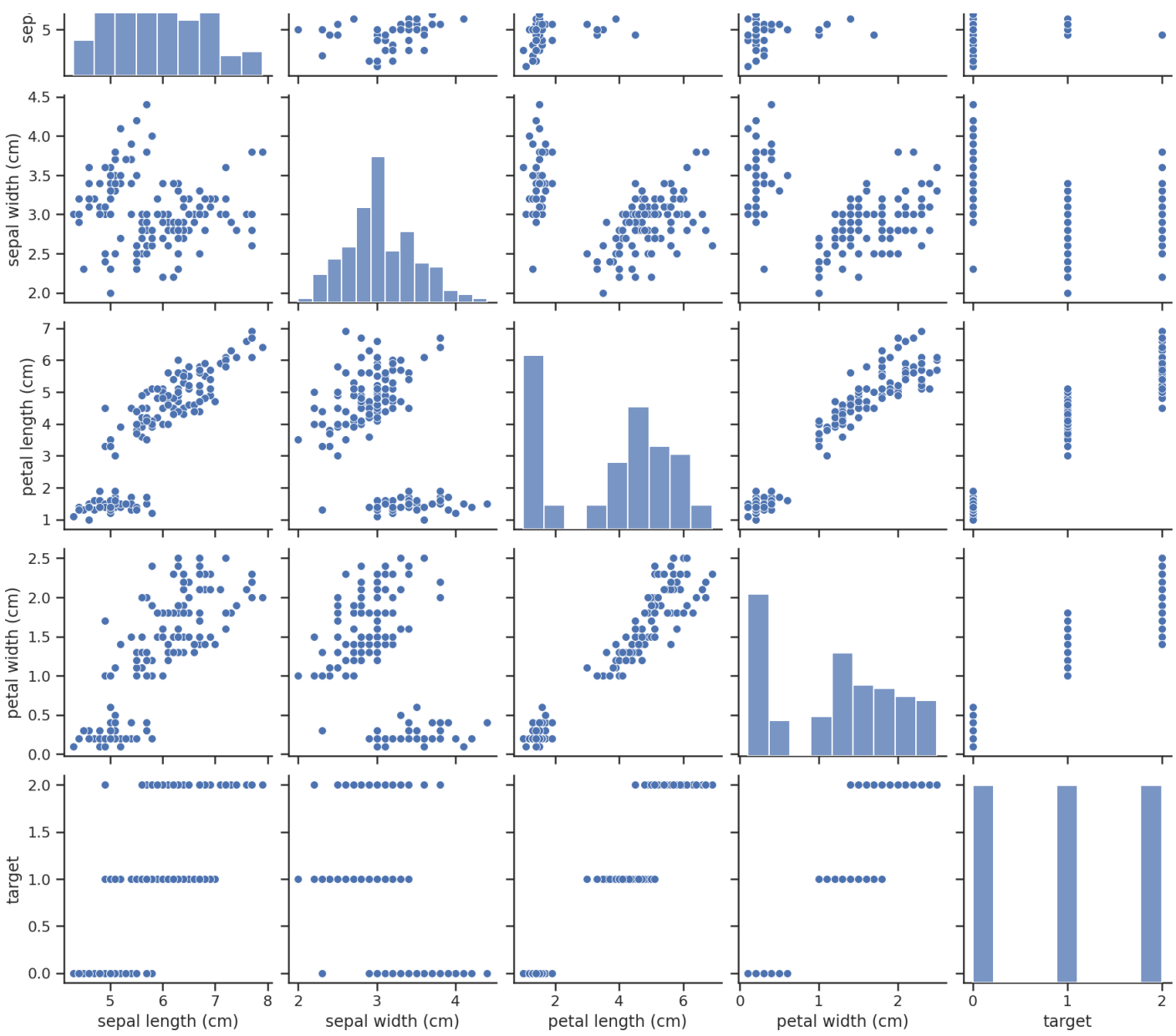
In [31]:

```
sns.pairplot(df)
```

Out[31]:

```
<seaborn.axisgrid.PairGrid at 0x7fb5fc3272e0>
```






С помощью парных диаграмм были легко получены различные гистограммы и диаграммы.

## Ящик с усами

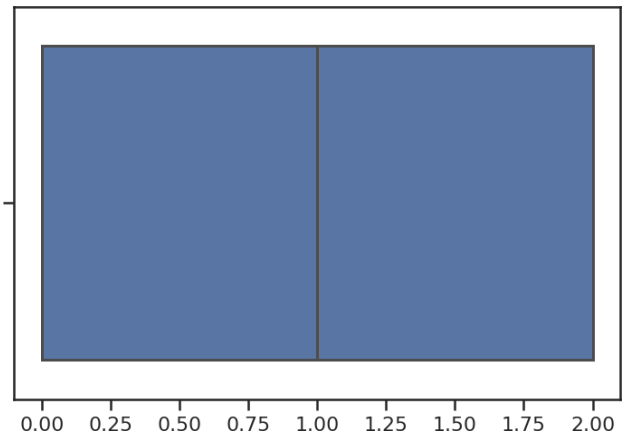
Вероятность получить определённый вид ириса

In [32]:

```
sns.boxplot(x=df['target'])
```

Out[32]:

```
<AxesSubplot:xlabel='target'>
```



target

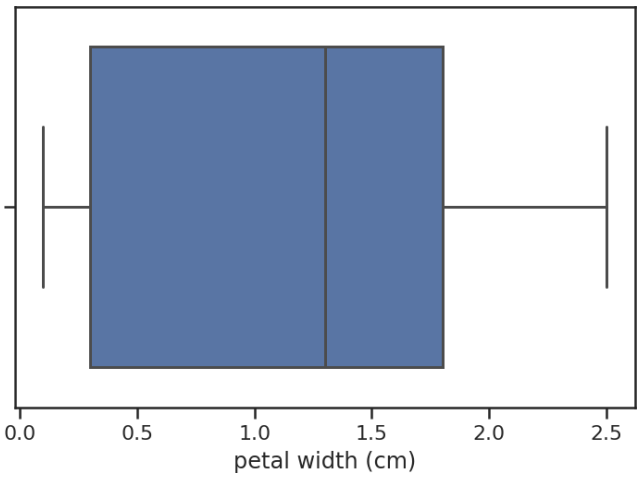
Вероятность найти лепесток определённой ширины

In [33]:

```
sns.boxplot(x=df['petal width (cm)'])
```

Out[33]:

```
<AxesSubplot:xlabel='petal width (cm)'>
```




## Информация о кореляции признаков

На основе коеффициента кореляции Пирса

In [34]:

```
df.corr()
```

Out[34]:

| | sepal length (cm) | sepal width (cm) | petal length (cm) | petal width (cm) | target |
|---|---|---|---|---|---|
| **sepal length (cm)** | 1.000000 | -0.117570 | 0.871754 | 0.817941 | 0.782561 |
| **sepal width (cm)** | -0.117570 | 1.000000 | -0.428440 | -0.366126 | -0.426658 |
| **petal length (cm)** | 0.871754 | -0.428440 | 1.000000 | 0.962865 | 0.949035 |
| **petal width (cm)** | 0.817941 | -0.366126 | 0.962865 | 1.000000 | 0.956547 |
| **target** | 0.782561 | -0.426658 | 0.949035 | 0.956547 | 1.000000 |

На основе корреляционной матрицы можно сделать следующие выводы:

- Целевой признак наиболее сильно коррелирует с шириной лепестка (0.95) и его длинной (0.94). Эти признаки обязательно следует оставить в модели.
- Целевой признак отчасти корелирует с длинной чашелистика (0.78). Этот признак стоит также оставить в модели.
- Целевой признак слабо корелирует с шириной чашелистика (-0.4). Скорее всего этот признак стоит исключить из модели, возможно он только ухудшит качество модели.
- Длинна чашелистика и длинна лепестка корелируют между собой (0.87). Возможно, стоит оставить один из этих признаков.

Визуалезируем кореляцию с помощью тёплой карты

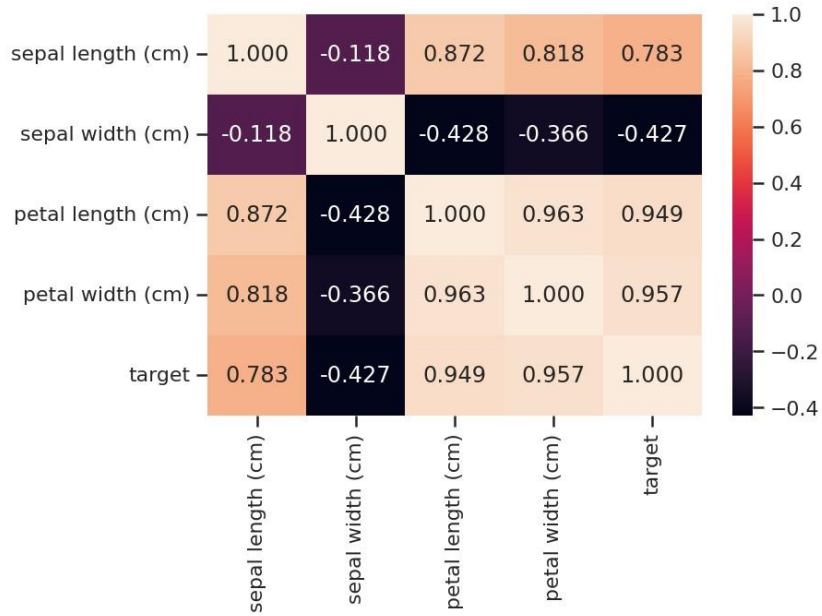
In [35]:

```
# Вывод значений в ячейках
```

```
sns.heatmap(df.corr(), annot=True, fmt='.3f')
```

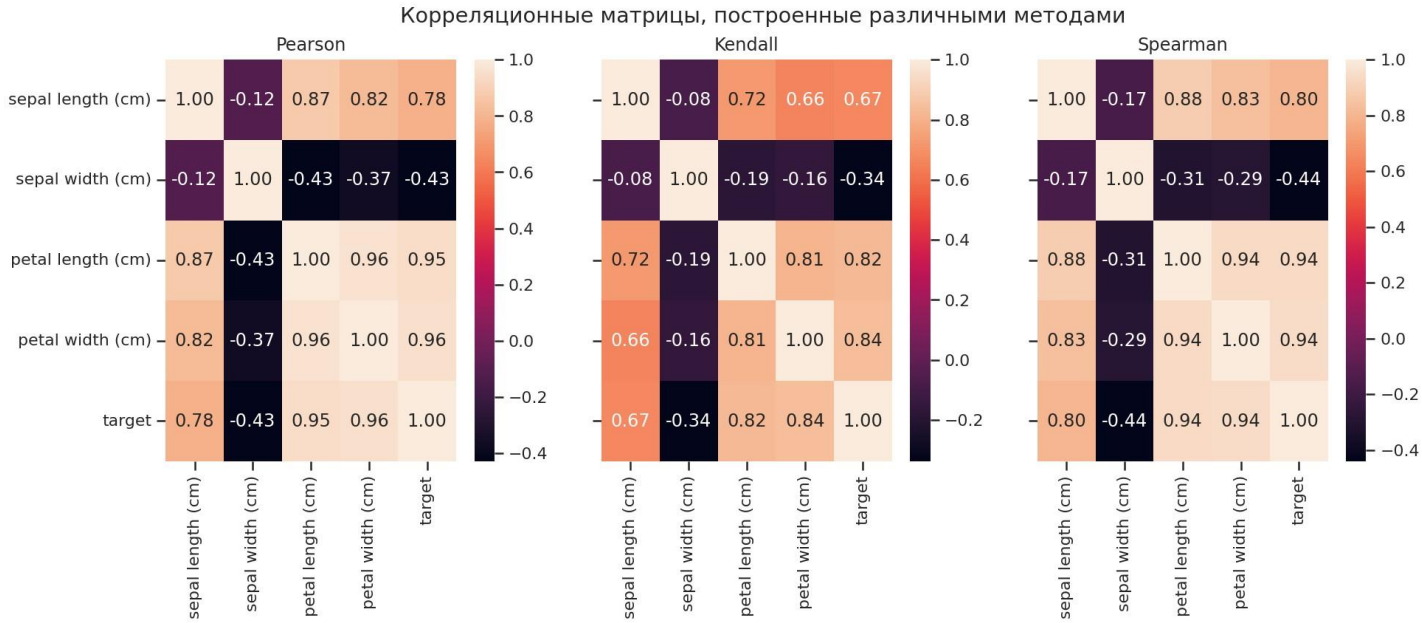
Out[35]:

<AxesSubplot:>




In [36]:

```
fig, ax = plt.subplots(1, 3, sharex='col', sharey='row', figsize=(15,5))
sns.heatmap(df.corr(method='pearson'), ax=ax[0], annot=True, fmt='.2f')
sns.heatmap(df.corr(method='kendall'), ax=ax[1], annot=True, fmt='.2f')
sns.heatmap(df.corr(method='spearman'), ax=ax[2], annot=True, fmt='.2f')
fig.suptitle('Корреляционные матрицы, построенные различными методами')
ax[0].title.set_text('Pearson')
ax[1].title.set_text('Kendall')
ax[2].title.set_text('Spearman')
```




Необходимо отметить, что тепловая карта не очень хорошо подходит для определения корреляции нецелевых признаков между собой.

In [ ]:

```
```