МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
им. Н.Э. Баумана

Кафедра «Систем обработки информации и управления»

# Лабораторная работа №1

по дисциплине

# «Методы машинного обучения»

ИСПОЛНИТЕЛЬ: Сукач Е.А.
ФИО

группа ИУ5-23М

_____
подпись

"__"_____2020 г.

ПРЕПОДАВАТЕЛЬ: Гапанюк Ю. Е.
ФИО

_____
подпись

"__"_____2020 г.

Москва – 2020

# 1) Текстовое описание набора данных

В качестве набора данных мы будем использовать набор данных для распознавания вина. https://archive.ics.uci.edu/ml/machine-learning-databases/wine/wine.data (https://archive.ics.uci.edu/ml/machine-learning-databases/wine/wine.data)

Данные представляют собой результаты химического анализа вин, выращенных в одном регионе Италии тремя различными культиваторами. Существует тринадцать различных измерений, проведенных для разных компонентов, найденных в трех типах вина.

Датасет состоит из одного файла:

- wine.data - набора характеристик вина

Характеристики датасета:

- Количество экземпляров: 178 (по 50 в каждом из трех классов)
- Количество атрибутов: 13 числовых, прогнозирующих атрибутов
- Информация об атрибутах:
  Алкоголь
  Яблочная кислота
  Зола
  Щелочность золы
  Магний
  Всего фенолов
  Флавоноиды
  Нефлаваноидные фенолы
  Proanthocyanins
  Интенсивность цвета
  Оттенок
  OD280 / OD315 разбавленных вин
  Пролин
- Классы: class_0 (59), class_1 (71), class_2 (48)

# Импорт библиотек

In [4]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style="ticks")
from sklearn.datasets import load_wine
```

# Загрузка данных

Преобразование наборов данных Scikit-learn в Pandas Dataframe

In [5]:
```
wine = load_wine()
type(wine)
```

Out[5]:
```
sklearn.utils.Bunch
```

In [6]:
```
for x in wine:
    print(x)
```

```
data
target
target_names
DESCR
feature_names
```

In [7]:
```
wine['target_names']
```

Out[7]:
```
array(['class_0', 'class_1', 'class_2'], dtype='<U7')
```

In [8]:
```
wine['feature_names']
```

Out[8]:
```
['alcohol',
 'malic_acid',
 'ash',
 'alcalinity_of_ash',
 'magnesium',
 'total_phenols',
 'flavanoids',
 'nonflavanoid_phenols',
 'proanthocyanins',
 'color_intensity',
 'hue',
 'od280/od315_of_diluted_wines',
 'proline']
```

In [9]:
```
wine['data'].shape
```

Out[9]: (178, 13)

In [10]:
```
wine['target'].shape
```

Out[10]: (178,)

Преобразование в Pandas DataFrame.

In [11]:
```
wine1 = pd.DataFrame(data= np.c_[wine['data'], wine['target']],
                     columns= wine['feature_names'] + ['target'])
```

In [12]:
```
wine1
```

Out[12]:

| | alcohol | malic_acid | ash | alcalinity_of_ash | magnesium | total_phenols | flavanoids | non |
|---|---|---|---|---|---|---|---|---|
| 0 | 14.23 | 1.71 | 2.43 | 15.6 | 127.0 | 2.80 | 3.06 | |
| 1 | 13.20 | 1.78 | 2.14 | 11.2 | 100.0 | 2.65 | 2.76 | |
| 2 | 13.16 | 2.36 | 2.67 | 18.6 | 101.0 | 2.80 | 3.24 | |
| 3 | 14.37 | 1.95 | 2.50 | 16.8 | 113.0 | 3.85 | 3.49 | |
| 4 | 13.24 | 2.59 | 2.87 | 21.0 | 118.0 | 2.80 | 2.69 | |
| ... | ... | ... | ... | ... | ... | ... | ... | |
| 173 | 13.71 | 5.65 | 2.45 | 20.5 | 95.0 | 1.68 | 0.61 | |
| 174 | 13.40 | 3.91 | 2.48 | 23.0 | 102.0 | 1.80 | 0.75 | |
| 175 | 13.27 | 4.28 | 2.26 | 20.0 | 120.0 | 1.59 | 0.69 | |
| 176 | 13.17 | 2.59 | 2.37 | 20.0 | 120.0 | 1.65 | 0.68 | |
| 177 | 14.13 | 4.10 | 2.74 | 24.5 | 96.0 | 2.05 | 0.76 | |

178 rows × 14 columns

# 2) Основные характеристики датасета

In [13]:
```
# Первые 5 строк датасета
wine1.head()
```

Out[13]:

| | alcohol | malic_acid | ash | alcalinity_of_ash | magnesium | total_phenols | flavanoids | nonfla |
|---|---|---|---|---|---|---|---|---|
| 0 | 14.23 | 1.71 | 2.43 | 15.6 | 127.0 | 2.80 | 3.06 | |
| 1 | 13.20 | 1.78 | 2.14 | 11.2 | 100.0 | 2.65 | 2.76 | |
| 2 | 13.16 | 2.36 | 2.67 | 18.6 | 101.0 | 2.80 | 3.24 | |
| 3 | 14.37 | 1.95 | 2.50 | 16.8 | 113.0 | 3.85 | 3.49 | |
| 4 | 13.24 | 2.59 | 2.87 | 21.0 | 118.0 | 2.80 | 2.69 | |

In [14]:
```
#Размер датасета - 178 строк, 14 колонок
wine1.shape
```

Out[14]:
```
(178, 14)
```

In [15]:
```
total_count = wine1.shape[0]
print('Всего строк: {}'.format(total_count))
```

```
Всего строк: 178
```

In [16]:
```
# Список колонок с типами данных
wine1.dtypes
```

Out[16]:
```
alcohol                         float64
malic_acid                      float64
ash                             float64
alcalinity_of_ash               float64
magnesium                       float64
total_phenols                   float64
flavanoids                      float64
nonflavanoid_phenols            float64
proanthocyanins                 float64
color_intensity                 float64
hue                             float64
od280/od315_of_diluted_wines    float64
proline                         float64
target                          float64
dtype: object
```

In [17]:
```
# Проверим наличие пустых значений
# Цикл по колонкам датасета
for col in wine1.columns:
    # Количество пустых значений - все значения заполнены
    temp_null_count = wine1[wine1[col].isnull()].shape[0]
    print('{} - {}'.format(col, temp_null_count))
```

```
alcohol - 0
malic_acid - 0
ash - 0
alcalinity_of_ash - 0
magnesium - 0
total_phenols - 0
flavanoids - 0
nonflavanoid_phenols - 0
proanthocyanins - 0
color_intensity - 0
hue - 0
od280/od315_of_diluted_wines - 0
proline - 0
target - 0
```

In [18]:
```
# Основные статистические характеристки набора данных
wine1.describe()
```

Out[18]:

| | alcohol | malic_acid | ash | alcalinity_of_ash | magnesium | total_phenols | fla |
|---|---|---|---|---|---|---|---|
| count | 178.000000 | 178.000000 | 178.000000 | 178.000000 | 178.000000 | 178.000000 | 178 |
| mean | 13.000618 | 2.336348 | 2.366517 | 19.494944 | 99.741573 | 2.295112 | 2 |
| std | 0.811827 | 1.117146 | 0.274344 | 3.339564 | 14.282484 | 0.625851 | 0 |
| min | 11.030000 | 0.740000 | 1.360000 | 10.600000 | 70.000000 | 0.980000 | 0 |
| 25% | 12.362500 | 1.602500 | 2.210000 | 17.200000 | 88.000000 | 1.742500 | 1 |
| 50% | 13.050000 | 1.865000 | 2.360000 | 19.500000 | 98.000000 | 2.355000 | 2 |
| 75% | 13.677500 | 3.082500 | 2.557500 | 21.500000 | 107.000000 | 2.800000 | 2 |
| max | 14.830000 | 5.800000 | 3.230000 | 30.000000 | 162.000000 | 3.880000 | 5 |

In [19]:
```
# Определим уникальные значения для target
wine1['target'].unique()
```

Out[19]:
```
array([0., 1., 2.])
```

# 3) Визуальное исследование датасета

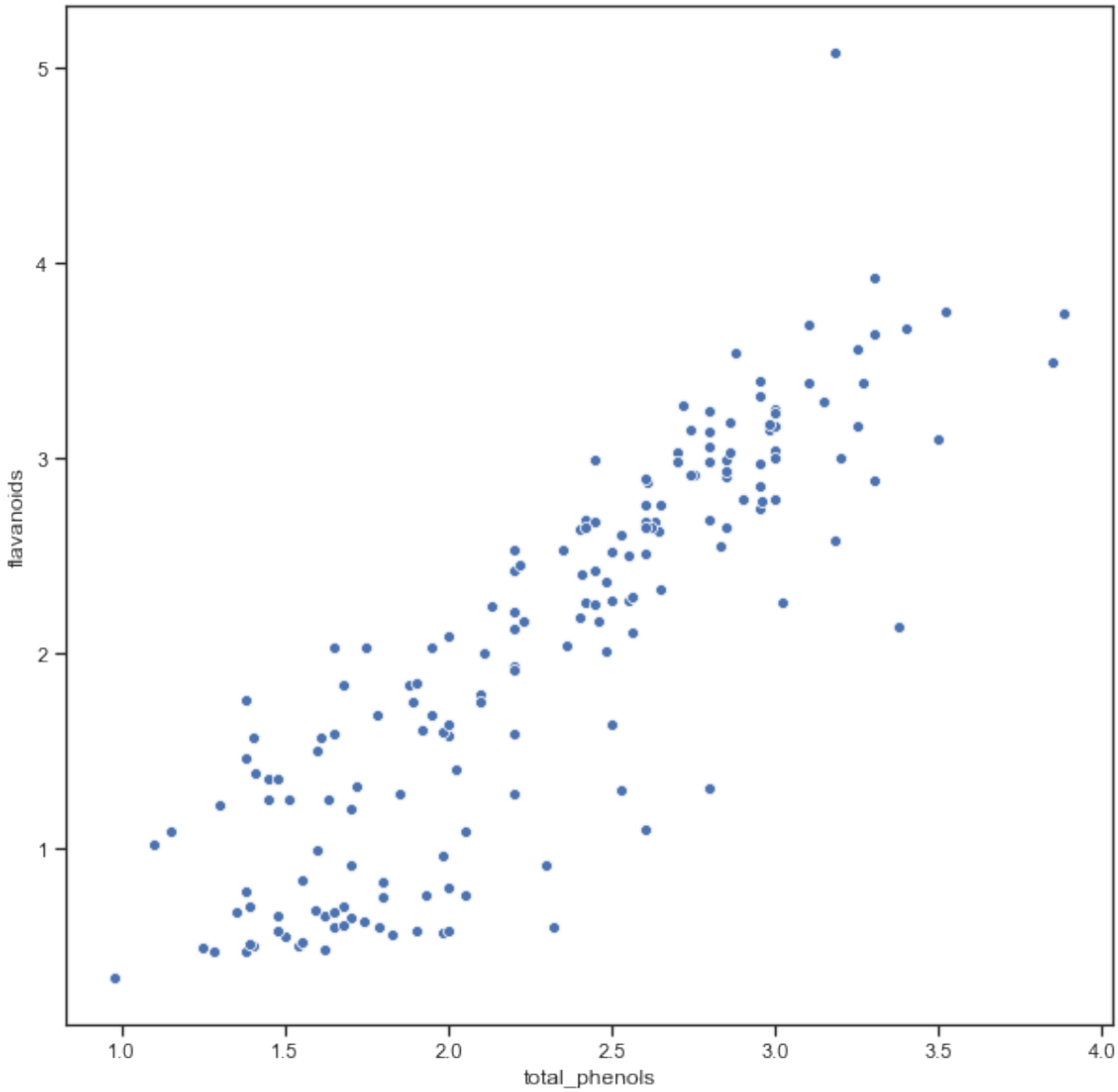
Для визуального исследования могут быть использованы различные виды диаграмм, мы построим только некоторые варианты диаграмм, которые используются достаточно часто.

## Диаграмма рассеяния (https://en.wikipedia.org/wiki/Scatter_plot)

Позволяет построить распределение двух колонок данных и визуально обнаружить наличие зависимости. Не предполагается, что значения упорядочены.

In [20]:
```
fig, ax = plt.subplots(figsize=(10,10))
sns.scatterplot(ax=ax, x='total_phenols', y='flavanoids', data=wine
```

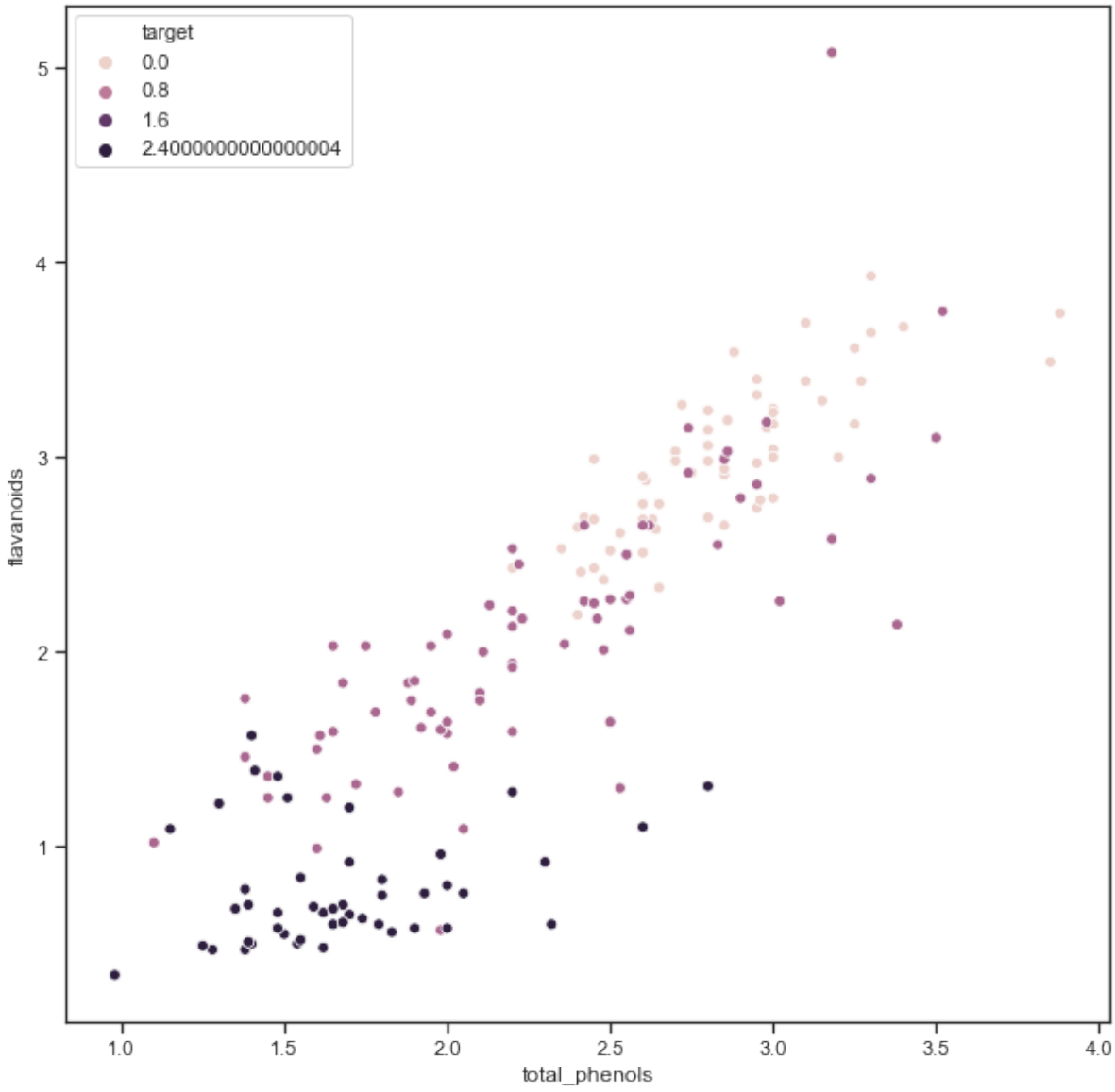
Out[20]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1a20cb48d0>
```




Посмотрим насколько на эту зависимость влияет целевой признак.

In [21]:
```
fig, ax = plt.subplots(figsize=(10,10))
sns.scatterplot(ax=ax, x='total_phenols', y='flavanoids', data=wine
```

Out[21]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1a2167e550>
```

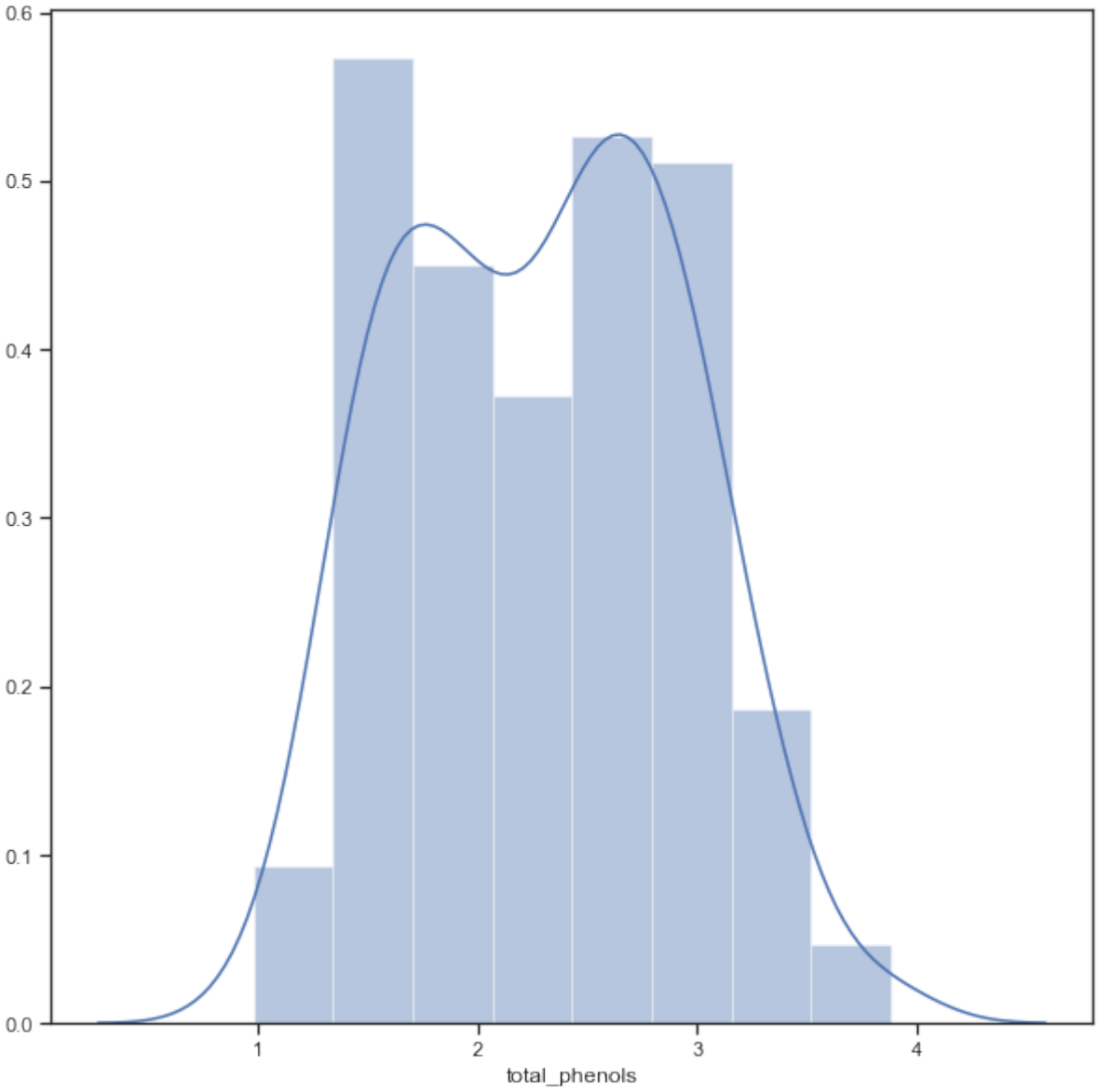



## Гистограмма (https://en.wikipedia.org/wiki/Histogram)

Позволяет оценить плотность вероятности распределения данных.

In [22]:
```
fig, ax = plt.subplots(figsize=(10,10))
sns.distplot(wine1['total_phenols'])
```

Out[22]: <matplotlib.axes._subplots.AxesSubplot at 0x1a218bf650>

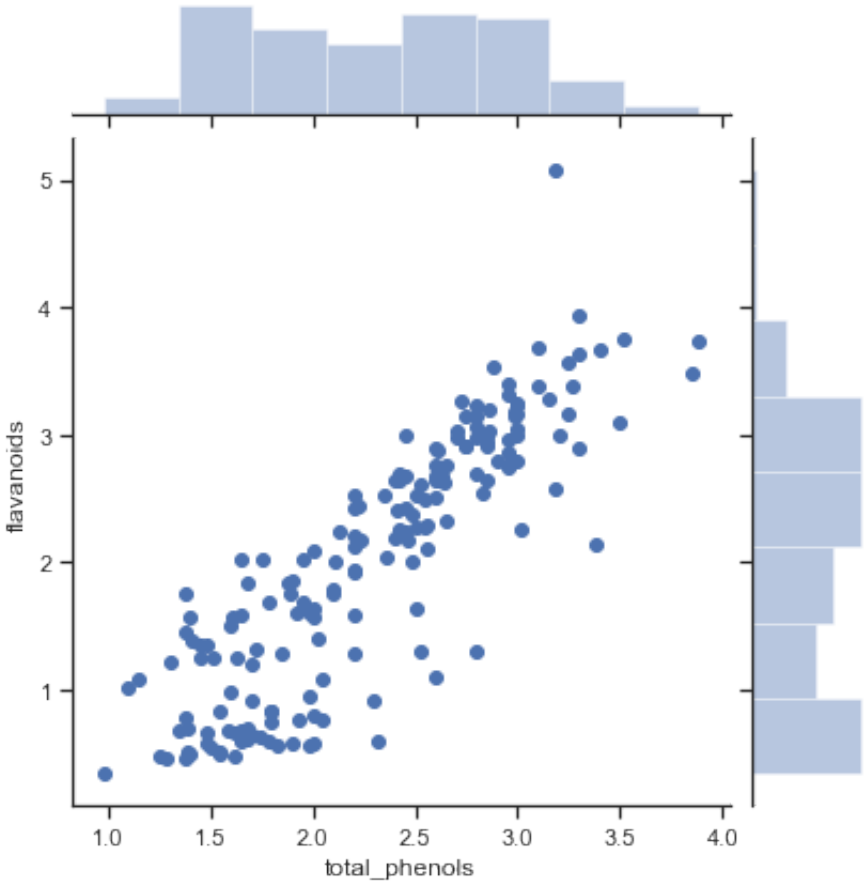



## Jointplot

Комбинация гистограмм и диаграмм рассеивания.

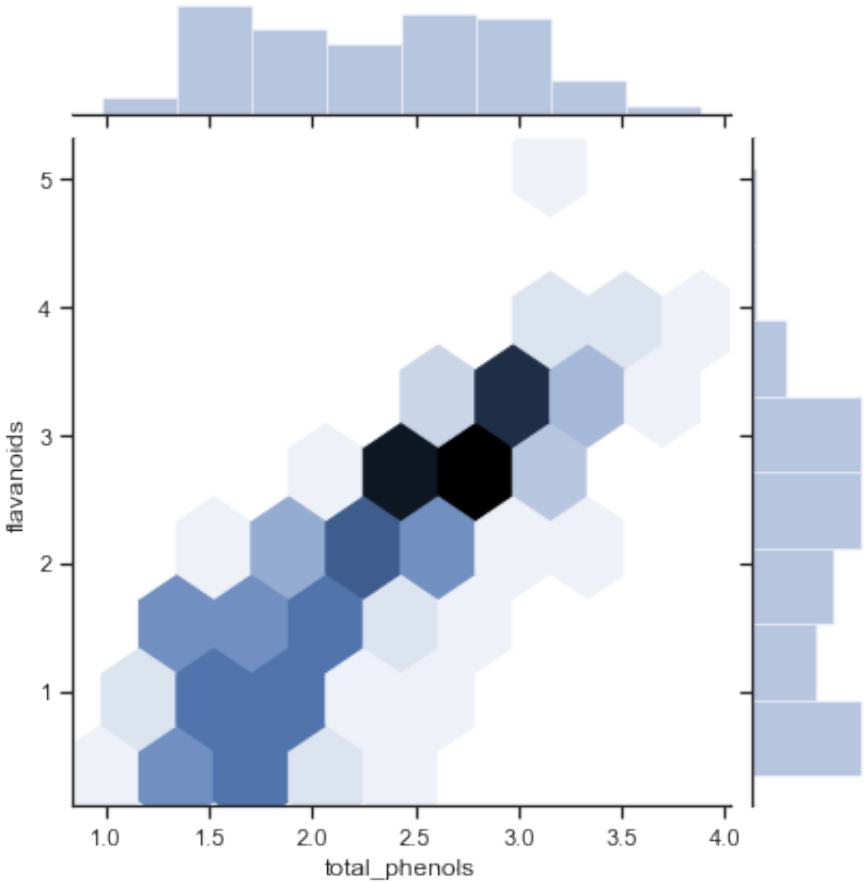
In [23]:
```
sns.jointplot(x='total_phenols', y='flavanoids', data=wine1)
```

Out[23]: <seaborn.axisgrid.JointGrid at 0x1a21abf110>



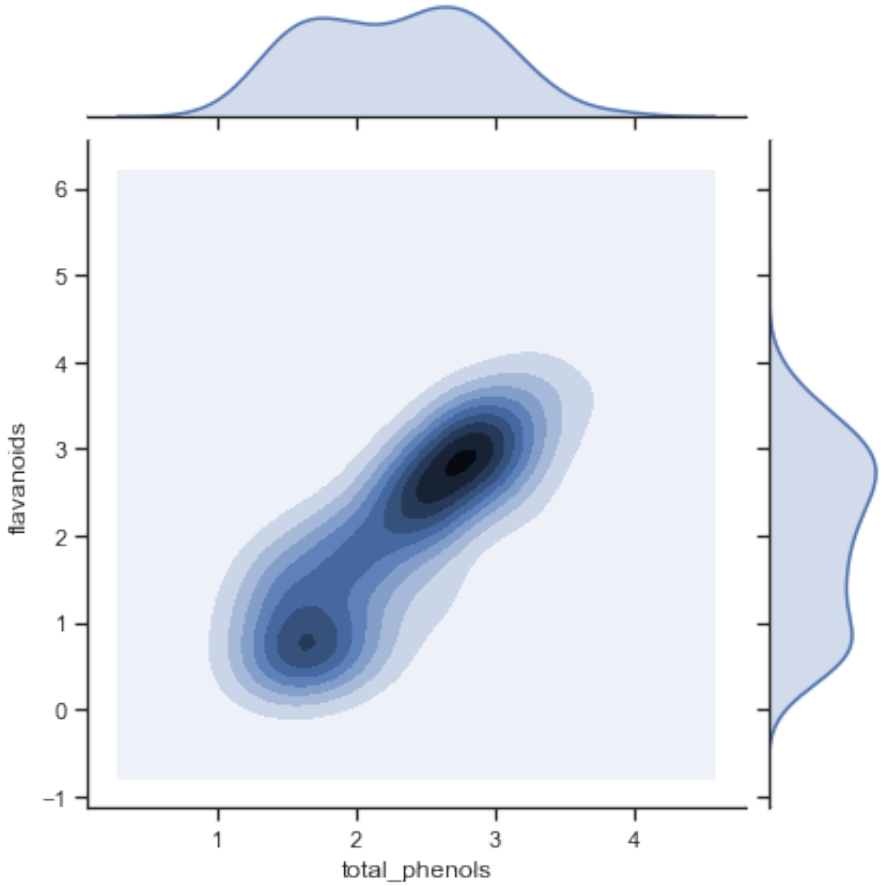
In [24]:
```
sns.jointplot(x='total_phenols', y='flavanoids', data=wine1,kind="h
```

Out[24]: <seaborn.axisgrid.JointGrid at 0x1a21d6b4d0>



In [25]:
```
sns.jointplot(x='total_phenols', y='flavanoids', data=wine1,kind="k
```

Out[25]:
```
<seaborn.axisgrid.JointGrid at 0x1a21d6bc50>
```



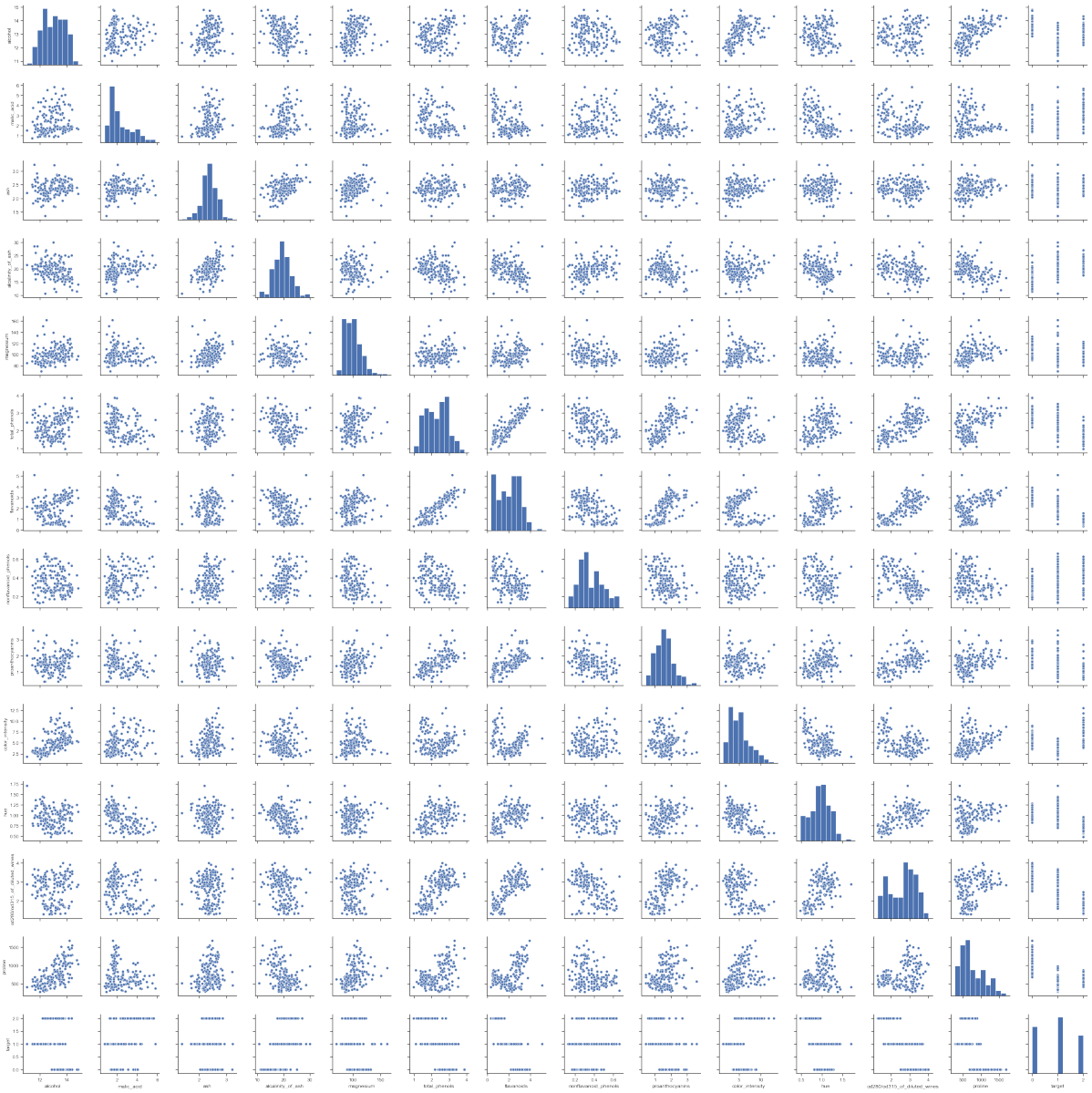

## "Парные диаграммы"

Комбинация гистограмм и диаграмм рассеивания для всего набора данных.

Выводится матрица графиков. На пересечении строки и столбца, которые соответстуют двум показателям, строится диаграмма рассеивания. В главной диагонали матрицы строятся гистограммы распределения соответствующих показателей.

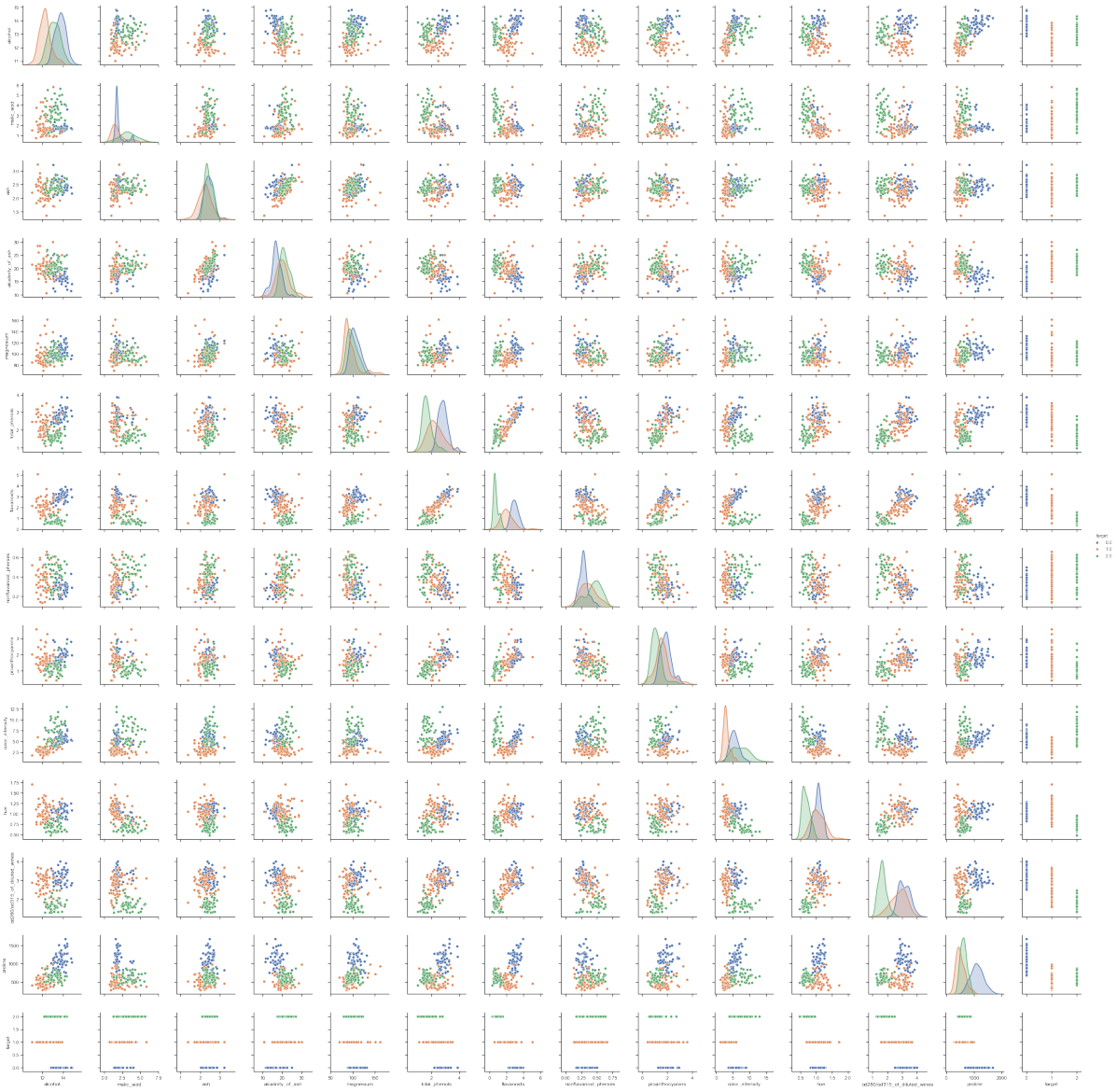
In [26]:
```
sns.pairplot(wine1)
```

Out[26]: <seaborn.axisgrid.PairGrid at 0x1a2042ec10>



In [27]:
```
sns.pairplot(wine1, hue="target")
```

Out[27]: <seaborn.axisgrid.PairGrid at 0x1a286a3590>

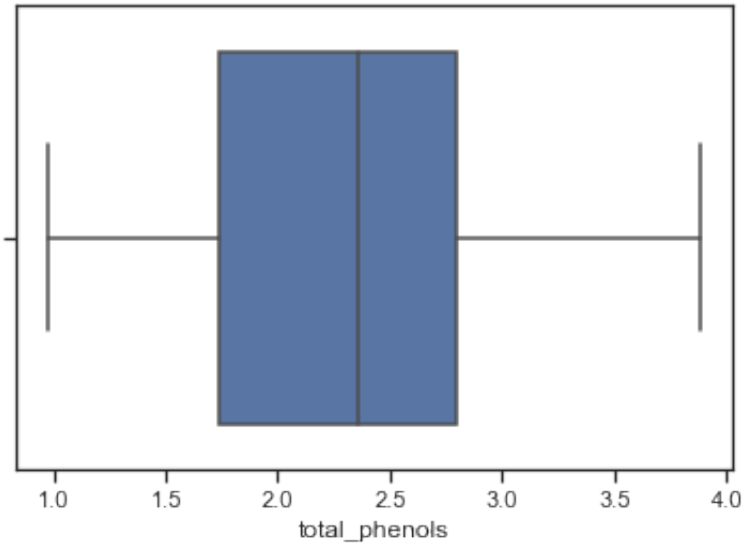


## [Ящик с усами (https://en.wikipedia.org/wiki/Box_plot)](https://en.wikipedia.org/wiki/Box_plot)

Отображает одномерное распределение вероятности.

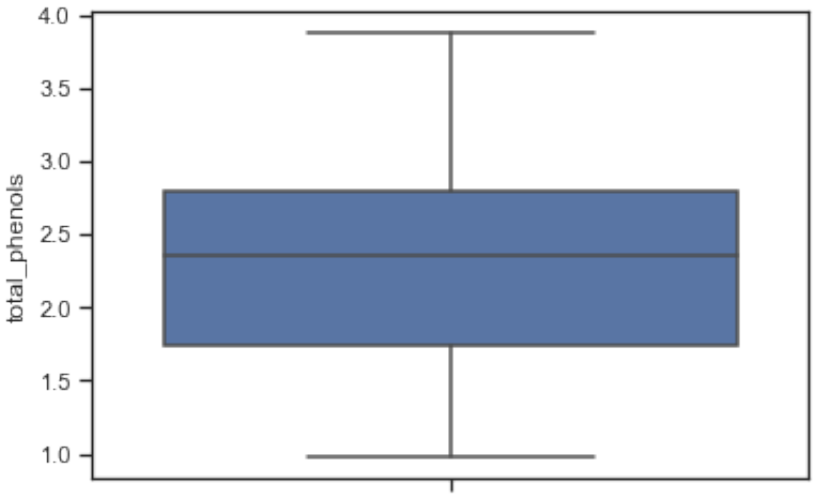
In [28]:
```
sns.boxplot(x=wine1['total_phenols'])
```

Out[28]: <matplotlib.axes._subplots.AxesSubplot at 0x1a3137ff50>



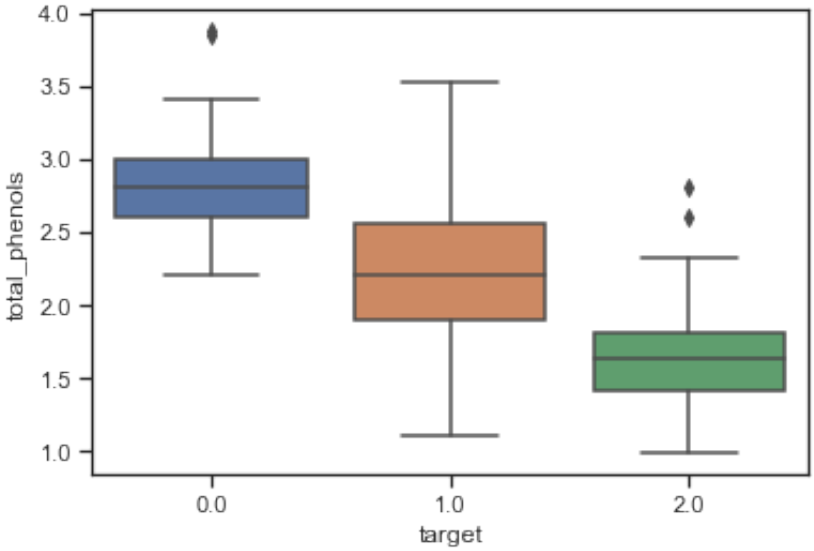

In [29]:
```
# По вертикали
sns.boxplot(y=wine1['total_phenols'])
```

Out[29]: <matplotlib.axes._subplots.AxesSubplot at 0x1a307a5510>



In [30]:
```
# Распределение параметра total_phenols сгруппированные по target.
sns.boxplot(x='target', y='total_phenols', data=wine1)
```

Out[30]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1a31303890>
```

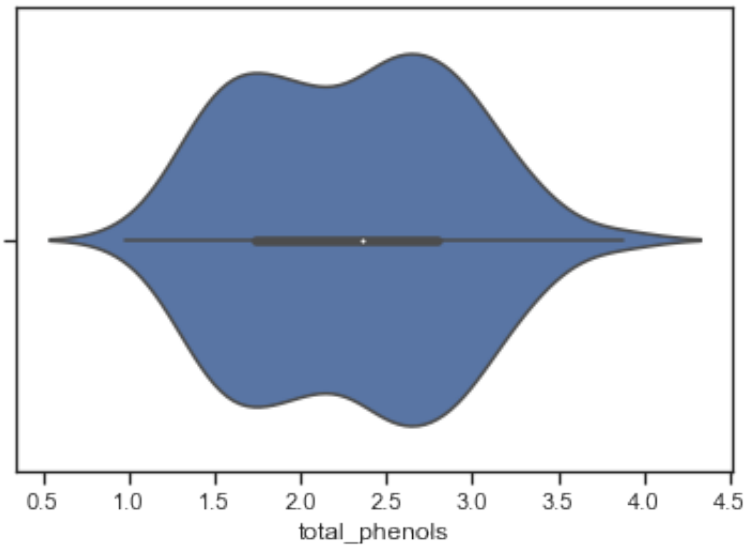



## Violin plot (https://en.wikipedia.org/wiki/Violin_plot)

Похоже на предыдущую диаграмму, но по краям отображаются распределения плотности - https://en.wikipedia.org/wiki/Kernel_density_estimation (https://en.wikipedia.org/wiki/Kernel_density_estimation)

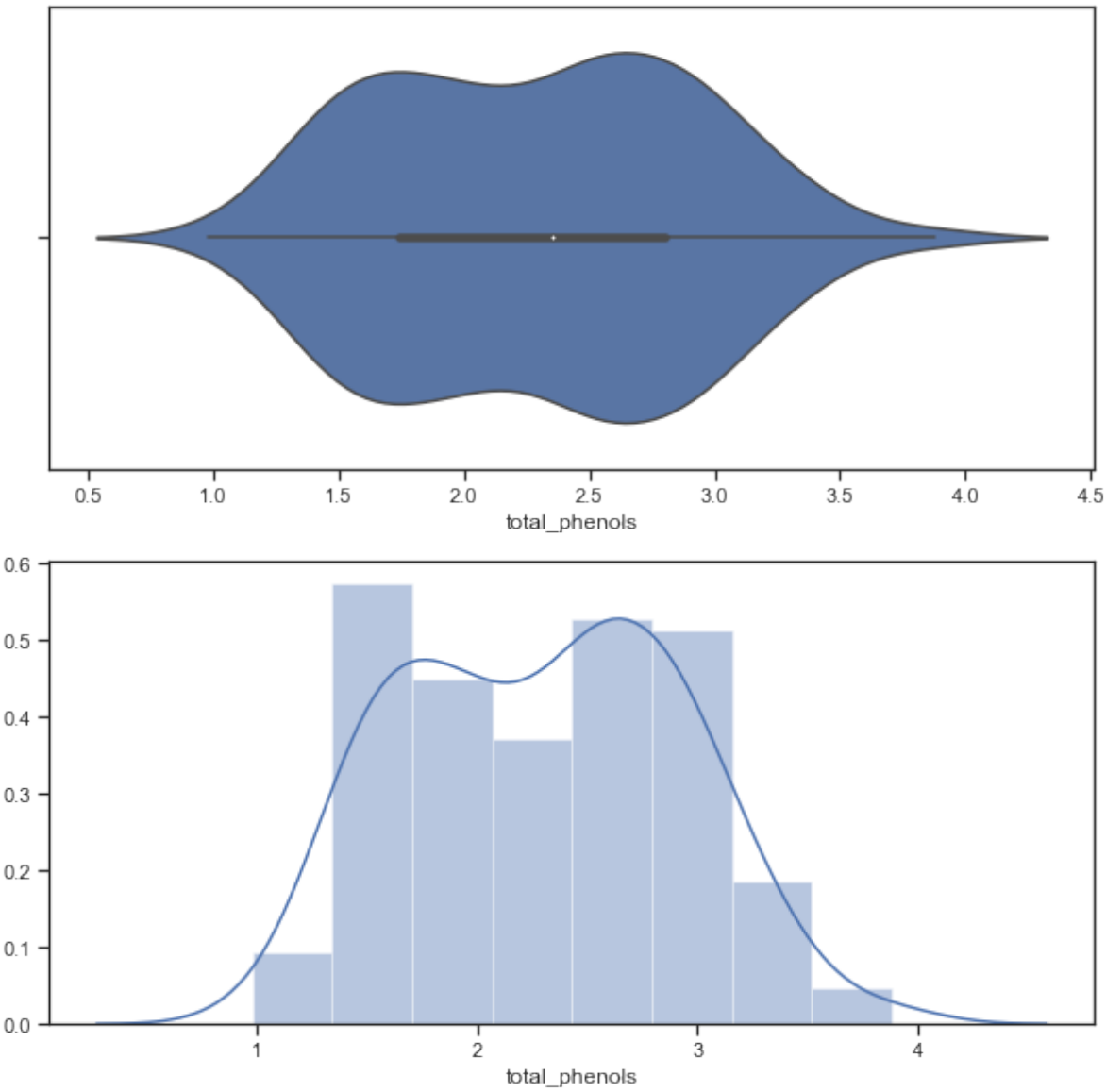
In [31]:
```
sns.violinplot(x=wine1['total_phenols'])
```

Out[31]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1a28955c50>
```



In [32]:
```
fig, ax = plt.subplots(2, 1, figsize=(10,10))
sns.violinplot(ax=ax[0], x=wine1['total_phenols'])
sns.distplot(wine1['total_phenols'], ax=ax[1])
```

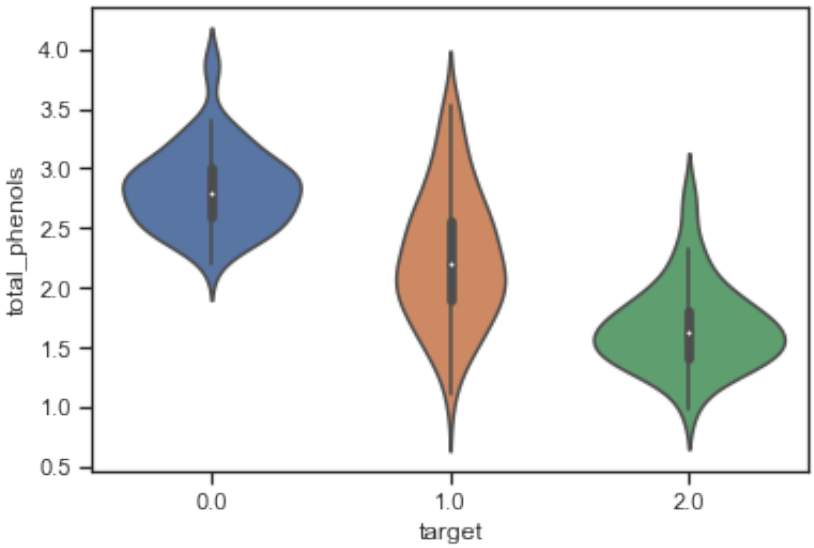
Out[32]: <matplotlib.axes._subplots.AxesSubplot at 0x1a28ef8410>




Из приведенных графиков видно, что violinplot действительно показывает распределение плотности.

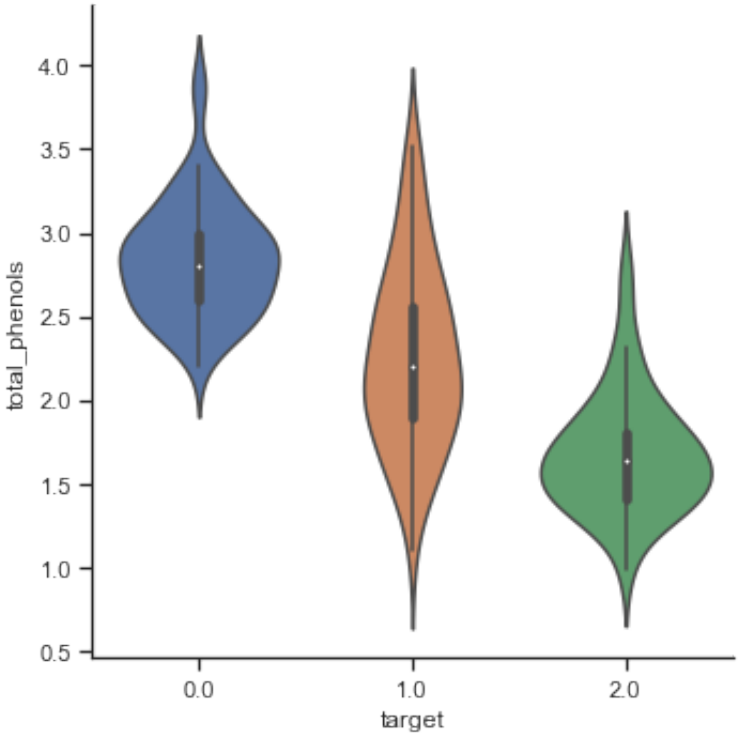
In [33]:
```
# Распределение параметра total_phenols сгруппированные по target.
sns.violinplot(x='target', y='total_phenols', data=wine1)
```

Out[33]: <matplotlib.axes._subplots.AxesSubplot at 0x1a29276e50>




In [34]:
```
sns.catplot(y='total_phenols', x='target', data=wine1, kind="violin
```

Out[34]: <seaborn.axisgrid.FacetGrid at 0x1a28fd38d0>




# 4) Информация о корреляции признаков

Проверка корреляции признаков позволяет решить две задачи:

1. Понять какие признаки (колонки датасета) наиболее сильно коррелируют с целевым признаком (в нашем примере это колонка "target"). Именно эти признаки будут наиболее информативными для моделей машинного обучения. Признаки, которые слабо коррелируют с целевым признаком, можно попробовать исключить из построения модели, иногда это повышает качество модели. Нужно отметить, что некоторые алгоритмы машинного обучения автоматически определяют ценность того или иного признака для построения модели.
2. Понять какие нецелевые признаки линейно зависимы между собой. Линейно зависимые признаки, как правило, очень плохо влияют на качество моделей. Поэтому если несколько признаков линейно зависимы, то для построения модели из них выбирают какой-то один признак.

In [35]:
```
wine1.corr()
```

Out[35]:

| | alcohol | malic_acid | ash | alcalinity_of_ash | magnesium |
|---|---|---|---|---|---|
| alcohol | 1.000000 | 0.094397 | 0.211545 | -0.310235 | 0.270798 |
| malic_acid | 0.094397 | 1.000000 | 0.164045 | 0.288500 | -0.054575 |
| ash | 0.211545 | 0.164045 | 1.000000 | 0.443367 | 0.286587 |
| alcalinity_of_ash | -0.310235 | 0.288500 | 0.443367 | 1.000000 | -0.083333 |
| magnesium | 0.270798 | -0.054575 | 0.286587 | -0.083333 | 1.000000 |
| total_phenols | 0.289101 | -0.335167 | 0.128980 | -0.321113 | 0.214401 |
| flavanoids | 0.236815 | -0.411007 | 0.115077 | -0.351370 | 0.195784 |
| nonflavanoid_phenols | -0.155929 | 0.292977 | 0.186230 | 0.361922 | -0.256294 |
| proanthocyanins | 0.136698 | -0.220746 | 0.009652 | -0.197327 | 0.236441 |
| color_intensity | 0.546364 | 0.248985 | 0.258887 | 0.018732 | 0.199950 |
| hue | -0.071747 | -0.561296 | -0.074667 | -0.273955 | 0.055398 |
| od280/od315_of_diluted_wines | 0.072343 | -0.368710 | 0.003911 | -0.276769 | 0.066004 |
| proline | 0.643720 | -0.192011 | 0.223626 | -0.440597 | 0.393351 |
| target | -0.328222 | 0.437776 | -0.049643 | 0.517859 | -0.209179 |

Корреляционная матрица содержит коэффициенты корреляции между всеми парами признаков.

Корреляционная матрица симметрична относительно главной диагонали. На главной диагонали расположены единицы (корреляция признака самого с собой).

На основе корреляционной матрицы можно сделать следующие выводы:

- Целевой признак наиболее сильно коррелирует с total_phenols (-0.719), flavanoids (-0.84), od280/od315_of_diluted_wines (-0,788). Эти признаки обязательно следует оставить в модели.
- Целевой признак отчасти коррелирует с alcalinity_of_ash (0.518), hue (-0,617), proline (-0,634), nonflavanoid_phenols (-0,489), proanthocyanins (-0,499) . Этот признак стоит также оставить в модели.
- Целевой признак слабо коррелирует с ash (-0,05), magnesium (-0,209), color_intensity (0,266) . Скорее всего эти признаки стоит исключить из модели, возможно они только ухудшат качество модели.
- Параметр color_intensity средне коррелирует с параметрами: alcohol (0,546) и hue (-0,522).

Описание метода corr - https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.corr.html (https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.corr.html)

По умолчанию при построении матрицы используется коэффициент корреляции Пирсона (https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D1%80%D1%80%D0%B5%D0%BB%
Возможно также построить корреляционную матрицу на основе коэффициентов корреляции Кендалла и Спирмена. На практике три метода редко дают значимые различия.

In [36]:
```
wine1.corr(method='pearson')
```

Out[36]:

| | alcohol | malic_acid | ash | alcalinity_of_ash | magnesium |
|---|---|---|---|---|---|
| alcohol | 1.000000 | 0.094397 | 0.211545 | -0.310235 | 0.270798 |
| malic_acid | 0.094397 | 1.000000 | 0.164045 | 0.288500 | -0.054575 |
| ash | 0.211545 | 0.164045 | 1.000000 | 0.443367 | 0.286587 |
| alcalinity_of_ash | -0.310235 | 0.288500 | 0.443367 | 1.000000 | -0.083333 |
| magnesium | 0.270798 | -0.054575 | 0.286587 | -0.083333 | 1.000000 |
| total_phenols | 0.289101 | -0.335167 | 0.128980 | -0.321113 | 0.214401 |
| flavanoids | 0.236815 | -0.411007 | 0.115077 | -0.351370 | 0.195784 |
| nonflavanoid_phenols | -0.155929 | 0.292977 | 0.186230 | 0.361922 | -0.256294 |
| proanthocyanins | 0.136698 | -0.220746 | 0.009652 | -0.197327 | 0.236441 |
| color_intensity | 0.546364 | 0.248985 | 0.258887 | 0.018732 | 0.199950 |
| hue | -0.071747 | -0.561296 | -0.074667 | -0.273955 | 0.055398 |
| od280/od315_of_diluted_wines | 0.072343 | -0.368710 | 0.003911 | -0.276769 | 0.066004 |
| proline | 0.643720 | -0.192011 | 0.223626 | -0.440597 | 0.393351 |
| target | -0.328222 | 0.437776 | -0.049643 | 0.517859 | -0.209179 |

In [37]:
```
wine1.corr(method='kendall')
```

Out[37]:

| | alcohol | malic_acid | ash | alcalinity_of_ash | magnesium |
|---|---|---|---|---|---|
| alcohol | 1.000000 | 0.093844 | 0.170154 | -0.212978 | 0.250506 |
| malic_acid | 0.093844 | 1.000000 | 0.158178 | 0.210119 | 0.050869 |
| ash | 0.170154 | 0.158178 | 1.000000 | 0.258352 | 0.254246 |
| alcalinity_of_ash | -0.212978 | 0.210119 | 0.258352 | 1.000000 | -0.121005 |
| magnesium | 0.250506 | 0.050869 | 0.254246 | -0.121005 | 1.000000 |
| total_phenols | 0.209099 | -0.174929 | 0.089855 | -0.256669 | 0.172195 |
| flavanoids | 0.191087 | -0.211918 | 0.049474 | -0.309865 | 0.161603 |
| nonflavanoid_phenols | -0.109554 | 0.175129 | 0.098937 | 0.278091 | -0.158361 |
| proanthocyanins | 0.133526 | -0.168714 | 0.018240 | -0.171404 | 0.117871 |
| color_intensity | 0.434353 | 0.195607 | 0.187786 | -0.057281 | 0.241781 |
| hue | -0.021717 | -0.388707 | -0.037234 | -0.239210 | 0.023760 |
| od280/od315_of_diluted_wines | 0.061513 | -0.162909 | -0.006341 | -0.226253 | 0.034307 |
| proline | 0.449387 | -0.044660 | 0.171574 | -0.313218 | 0.343016 |
| target | -0.238984 | 0.247494 | -0.038085 | 0.449402 | -0.184992 |

In [38]:
```
wine1.corr(method='spearman')
```

Out[38]:

| | alcohol | malic_acid | ash | alcalinity_of_ash | magnesium |
|---|---|---|---|---|---|
| alcohol | 1.000000 | 0.140430 | 0.243722 | -0.306598 | 0.365503 |
| malic_acid | 0.140430 | 1.000000 | 0.230674 | 0.304069 | 0.080188 |
| ash | 0.243722 | 0.230674 | 1.000000 | 0.366374 | 0.361488 |
| alcalinity_of_ash | -0.306598 | 0.304069 | 0.366374 | 1.000000 | -0.169558 |
| magnesium | 0.365503 | 0.080188 | 0.361488 | -0.169558 | 1.000000 |
| total_phenols | 0.310920 | -0.280225 | 0.132193 | -0.376657 | 0.246417 |
| flavanoids | 0.294740 | -0.325202 | 0.078796 | -0.443770 | 0.233167 |
| nonflavanoid_phenols | -0.162207 | 0.255236 | 0.145583 | 0.389390 | -0.236786 |
| proanthocyanins | 0.192734 | -0.244825 | 0.024384 | -0.253695 | 0.173647 |
| color_intensity | 0.635425 | 0.290307 | 0.283047 | -0.073776 | 0.357029 |
| hue | -0.024203 | -0.560265 | -0.050183 | -0.352507 | 0.036095 |
| od280/od315_of_diluted_wines | 0.103050 | -0.255185 | -0.007500 | -0.325890 | 0.056963 |
| proline | 0.633580 | -0.057466 | 0.253163 | -0.456090 | 0.507575 |
| target | -0.354167 | 0.346913 | -0.053988 | 0.569792 | -0.250498 |

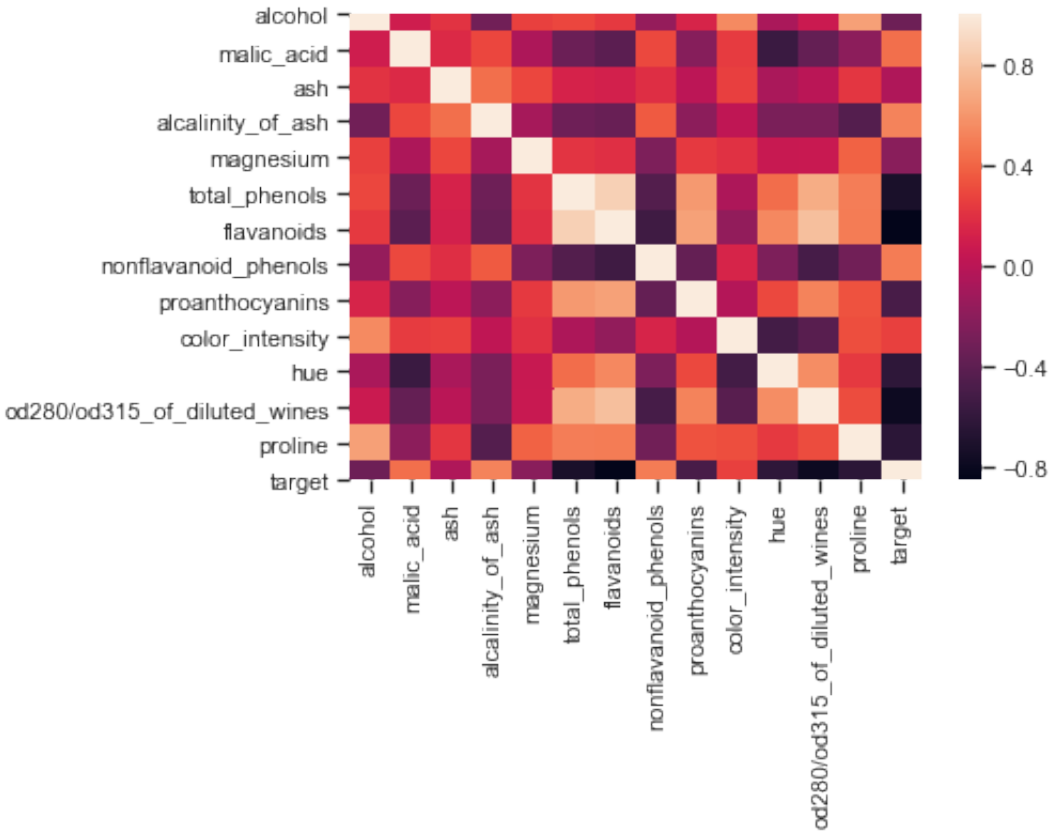
В случае большого количества признаков анализ числовой корреляционной матрицы становится неудобен.

Для визуализации корреляционной матрицы будем использовать "тепловую карту" heatmap которая показывает степень корреляции различными цветами.

Используем метод heatmap библиотеки seaborn - https://seaborn.pydata.org/generated/seaborn.heatmap.html (https://seaborn.pydata.org/generated/seaborn.heatmap.html)

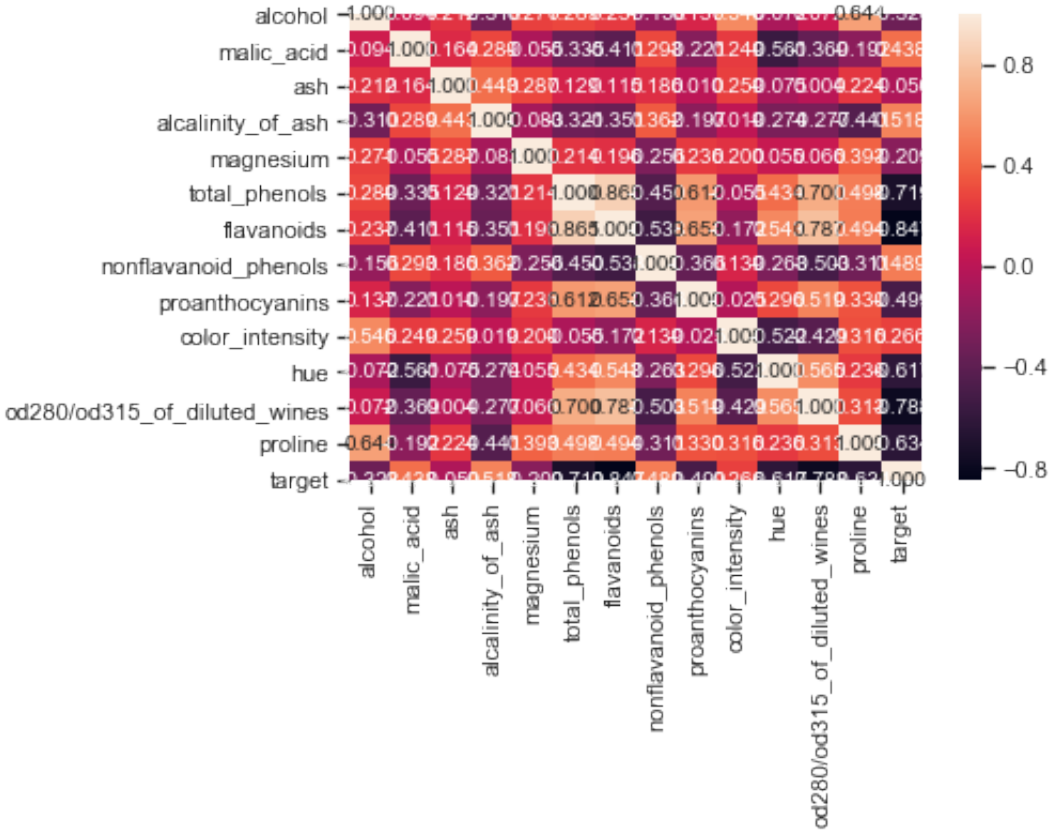
In [39]:
```
sns.heatmap(wine1.corr())
```

Out[39]: <matplotlib.axes._subplots.AxesSubplot at 0x1a2ad18b10>



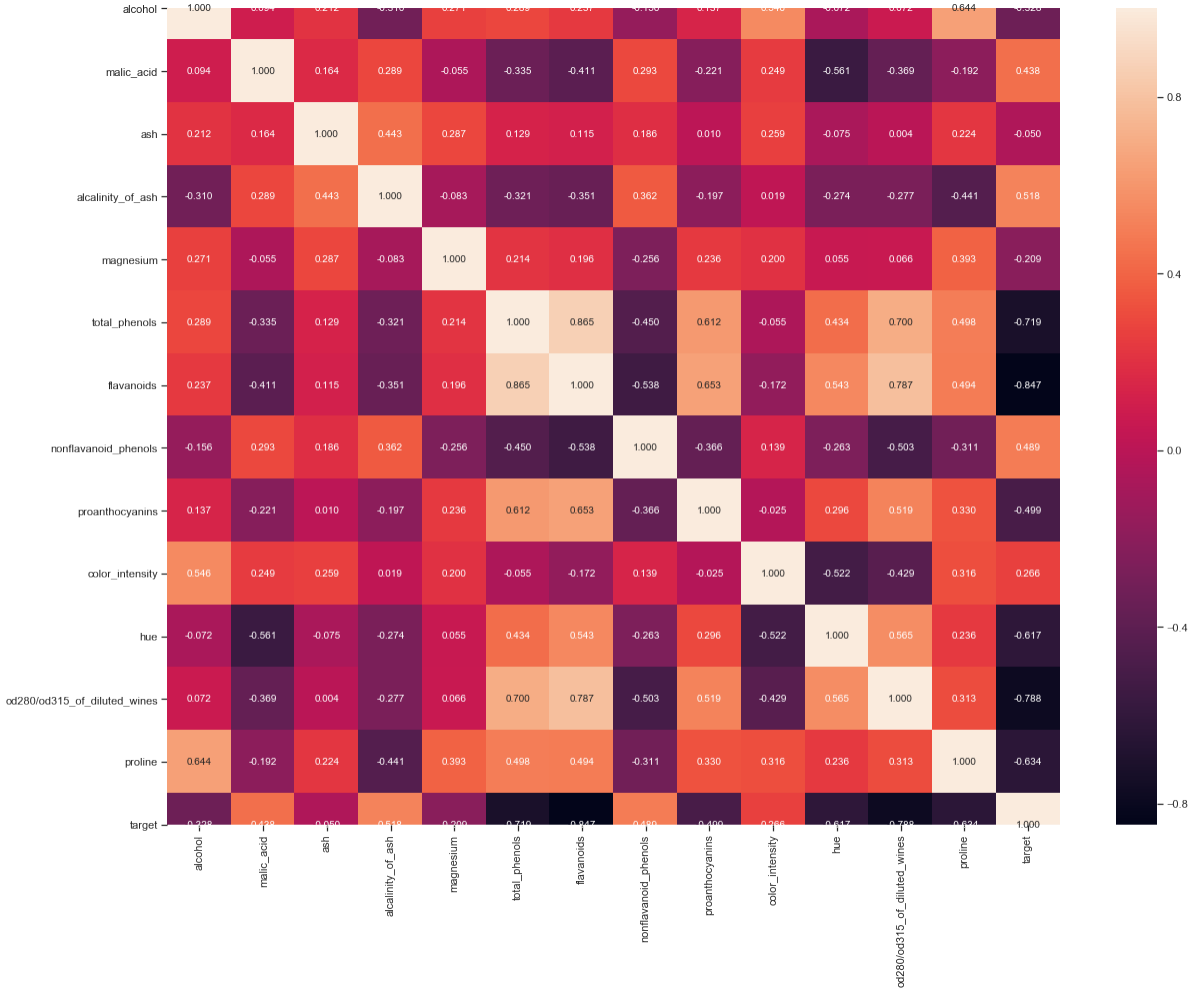
In [40]:
```
# Вывод значений в ячейках
sns.heatmap(wine1.corr(), annot=True, fmt='.3f')
```

Out[40]: <matplotlib.axes._subplots.AxesSubplot at 0x1a2aed2810>



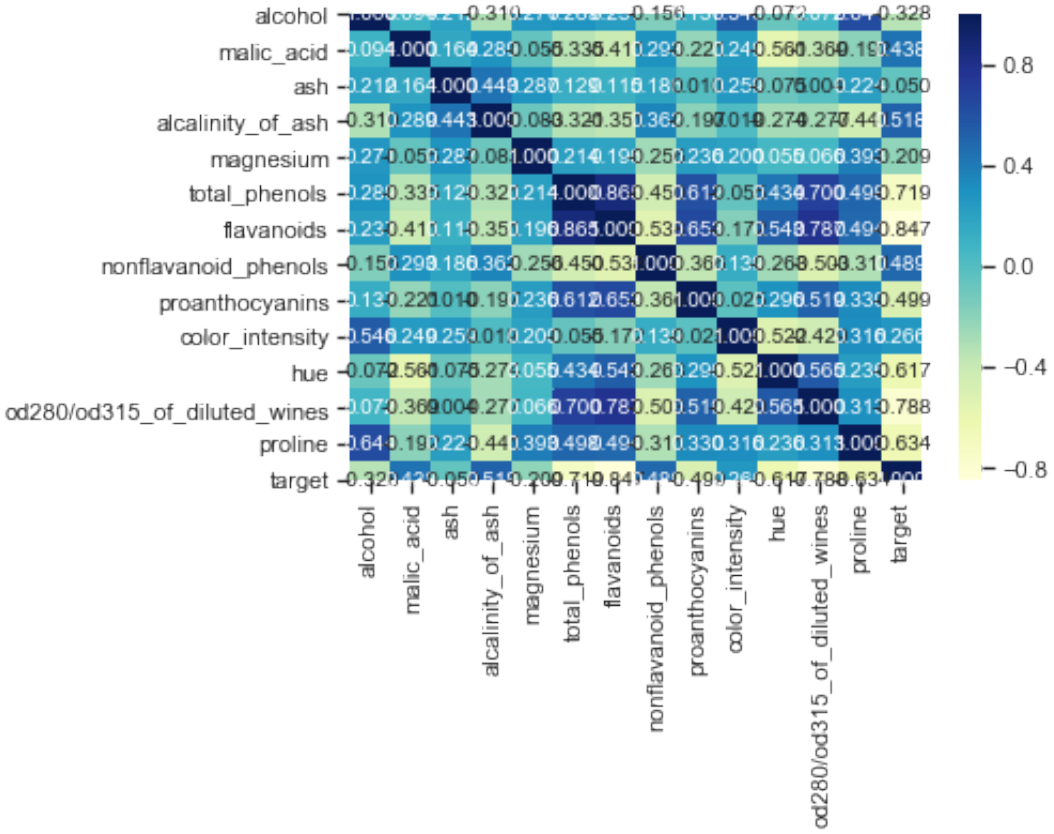
In [41]:
```
plt.subplots(figsize=(20,15))
sns.heatmap(wine1.corr(), annot=True, fmt='.3f')
```

Out[41]: <matplotlib.axes._subplots.AxesSubplot at 0x1a2b7acd10>



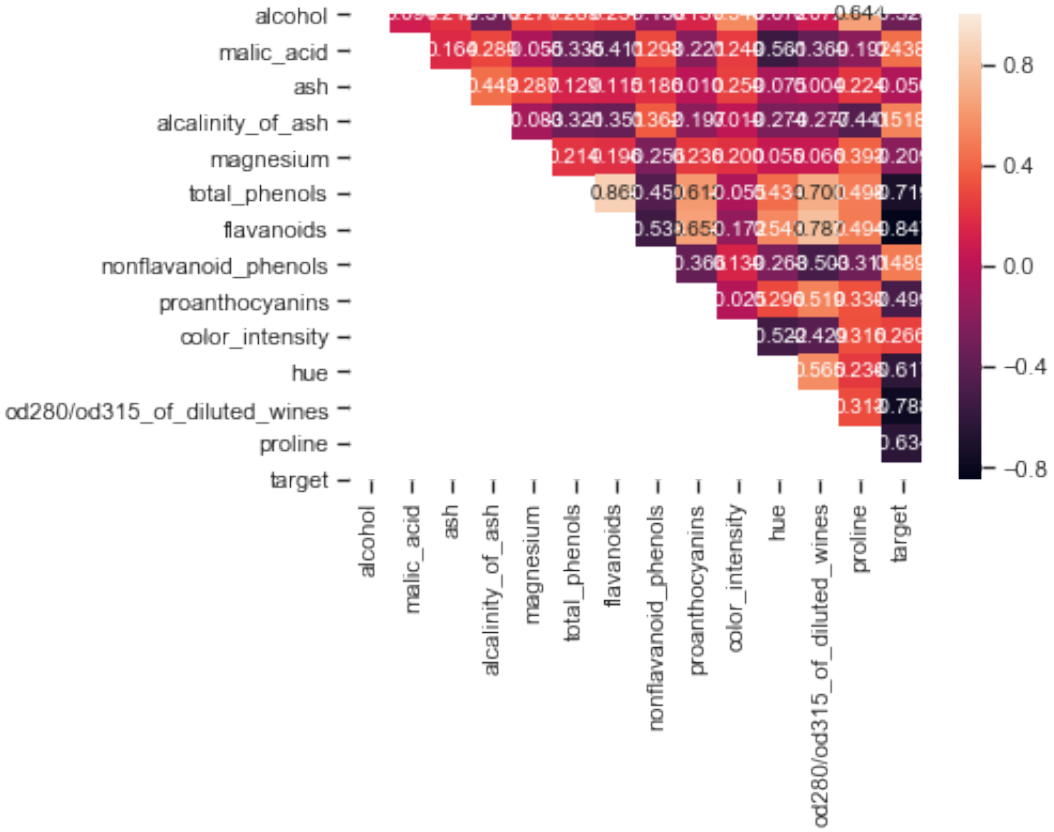
In [42]:
```
# Изменение цветовой гаммы
sns.heatmap(wine1.corr(), cmap='YlGnBu', annot=True, fmt='.3f')
```

Out[42]: <matplotlib.axes._subplots.AxesSubplot at 0x1a315b7b90>

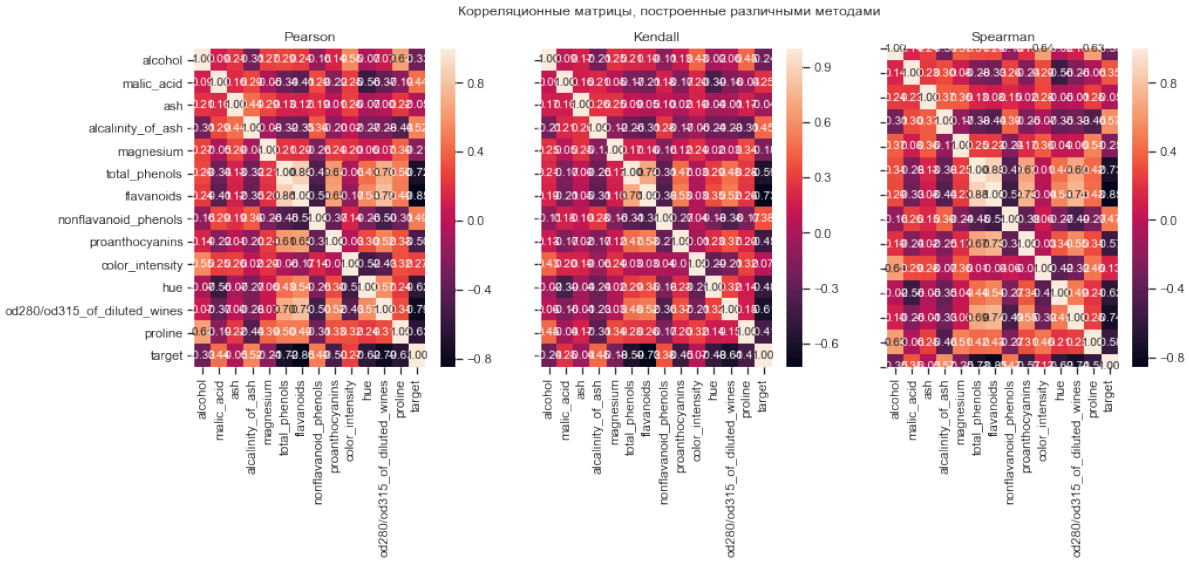


In [43]:
```
# Треугольный вариант матрицы
mask = np.zeros_like(wine1.corr(), dtype=np.bool)
# чтобы оставить нижнюю часть матрицы
# mask[np.triu_indices_from(mask)] = True
# чтобы оставить верхнюю часть матрицы
mask[np.tril_indices_from(mask)] = True
sns.heatmap(wine1.corr(), mask=mask, annot=True, fmt='.3f')
```

Out[43]: <matplotlib.axes._subplots.AxesSubplot at 0x1a2ba31750>



In [44]:
```
fig, ax = plt.subplots(1, 3, sharex='col', sharey='row', figsize=(1
sns.heatmap(wine1.corr(method='pearson'), ax=ax[0], annot=True, fmt
sns.heatmap(wine1.corr(method='kendall'), ax=ax[1], annot=True, fmt
sns.heatmap(wine1.corr(method='spearman'), ax=ax[2], annot=True, fm
fig.suptitle('Корреляционные матрицы, построенные различными метода
ax[0].title.set_text('Pearson')
ax[1].title.set_text('Kendall')
ax[2].title.set_text('Spearman')
```




In [ ]: