

Разведочный анализ данных. Исследование и визуализация данных.

1) Текстовое описание набора данных

В качестве набора данных мы будем использовать набор данных по распознаванию вин - <https://archive.ics.uci.edu/ml/machine-learning-databases/wine/wine.data>. Эти данные являются результатами химического анализа вин, выращенных в одном регионе Италии тремя разными производителями. Проведено тридцать различных измерений для различных компонентов, обнаруженных в трех типах вина.

Файл содержит следующие колонки:

- Алкоголь
- Яблочная кислота
- Пепел
- Щелочность зоны
- Магний
- Общее количество фенолов
- Флавоноиды
- Нефлавоноидные фенолы
- Проантоцианы
- Интенсивность цвета
- Оттенок
- OD280/OD315 разбавленных вин
- Пролин

Импорт библиотек

Импортируем библиотеки с помощью команды `import`. Как правило, все команды `import` размещают в первой ячейке ноутбука, но мы в этом примере будем подключать все библиотеки последовательно, по мере их использования.

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

```
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style="ticks")
```

Загрузка данных

Загрузим файл датасета из https://scikit-learn.org/stable/datasets/toy_dataset.html:

```
from sklearn import datasets
wine = datasets.load_wine()
```

Создадим датафрейм wine_df в библиотеке Pandas на основе загруженного файла:

```
wine_df = pd.DataFrame(wine.data, columns=wine.feature_names)
```

2) Основные характеристики датасета

```
# Первые пять строк датасета
wine_df.head()
```

	alcohol	malic_acid	ash	alcalinity_of_ash	magnesium	total_phenols	flavanoids	nonflavar
0	14.23	1.71	2.43	15.6	127.0	2.80	3.06	0.28
1	13.20	1.78	2.14	11.2	100.0	2.65	2.76	0.26
2	13.16	2.36	2.67	18.6	101.0	2.80	3.24	0.30
3	14.37	1.95	2.50	16.8	113.0	3.85	3.49	0.24
4	13.24	2.59	2.87	21.0	118.0	2.80	2.69	0.39

```
# Размер датасета - 178 строк, 13 колонок
```

```
wine_df.shape
```

```
(178, 13)
```

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

```
Всего строк: 178
```

```
# Список колонок
wine_df.columns
```

```
Index(['alcohol', 'malic_acid', 'ash', 'alcalinity_of_ash', 'magn
       'total_phenols', 'flavanoids', 'nonflavanoid_phenols',
       'proanthocyanins', 'color_intensity', 'hue',
       'od280/od315_of_diluted_wines', 'proline'],
     dtype='object')
```

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

```
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
dtype: object
```

```
# Проверим наличие пустых значений
# Цикл по колонкам датасета
for col in wine_df.columns:
```

```
# Количество пустых значений - все значения заполнены
temp_null_count = wine_df[wine_df[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
```

```
# Основные статистические характеристики набора данных
wine_df.describe()
```

	alcohol	malic_acid	ash	alcalinity_of_ash	magnesium	total_phenols	flav
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	2.0
std	0.811827	1.117146	0.274344	3.339564	14.282484	0.625851	0.9
min	11.030000	0.740000	1.360000	10.600000	70.000000	0.980000	0.3
25%	12.362500	1.602500	2.210000	17.200000	88.000000	1.742500	1.2
50%	13.050000	1.865000	2.360000	19.500000	98.000000	2.355000	2.1
75%	13.677500	3.082500	2.557500	21.500000	107.000000	2.800000	2.8
max	14.830000	5.800000	3.230000	30.000000	162.000000	3.880000	5.0

```
# Определим уникальные значения для пепела
wine_df['ash'].unique()
```

```
array([2.43, 2.14, 2.67, 2.5 , 2.87, 2.45, 2.61, 2.17, 2.27, 2.3 , 2.32,
       2.41, 2.39, 2.38, 2.7 , 2.72, 2.62, 2.48, 2.56, 2.28, 2.65, 2.36,
       2.52, 3.22, 2.8 , 2.21, 2.84, 2.55, 2.1 , 2.51, 2.31, 2.12, 2.59,
       2.29, 2.44, 2.4 , 2.04, 2.6 , 2.42, 2.68, 2.25, 2.46, 1.36, 2.02,
       1.02 2.16 2.53 1.7 1.75 2.21 1.71 2.22 1.05 2.22]
```

```
2.58, 2.26, 2.22, 2.74, 1.98, 1.9 , 1.88, 1.94, 1.82, 2.92, 1.99,
2.19, 3.23, 2.73, 2.13, 2.78, 2.54, 2.64, 2.35, 2.15, 2.75, 2.69,
2.86, 2.37])
```

3) Визуальное исследование датасета

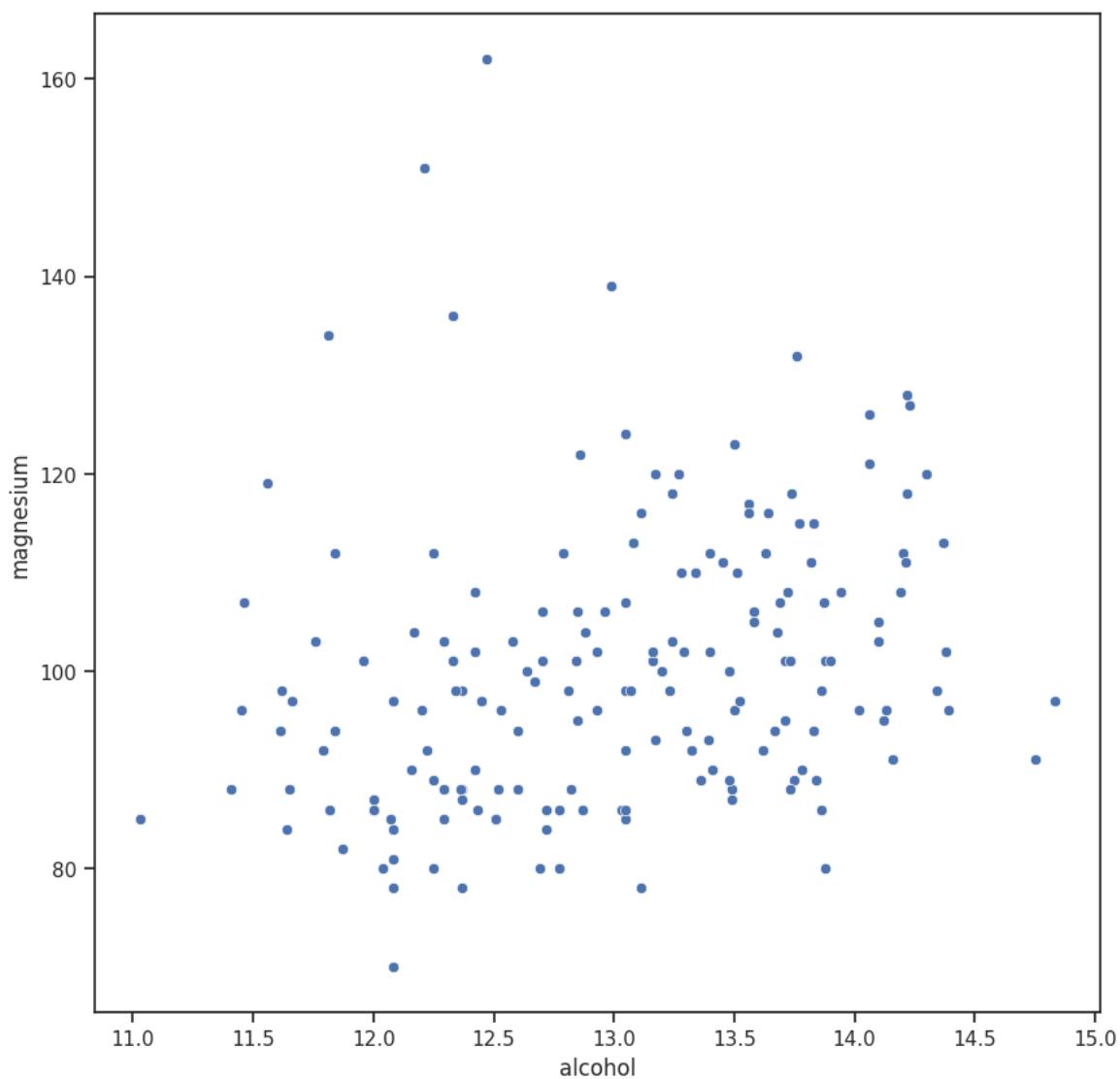
Диаграмма рассеивания

Позволяет построить распределение двух колонок данных и визуально обнаружить наличие зависимости. Не предполагается, что значения упорядочены (например, по времени).

```
fig, ax = plt.subplots(figsize=(10,10))
sns.scatterplot(ax=ax, x='alcohol', y='magnesium', data=wine_df)
```

```
<Axes: xlabel='alcohol', ylabel='magnesium'>
```

[!\[\]\(0b5e7e25e8775f7e7e80906ada4f0021_img.jpg\) Download](#)

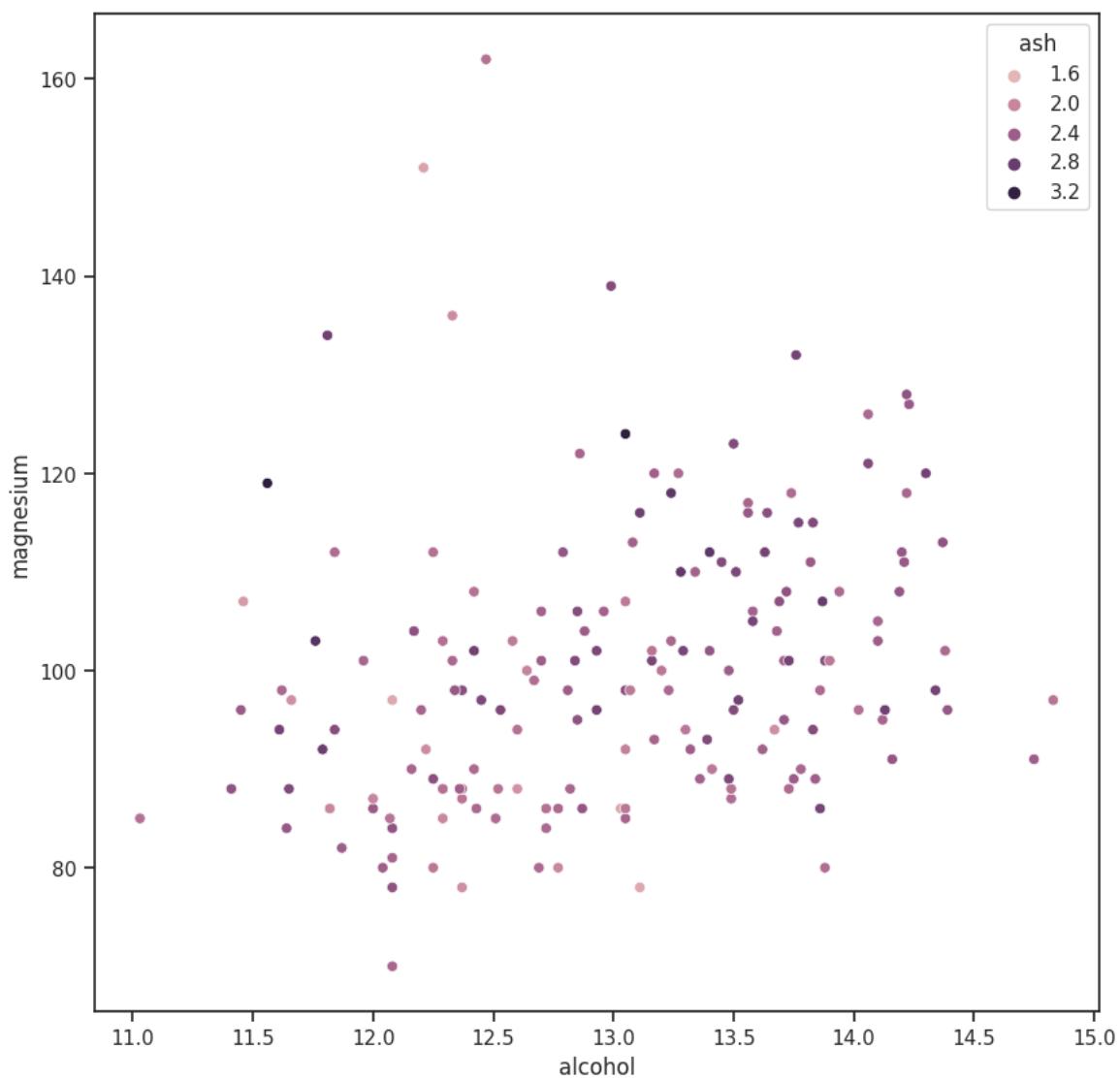


Посмотрим насколько на эту зависимость влияет пепел:

```
fig, ax = plt.subplots(figsize=(10,10))
sns.scatterplot(ax=ax, x='alcohol', y='magnesium', data=wine_df, hu
```

```
<Axes: xlabel='alcohol', ylabel='magnesium'>
```

[Download](#)



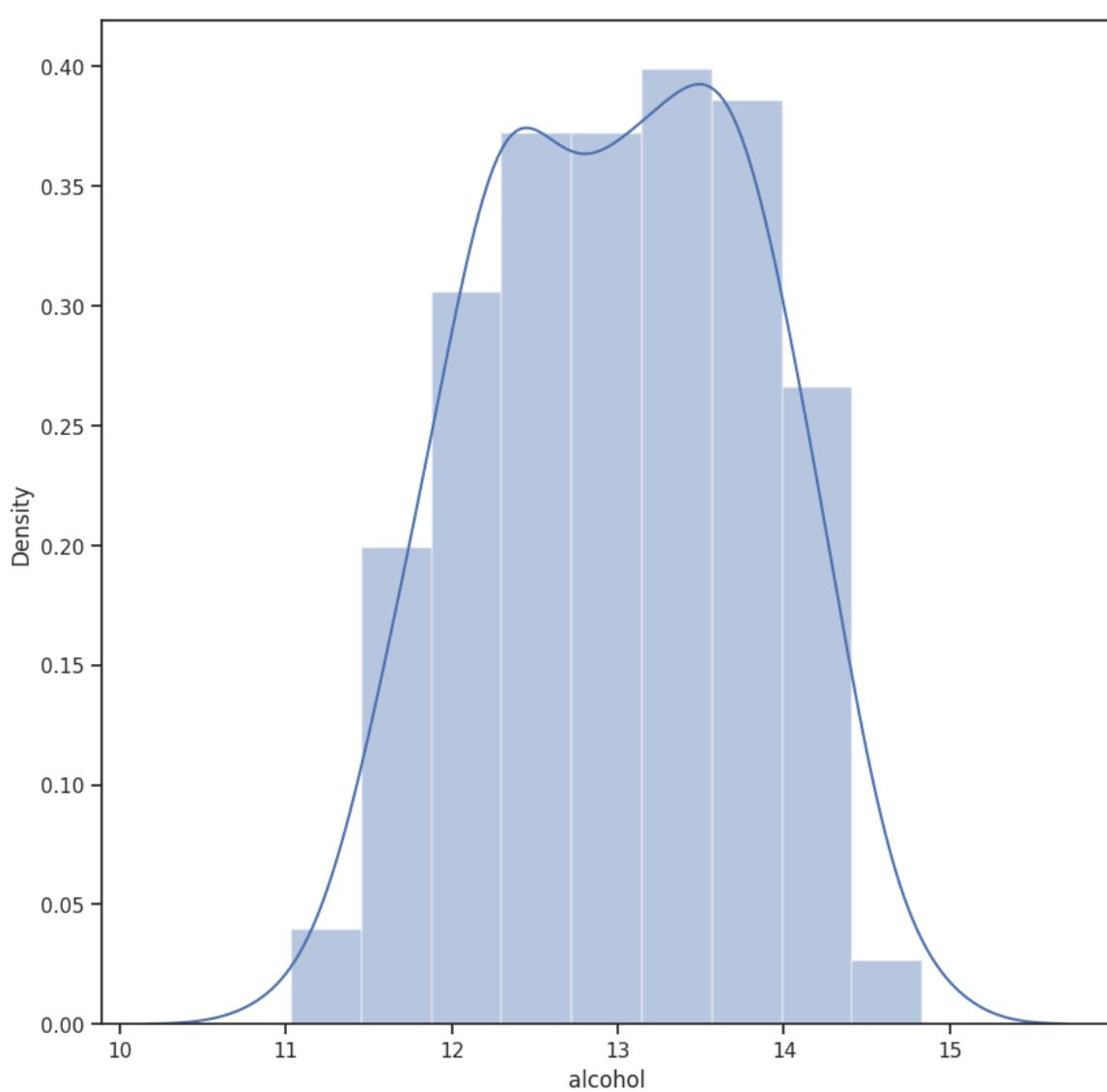
Гистограмма

Позволяет оценить плотность вероятности распределения данных.

```
fig, ax = plt.subplots(figsize=(10,10))
sns.distplot(wine_df['alcohol'])
```

```
<Axes: xlabel='alcohol', ylabel='Density'>
```

[Download](#)



```
<ipython-input-31-0c268acb8e01>:2: UserWarning:  
`distplot` is a deprecated function and will be removed in seabor  
Please adapt your code to use either `displot` (a figure-level fu  
similar flexibility) or `histplot` (an axes-level function for hi  
For a guide to updating your code to use the new functions, pleas  
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751  
sns.distplot(wine_df['alcohol'])
```

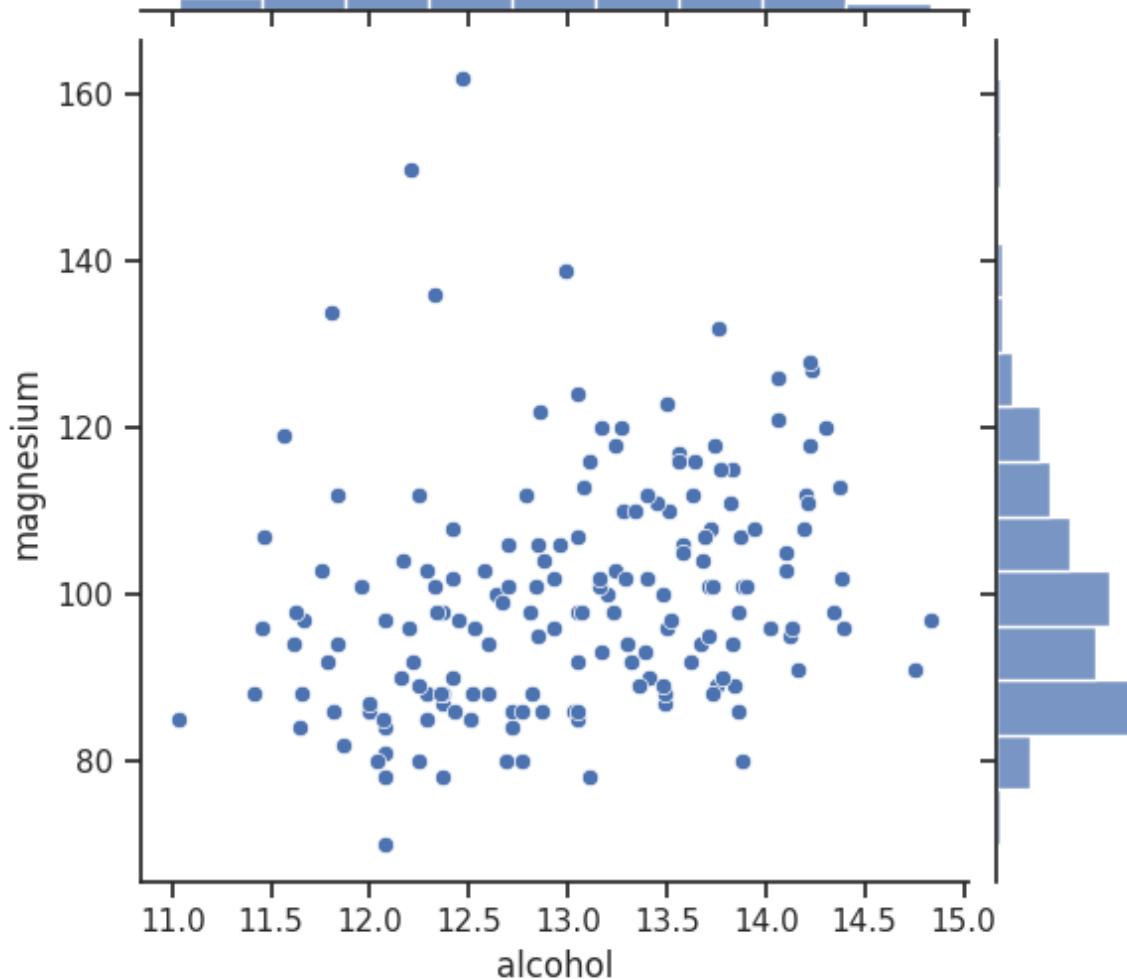
Jointplot

Комбинация гистограмм и диаграмм рассеивания.

```
sns.jointplot(x='alcohol', y='magnesium', data=wine_df)
```

```
<seaborn.axisgrid.JointGrid at 0x7f2e920b7910>
```

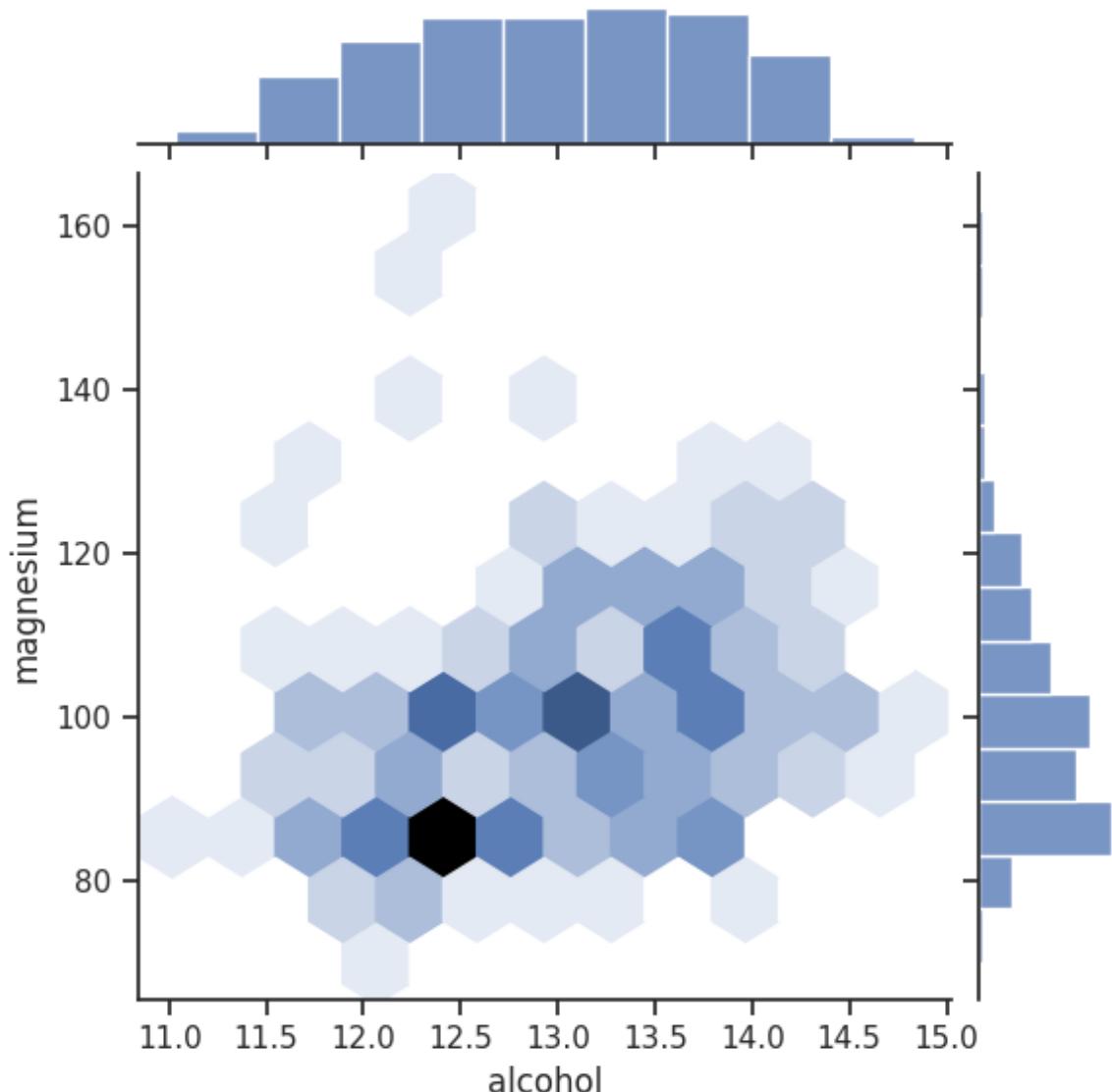
[Download](#)



```
sns.jointplot(x='alcohol', y='magnesium', data=wine_df, kind="hex")
```

```
<seaborn.axisgrid.JointGrid at 0x7f2e91b49610>
```

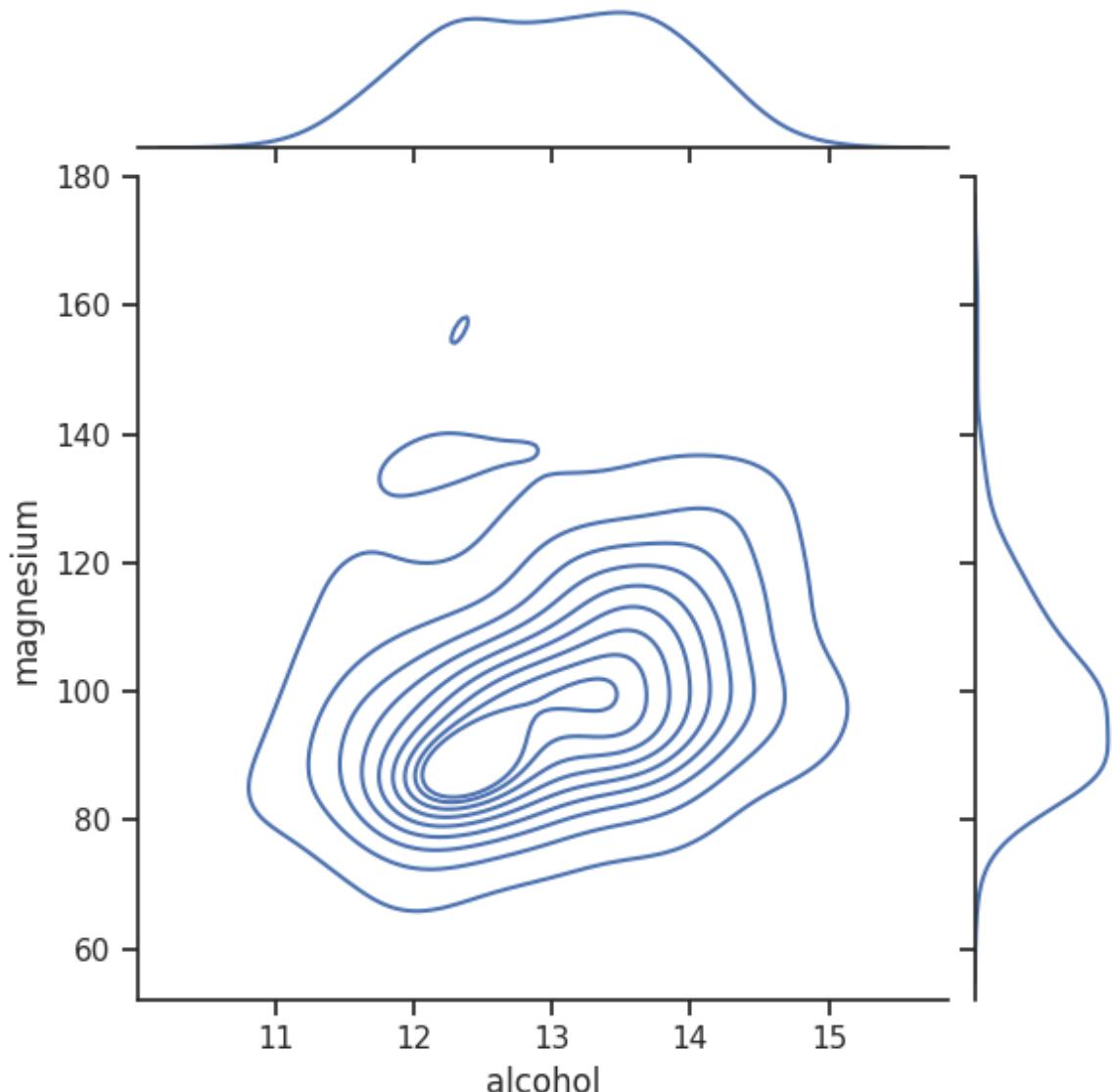
[Download](#)



```
sns.jointplot(x='alcohol', y='magnesium', data=wine_df, kind="kde")
```

```
<seaborn.axisgrid.JointGrid at 0x7f2e91a37af0>
```

[Download](#)



"Парные диаграммы"

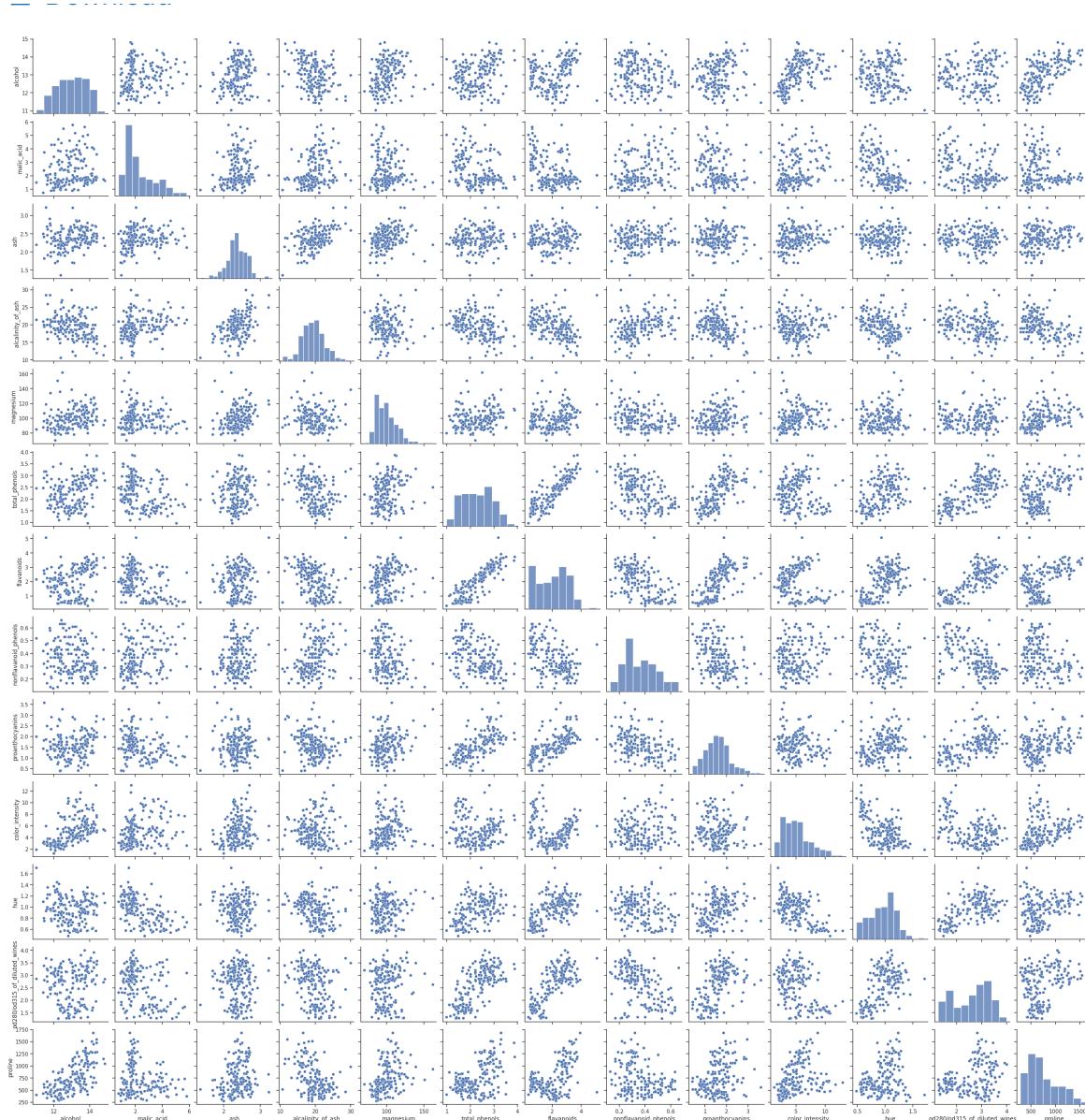
Комбинация гистограмм и диаграмм рассеивания для всего набора данных.

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

```
sns.pairplot(wine_df)
```

```
<seaborn.axisgrid.PairGrid at 0x7f2e9235cb20>
```

[Download](#)

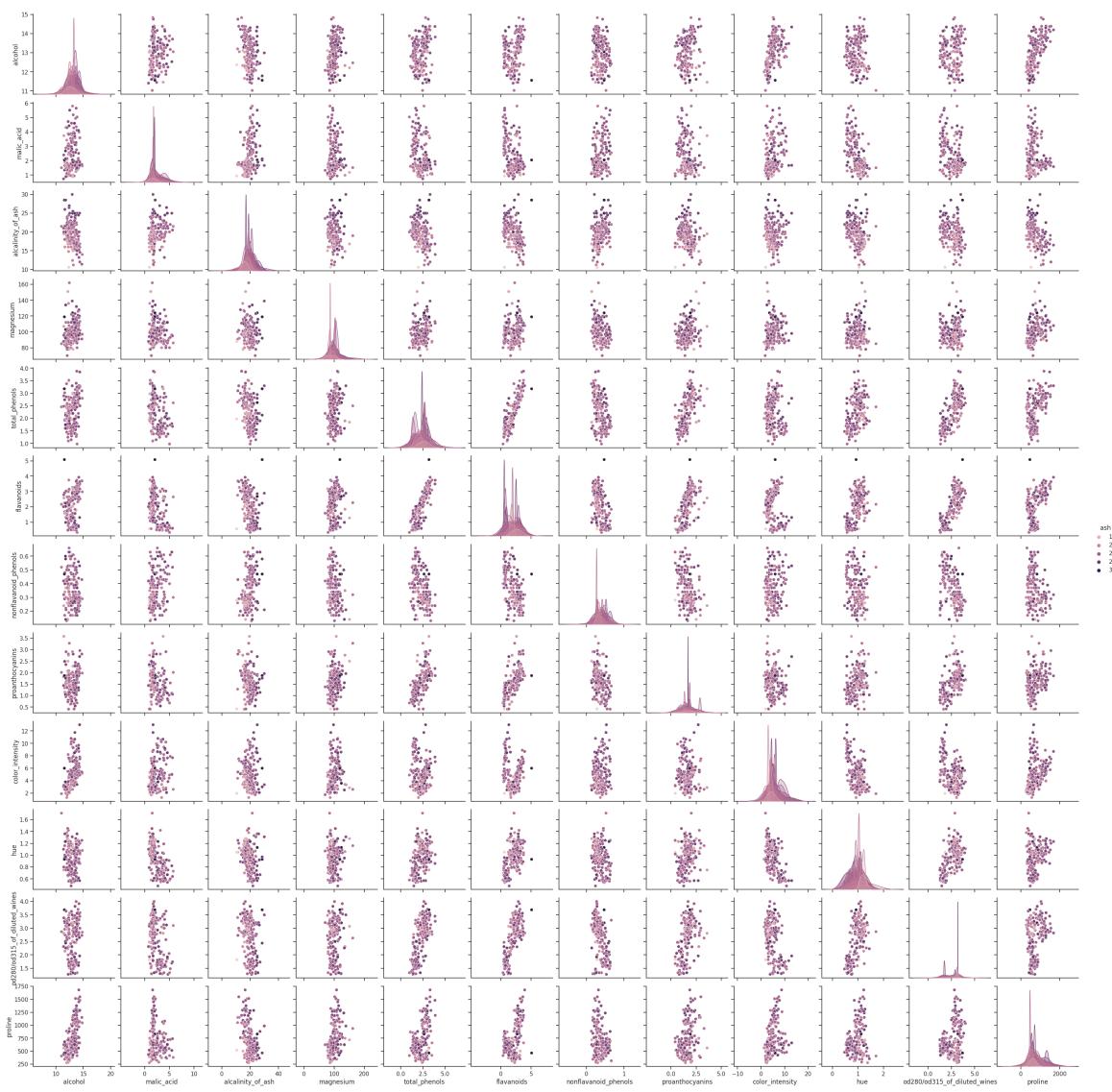


С помощью параметра "hue" возможна группировка по значениям какого-либо признака.

```
sns.pairplot(wine_df, hue="ash")
```

```
<seaborn.axisgrid.PairGrid at 0x7f2e882cc4f0>
```

[Download](#)



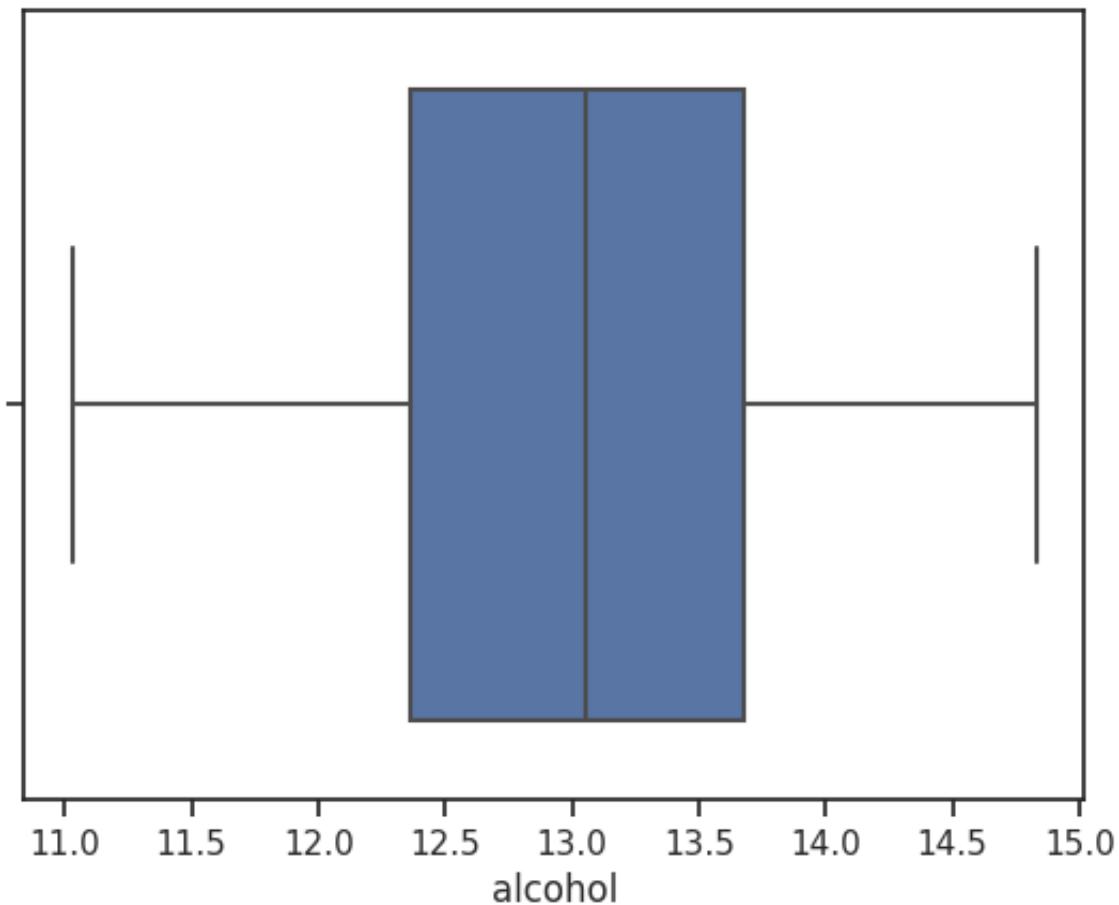
Ящик с усами

Отображает одномерное распределение вероятности.

```
sns.boxplot(x=wine_df['alcohol'])
```

```
<Axes: xlabel='alcohol'>
```

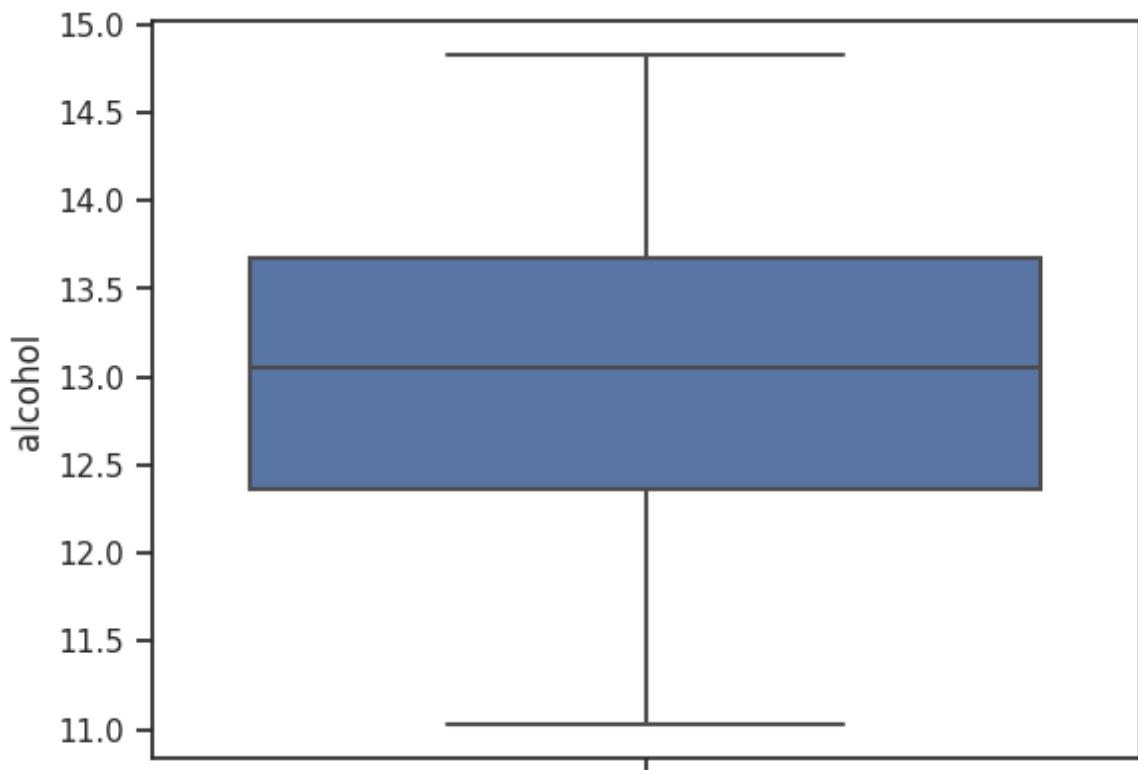
[Download](#)



```
# По вертикали  
sns.boxplot(y=wine_df['alcohol'])
```

```
<Axes: ylabel='alcohol'>
```

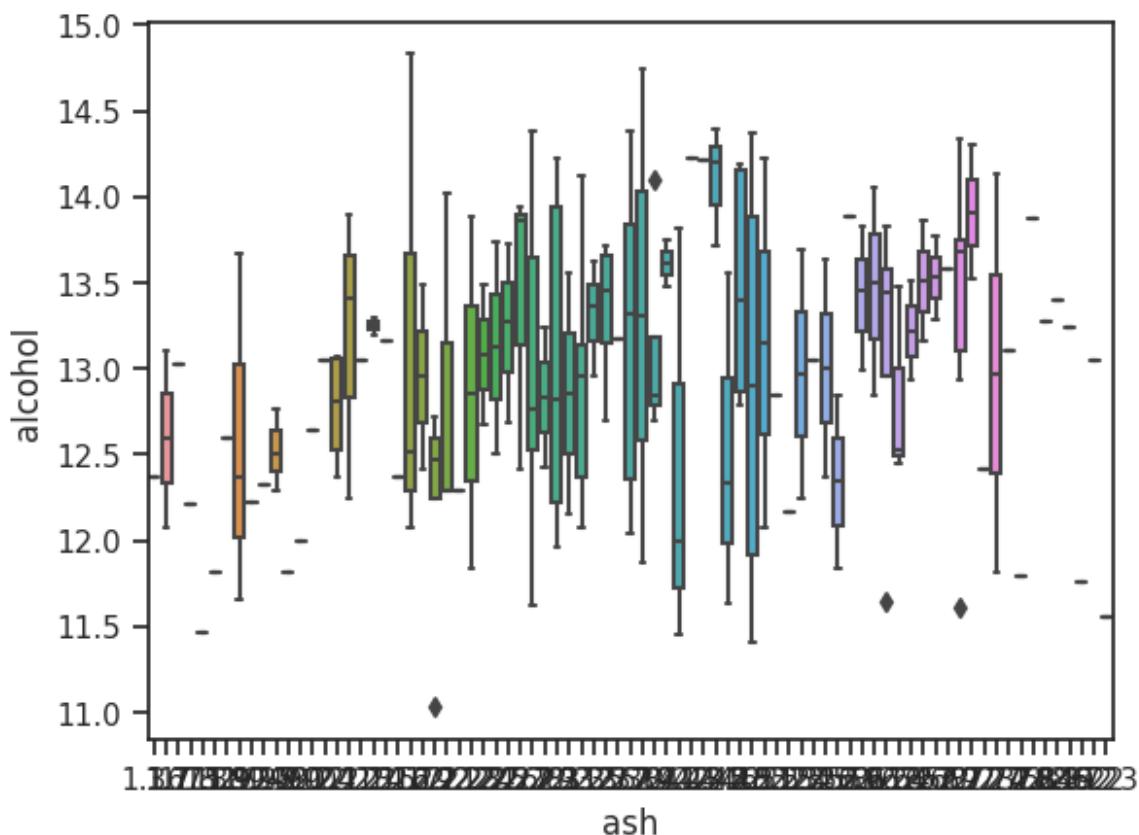
[Download](#)



```
# Распределение параметра alcohol сгруппированные по ash.  
sns.boxplot(x='ash', y='alcohol', data=wine_df)
```

```
<Axes: xlabel='ash', ylabel='alcohol'>
```

[Download](#)



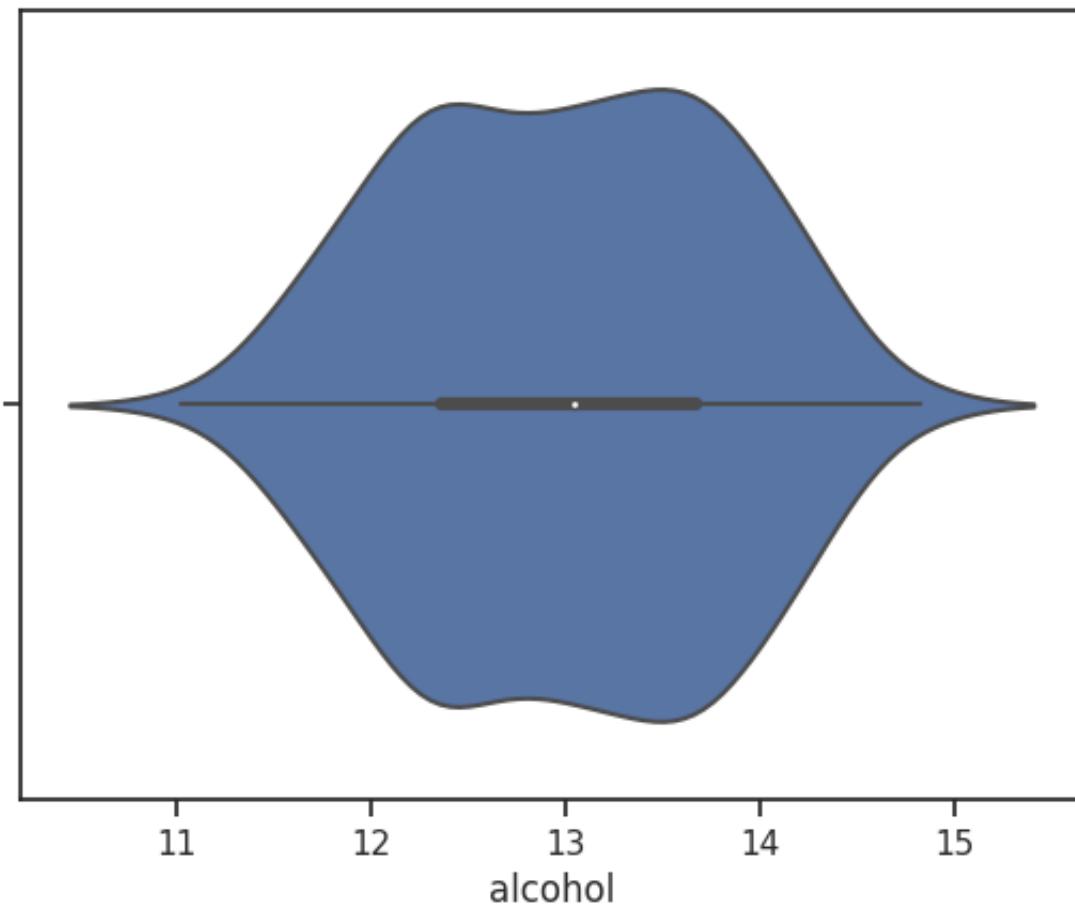
Violin plot

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

```
sns.violinplot(x=wine_df['alcohol'])
```

```
<Axes: xlabel='alcohol'>
```

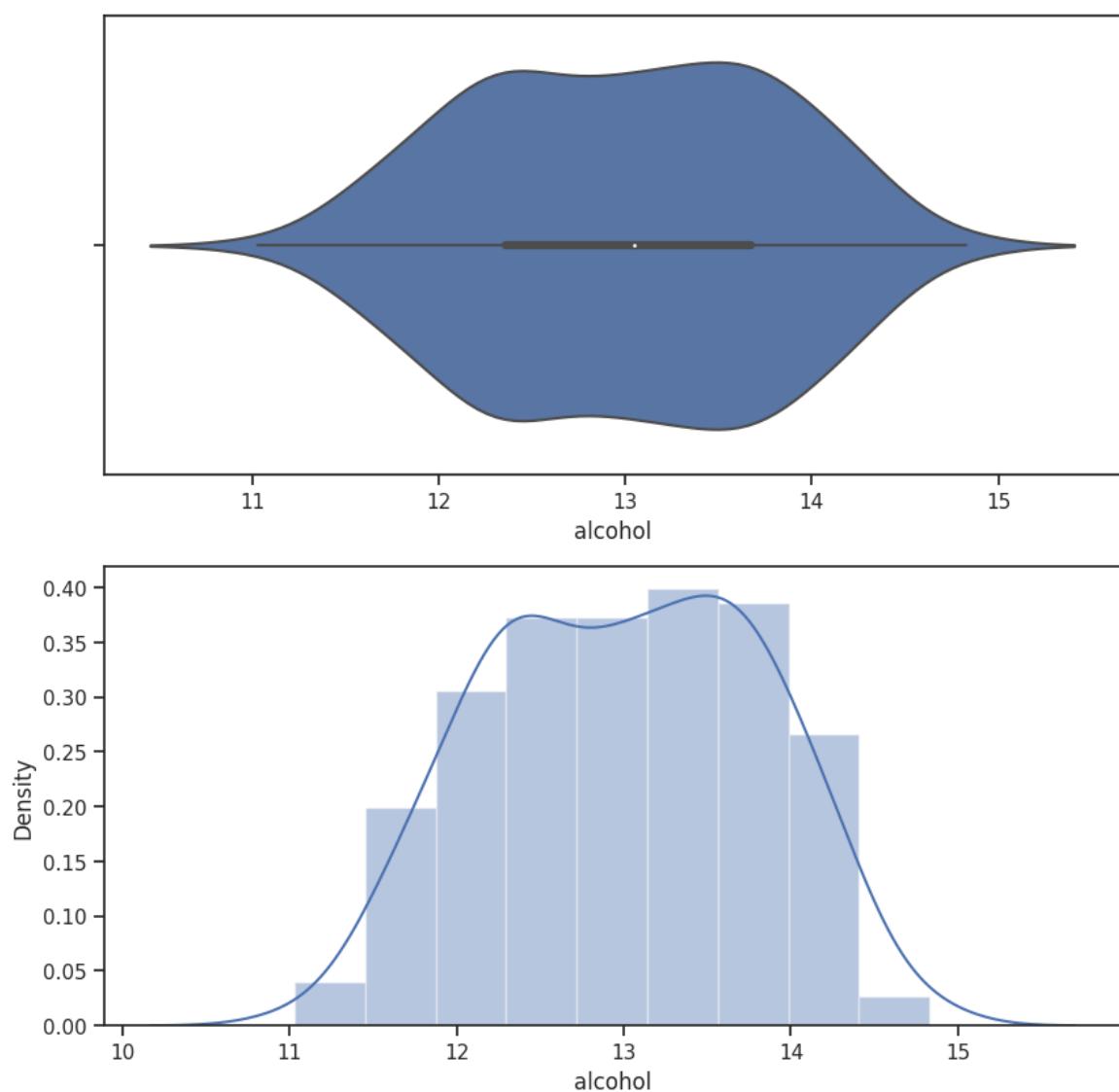
[Download](#)



```
fig, ax = plt.subplots(2, 1, figsize=(10,10))
sns.violinplot(ax=ax[0], x=wine_df['alcohol'])
sns.distplot(wine_df['alcohol'], ax=ax[1])
```

<Axes: xlabel='alcohol', ylabel='Density'>

[Download](#)



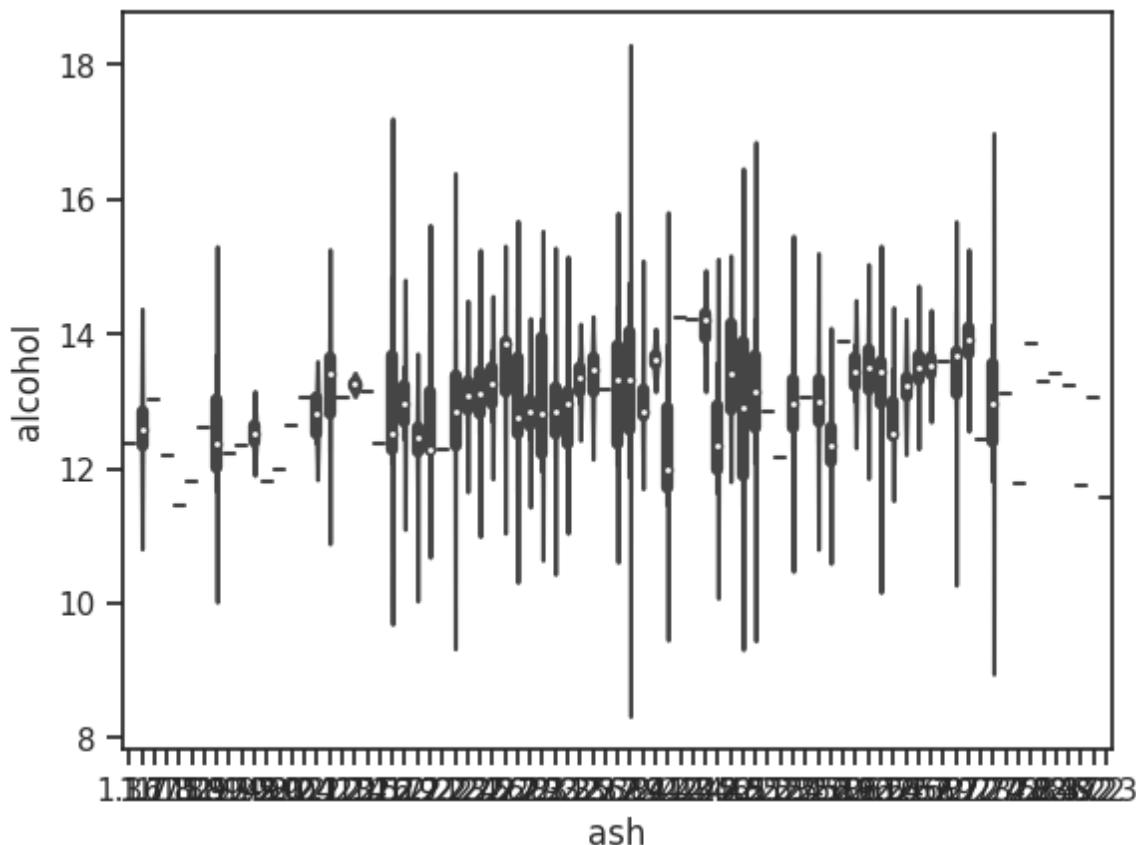
```
<ipython-input-41-b9c5e744cb77>:3: UserWarning:  
`distplot` is a deprecated function and will be removed in seabor  
Please adapt your code to use either `displot` (a figure-level fu  
similar flexibility) or `histplot` (an axes-level function for hi  
For a guide to updating your code to use the new functions, pleas  
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751  
sns.distplot(wine_df['alcohol'], ax=ax[1])
```

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

```
# Распределение параметра alcohol сгруппированные по ash.  
sns.violinplot(x='ash', y='alcohol', data=wine_df)
```

```
<Axes: xlabel='ash', ylabel='alcohol'>
```

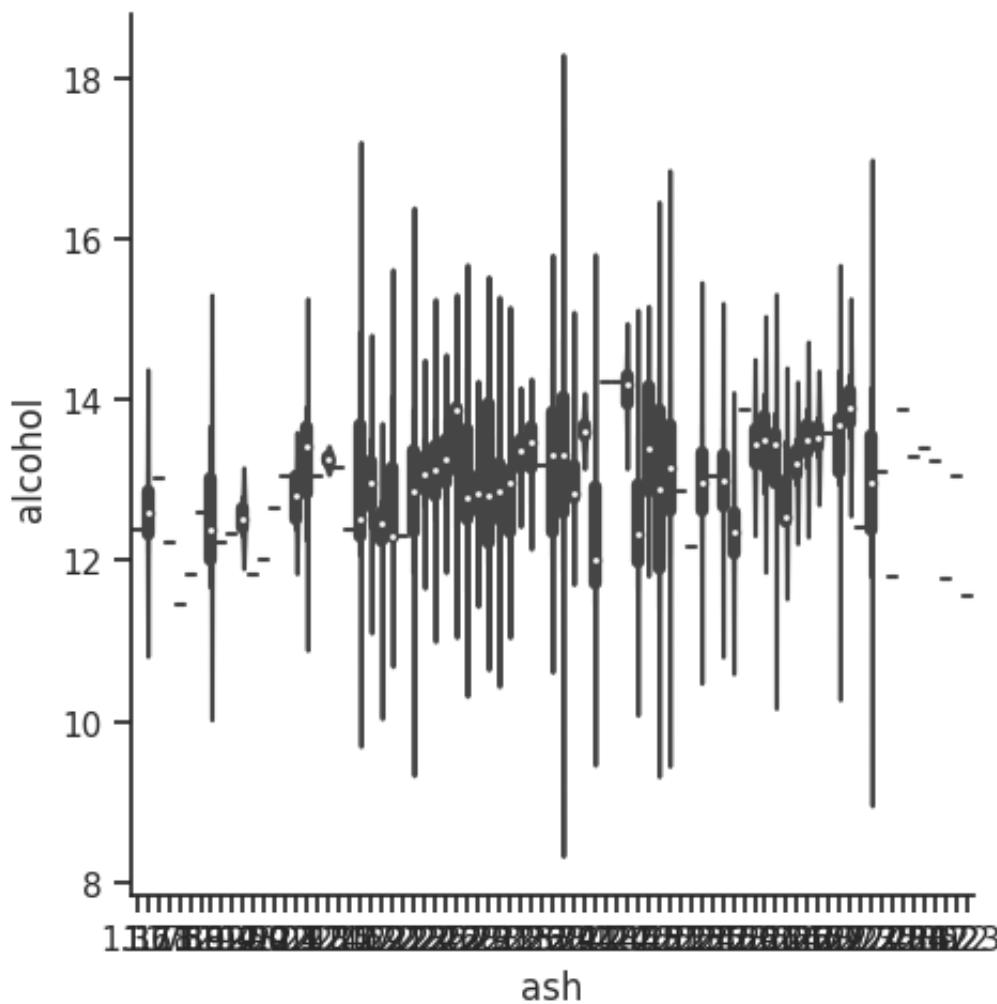
[Download](#)



```
sns.catplot(y='alcohol', x='ash', data=wine_df, kind="violin", split=True)
```

```
<seaborn.axisgrid.FacetGrid at 0x7f2e7d2b3c70>
```

[Download](#)



4) Информация о корреляции признаков

Проверка корреляции признаков позволяет решить две задачи:

Понять какие признаки (колонки датасета) наиболее сильно коррелируют с целевым признаком (в нашем примере это колонка "ash"). 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.

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.

```
wine_df.corr()
```

	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

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).

- Флаваноиды наиболее сильно коррелируют с общим количеством фенолов (0.86), OD280/OD315 разбавленных вин (0.79) и проантоксианами (0.65). Эти признаки обязательно следует оставить в модели.
- Флаваноиды отчасти коррелируют с нефлановоидными фенолами (-0.54), пролином (0.49) и оттенком (0.54). Эти признаки стоит также оставить в модели.
- Флаваноиды слабо коррелируют с пепелом (0.11) и интенсивностью цвета (-0.17). Скорее всего эти признаки стоит исключить из модели, возможно они только ухудшат качество модели.

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

По умолчанию при построении матрицы используется коэффициент корреляции Пирсона. Возможно также построить корреляционную матрицу на основе коэффициентов корреляции Кендалла и Спирмена. На практике три

метода редко дают значимые различия.

```
wine_df.corr(method='pearson')
```

	alcohol	malic_acid	ash	alcalinity_of_ash	magnesiu
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

```
wine_df.corr(method='kendall')
```

	alcohol	malic_acid	ash	alcalinity_of_ash	magnesiu
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

```
wine_df.corr(method='spearman')
```

	alcohol	malic_acid	ash	alcalinity_of_ash	magnesiu
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	-0.236781
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

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

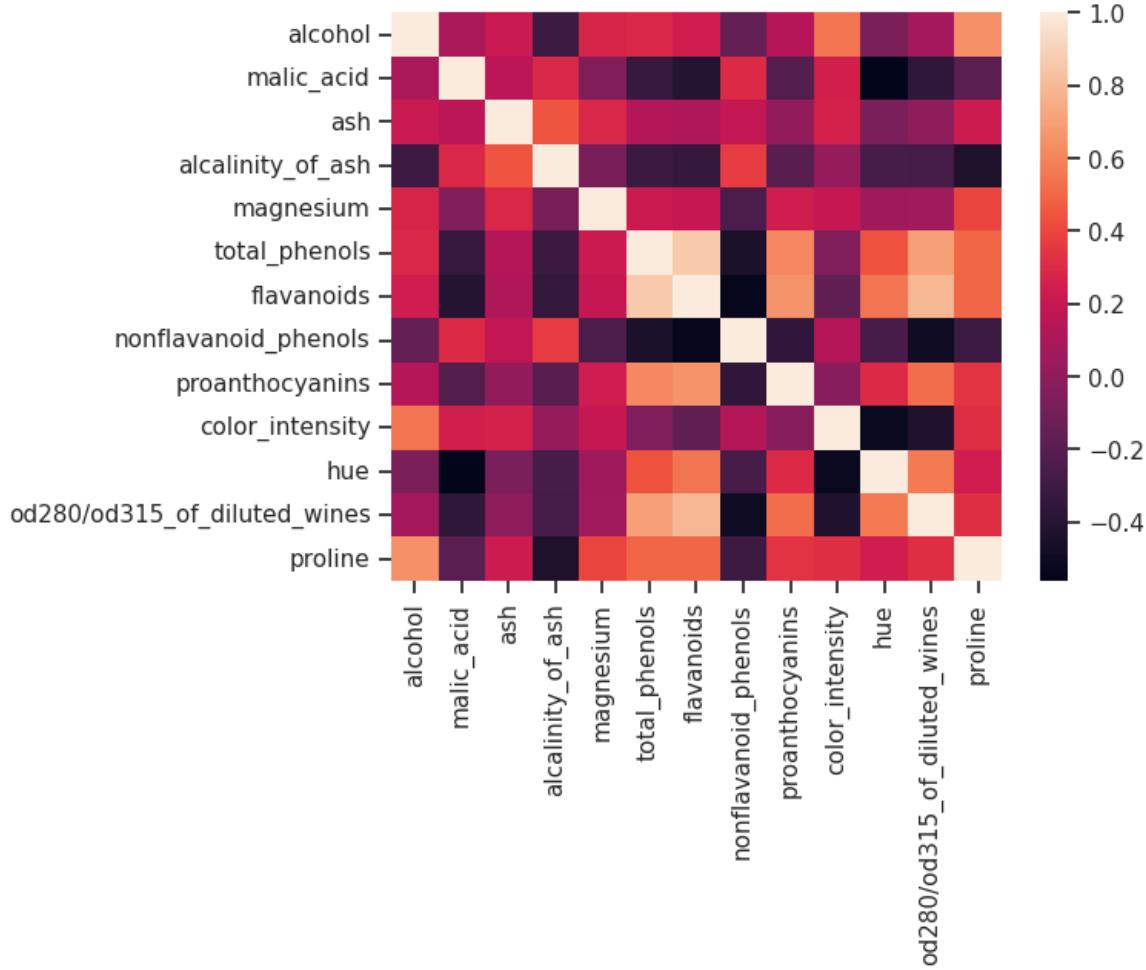
Для визуализации корреляционной матрицы будем использовать "тепловую карту" heatmap которая показывает степень корреляции различными цветами.

Используем метод heatmap библиотеки seaborn -
<https://seaborn.pydata.org/generated/seaborn.heatmap.html>

```
sns.heatmap(wine_df.corr())
```

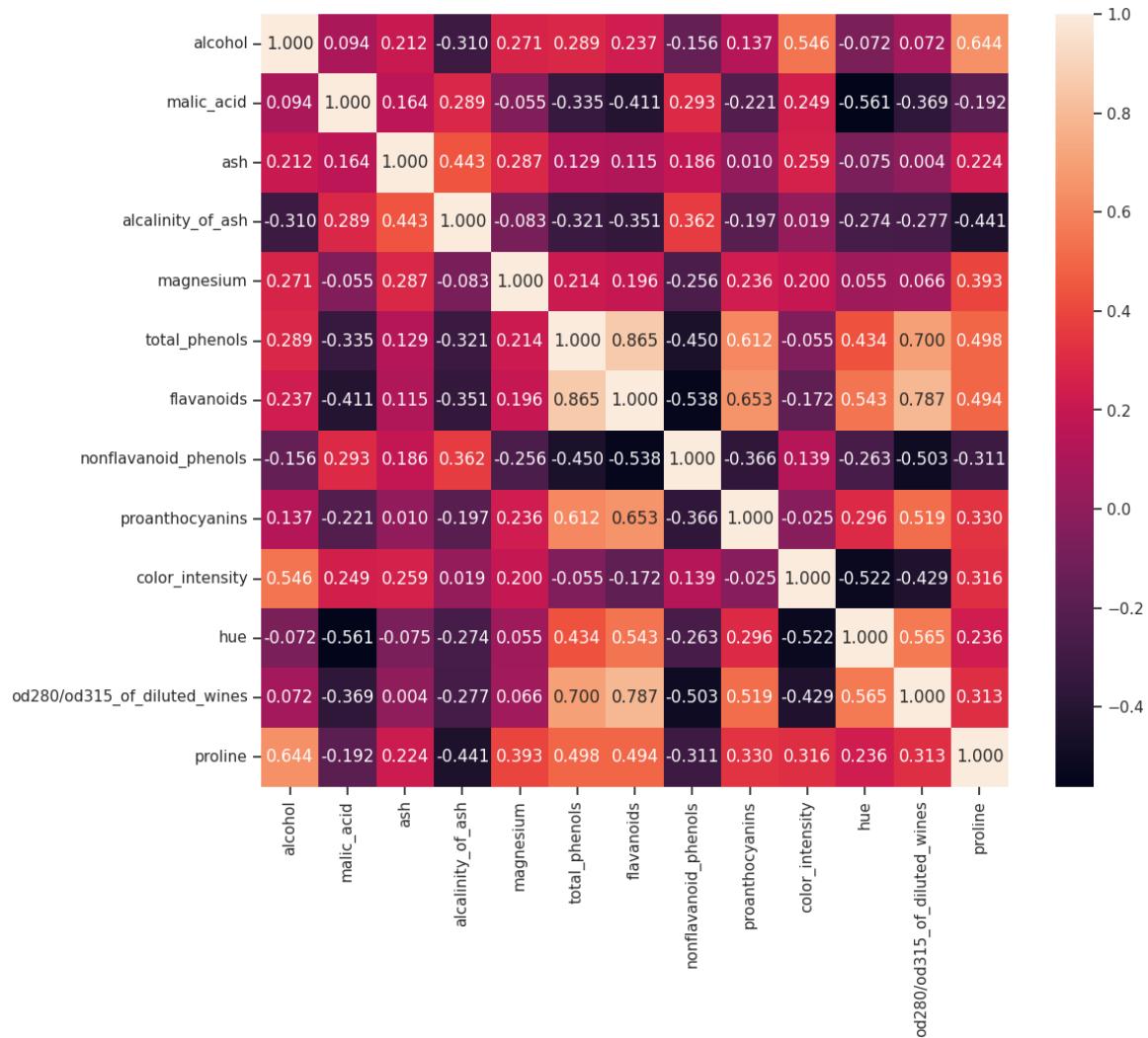
<Axes: >

[Download](#)



```
# Вывод значений в ячейках
plt.figure(figsize=(12, 10)) # Устанавливаем размер фигуры
sns.heatmap(wine_df.corr(), annot=True, fmt='.{3f}')
plt.show()
```

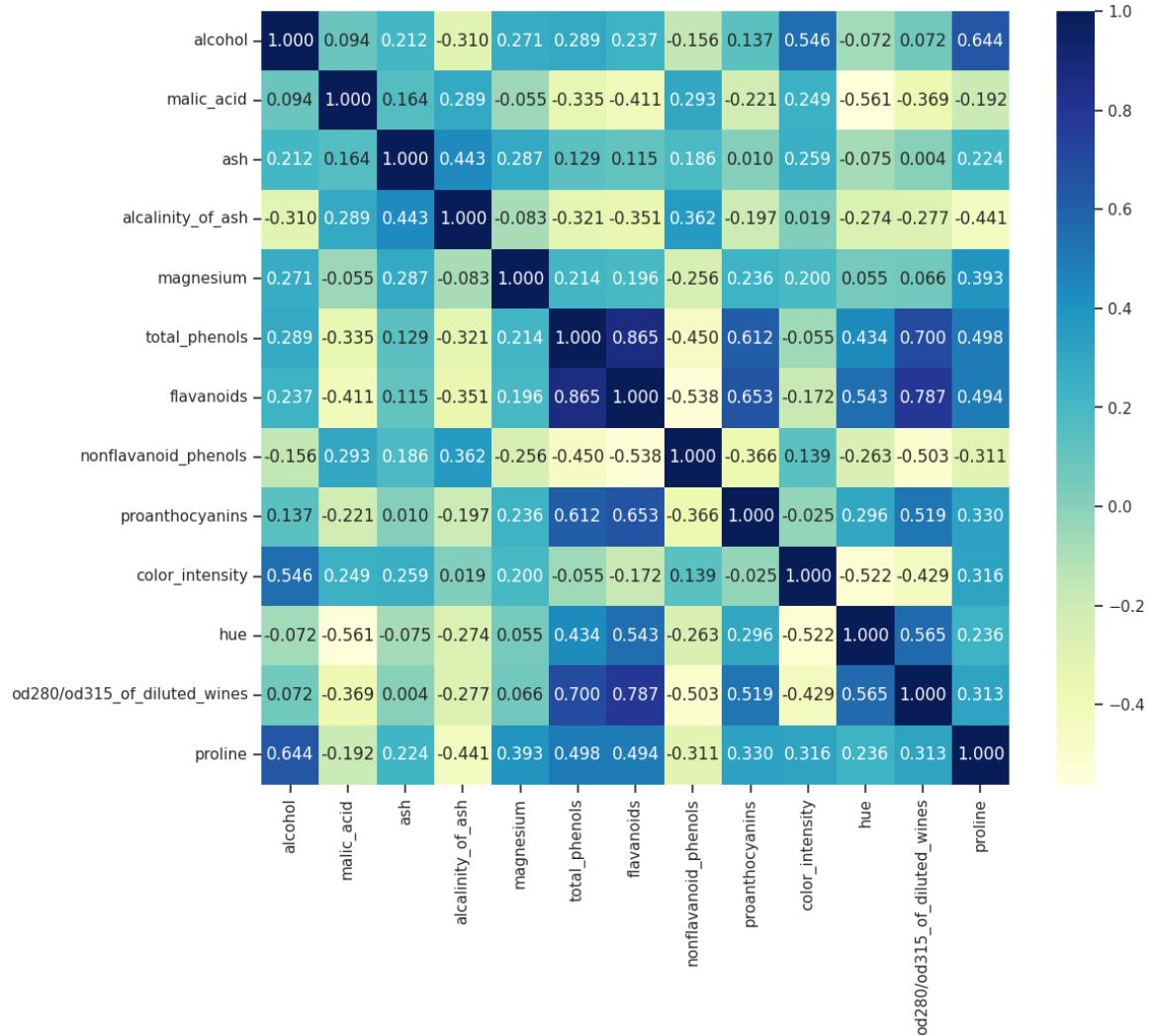
[Download](#)



При тренировке модели следует выбрать один из двух признаков, наиболее коррелирующих с целевым flavanoids: total_phenols (0,865) и od280/od315_of_diluted_wines (0,787). Одновременно вдвоем их использовать нельзя для тренировки, так как они также сильно коррелируют между собой (0,7).

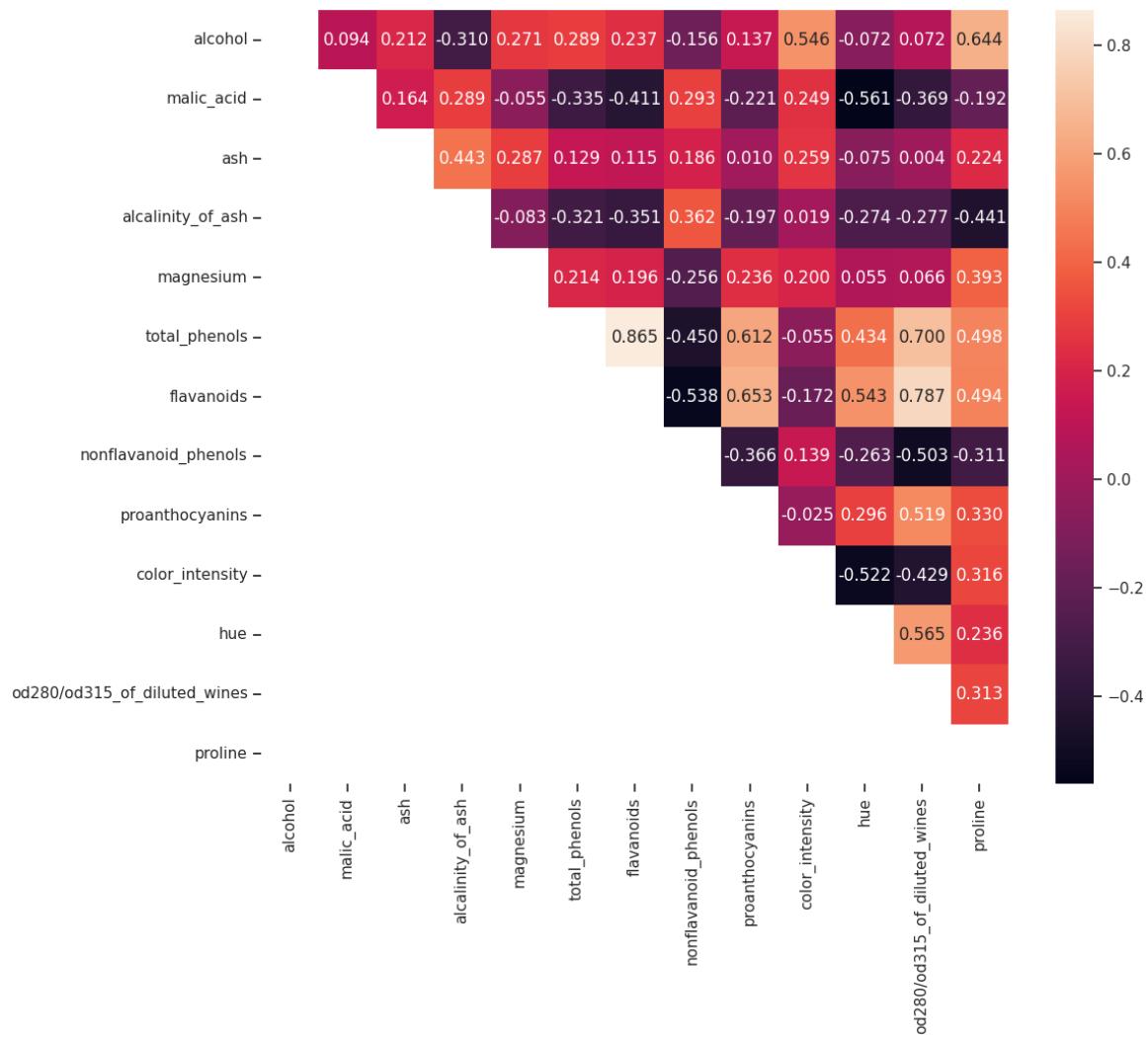
```
# Изменение цветовой гаммы
plt.figure(figsize=(12, 10))
sns.heatmap(wine_df.corr(), cmap='YlGnBu', annot=True, fmt='.3f')
plt.show()
```

[Download](#)



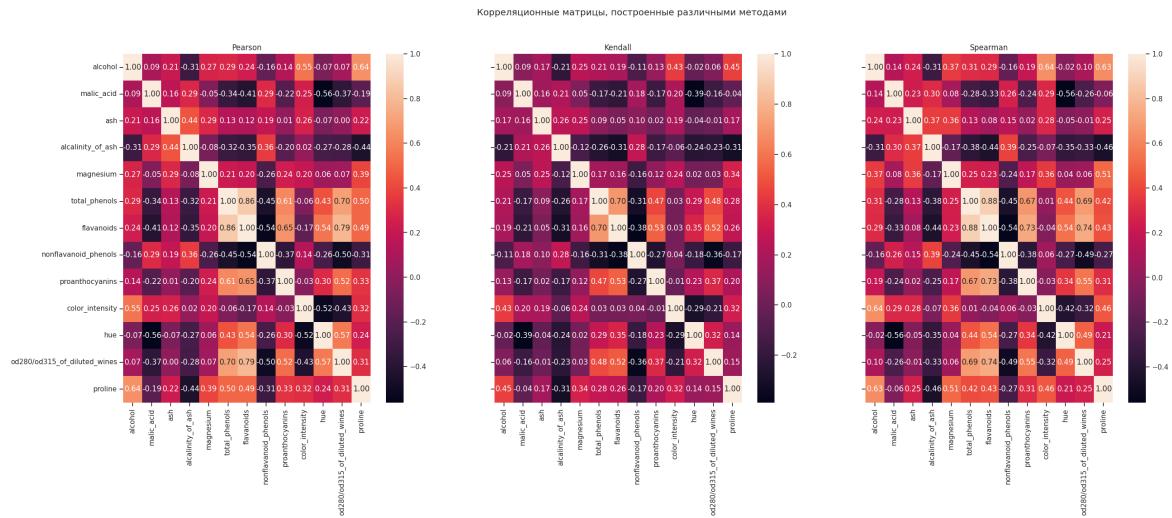
```
# Треугольный вариант матрицы
plt.figure(figsize=(12, 10))
mask = np.zeros_like(wine_df.corr(), dtype=bool)
# чтобы оставить нижнюю часть матрицы
# mask[np.triu_indices_from(mask)] = True
# чтобы оставить верхнюю часть матрицы
mask[np.tril_indices_from(mask)] = True
sns.heatmap(wine_df.corr(), mask=mask, annot=True, fmt='.3f')
plt.show()
```

[Download](#)



```
fig, ax = plt.subplots(1, 3, sharex='col', sharey='row', figsize=(3
sns.heatmap(wine_df.corr(method='pearson'), ax=ax[0], annot=True, f
sns.heatmap(wine_df.corr(method='kendall'), ax=ax[1], annot=True, f
sns.heatmap(wine_df.corr(method='spearman'), ax=ax[2], annot=True,
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')
```

[Download](#)



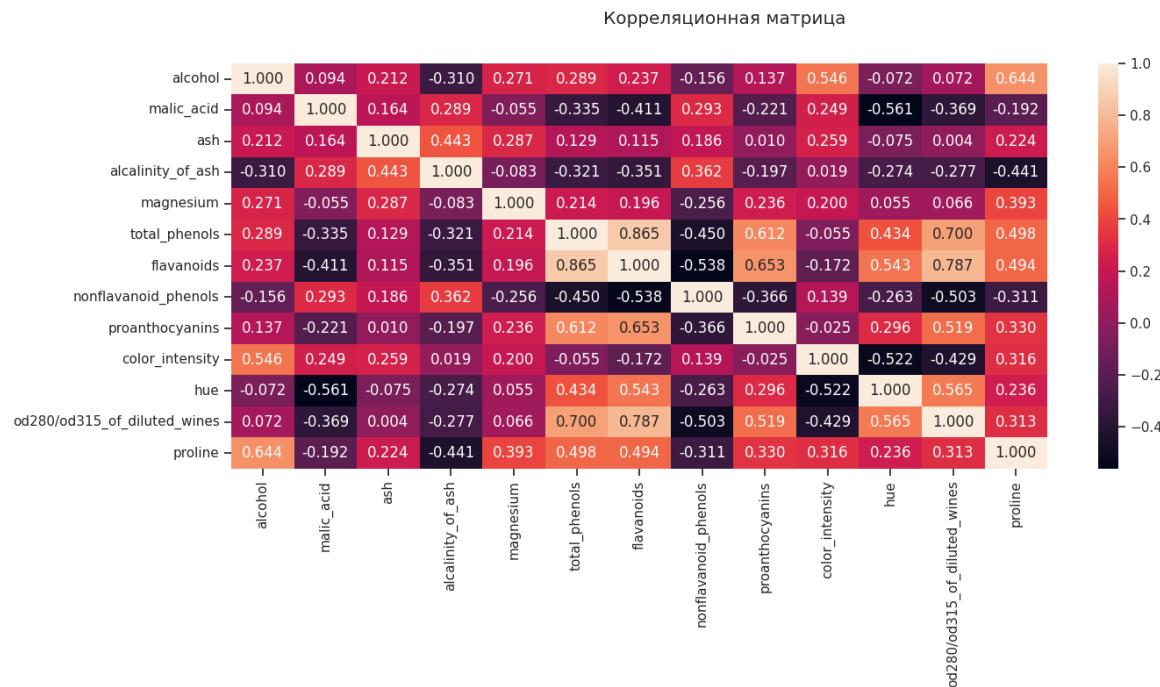
Тепловая карта с указанием размера

- Функция plt.subplots создает область окна нужного размера, в которую может выводиться график.
- Параметр figsize=(размер по горизонтали, размер по вертикали).
- По умолчанию размер задается в дюймах, но возможно использование и других единиц измерения – https://matplotlib.org/devdocs/gallery/subplots_axes_and_figures/figure_size_units
- Функция sns.heatmap содержит параметр ax=ax, который ссылается на область, созданную plt.subplots, поэтому график выводится в данной области.

```
fig, ax = plt.subplots(1, 1, sharex='col', sharey='row', figsize=(1
fig.suptitle('Корреляционная матрица')
sns.heatmap(wine_df.corr(), ax=ax, annot=True, fmt='.3f')
```

<Axes: >

[Download](#)



Необходимо отметить, что тепловая карта не очень хорошо подходит для определения корреляции нецелевых признаков между собой. В реальной модели могут быть сотни признаков и коррелирующие признаки могут образовывать группы, состоящие более чем из двух признаков. Увидеть такие группы с помощью тепловой карты сложно.

Для решения задачи предлагается новый вариант визуализации – "Солнечная корреляционная карта" Solar correlation map.

К сожалению, данная библиотека пока работает только через файловый интерфейс и не предназначена для встраивания в ноутбук.

Примеры статей с описанием работы библиотеки:

<https://www.oreilly.com/learning/a-new-visualization-to-beautifully-explore-correlations>

<https://www.mtab.com/the-puzzle-of-visualizing-correlations/>