МГТУ им. Н. Э. Баумана

Кафедра «Системы обработки информации и управления»

Лабораторная работа №1

по курсу «Технологии машинного обучения»

«Разведочный анализ данных. Исследование и визуализация данных»

Выполнила:

Ларионова А.П., ИУ5-63Б

Преподаватель:

Гапанюк Ю.Е.

Москва, 2022 год

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.

Выполнение работы:

1) Текстовое описание выбранного набора данных

Для первой лабораторной работы я выбрала Toy-dataset “Wine recognition dataset”, который содержит следующие колонки:

- Alcohol- алкоголь
- Malic acid-яблочная кислота
- Ash-зола
- Alcalinity of ash-щелочность золы
- Magnesium-магний
- Total phenols-общие фенолы
- Flavanoids-флаваноиды
- Nonflavanoid phenols-нефлавоидные фенолы
- Proanthocyanins-проантоцианы
- Color intensity-интенсивность цвета
- Hue- оттенок
- OD280/OD315 of diluted wines
- Proline- пролин
- Target-целевой признак

class:

  - class_0
  - class_1

- class_2

Импортируем библиотеки с помощью команды import

```
In [48]: import numpy as np
         import pandas as pd
         import seaborn as sns
         import matplotlib.pyplot as plt
         from sklearn.datasets import *

         wine=load_wine()
         type(wine)

Out[48]: sklearn.utils.Bunch
```

Загрузка данных. Преобразование наборов данных Scikit-learn в Pandas Dataframe.

```
In [49]: for x in wine:
             print(x)

data
target
frame
target_names
DESCR
feature_names

In [50]: wine['target_names']

Out[50]: array(['class_0', 'class_1', 'class_2'], dtype='<U7')

In [51]: wine['feature_names']

Out[51]: ['alcohol',
          'malic_acid',
          'ash',
          'alcalinity_of_ash',
          'magnesium',
          'total_phenols',
          'flavanoids',
          'nonflavanoid_phenols',
          'proanthocyanins',
          'color_intensity',
          'hue',
          'od280/od315_of_diluted_wines',
          'proline']

In [52]: wine['data'].shape

Out[52]: (178, 13)

In [53]: wine['target'].shape

Out[53]: (178,)

In [85]: df=pd.DataFrame(data=np.c_[wine['data'],wine['target']],
                         columns=list(wine['feature_names'])+['target'])
```

In [55]:
```
df
```

Out[55]:

	alcohol	malic_acid	ash	alcalinity_of_ash	magnesium	total_phenols	flavanoids	nonflavanoid_phenols	proanthocyanins	color_intensity	hue	od280/od3
0	14.23	1.71	2.43	15.6	127.0	2.80	3.06	0.28	2.29	5.64	1.04	
1	13.20	1.78	2.14	11.2	100.0	2.65	2.76	0.26	1.28	4.38	1.05	
2	13.16	2.36	2.67	18.6	101.0	2.80	3.24	0.30	2.81	5.68	1.03	
3	14.37	1.95	2.50	16.8	113.0	3.85	3.49	0.24	2.18	7.80	0.86	
4	13.24	2.59	2.87	21.0	118.0	2.80	2.69	0.39	1.82	4.32	1.04	
...	...	...	...	...	...	...	...	...	...	...	...	
173	13.71	5.65	2.45	20.5	95.0	1.68	0.61	0.52	1.06	7.70	0.64	
174	13.40	3.91	2.48	23.0	102.0	1.80	0.75	0.43	1.41	7.30	0.70	
175	13.27	4.28	2.26	20.0	120.0	1.59	0.69	0.43	1.35	10.20	0.59	
176	13.17	2.59	2.37	20.0	120.0	1.65	0.68	0.53	1.46	9.30	0.60	
177	14.13	4.10	2.74	24.5	96.0	2.05	0.76	0.56	1.35	9.20	0.61	

178 rows × 14 columns

2) Основные характеристики Dataset

In [56]:
```
df.head()
```

Out[56]:

	alcohol	malic_acid	ash	alcalinity_of_ash	magnesium	total_phenols	flavanoids	nonflavanoid_phenols	proanthocyanins	color_intensity	hue	od280/od315
0	14.23	1.71	2.43	15.6	127.0	2.80	3.06	0.28	2.29	5.64	1.04	
1	13.20	1.78	2.14	11.2	100.0	2.65	2.76	0.26	1.28	4.38	1.05	
2	13.16	2.36	2.67	18.6	101.0	2.80	3.24	0.30	2.81	5.68	1.03	
3	14.37	1.95	2.50	16.8	113.0	3.85	3.49	0.24	2.18	7.80	0.86	
4	13.24	2.59	2.87	21.0	118.0	2.80	2.69	0.39	1.82	4.32	1.04	

In [57]:
```
df.shape
```

Out[57]:
```
(178, 14)
```

In [58]:
```
total_count=df.shape[0]
print('Всего строк: {}'.format(total_count))
```

```
Всего строк: 178
```

In [59]:
```
df.columns
```

Out[59]:
```
Index(['alcohol', 'malic_acid', 'ash', 'alcalinity_of_ash', 'magnesium',
       'total_phenols', 'flavanoids', 'nonflavanoid_phenols',
       'proanthocyanins', 'color_intensity', 'hue',
       'od280/od315_of_diluted_wines', 'proline', 'target'],
      dtype='object')
```

In [60]:
```
df.dtypes
```

Out[60]:
```
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 [61]:
```
# проверим наличие пустых значений
# цикл по колонкам датасета
for col in df.columns:
    temp_null_count=df[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
target - 0
```

In [62]:
```
# основные статистические характеристики набора данных
df.describe()
```

Out[62]:

	alcohol	malic_acid	ash	alcalinity_of_ash	magnesium	total_phenols	flavanoids	nonflavanoid_phenols	proanthocyanins	color_intensity	
count	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	17
mean	13.000618	2.336348	2.366517	19.494944	99.741573	2.295112	2.029270	0.361854	1.590899	5.058090	
std	0.811827	1.117146	0.274344	3.339564	14.282484	0.625851	0.998859	0.124453	0.572359	2.318286	
min	11.030000	0.740000	1.360000	10.600000	70.000000	0.980000	0.340000	0.130000	0.410000	1.280000	
25%	12.362500	1.602500	2.210000	17.200000	88.000000	1.742500	1.205000	0.270000	1.250000	3.220000	
50%	13.050000	1.865000	2.360000	19.500000	98.000000	2.355000	2.135000	0.340000	1.555000	4.690000	
75%	13.677500	3.082500	2.557500	21.500000	107.000000	2.800000	2.875000	0.437500	1.950000	6.200000	
max	14.830000	5.800000	3.230000	30.000000	162.000000	3.880000	5.080000	0.660000	3.580000	13.000000	

In [65]:
```
# целевой признак
df['target'].unique()
```

Out[65]:
```
array([0., 1., 2.])
```

3)Визуальное исследование Dataset

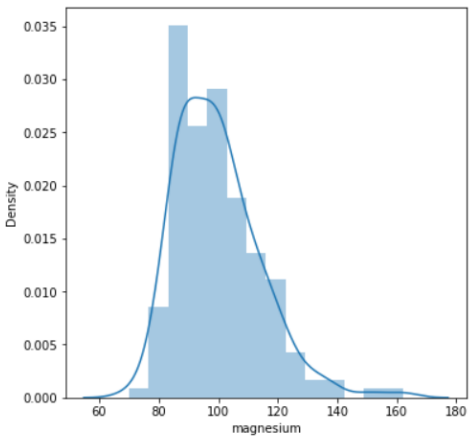
Гистограмма

Позволяет оценить плотность вероятности распределения данных

In [28]:
```
fig, ax = plt.subplots(figsize=(6,6))
sns.distplot(df['magnesium'])
```

```
C:\Users\amina\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function an
d will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar fle
xibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[28]:
```
<AxesSubplot:xlabel='magnesium', ylabel='Density'>
```



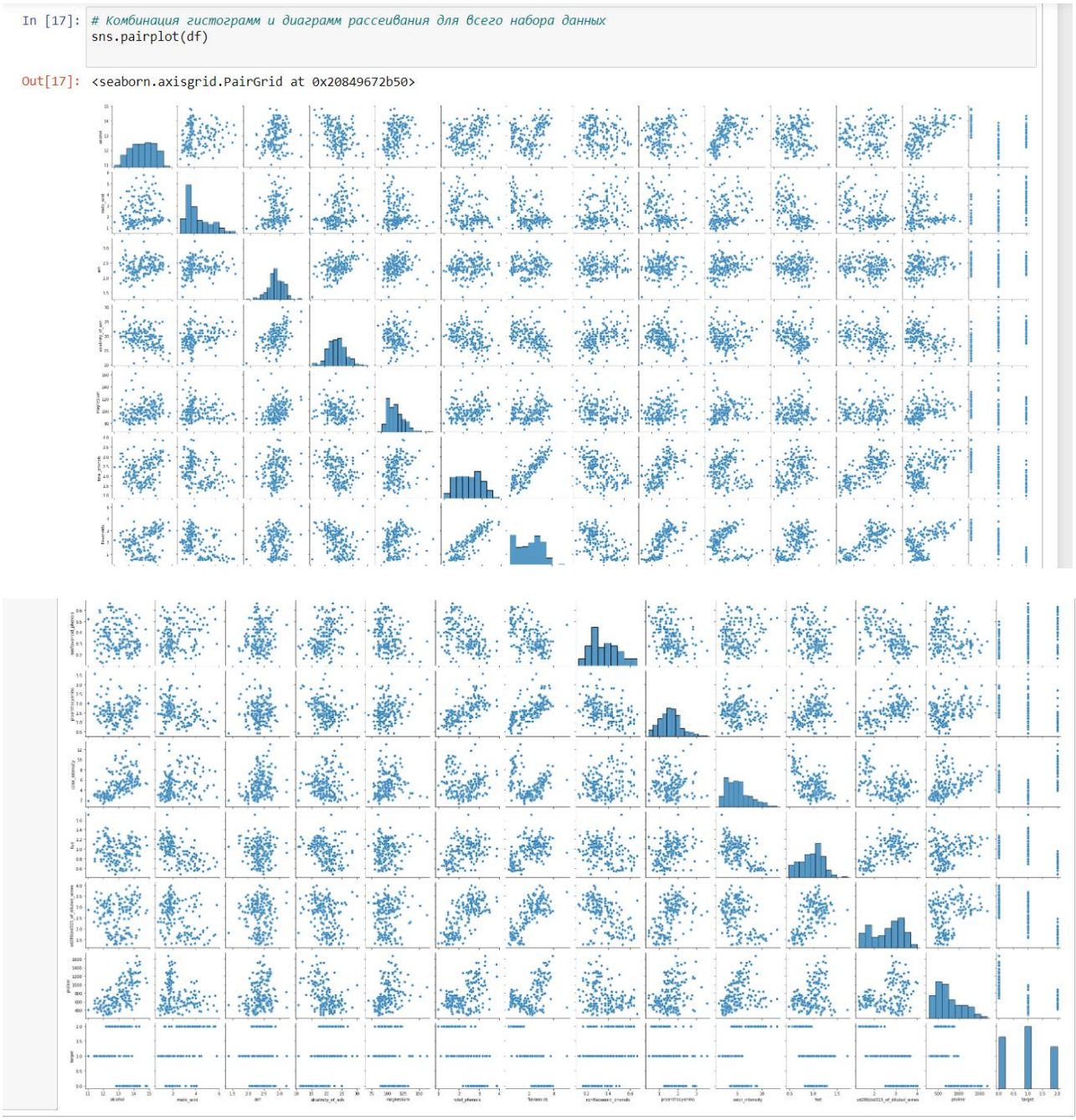
"Парные диаграммы"

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.

```
In [17]: # Комбинация гистограмм и диаграмм рассеивания для всего набора данных
         sns.pairplot(df)

Out[17]: <seaborn.axisgrid.PairGrid at 0x20849672b50>
```

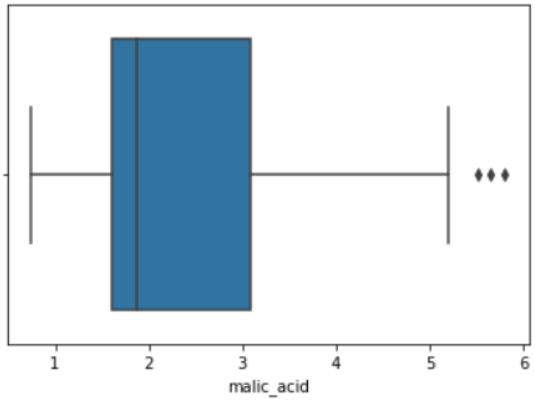


Ящик с усами

Отображает одномерное распределение вероятности

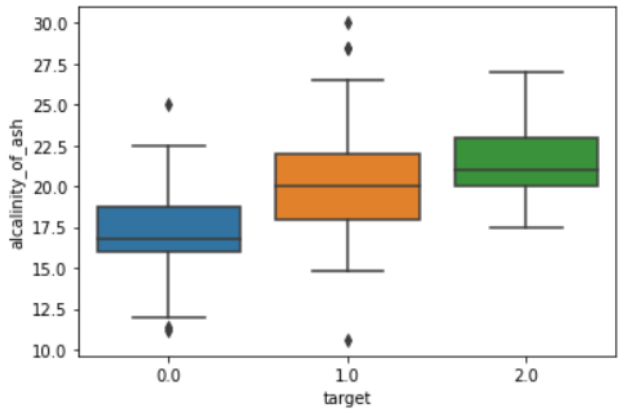
```
In [19]: sns.boxplot(x=df['malic_acid'])
```

```
Out[19]: <AxesSubplot:xlabel='malic_acid'>
```




```
In [20]: sns.boxplot(x='target', y='alcalinity_of_ash', data=df)
```

```
Out[20]: <AxesSubplot:xlabel='target', ylabel='alcalinity_of_ash'>
```

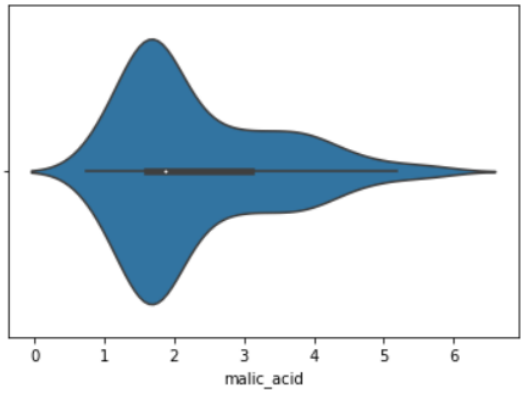


Violin plot

Похоже на предыдущую диаграмму, но по краям отображаются распределения плотности

```
In [21]: sns.violinplot(x=df['malic_acid'])
```

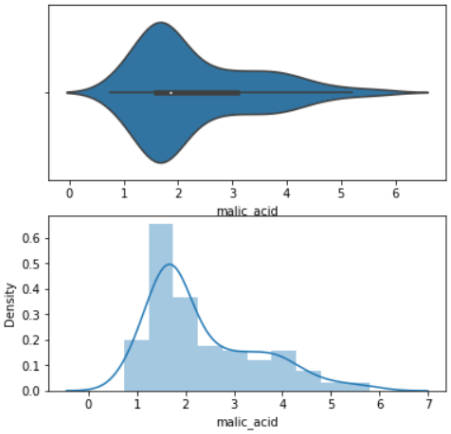
```
Out[21]: <AxesSubplot:xlabel='malic_acid'>
```



In [29]:
```
fig, ax = plt.subplots(2, 1, figsize=(6,6))
sns.violinplot(ax=ax[0], x=df['malic_acid'])
sns.distplot(df['malic_acid'], ax=ax[1])
```

```
C:\Users\amina\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function an
d will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar fle
xibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[29]:
```
<AxesSubplot:xlabel='malic_acid', ylabel='Density'>
```



4) Информация о корреляции признаков

Проверка корреляции признаков позволяет решить две задачи:

1.Понять какие признаки (колонки датасета) наиболее сильно коррелируют с целевым признаком. 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 [23]: df.corr()
```

Out[23]:

	alcohol	malic_acid	ash	alcalinity_of_ash	magnesium	total_phenols	flavanoids	nonflavanoid_phenols	proanthocyanins
alcohol	1.000000	0.094397	0.211545	-0.310235	0.270798	0.289101	0.236815	-0.155929	0.136698
malic_acid	0.094397	1.000000	0.164045	0.288500	-0.054575	-0.335167	-0.411007	0.292977	-0.220746
ash	0.211545	0.164045	1.000000	0.443367	0.286587	0.128980	0.115077	0.186230	0.009652
alcalinity_of_ash	-0.310235	0.288500	0.443367	1.000000	-0.083333	-0.321113	-0.351370	0.361922	-0.197327
magnesium	0.270798	-0.054575	0.286587	-0.083333	1.000000	0.214401	0.195784	-0.256294	0.236441
total_phenols	0.289101	-0.335167	0.128980	-0.321113	0.214401	1.000000	0.864564	-0.449935	0.612413
flavanoids	0.236815	-0.411007	0.115077	-0.351370	0.195784	0.864564	1.000000	-0.537900	0.652692
nonflavanoid_phenols	-0.155929	0.292977	0.186230	0.361922	-0.256294	-0.449935	-0.537900	1.000000	-0.365845
proanthocyanins	0.136698	-0.220746	0.009652	-0.197327	0.236441	0.612413	0.652692	-0.365845	1.000000
color_intensity	0.546364	0.248985	0.258887	0.018732	0.199950	-0.055136	-0.172379	0.139057	-0.025250
hue	-0.071747	-0.561296	-0.074667	-0.273955	0.055398	0.433681	0.543479	-0.262640	0.295544
od280/od315_of_diluted_wines	0.072343	-0.368710	0.003911	-0.276769	0.066004	0.699949	0.787194	-0.503270	0.519067
proline	0.643720	-0.192011	0.223626	-0.440597	0.393351	0.498115	0.494193	-0.311385	0.330417
target	-0.328222	0.437776	-0.049643	0.517859	-0.209179	-0.719163	-0.847498	0.489109	-0.499130

На основе корреляционной матрицы можно сделать следующие выводы:

- Целевой признак target наиболее сильно коррелирует с flavonoids (-0.85), с OD280/OD315 of diluted wines (-0.79) и с total_phenols (-0,72). Эти признаки обязательно следует оставить в модели.
- Целевой признак очень слабо коррелирует с ash (-0.05). Скорее всего этот признак стоит исключить из модели, возможно он только ухудшат её качество.
- Flavonoids и total_phenols сильно коррелируют между собой (0.86).
- Также flavonoids достаточно сильно коррелирует с OD280/OD315 of diluted wines (0,79).
- Можно сделать вывод, что выбирая из признаков Flavonoids, OD280/OD315 of diluted wines и total_phenols лучше выбрать flavonoids, потому что он сильнее коррелирован с целевым признаком.

Если линейно зависимые признаки сильно коррелированы с целевым, то оставляют именно тот признак, который коррелирован с целевым сильнее.

Построение корреляционной матрицы методом Пирсона:

In [24]:
```
df.corr(method='pearson')
```

Out[24]:

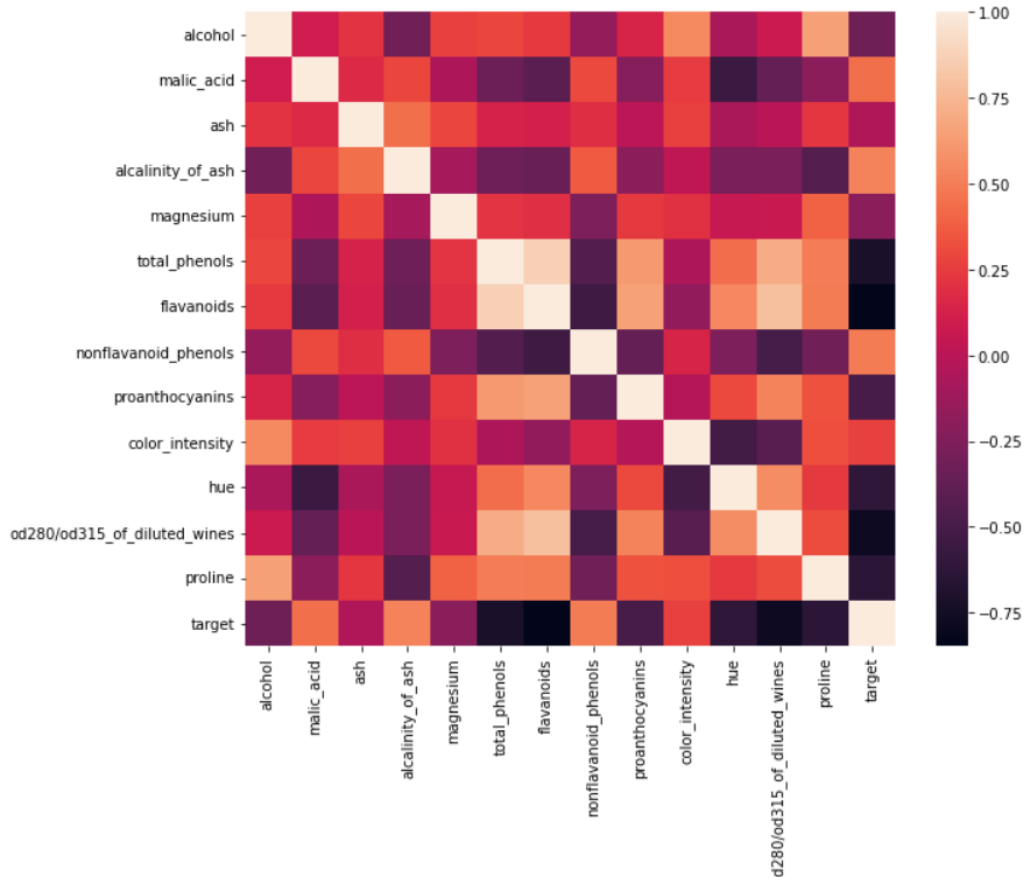
	alcohol	malic_acid	ash	alcalinity_of_ash	magnesium	total_phenols	flavanoids	nonflavanoid_phenols	proanthocyanins
alcohol	1.000000	0.094397	0.211545	-0.310235	0.270798	0.289101	0.236815	-0.155929	0.136698
malic_acid	0.094397	1.000000	0.164045	0.288500	-0.054575	-0.335167	-0.411007	0.292977	-0.220746
ash	0.211545	0.164045	1.000000	0.443367	0.286587	0.128980	0.115077	0.186230	0.009652
alcalinity_of_ash	-0.310235	0.288500	0.443367	1.000000	-0.083333	-0.321113	-0.351370	0.361922	-0.197327
magnesium	0.270798	-0.054575	0.286587	-0.083333	1.000000	0.214401	0.195784	-0.256294	0.236441
total_phenols	0.289101	-0.335167	0.128980	-0.321113	0.214401	1.000000	0.864564	-0.449935	0.612413
flavanoids	0.236815	-0.411007	0.115077	-0.351370	0.195784	0.864564	1.000000	-0.537900	0.652692
nonflavanoid_phenols	-0.155929	0.292977	0.186230	0.361922	-0.256294	-0.449935	-0.537900	1.000000	-0.365845
proanthocyanins	0.136698	-0.220746	0.009652	-0.197327	0.236441	0.612413	0.652692	-0.365845	1.000000
color_intensity	0.546364	0.248985	0.258887	0.018732	0.199950	-0.055136	-0.172379	0.139057	-0.025250
hue	-0.071747	-0.561296	-0.074667	-0.273955	0.055398	0.433681	0.543479	-0.262640	0.295544
od280/od315_of_diluted_wines	0.072343	-0.368710	0.003911	-0.276769	0.066004	0.699949	0.787194	-0.503270	0.519067
proline	0.643720	-0.192011	0.223626	-0.440597	0.393351	0.498115	0.494193	-0.311385	0.330417
target	-0.328222	0.437776	-0.049643	0.517859	-0.209179	-0.719163	-0.847498	0.489109	-0.499130

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.

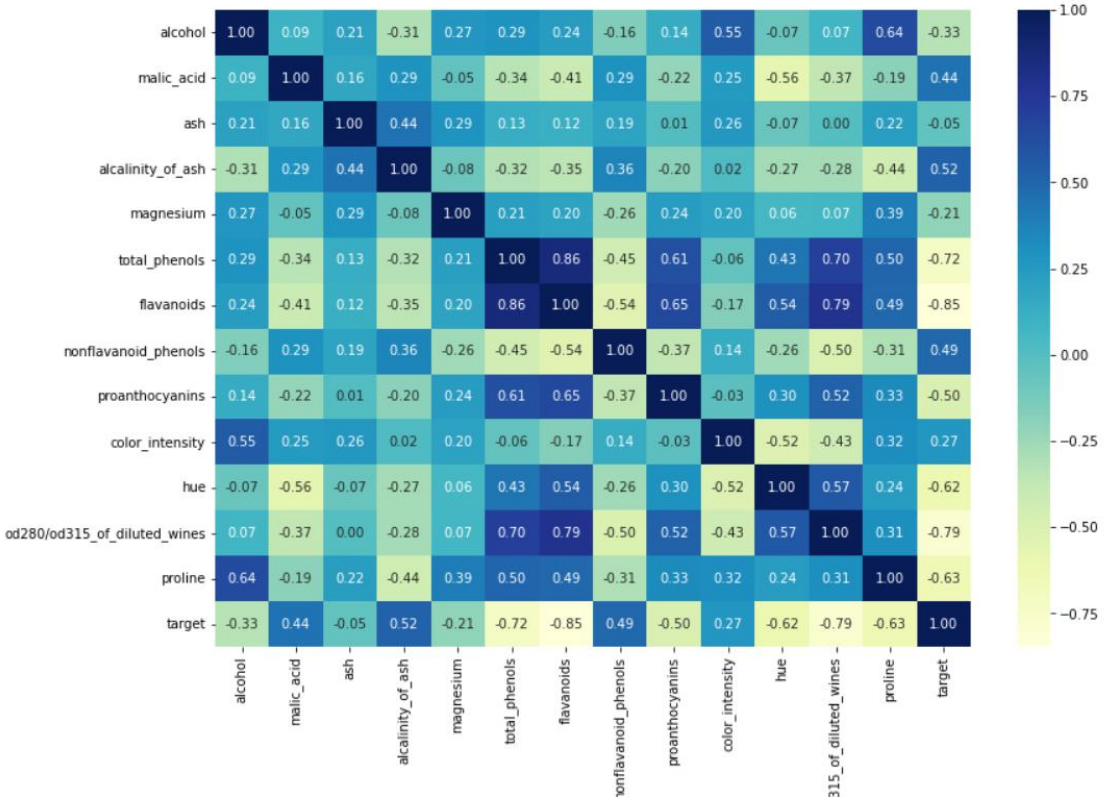
In [32]:
```
# Для визуализации корреляционной матрицы будем использовать 'тепловую карту'
plt.figure(figsize=(10,8))
sns.heatmap(df.corr())
```

Out[32]: <AxesSubplot:>



In [26]:
```
# Вывод значений в ячейках
plt.figure(figsize=(13,9))
sns.heatmap( df.corr(), cmap='YlGnBu', annot=True, fmt='.2f')
```

Out[26]: <AxesSubplot:>



In [27]:
```
# Треугольный вариант матрицы
plt.figure(figsize=(13,9))
mask = np.zeros_like(df.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(df.corr(), mask=mask, annot=True, fmt='.2f')
```

Out[27]: <AxesSubplot:>