МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
им. Н.Э. Баумана

Кафедра «Систем обработки информации и управления»

Лабораторная работа №1

по курсу «Методы машинного обучения»
на тему« Разведочный анализ данных. Исследование и визуализация данных »

Выполнил: Сефербеков М.С
группа ИУ5-21М

Москва - 2020

In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.datasets import load_diabetes
%matplotlib inline
sns.set(style="ticks")
```

In [2]:
```
data = pd.read_csv('heart.csv')
```

In [3]:
```
data.head()
```

Out[3]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1

In [4]:
```
data.shape
```

Out[4]:
```
(303, 14)
```

In [5]:
```
data.columns
```

Out[5]:
```
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
      dtype='object')
```

In [6]:
```
data.dtypes
```

Out[6]:
```
age           int64
sex           int64
cp            int64
trestbps      int64
chol          int64
fbs           int64
restecg       int64
thalach       int64
exang         int64
oldpeak     float64
slope         int64
ca            int64
thal          int64
target        int64
dtype: object
```

In [7]:
```
for col in data.columns:
    # Количество пустых значений - все значения заполнены
    temp_null_count = data[data[col].isnull()].shape[0]
    print('{} - {}'.format(col, temp_null_count))
```

```
age - 0
sex - 0
cp - 0
trestbps - 0
chol - 0
fbs - 0
restecg - 0
thalach - 0
exang - 0
oldpeak - 0
slope - 0
ca - 0
thal - 0
target - 0
```

In [8]:
```
data.describe()
```

Out[8]:

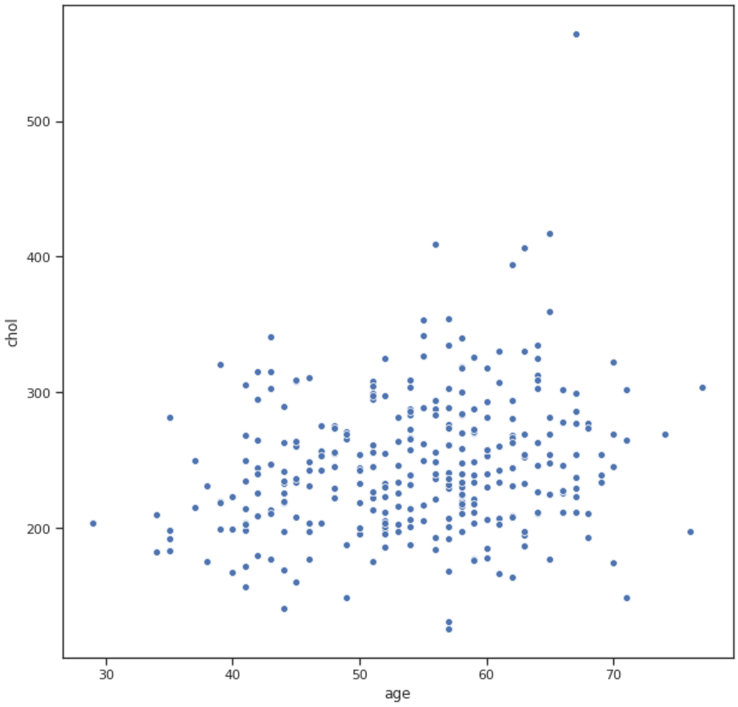
	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.
mean	54.366337	0.683168	0.966997	131.623762	246.264026	0.148515	0.528053	149.646865	0.326733	1.039604	1.399340	0.729373	2.313531	0.
std	9.082101	0.466011	1.032052	17.538143	51.830751	0.356198	0.525860	22.905161	0.469794	1.161075	0.616226	1.022606	0.612277	0.
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.
25%	47.500000	0.000000	0.000000	120.000000	211.000000	0.000000	0.000000	133.500000	0.000000	0.000000	1.000000	0.000000	2.000000	0.
50%	55.000000	1.000000	1.000000	130.000000	240.000000	0.000000	1.000000	153.000000	0.000000	0.800000	1.000000	0.000000	2.000000	1.
75%	61.000000	1.000000	2.000000	140.000000	274.500000	0.000000	1.000000	166.000000	1.000000	1.600000	2.000000	1.000000	3.000000	1.
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000	202.000000	1.000000	6.200000	2.000000	4.000000	3.000000	1.

In [9]:
```
# Определим уникальные значения для целевого признака
data['target'].unique()
```

Out[9]:
```
array([1, 0])
```

```
In [10]: fig, ax = plt.subplots(figsize=(10,10))
         sns.scatterplot(ax=ax, x='age', y='chol', data=data)
```

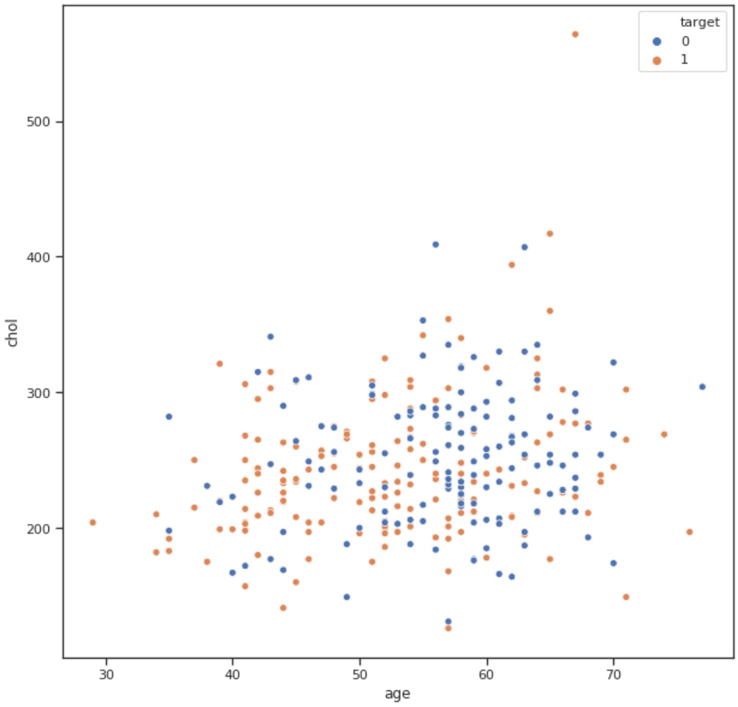
```
Out[10]: <matplotlib.axes._subplots.AxesSubplot at 0x7f69c0ae9c90>
```



```
In [11]: fig, ax = plt.subplots(figsize=(10,10))

         sns.scatterplot(ax=ax, x='age', y='chol', data=data,hue='target')
```

```
Out[11]: <matplotlib.axes._subplots.AxesSubplot at 0x7f69c05296d0>
```

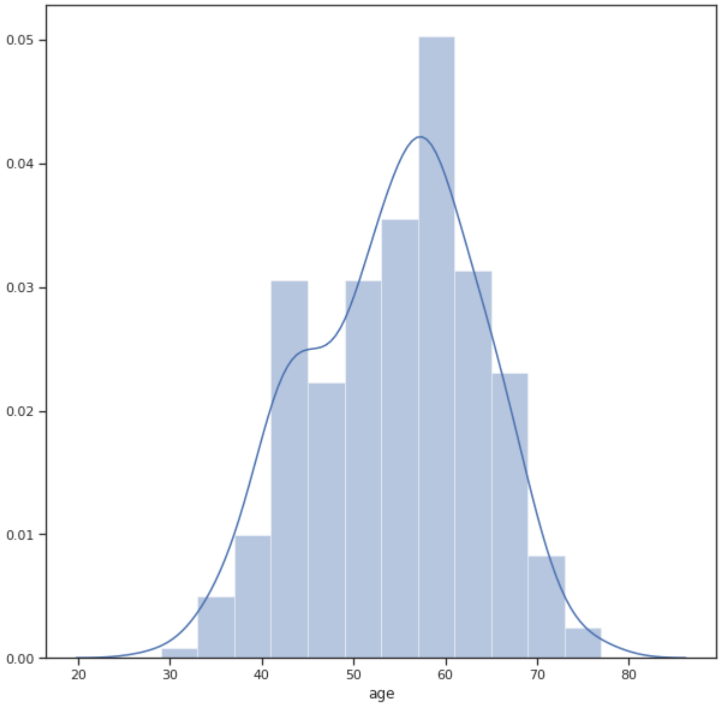


In [12]:

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

Out[12]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f69c06e9e10>
```

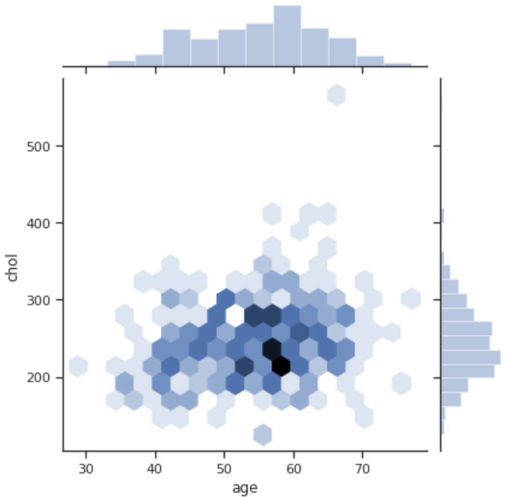


In [13]:

```
sns.jointplot(x='age', y='chol', data=data,kind="hex")
```

Out[13]:

```
<seaborn.axisgrid.JointGrid at 0x7f69c0614050>
```

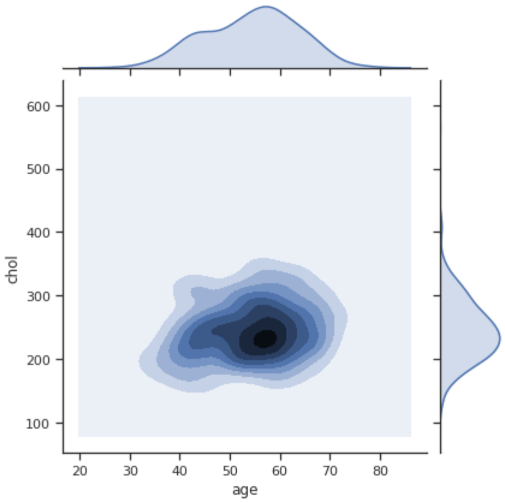


In [14]:

```
sns.jointplot(x='age', y='chol', data=data,kind="kde")
```

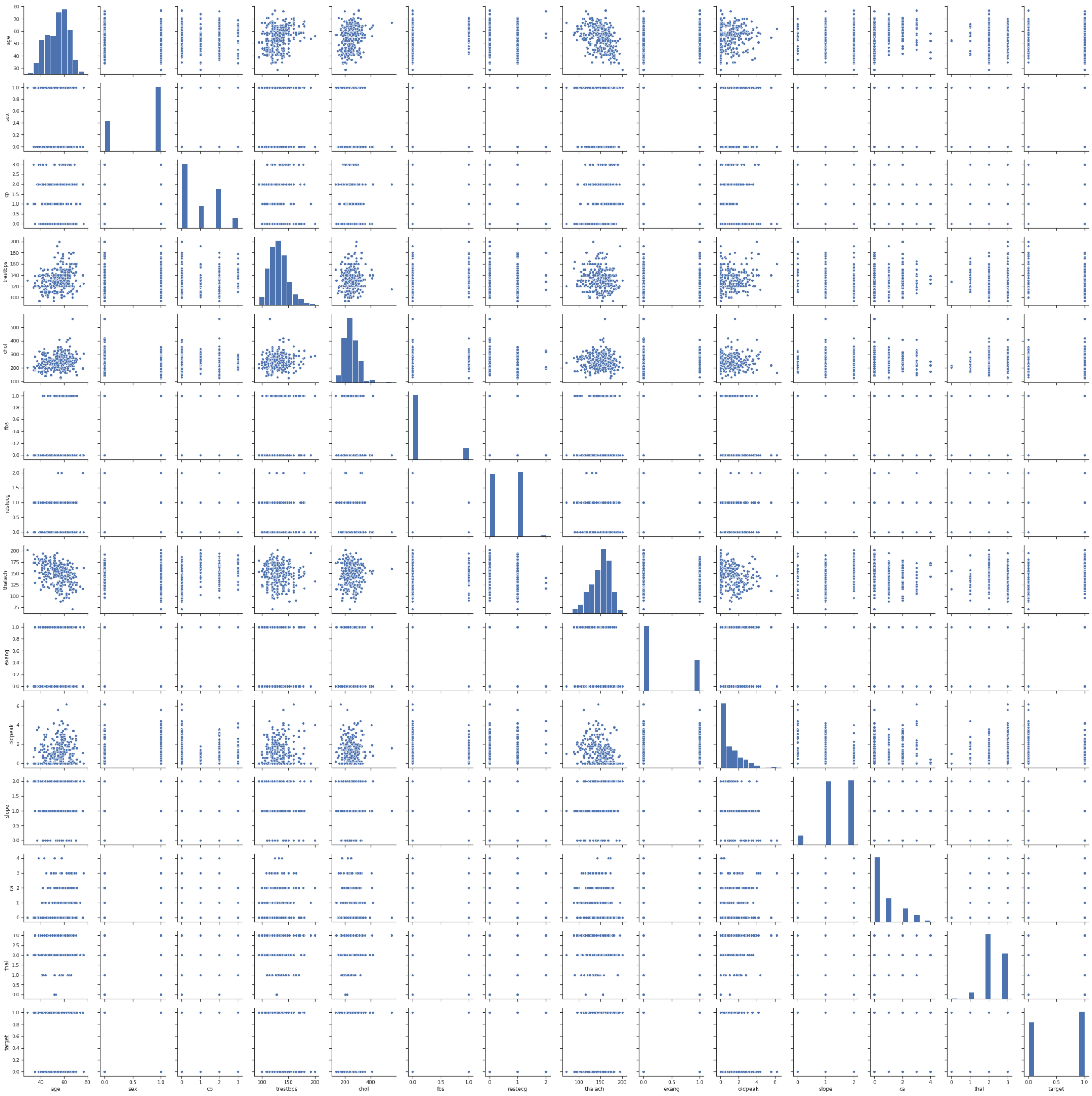
Out[14]:

```
<seaborn.axisgrid.JointGrid at 0x7f69c0308490>
```



```
In [15]: sns.pairplot(data)

Out[15]: <seaborn.axisgrid.PairGrid at 0x7f69c01c1890>
```



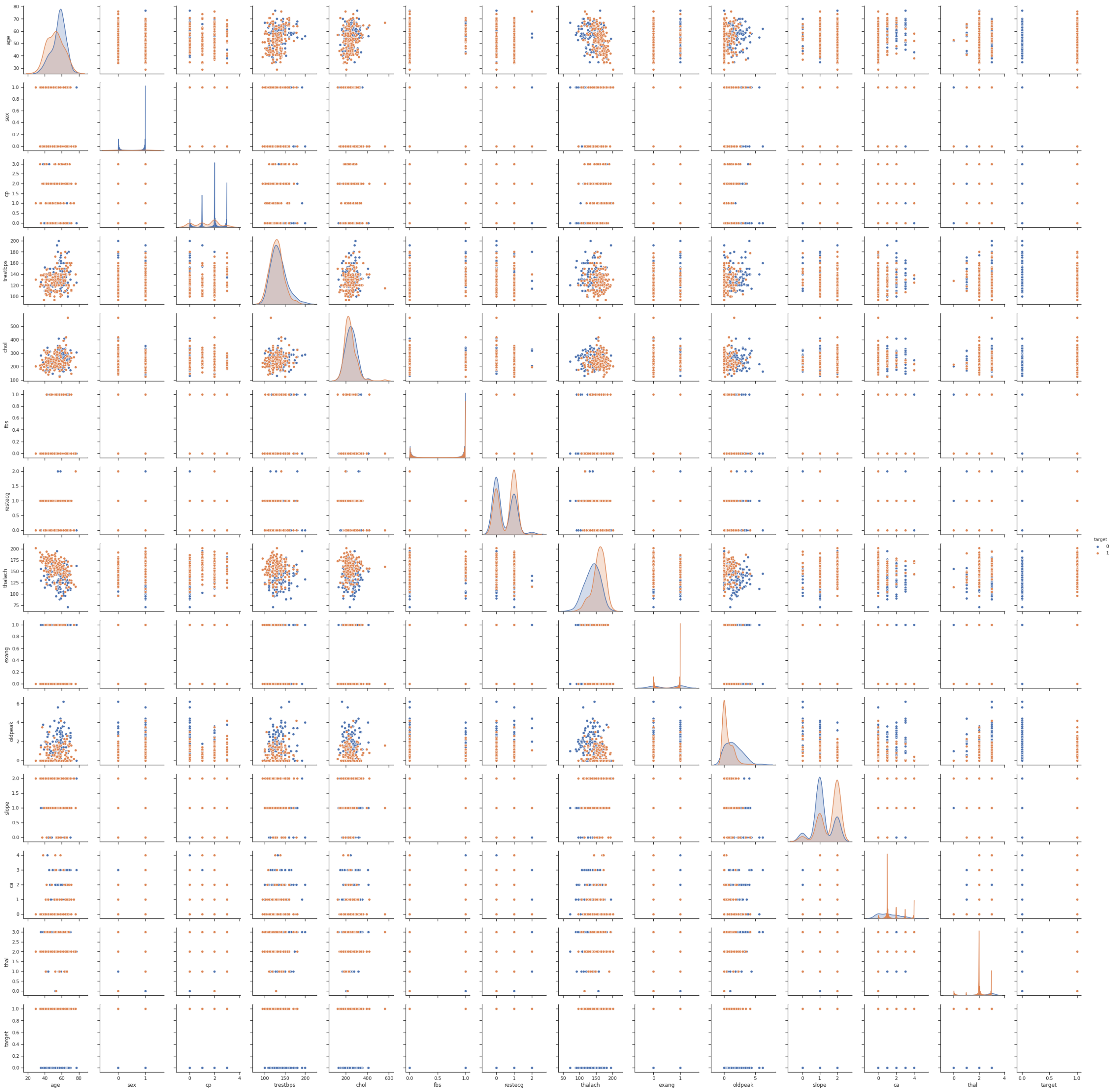
In [16]:

```
sns.pairplot(data,hue='target')
```

```
/home/darum/anaconda3/lib/python3.7/site-packages/statsmodels/nonparametric/kde.py:487: RuntimeWarning: invalid value encountered in true
_divide
  binned = fast_linbin(X, a, b, gridsize) / (delta * nobs)
/home/darum/anaconda3/lib/python3.7/site-packages/statsmodels/nonparametric/kdetools.py:34: RuntimeWarning: invalid value encountered in
double_scalars
  FAC1 = 2*(np.pi*bw/RANGE)**2
```

Out[16]:

```
<seaborn.axisgrid.PairGrid at 0x7f69ba00fad0>
```

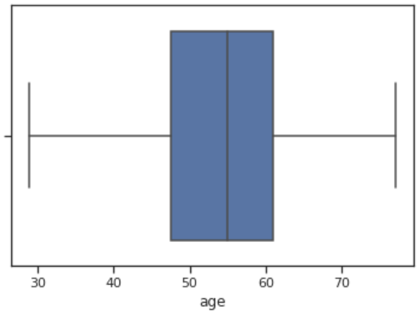


In [17]:

```
sns.boxplot(x=data['age'])
```

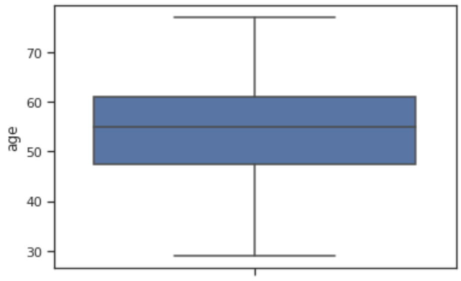
Out[17]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f69ad0cb550>
```



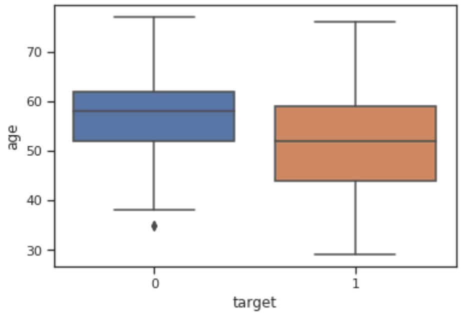
```
In [18]: # По вертикали
         sns.boxplot(y=data['age'])
```

```
Out[18]: <matplotlib.axes._subplots.AxesSubplot at 0x7f69ab610f10>
```



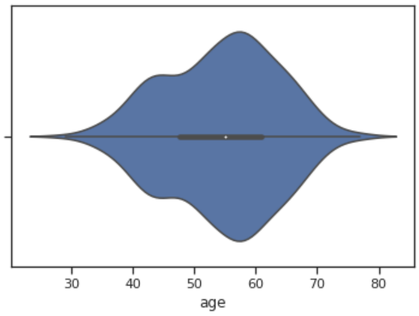
```
In [19]: sns.boxplot(x='target', y='age', data=data)
```

```
Out[19]: <matplotlib.axes._subplots.AxesSubplot at 0x7f69ab51c7d0>
```



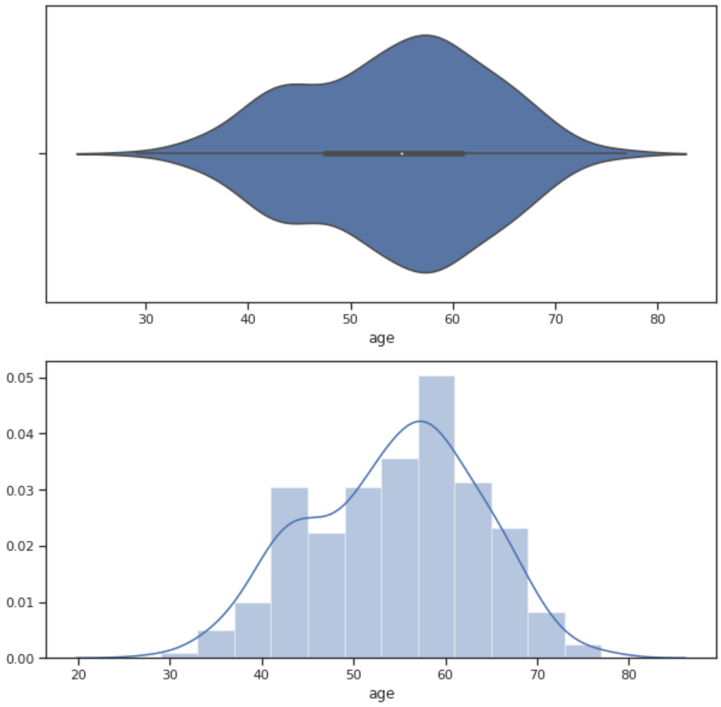
```
In [20]: sns.violinplot(x=data['age'])
```

```
Out[20]: <matplotlib.axes._subplots.AxesSubplot at 0x7f69acf3bc50>
```



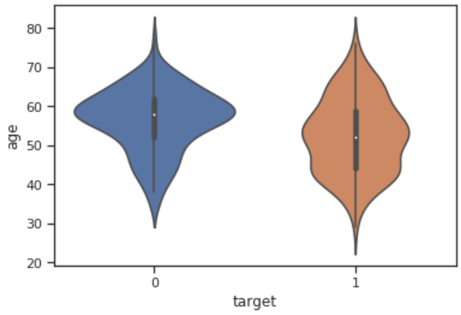
In [21]:
```
fig, ax = plt.subplots(2, 1, figsize=(10,10))
sns.violinplot(ax=ax[0], x=data['age'])
sns.distplot(data['age'], ax=ax[1])
```

Out[21]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f69acea24d0>
```



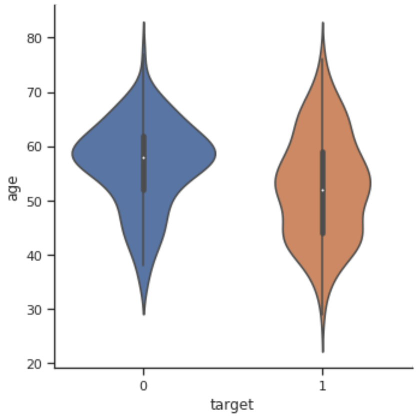
In [22]:
```
# Распределение параметра Humidity сгруппированные по Occupancy.
sns.violinplot(x='target', y='age', data=data)
```

Out[22]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f69ace3fe90>
```



In [23]:
```
sns.catplot(y='age', x='target', data=data, kind="violin", split=True)
```

Out[23]:
```
<seaborn.axisgrid.FacetGrid at 0x7f69acf38e10>
```



In [24]:

```
data.corr()
```

Out[24]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
age	1.000000	-0.098447	-0.068653	0.279351	0.213678	0.121308	-0.116211	-0.398522	0.096801	0.210013	-0.168814	0.276326	0.068001	-0.225439
sex	-0.098447	1.000000	-0.049353	-0.056769	-0.197912	0.045032	-0.058196	-0.044020	0.141664	0.096093	-0.030711	0.118261	0.210041	-0.280937
cp	-0.068653	-0.049353	1.000000	0.047608	-0.076904	0.094444	0.044421	0.295762	-0.394280	-0.149230	0.119717	-0.181053	-0.161736	0.433798
trestbps	0.279351	-0.056769	0.047608	1.000000	0.123174	0.177531	-0.114103	-0.046698	0.067616	0.193216	-0.121475	0.101389	0.062210	-0.144931
chol	0.213678	-0.197912	-0.076904	0.123174	1.000000	0.013294	-0.151040	-0.009940	0.067023	0.053952	-0.004038	0.070511	0.098803	-0.085239
fbs	0.121308	0.045032	0.094444	0.177531	0.013294	1.000000	-0.084189	-0.008567	0.025665	0.005747	-0.059894	0.137979	-0.032019	-0.028046
restecg	-0.116211	-0.058196	0.044421	-0.114103	-0.151040	-0.084189	1.000000	0.044123	-0.070733	-0.058770	0.093045	-0.072042	-0.011981	0.137230
thalach	-0.398522	-0.044020	0.295762	-0.046698	-0.009940	-0.008567	0.044123	1.000000	-0.378812	-0.344187	0.386784	-0.213177	-0.096439	0.421741
exang	0.096801	0.141664	-0.394280	0.067616	0.067023	0.025665	-0.070733	-0.378812	1.000000	0.288223	-0.257748	0.115739	0.206754	-0.436757
oldpeak	0.210013	0.096093	-0.149230	0.193216	0.053952	0.005747	-0.058770	-0.344187	0.288223	1.000000	-0.577537	0.222682	0.210244	-0.430696
slope	-0.168814	-0.030711	0.119717	-0.121475	-0.004038	-0.059894	0.093045	0.386784	-0.257748	-0.577537	1.000000	-0.080155	-0.104764	0.345877
ca	0.276326	0.118261	-0.181053	0.101389	0.070511	0.137979	-0.072042	-0.213177	0.115739	0.222682	-0.080155	1.000000	0.151832	-0.391724
thal	0.068001	0.210041	-0.161736	0.062210	0.098803	-0.032019	-0.011981	-0.096439	0.206754	0.210244	-0.104764	0.151832	1.000000	-0.344029
target	-0.225439	-0.280937	0.433798	-0.144931	-0.085239	-0.028046	0.137230	0.421741	-0.436757	-0.430696	0.345877	-0.391724	-0.344029	1.000000

In [28]:

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

Out[28]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
age	1.000000	-0.098447	-0.068653	0.279351	0.213678	0.121308	-0.116211	-0.398522	0.096801	0.210013	-0.168814	0.276326	0.068001	-0.225439
sex	-0.098447	1.000000	-0.049353	-0.056769	-0.197912	0.045032	-0.058196	-0.044020	0.141664	0.096093	-0.030711	0.118261	0.210041	-0.280937
cp	-0.068653	-0.049353	1.000000	0.047608	-0.076904	0.094444	0.044421	0.295762	-0.394280	-0.149230	0.119717	-0.181053	-0.161736	0.433798
trestbps	0.279351	-0.056769	0.047608	1.000000	0.123174	0.177531	-0.114103	-0.046698	0.067616	0.193216	-0.121475	0.101389	0.062210	-0.144931
chol	0.213678	-0.197912	-0.076904	0.123174	1.000000	0.013294	-0.151040	-0.009940	0.067023	0.053952	-0.004038	0.070511	0.098803	-0.085239
fbs	0.121308	0.045032	0.094444	0.177531	0.013294	1.000000	-0.084189	-0.008567	0.025665	0.005747	-0.059894	0.137979	-0.032019	-0.028046
restecg	-0.116211	-0.058196	0.044421	-0.114103	-0.151040	-0.084189	1.000000	0.044123	-0.070733	-0.058770	0.093045	-0.072042	-0.011981	0.137230
thalach	-0.398522	-0.044020	0.295762	-0.046698	-0.009940	-0.008567	0.044123	1.000000	-0.378812	-0.344187	0.386784	-0.213177	-0.096439	0.421741
exang	0.096801	0.141664	-0.394280	0.067616	0.067023	0.025665	-0.070733	-0.378812	1.000000	0.288223	-0.257748	0.115739	0.206754	-0.436757
oldpeak	0.210013	0.096093	-0.149230	0.193216	0.053952	0.005747	-0.058770	-0.344187	0.288223	1.000000	-0.577537	0.222682	0.210244	-0.430696
slope	-0.168814	-0.030711	0.119717	-0.121475	-0.004038	-0.059894	0.093045	0.386784	-0.257748	-0.577537	1.000000	-0.080155	-0.104764	0.345877
ca	0.276326	0.118261	-0.181053	0.101389	0.070511	0.137979	-0.072042	-0.213177	0.115739	0.222682	-0.080155	1.000000	0.151832	-0.391724
thal	0.068001	0.210041	-0.161736	0.062210	0.098803	-0.032019	-0.011981	-0.096439	0.206754	0.210244	-0.104764	0.151832	1.000000	-0.344029
target	-0.225439	-0.280937	0.433798	-0.144931	-0.085239	-0.028046	0.137230	0.421741	-0.436757	-0.430696	0.345877	-0.391724	-0.344029	1.000000

In [29]:

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

Out[29]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
age	1.000000	-0.082272	-0.071577	0.201071	0.135062	0.094595	-0.109349	-0.280009	0.074427	0.193269	-0.147713	0.273255	0.070722	-0.197857
sex	-0.082272	1.000000	-0.057955	-0.044438	-0.124104	0.045032	-0.048085	-0.032817	0.141664	0.086437	-0.024333	0.112199	0.244164	-0.280937
cp	-0.071577	-0.057955	1.000000	0.027548	-0.069899	0.083862	0.060839	0.246160	-0.390708	-0.125081	0.145796	-0.189400	-0.188999	0.430506
trestbps	0.201071	-0.044438	0.027548	1.000000	0.086474	0.127574	-0.105147	-0.027760	0.044419	0.109103	-0.070360	0.070387	0.049028	-0.102064
chol	0.135062	-0.124104	-0.069899	0.086474	1.000000	0.015140	-0.132664	-0.031437	0.075044	0.035176	-0.010039	0.088549	0.066255	-0.099131
fbs	0.094595	0.045032	0.083862	0.127574	0.015140	1.000000	-0.080996	-0.011749	0.025665	0.024342	-0.044546	0.126434	-0.006559	-0.028046
restecg	-0.109349	-0.048085	0.060839	-0.105147	-0.132664	-0.080996	1.000000	0.072481	-0.076913	-0.066262	0.110042	-0.091541	-0.010692	0.147678
thalach	-0.280009	-0.032817	0.246160	-0.027760	-0.031437	-0.011749	0.072481	1.000000	-0.329965	-0.306843	0.349702	-0.198407	-0.130239	0.352609
exang	0.074427	0.141664	-0.390708	0.044419	0.075044	0.025665	-0.076913	-0.329965	1.000000	0.255042	-0.267046	0.152294	0.240555	-0.436757
oldpeak	0.193269	0.086437	-0.125081	0.109103	0.035176	0.024342	-0.066262	-0.306843	0.255042	1.000000	-0.508539	0.183166	0.213656	-0.361731
slope	-0.147713	-0.024333	0.145796	-0.070360	-0.010039	-0.044546	0.110042	0.349702	-0.267046	-0.508539	1.000000	-0.092013	-0.147382	0.361406
ca	0.273255	0.112199	-0.189400	0.070387	0.088549	0.126434	-0.091541	-0.198407	0.152294	0.183166	-0.092013	1.000000	0.173361	-0.430124
thal	0.070722	0.244164	-0.188999	0.049028	0.066255	-0.006559	-0.010692	-0.130239	0.240555	0.213656	-0.147382	0.173361	1.000000	-0.392595
target	-0.197857	-0.280937	0.430506	-0.102064	-0.099131	-0.028046	0.147678	0.352609	-0.436757	-0.361731	0.361406	-0.430124	-0.392595	1.000000

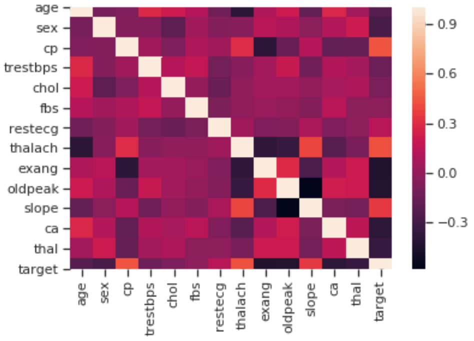
In [27]:
```
data.corr(method='spearman')
```

Out[27]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
age	1.000000	-0.099131	-0.087494	0.285617	0.195786	0.113978	-0.132769	-0.398052	0.089679	0.268291	-0.184048	0.340955	0.087254	-0.238400
sex	-0.099131	1.000000	-0.062041	-0.052941	-0.151342	0.045032	-0.048389	-0.039868	0.141664	0.100715	-0.025010	0.119368	0.250821	-0.280937
cp	-0.087494	-0.062041	1.000000	0.035413	-0.091721	0.089775	0.065640	0.324013	-0.418256	-0.161449	0.159478	-0.216006	-0.207840	0.460860
trestbps	0.285617	-0.052941	0.035413	1.000000	0.126562	0.151984	-0.125841	-0.040407	0.052918	0.154267	-0.086570	0.090140	0.059673	-0.121593
chol	0.195786	-0.151342	-0.091721	0.126562	1.000000	0.018463	-0.161933	-0.046766	0.091514	0.045260	-0.012551	0.111981	0.083628	-0.120888
fbs	0.113978	0.045032	0.089775	0.151984	0.018463	1.000000	-0.081508	-0.014273	0.025665	0.028363	-0.045786	0.134513	-0.006737	-0.028046
restecg	-0.132769	-0.048389	0.065640	-0.125841	-0.161933	-0.081508	1.000000	0.087863	-0.077399	-0.077372	0.113661	-0.097862	-0.010982	0.148612
thalach	-0.398052	-0.039868	0.324013	-0.040407	-0.046766	-0.014273	0.087863	1.000000	-0.400860	-0.433241	0.436968	-0.257347	-0.160581	0.428370
exang	0.089679	0.141664	-0.418256	0.052918	0.091514	0.025665	-0.077399	-0.400860	1.000000	0.297173	-0.274475	0.162025	0.247113	-0.436757
oldpeak	0.268291	0.100715	-0.161449	0.154267	0.045260	0.028363	-0.077372	-0.433241	0.297173	1.000000	-0.594847	0.224895	0.255026	-0.421487
slope	-0.184048	-0.025010	0.159478	-0.086570	-0.012551	-0.045786	0.113661	0.436968	-0.274475	-0.594847	1.000000	-0.099901	-0.154886	0.371460
ca	0.340955	0.119368	-0.216006	0.090140	0.111981	0.134513	-0.097862	-0.257347	0.162025	0.224895	-0.099901	1.000000	0.189103	-0.457607
thal	0.087254	0.250821	-0.207840	0.059673	0.083628	-0.006737	-0.010982	-0.160581	0.247113	0.255026	-0.154886	0.189103	1.000000	-0.403299
target	-0.238400	-0.280937	0.460860	-0.121593	-0.120888	-0.028046	0.148612	0.428370	-0.436757	-0.421487	0.371460	-0.457607	-0.403299	1.000000

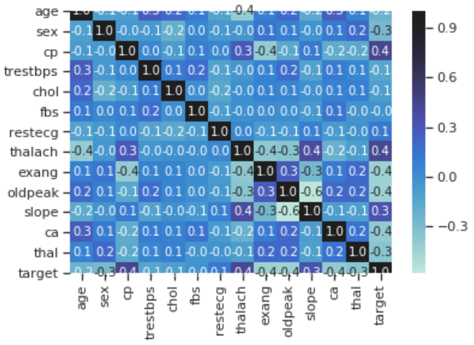
In [49]:
```
sns.heatmap(data.corr())
```

Out[49]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7ffba50be090>
```



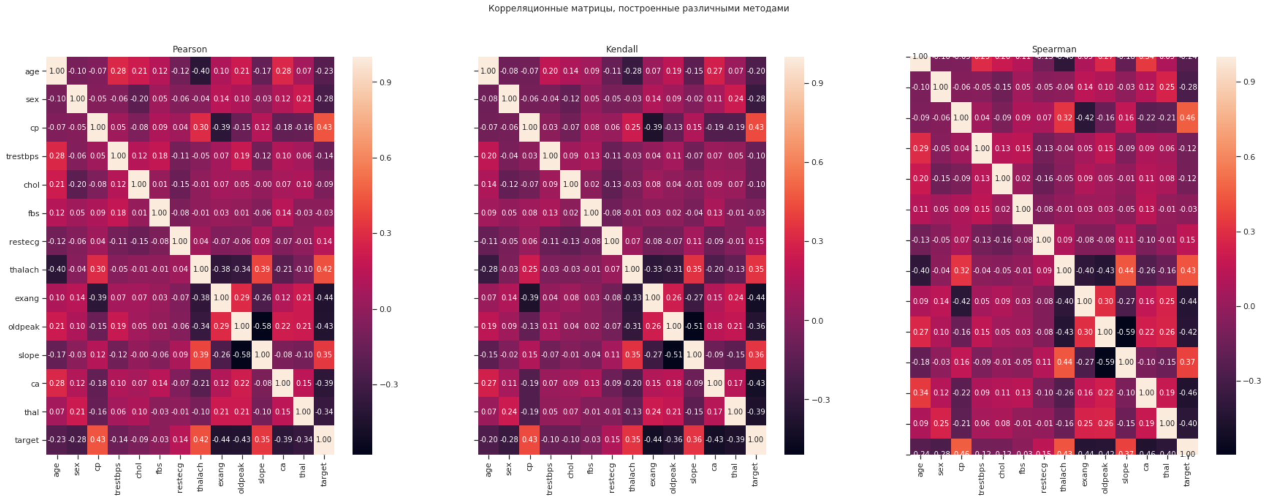
In [59]:
```
sns.heatmap(data.corr(), annot=True, fmt='.1f',center=1)
```

Out[59]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7ffb9e4f7c90>
```



In [38]:

```
fig, ax = plt.subplots(1, 3, sharex='col', sharey='row', figsize=(30,10))
sns.heatmap(data.corr(method='pearson'), ax=ax[0], annot=True, fmt='.2f')
sns.heatmap(data.corr(method='kendall'), ax=ax[1], annot=True, fmt='.2f')
sns.heatmap(data.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')
```


Корреляционные матрицы, построенные различными методами

In [ ]: