Не удается связаться с сервисом reCAPTCHA. Проверьте подключение к Интернету и перезагрузите страницу.

```
import pandas as pd
import numpy as np

import seaborn as sns
import matplotlib.pyplot as plt

%matplotlib inline


df = pd.read_csv('heart.csv')


columns_names = {'trtbps': 'blood_pressure', 'fbs': 'blood_sugar', 'thalachh': 'max


df = df.rename(columns=columns_names)


df.head()
```

	age	sex	cp	blood_pressure	chol	blood_sugar	restecg	max_hr	exng	oldp
0	63	1	3	145	233	1	0	150	0	
1	37	1	2	130	250	0	1	187	0	
2	41	0	1	130	204	0	0	172	0	
3	56	1	1	120	236	0	1	178	0	
4	57	0	0	120	354	0	1	163	1	

```
df['output'].sum(), len(df)
```

```
(165, 303)
```

Количество измерений в каждом классе примерно одинаковое

```
df.describe()
```

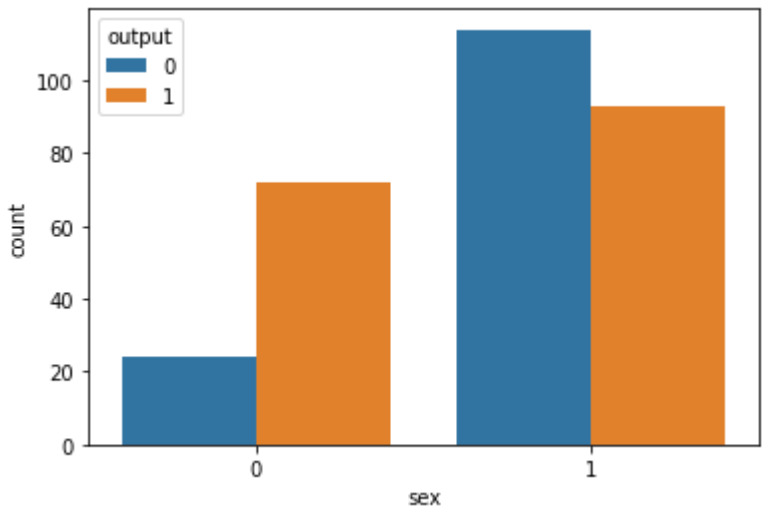
	age	sex	cp	blood_pressure	chol	blood_sugar
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000
mean	54.366337	0.683168	0.966997	131.623762	246.264026	0.148515
std	9.082101	0.466011	1.032052	17.538143	51.830751	0.356198
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000
25%	47.500000	0.000000	0.000000	120.000000	211.000000	0.000000

```
df.groupby(['caa']).count()['output']
```

```
caa
0    175
1     65
2     38
3     20
4      5
Name: output, dtype: int64
```

```
# sns.kdeplot(data=df, x='age', hue='sex')
sns.countplot(data=df, x='sex', hue='output')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f51349c7fa0>
```



На этой гистограмме видно, что принадлежность к полу 0(в датасете не описано, что это за пол) несет большие риски сердечного приступа

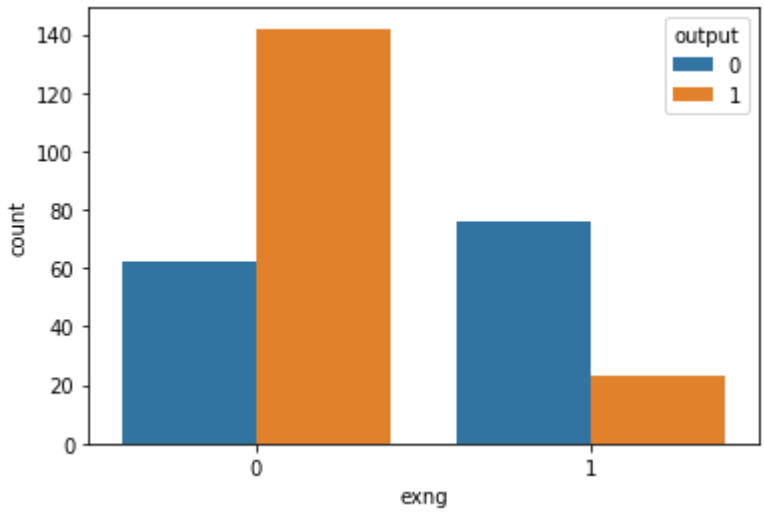
```
sns.countplot(data=df, x='blood_sugar', hue='output')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f51344eaca0>
```

При этом количество сахара в крови не очень принципиально как отдельный признак

```
sns.countplot(data=df, x='exng', hue='output')
```

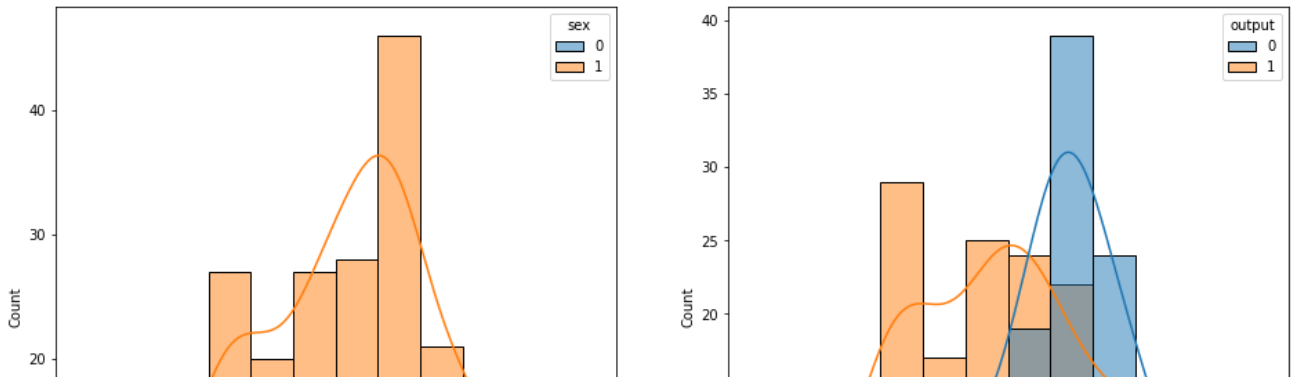
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5134a55dc0>
```



Также видно, что риск сердечного приступа выше у тех, у кого часто стенакордия вызвана не физическими нагрузками

```
fig, ax = plt.subplots(1, 2, figsize=(16, 8))


sns.histplot(data=df, x='age', hue='sex', kde=True, ax=ax[0])
sns.histplot(data=df, x='age', hue='output', kde=True, ax=ax[1])
```

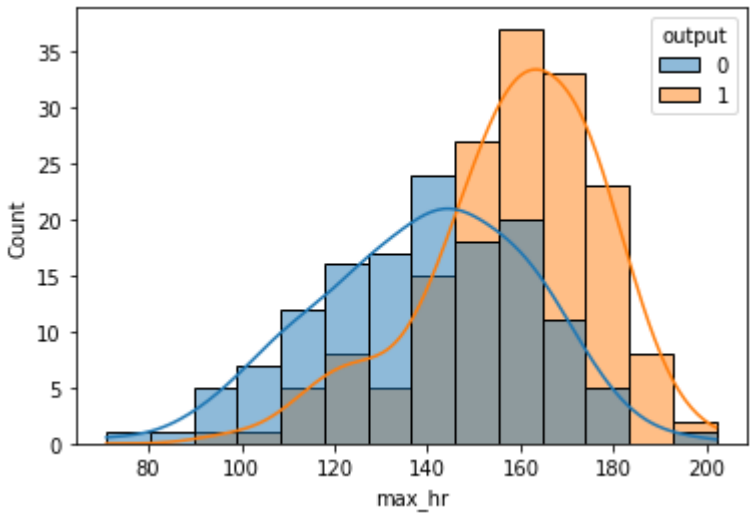
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f512ebfe7f0>
```



По этим распределениям можно видеть, возрастная группа преимущественно состоит из людей за 50(в общем то не удивительно). Также стоит отметить дизбаланст классов по полу. Людей пола 0 гораздо меньше, чем пола 1. В правом распределении видно различия в пиках, которое показывает, что довольно большая группа риска это медианная возрастная группа, хотя интуитивно казалось, если человек старее, то и риск больше, соответственно.

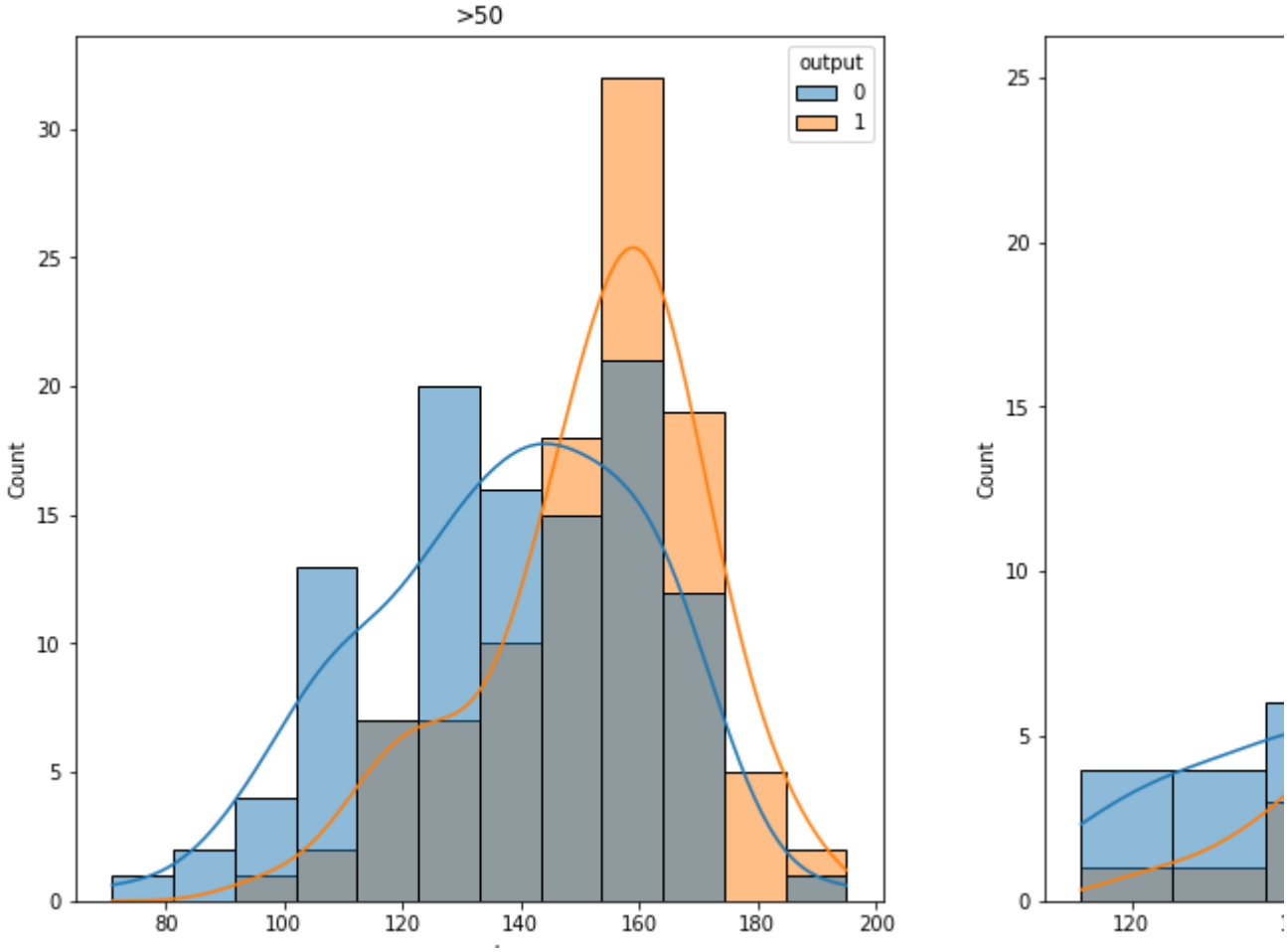
```
sns.histplot(data=df, x='max_hr', hue='output', kde=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f51344d00a0>
```



Здесь видно, что чем выше максимальный пульс тем риск сердечного приступа выше, что опять же контринтуитивно. Интересно посмотреть, какое распределение у "старших" и у "младших"

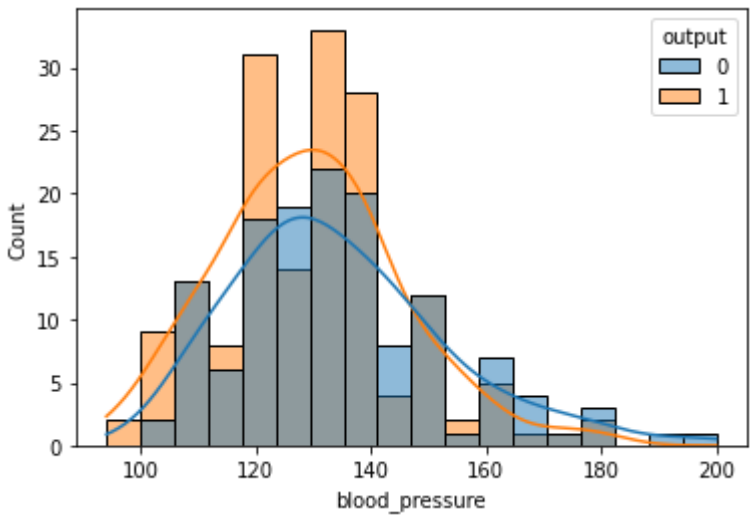
```
fig, ax = plt.subplots(1, 2, figsize=(16, 8))

sns.histplot(data=df[df['age'] >= 50], x='max_hr', hue='output', kde=True, ax=ax[0]
sns.histplot(data=df[df['age'] < 50], x='max_hr', hue='output', kde=True, ax=ax[1])
ax[0].set_title('>50')
ax[1].set_title('<50')
plt.show()
```



Интересно, что в двух возрастных категориях максимальный пульс должен быть не особо высоким, чтобы минимизировать риски сердечного приступа

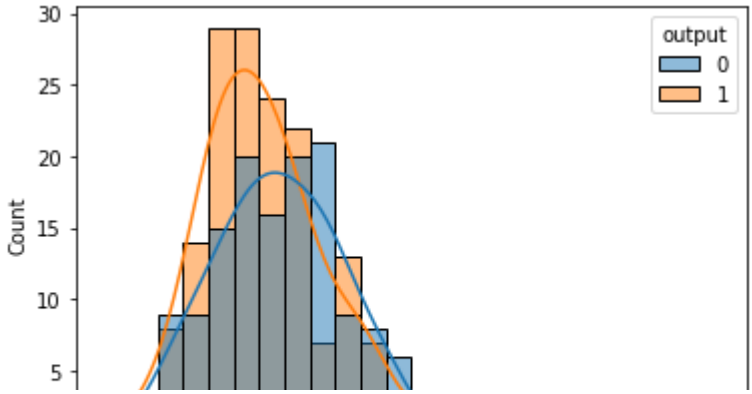
```
sns.histplot(data=df, x='blood_pressure', hue='output', kde=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5136aa72e0>
```



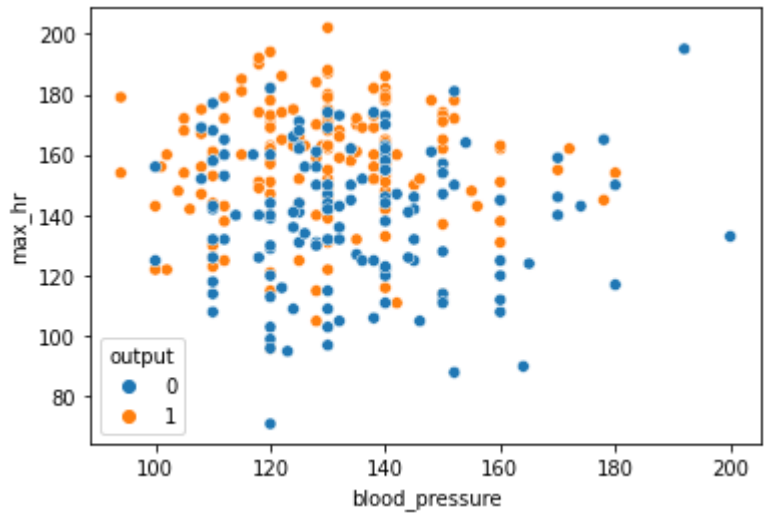

```
sns.histplot(data=df, x='chol', hue='output', kde=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5134317a90>
```




```
sns.scatterplot(x=df['blood_pressure'], y=df['max_hr'], hue=df['output'])
```

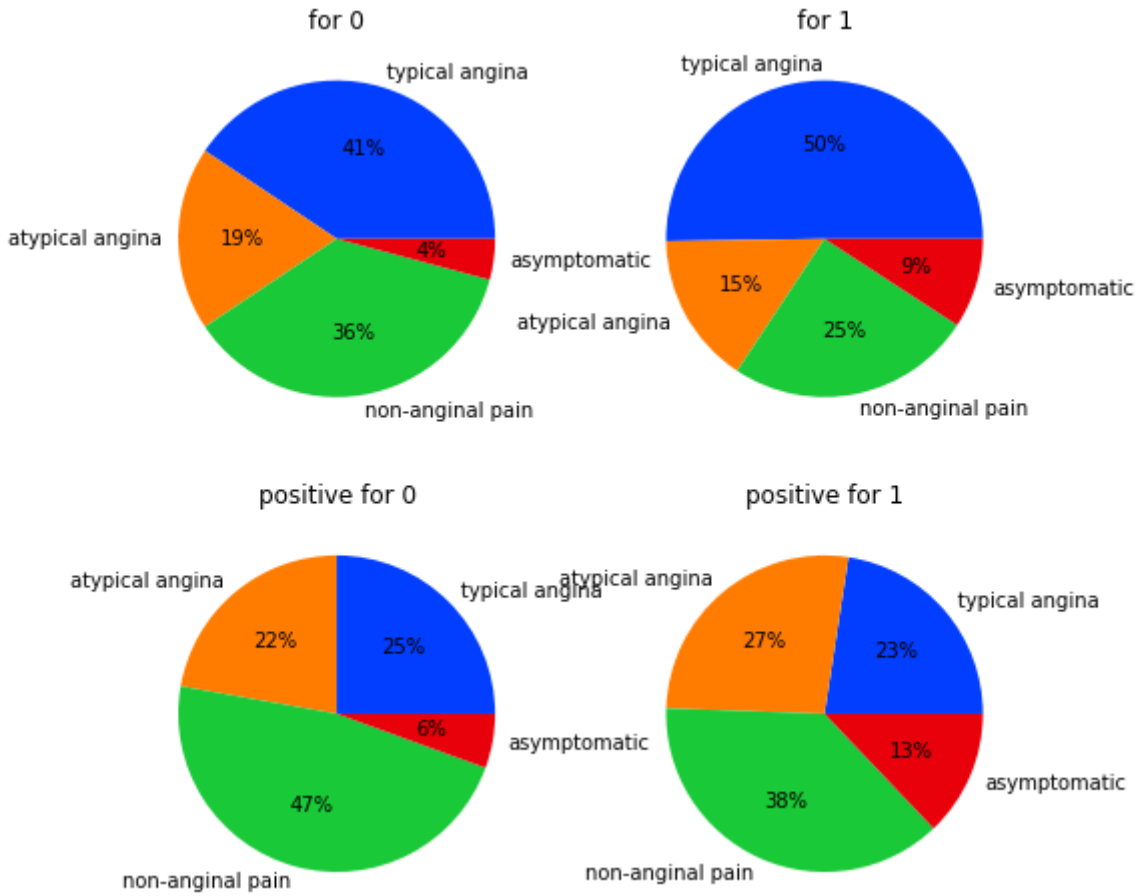
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f51341f8760>
```



Посмотрим теперь на категориальные признаки

```
colors = sns.color_palette('bright')
filtered_1 = df[(df['sex'] == 0)].groupby(['cp']).count()['age']
filtered_2 = df[(df['sex'] == 1)].groupby(['cp']).count()['age']

filtered_3 = df[(df['sex'] == 0) & (df['output'] == 1)].groupby(['cp']).count()['ag
filtered_4 = df[(df['sex'] == 1) & (df['output'] == 1)].groupby(['cp']).count()['ag
labels = ['typical angina', 'atypical angina', 'non-anginal pain', 'asymptomatic']

fig, ax = plt.subplots(2, 2, figsize=(8, 8))

ax[0][0].pie(filtered_1.values, labels=labels, colors=colors, autopct = '%0.0f%%')
ax[0][0].set_title('for 0')



ax[0][1].pie(filtered_2.values, labels=labels, colors=colors, autopct = '%0.0f%%')
ax[0][1].set_title('for 1')


ax[1][0].pie(filtered_3.values, labels=labels, colors=colors, autopct = '%0.0f%%')
ax[1][0].set_title('positive for 0')
```

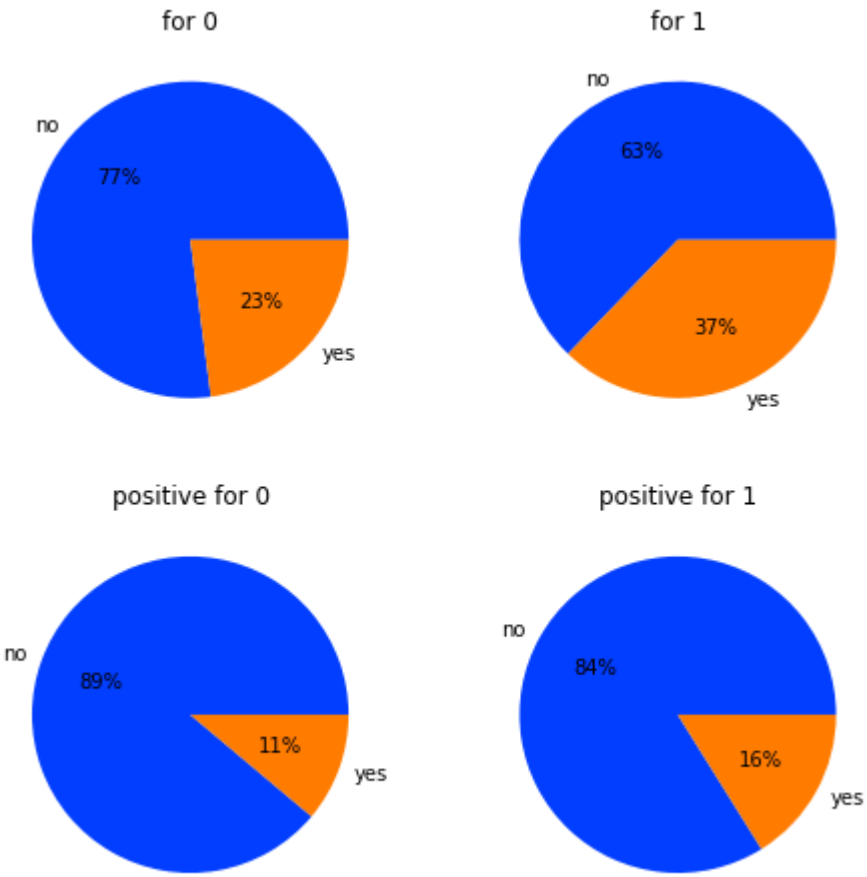
```
ax[1][1].pie(filtered_4.values, labels=labels, colors=colors, autopct = '%0.0f%%')
ax[1][1].set_title('positive for 1')

plt.show()
```



На этих круговых диаграмах показаны доли типов болей в грудной клетке у разных полов. Можно видеть, что у тех людей, у которых высокий риск сердечного приступа преобладает неангинальная боль, в то время как типичная стенокардия уменьшает свое влияние.

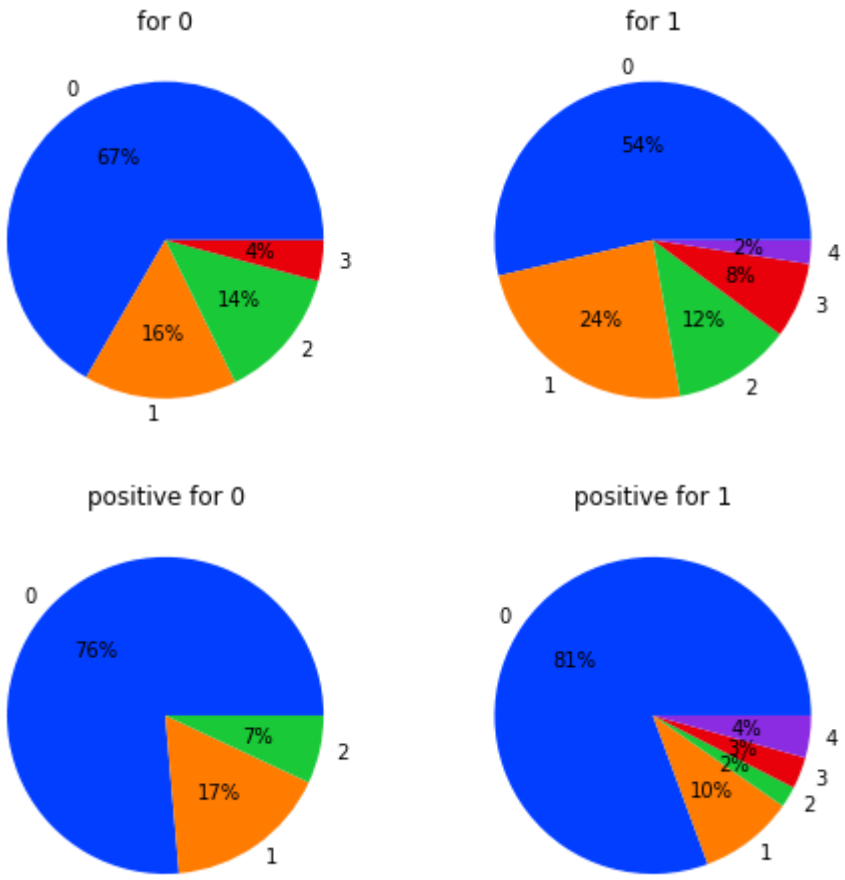
```
colors = sns.color_palette('bright')
filtered_1 = df[(df['sex'] == 0)].groupby(['exng']).count()['age']
filtered_2 = df[(df['sex'] == 1)].groupby(['exng']).count()['age']
filtered_3 = df[(df['sex'] == 0) & (df['output'] == 1)].groupby(['exng']).count()[
filtered_4 = df[(df['sex'] == 1) & (df['output'] == 1)].groupby(['exng']).count()[

labels = ['no', 'yes']
labels_for_last = labels[:-1]

fig, ax = plt.subplots(2, 2, figsize=(8, 8))

ax[0][0].pie(filtered_1.values, labels=labels, colors=colors, autopct = '%0.0f%%')
ax[0][0].set_title('for 0')
```

```
ax[0][1].pie(filtered_2.values, labels=labels, colors=colors, autopct = '%0.0f%%')
ax[0][1].set_title('for 1')


ax[1][0].pie(filtered_3.values, labels=labels, colors=colors, autopct = '%0.0f%%')
ax[1][0].set_title('positive for 0')



ax[1][1].pie(filtered_4.values, labels=labels, colors=colors, autopct = '%0.0f%%')
ax[1][1].set_title('positive for 1')

plt.show()
```




```
colors = sns.color_palette('bright')
filtered_1 = df[(df['sex'] == 0)].groupby(['caa']).count()['age']
filtered_2 = df[(df['sex'] == 1)].groupby(['caa']).count()['age']
filtered_3 = df[(df['sex'] == 0) & (df['output'] == 1)].groupby(['caa']).count()['a
filtered_4 = df[(df['sex'] == 1) & (df['output'] == 1)].groupby(['caa']).count()['a

labels = ['0', '1', '2', '3', '4']
labels_1 = ['0', '1', '2', '3']
labels_3 = ['0', '1', '2']

fig, ax = plt.subplots(2, 2, figsize=(8, 8))
```

```
ax[0][0].pie(filtered_1.values, labels=labels_1, colors=colors, autopct = '%0.0f%%
ax[0][0].set_title('for 0')



ax[0][1].pie(filtered_2.values, labels=labels, colors=colors, autopct = '%0.0f%%')
ax[0][1].set_title('for 1')


ax[1][0].pie(filtered_3.values, labels=labels_3, colors=colors, autopct = '%0.0f%%
ax[1][0].set_title('positive for 0')



ax[1][1].pie(filtered_4.values, labels=labels, colors=colors, autopct = '%0.0f%%')
ax[1][1].set_title('positive for 1')

plt.show()
```



Посмотрим еще на непрерывные признаки

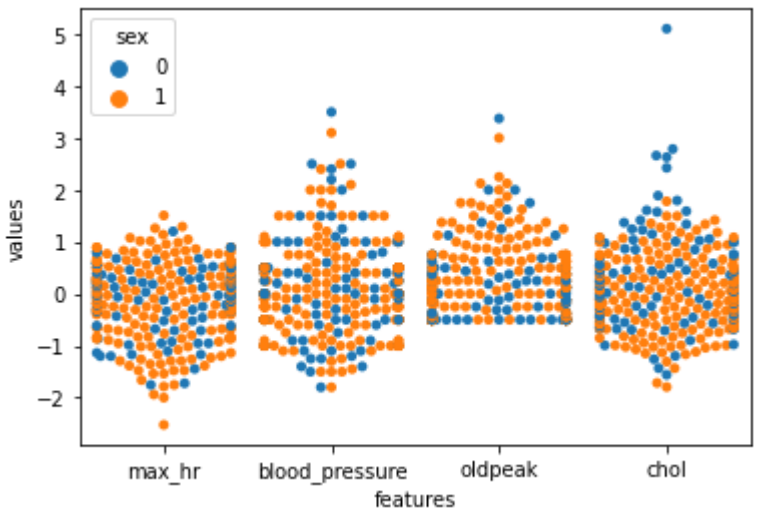
```
melt_df = pd.melt(df[['max_hr', 'blood_pressure', 'oldpeak', 'chol', 'sex']],
                  var_name='features', value_name='values', id_vars='sex')


df_cont = df[['max_hr', 'blood_pressure', 'oldpeak', 'chol', 'sex']].copy()


from sklearn.preprocessing import RobustScaler
```

```
scaler = RobustScaler()
cont_norm_matrix = scaler.fit_transform(df_cont.drop(columns='sex'))
df_cont_norm = pd.DataFrame(cont_norm_matrix, columns=['max_hr', 'blood_pressure',
df_cont_norm = pd.concat([df_cont_norm, df_cont.loc[:, 'sex']], axis=1)
```

```
melt_df = pd.melt(df_cont_norm,
                  var_name='features', value_name='values', id_vars='sex')
```

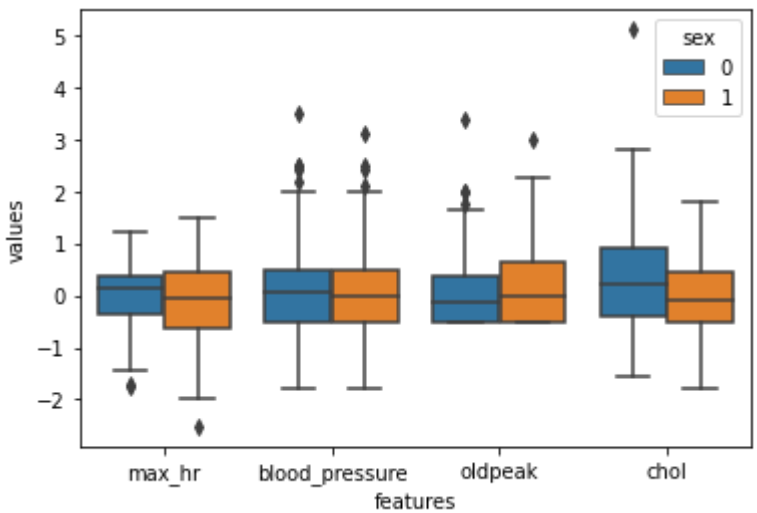
```
sns.swarmplot(data=melt_df, x='features', y='values', hue='sex')
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/categorical.py:1296: UserWarni
  warnings.warn(msg, UserWarning)
/usr/local/lib/python3.8/dist-packages/seaborn/categorical.py:1296: UserWarni
  warnings.warn(msg, UserWarning)
/usr/local/lib/python3.8/dist-packages/seaborn/categorical.py:1296: UserWarni
  warnings.warn(msg, UserWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f512c110130>
```




```
sns.boxplot(data=melt_df, x='features', y='values', hue='sex')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f512bfe83d0>
```



В принципе можно видеть, что зависимости от пола в явновыраженном виде нет. Можно сделать вывод из всего вышеперечисленного, что датасет просто несбалансированный по полу.

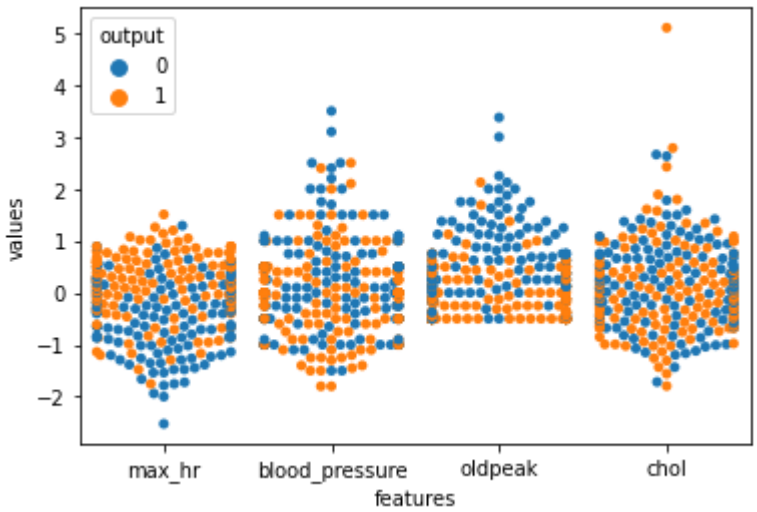
```
df_cont = df[['max_hr', 'blood_pressure', 'oldpeak', 'chol', 'output']].copy()
```

```
scaler = RobustScaler()
cont_norm_matrix = scaler.fit_transform(df_cont.drop(columns='output'))
df_cont_norm = pd.DataFrame(cont_norm_matrix, columns=['max_hr', 'blood_pressure',
df_cont_norm = pd.concat([df_cont_norm, df_cont.loc[:, 'output']], axis=1)
```

```
melt_df = pd.melt(df_cont_norm,
                  var_name='features', value_name='values', id_vars='output')
```

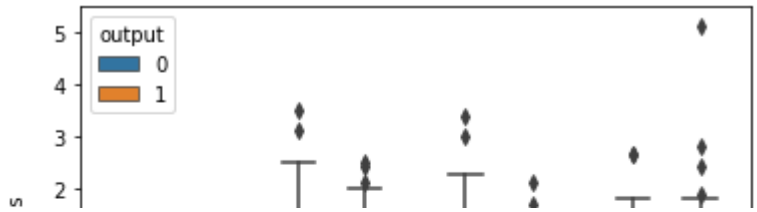
```
sns.swarmplot(data=melt_df, x='features', y='values', hue='output')
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/categorical.py:1296: UserWarni
  warnings.warn(msg, UserWarning)
/usr/local/lib/python3.8/dist-packages/seaborn/categorical.py:1296: UserWarni
  warnings.warn(msg, UserWarning)
/usr/local/lib/python3.8/dist-packages/seaborn/categorical.py:1296: UserWarni
  warnings.warn(msg, UserWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f512bec5a30>
```



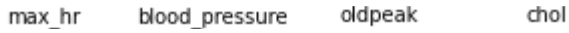

```
sns.boxplot(data=melt_df, x='features', y='values', hue='output')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f512be177f0>
```



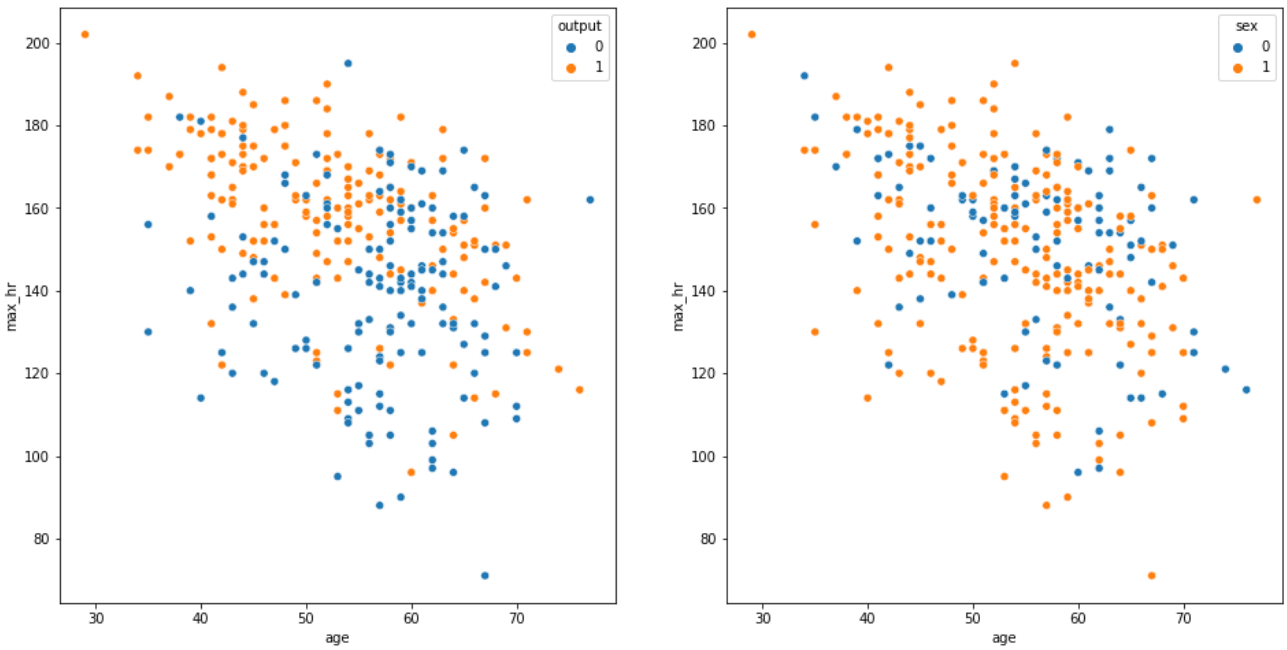
При этом можно видеть, что хорошо разделяют такие признаки как максимальный пульс и oldpeak.

Посмотрим попарно на непрерывные признаки



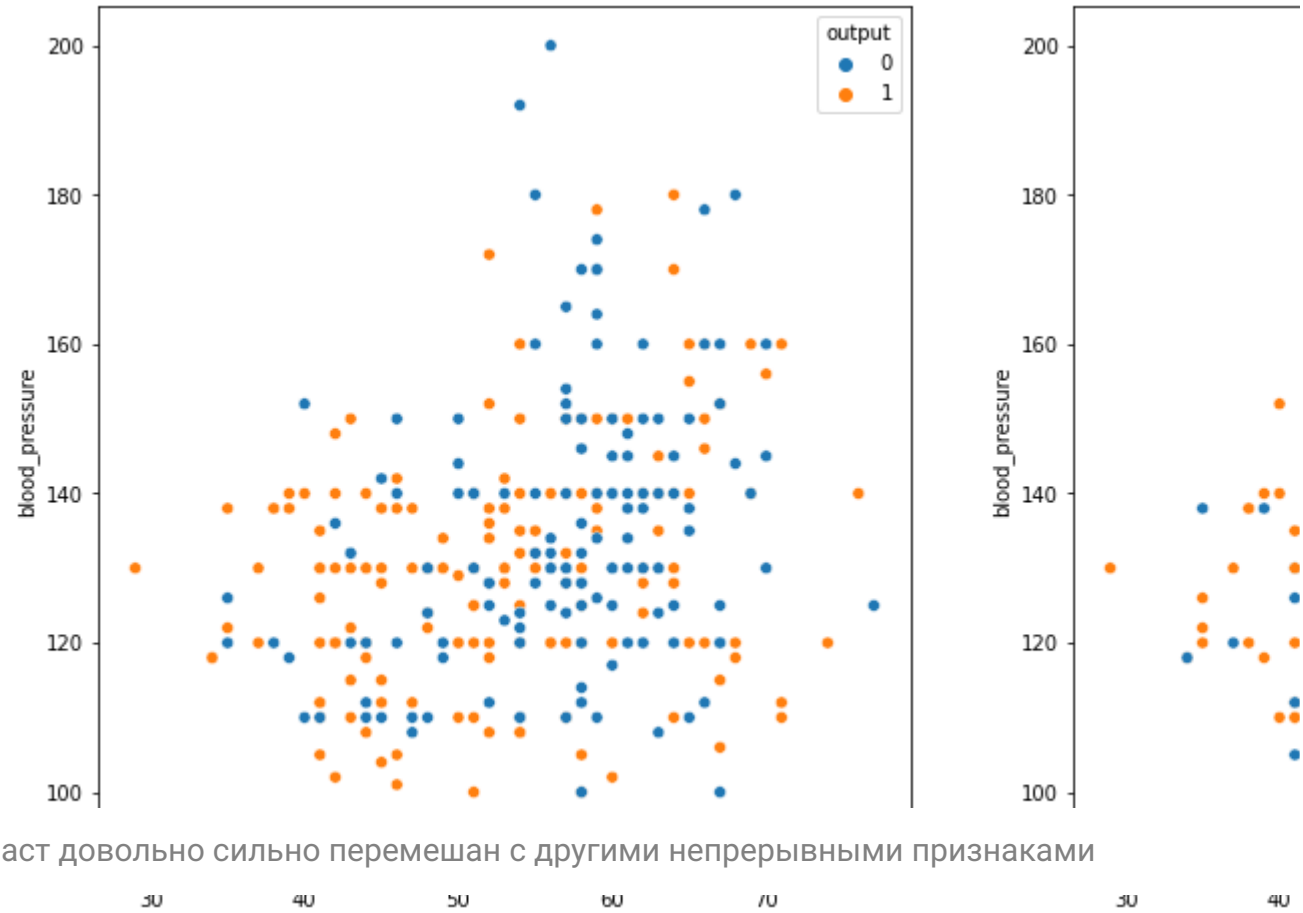

```
fig, ax = plt.subplots(1, 2, figsize=(16, 8))
sns.scatterplot(x=df['age'], y=df['max_hr'], hue=df['output'], ax=ax[0])
sns.scatterplot(x=df['age'], y=df['max_hr'], hue=df['sex'], ax=ax[1])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f513415de50>
```




```
fig, ax = plt.subplots(1, 2, figsize=(16, 8))
sns.scatterplot(x=df['age'], y=df['blood_pressure'], hue=df['output'], ax=ax[0])
sns.scatterplot(x=df['age'], y=df['blood_pressure'], hue=df['sex'], ax=ax[1])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f512ee813a0>
```



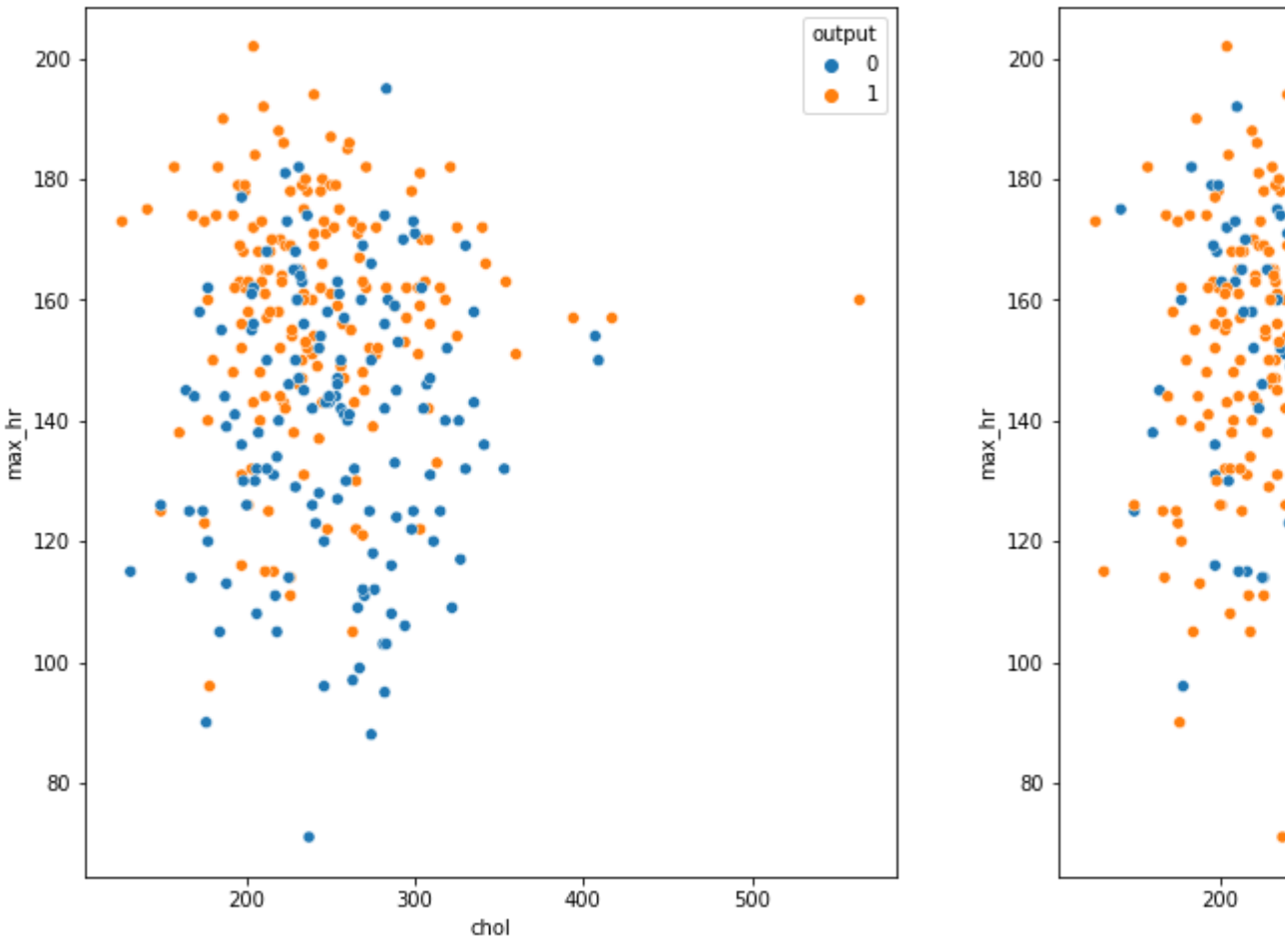
Возраст довольно сильно перемешан с другими непрерывными признаками

```
fig, ax = plt.subplots(1, 2, figsize=(16, 8))
sns.scatterplot(x=df['chol'], y=df['blood_pressure'], hue=df['output'], ax=ax[0])
sns.scatterplot(x=df['chol'], y=df['blood_pressure'], hue=df['sex'], ax=ax[1])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5133f52100>
```

```
fig, ax = plt.subplots(1, 2, figsize=(16, 8))
sns.scatterplot(x=df['chol'], y=df['max_hr'], hue=df['output'], ax=ax[0])
sns.scatterplot(x=df['chol'], y=df['max_hr'], hue=df['sex'], ax=ax[1])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f512ed43d00>
```



В принципе количество холестерина и максимальный пульс можно разделить прямой

Попробуем теперь обучить модельки по выбранным признакам

```
cat_features = ['sex', 'slp', 'restecg','caa', 'exng', 'cp', 'thall']
cont_features = ['age', 'blood_pressure', 'chol', 'max_hr', 'oldpeak']
target = 'output'
```

```
from sklearn.utils import shuffle
df = shuffle(df)
```

```
filtered_df = df[cont_features + cat_features + [target]]
```

```
oh_df = pd.get_dummies(filtered_df, columns=cat_features, drop_first = True)
```

```
oh_df[cont_features] = scaler.fit_transform(oh_df[cont_features])

X_data = oh_df.drop(columns='output').values
y_data = df['output']


from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, roc_auc_score, roc_curve, f1_score, pre


X_train, X_test, y_train, y_test = train_test_split(X_data, y_data, train_size=0.8,


# !pip install catboost


from catboost import CatBoostClassifier


catboost_model = CatBoostClassifier(iterations=1000,
                            depth=5,
                            learning_rate=1e-4,
                            eval_metric='AUC',
                            loss_function='Logloss',
                            verbose=False)


catboost_model.fit(X_train, y_train)
```

```
<catboost.core.CatBoostClassifier at 0x7f5128e34d30>
```

```
preds = catboost_model.predict(X_test)


accuracy_score(y_test, preds)
```

```
0.8360655737704918
```

```
from sklearn.svm import SVC
from sklearn.model_selection import GridSearchCV

# ['linear', 'poly', 'rbf', 'sigmoid', 'precomputed']
grid = {'C': np.linspace(0.0001, 10, 10), 'kernel': ['rbf', 'linear'],
        'gamma': np.linspace(1e-5, 10, 10)}

svm = SVC(random_state=42)
search = GridSearchCV(svm, grid, cv=3, scoring='accuracy', refit=True, n_jobs=2)


search.fit(X_train, y_train)
```

```
GridSearchCV(cv=3, estimator=SVC(random_state=42), n_jobs=2,
             param_grid={'C': array([1.0000e-04, 1.1112e+00, 2.2223e+00,
```

```
       3.3334e+00, 4.4445e+00,
              5.5556e+00, 6.6667e+00, 7.7778e+00, 8.8889e+00, 1.0000e+01]),
                                  'gamma': array([1.00000e-05, 1.11112e+00,
       2.22223e+00, 3.33334e+00, 4.44445e+00,
              5.55556e+00, 6.66667e+00, 7.77778e+00, 8.88889e+00, 1.00000e+01]),
                                  'kernel': ['rbf', 'linear']},
                     scoring='accuracy')
```

```
search.best_params_
```

```
{'C': 4.4445, 'gamma': 1e-05, 'kernel': 'linear'}
```

```
svm = SVC(C=5.5, kernel='linear', gamma=1e-05, random_state=42)
```

```
svm.fit(X_train, y_train)
```

```
SVC(C=5.5, gamma=1e-05, kernel='linear', random_state=42)
```

```
preds = svm.predict(X_test)
```

```
accuracy_score(y_test, preds)
```

```
0.9344262295081968
```

✓ 0 сек. выполнено в 15:22

Не удается связаться с сервисом reCAPTCHA. Проверьте подключение к Интернету и перезагрузите страницу.