```
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
```

```
data=pd.read_excel('/content/data.xlsx')
```

```
data.head()
```

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

```
data.isnull().sum()
```

```
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
dtype: int64
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       303 non-null    int64
 1   sex       303 non-null    int64
 2   cp        303 non-null    int64
 3   trestbps  303 non-null    int64
 4   chol      303 non-null    int64
 5   fbs       303 non-null    int64
 6   restecg   303 non-null    int64
```

```
 7   thalach   303 non-null    int64
 8   exang     303 non-null    int64
 9   oldpeak   303 non-null    float64
 10  slope     303 non-null    int64
 11  ca        303 non-null    int64
 12  thal      303 non-null    int64
 13  target    303 non-null    int64
dtypes: float64(1), int64(13)
memory usage: 33.3 KB
```

```
data.nunique()
```

```
age          41
sex           2
cp            4
trestbps     49
chol        152
fbs           2
restecg       3
thalach      91
exang         2
oldpeak      40
slope         3
ca            5
thal          4
target        2
dtype: int64
```

```
data['sex']=data['sex'].replace({1 : 'male',0 : 'female'})
```

```
data.head()
```

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

```
data['thal'].nunique()
```

```
4
```

```
data['thal'].value_counts()
```

```
thal
2    166
3    117
1     18
```

```
0      2
Name: count, dtype: int64
```

```
data.describe()
```

	age	cp	trestbps	chol	fbs	restecg	thalac
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.00000
mean	54.366337	0.966997	131.623762	246.264026	0.148515	0.528053	149.64686
std	9.082101	1.032052	17.538143	51.830751	0.356198	0.525860	22.90516
min	29.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.00000
25%	47.500000	0.000000	120.000000	211.000000	0.000000	0.000000	133.50000
50%	55.000000	1.000000	130.000000	240.000000	0.000000	1.000000	153.00000
75%	61.000000	2.000000	140.000000	274.500000	0.000000	1.000000	166.00000
max	77.000000	3.000000	200.000000	564.000000	1.000000	2.000000	202.00000

```
data1=pd.read_excel('/content/data.xlsx')
```

```
data1.agg('mode')
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca
0	58.0	1.0	0.0	120.0	197	0.0	1.0	162.0	0.0	0.0	2.0	0.0
1	NaN	NaN	NaN	NaN	204	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	NaN	NaN	NaN	NaN	234	NaN	NaN	NaN	NaN	NaN	NaN	NaN

```
data1.agg('median')
```

```
age          55.0
sex           1.0
cp            1.0
trestbps    130.0
chol        240.0
fbs           0.0
restecg       1.0
thalach     153.0
exang         0.0
oldpeak       0.8
slope         1.0
ca            0.0
thal          2.0
target        1.0
dtype: float64
```

```
data1.agg('var')
```

```
age           82.484558
sex            0.217166
cp             1.065132
trestbps     307.586453
chol        2686.426748
fbs            0.126877
restecg        0.276528
thalach      524.646406
exang          0.220707
oldpeak        1.348095
slope          0.379735
ca             1.045724
thal           0.374883
target         0.248836
dtype: float64
```

```
data1.agg('skew')
```

```
age        -0.202463
sex        -0.791335
cp          0.484732
trestbps    0.713768
chol        1.143401
fbs         1.986652
restecg     0.162522
thalach    -0.537410
exang       0.742532
oldpeak     1.269720
slope      -0.508316
ca          1.310422
thal       -0.476722
target     -0.179821
dtype: float64
```

```
# prompt: identify the data variables which might be categorical in nature

categorical_variables = data.select_dtypes(include=['object']).columns.tolist()
print("Categorical variables:", categorical_variables)
```

```
Categorical variables: ['sex']
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       303 non-null    int64
 1   sex       303 non-null    object
 2   cp        303 non-null    int64
 3   trestbps  303 non-null    int64
 4   chol      303 non-null    int64
```

```
 5   fbs       303 non-null    int64
 6   restecg   303 non-null    int64
 7   thalach   303 non-null    int64
 8   exang     303 non-null    int64
 9   oldpeak   303 non-null    float64
 10  slope     303 non-null    int64
 11  ca        303 non-null    int64
 12  thal      303 non-null    int64
 13  target    303 non-null    int64
dtypes: float64(1), int64(12), object(1)
memory usage: 33.3+ KB
```

```
data['thal'].value_counts()
```

```
thal
2    166
3    117
1     18
0      2
Name: count, dtype: int64
```

```
data['thal']=data['thal'].replace({0 : np.NAN,1 : 'normal',2 : 'fixed defect',3 : 'revers
```

```
data['thal'].value_counts()
```

```
thal
fixed defect         166
reversible defect    117
normal                18
Name: count, dtype: int64
```

```
data['thal']=data['thal'].astype('object')
```

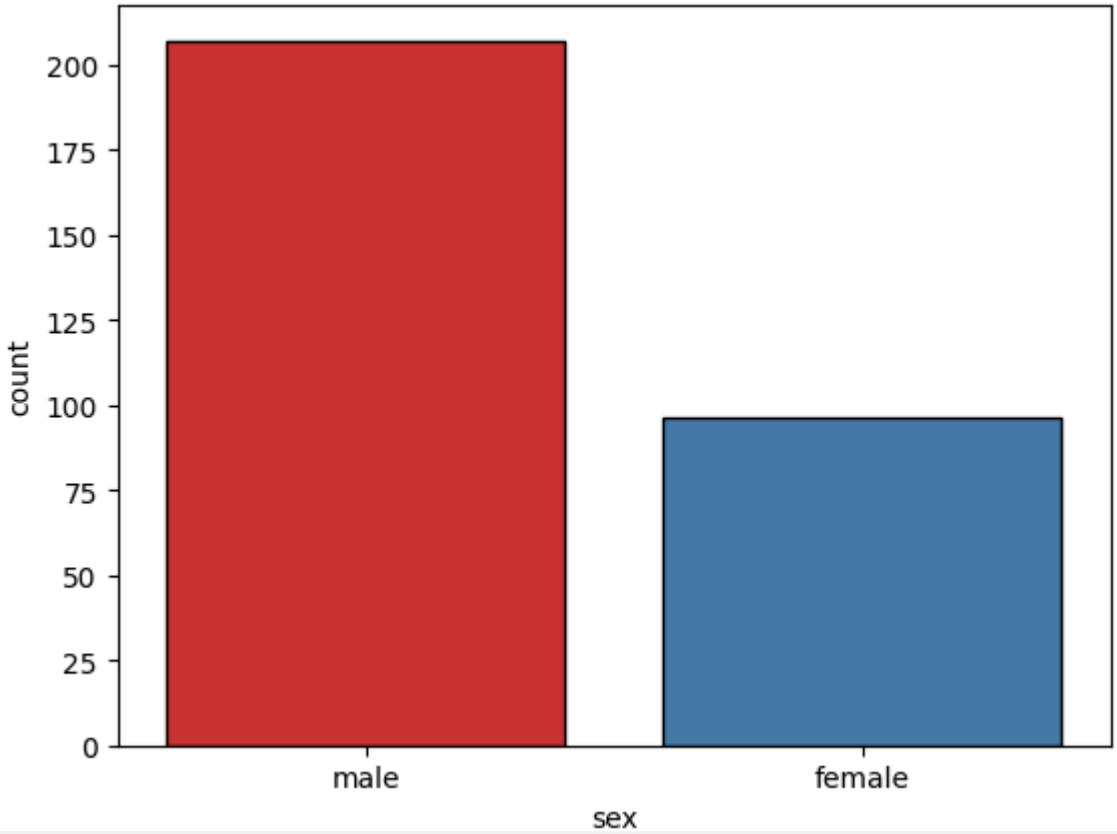
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       303 non-null    int64
 1   sex       303 non-null    object
 2   cp        303 non-null    int64
 3   trestbps  303 non-null    int64
 4   chol      303 non-null    int64
 5   fbs       303 non-null    int64
 6   restecg   303 non-null    int64
 7   thalach   303 non-null    int64
 8   exang     303 non-null    int64
 9   oldpeak   303 non-null    float64
 10  slope     303 non-null    int64
 11  ca        303 non-null    int64
 12  thal      301 non-null    object
 13  target    303 non-null    int64
dtypes: float64(1), int64(11), object(2)
```

```
memory usage: 33.3+ KB
```

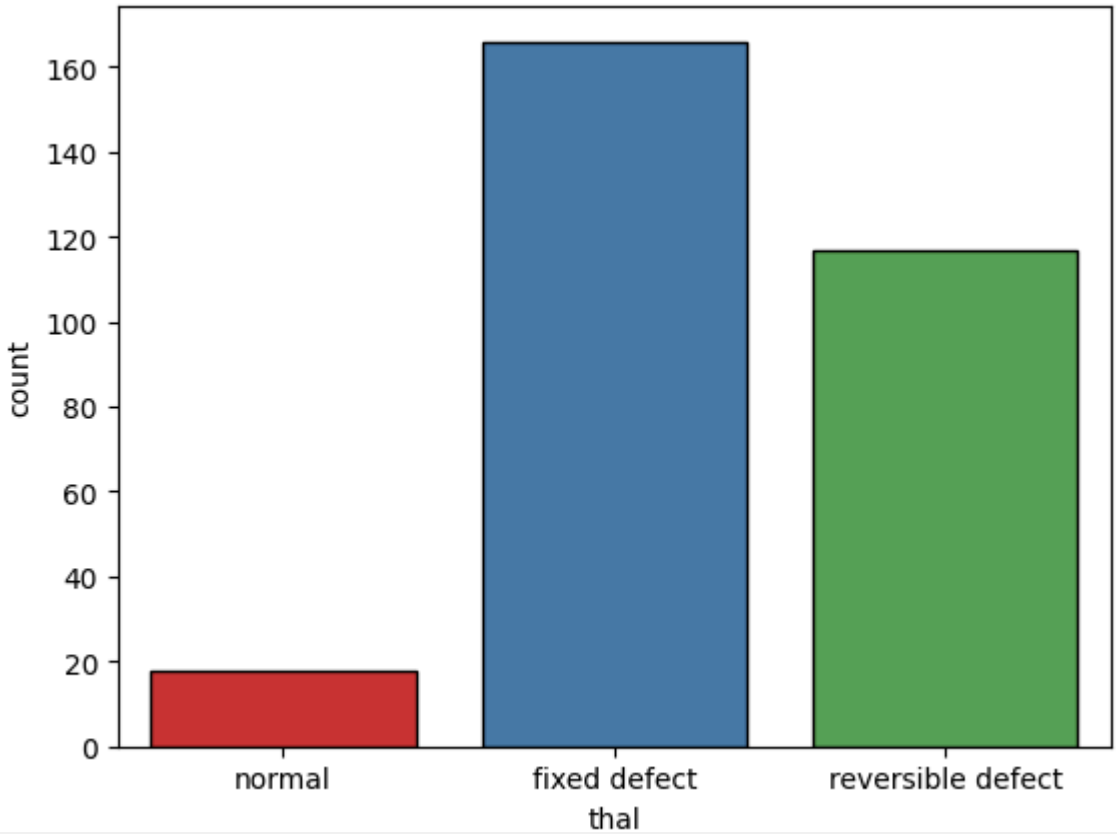
```
sns.countplot(x=data['sex'],palette='Set1',edgecolor='Black')
plt.show()
```

```
<ipython-input-48-103507cb145b>:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.

  sns.countplot(x=data['sex'],palette='Set1',edgecolor='Black')
```




```
sns.countplot(x=data['thal'],palette='Set1',edgecolor='Black')
plt.show()
```

```
<ipython-input-49-0bce94f3ec42>:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.

  sns.countplot(x=data['thal'],palette='Set1',edgecolor='Black')
```



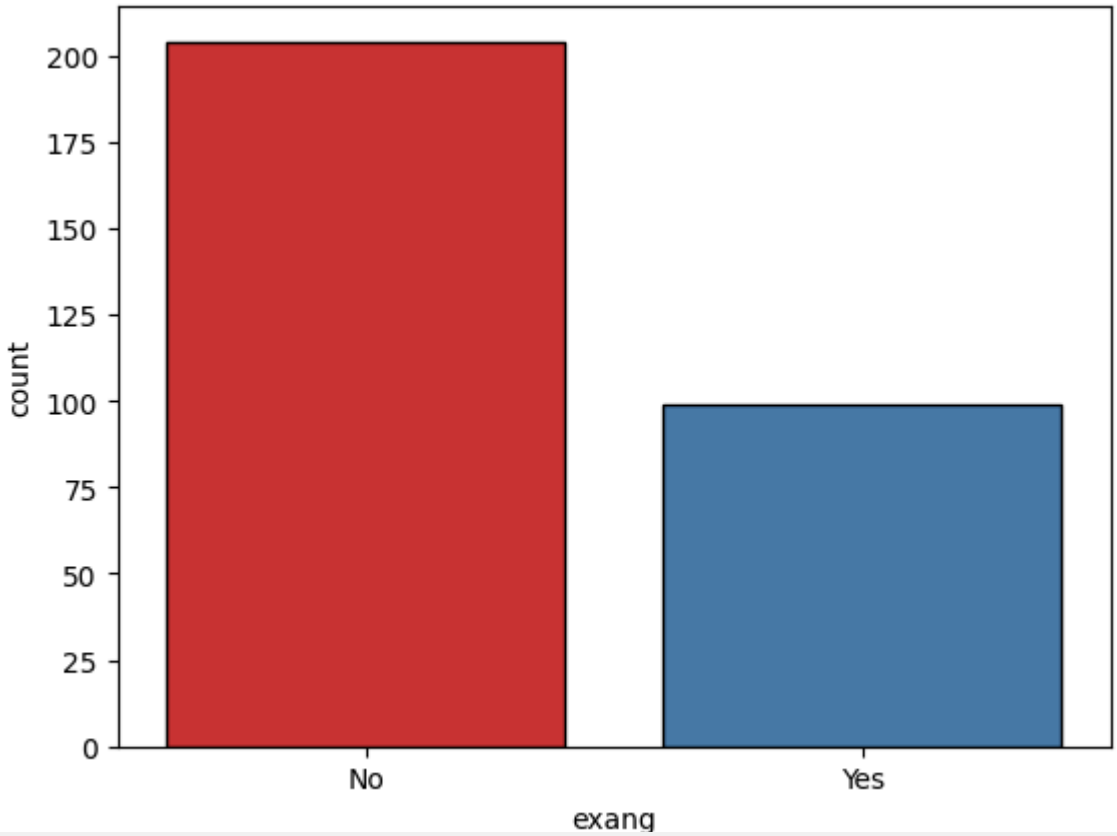

```
data['exang']=data['exang'].replace({0 : 'No',1 : 'Yes'})
```

```
data.info()
```

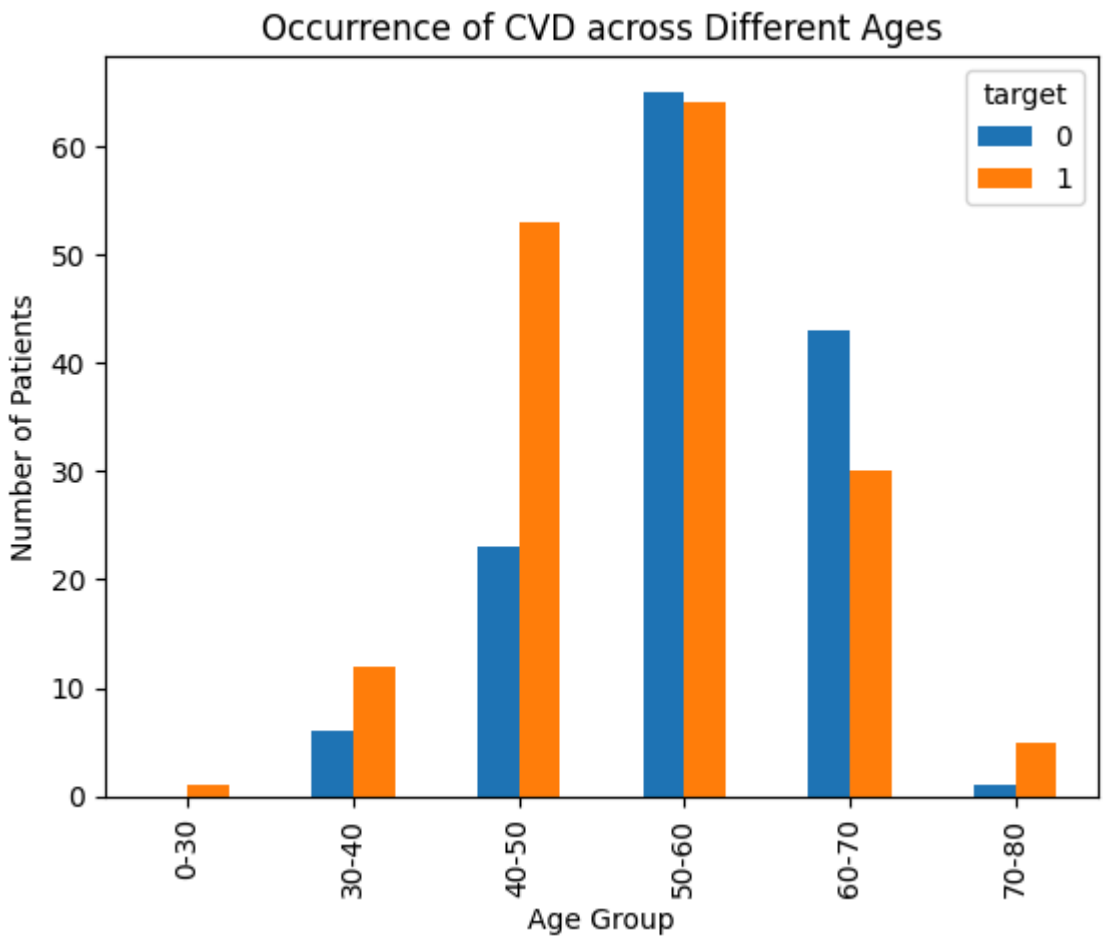
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       303 non-null    int64
 1   sex       303 non-null    object
 2   cp        303 non-null    int64
 3   trestbps  303 non-null    int64
 4   chol      303 non-null    int64
 5   fbs       303 non-null    int64
 6   restecg   303 non-null    int64
 7   thalach   303 non-null    int64
 8   exang     303 non-null    object
 9   oldpeak   303 non-null    float64
 10  slope     303 non-null    int64
 11  ca        303 non-null    int64
 12  thal      301 non-null    object
 13  target    303 non-null    int64
dtypes: float64(1), int64(10), object(3)
memory usage: 33.3+ KB
```

```
sns.countplot(x=data['exang'],palette='Set1',edgecolor='Black')
plt.show()
```

```
<ipython-input-52-9252fc7459ef>:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.

  sns.countplot(x=data['exang'],palette='Set1',edgecolor='Black')
```




```
# prompt: study the occurrence of cvd across different ages

import pandas as pd
import matplotlib.pyplot as plt
data['age_group'] = pd.cut(data['age'], bins=[0, 30, 40, 50, 60, 70, 80], labels=['0-30',

cvd_counts = data.groupby('age_group')['target'].value_counts()

# Create a bar chart to visualize the occurrence of CVD across different age groups.
cvd_counts.unstack().plot(kind='bar')
plt.xlabel('Age Group')
plt.ylabel('Number of Patients')
plt.title('Occurrence of CVD across Different Ages')
plt.show()
```




```
data['age_group1']=pd.cut(data['age'] , bins=[0,30,40,50,60,70,80], labels=['0-30','30-40
```

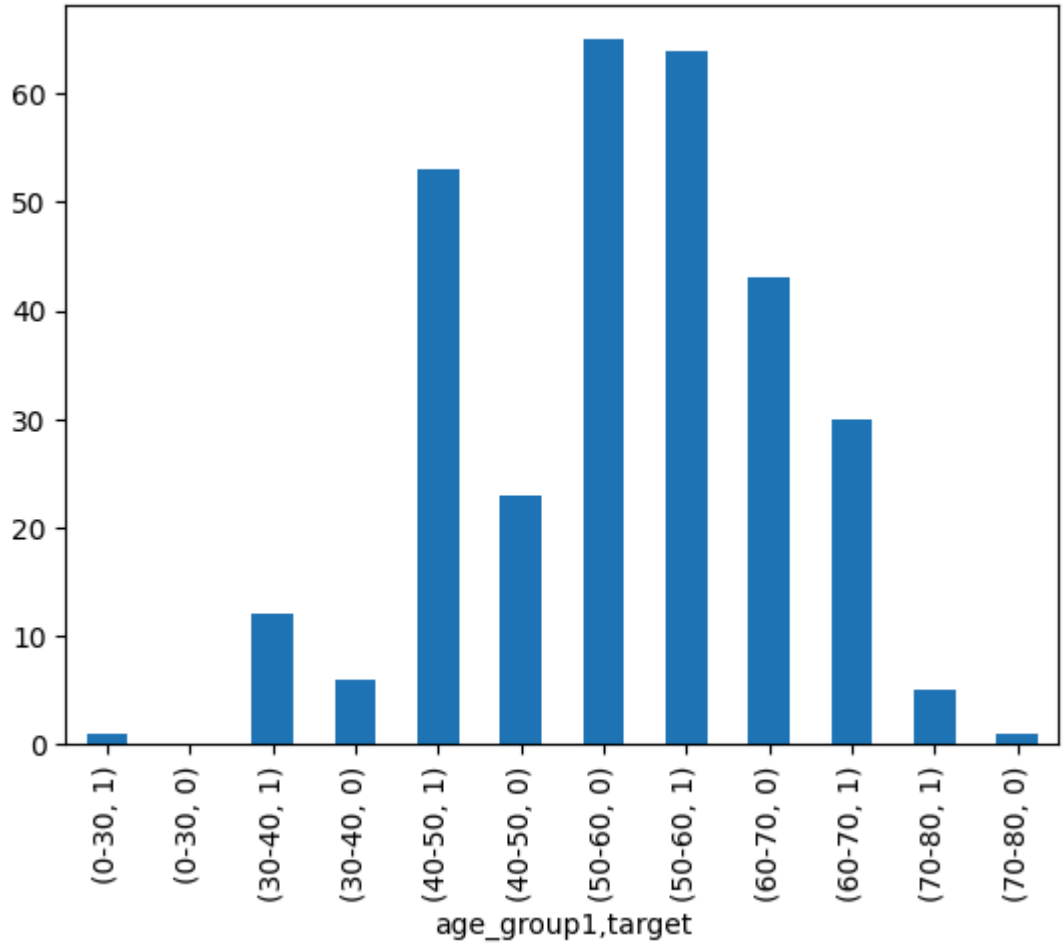
```
cvd_counts1=data.groupby('age_group1')['target'].value_counts()
```

```
cvd_counts
```

```
age_group  target
0-30       1          1
           0          0
30-40      1         12
           0          6
40-50      1         53
           0         23
50-60      0         65
           1         64
60-70      0         43
           1         30
70-80      1          5
           0          1
Name: count, dtype: int64
```

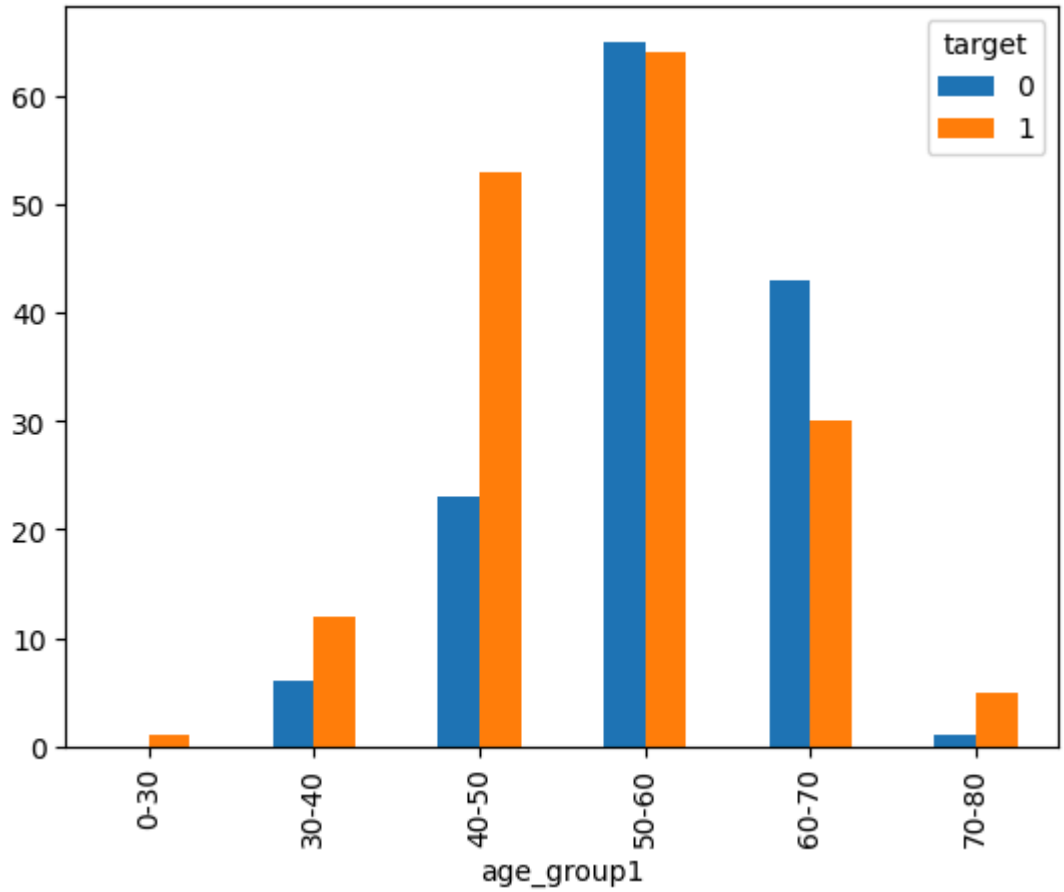
```
cvd_counts1.plot(kind='bar')
```

```
<Axes: xlabel='age_group1,target'>
```



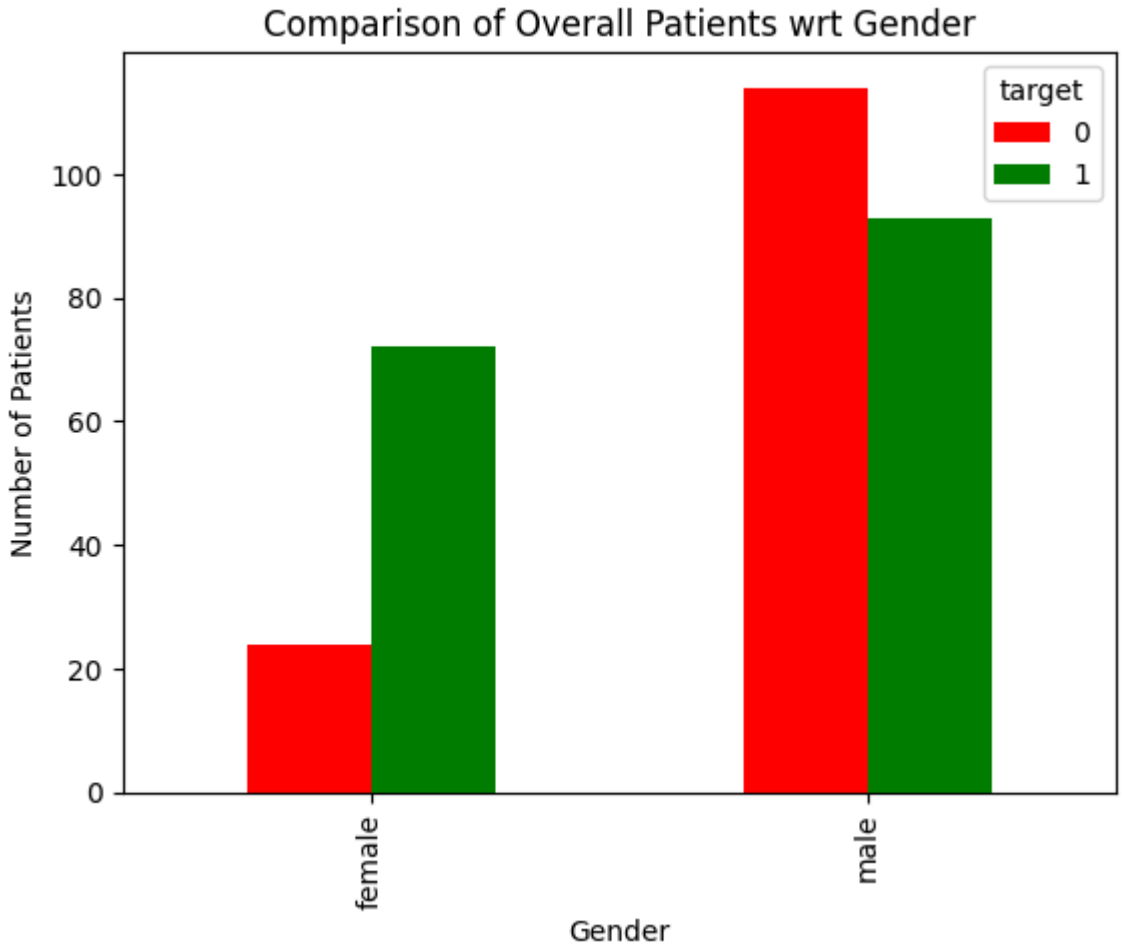

```
cvd_counts1.unstack().plot(kind='bar')
```

```
<Axes: xlabel='age_group1'>
```




```
# prompt: study the comparison of overall patients wrt gender

import pandas as pd
import matplotlib.pyplot as plt
# Create a crosstab table to compare the overall number of patients with respect to gende
gender_counts = pd.crosstab(data['sex'], data['target'])

# Print the crosstab table.
print(gender_counts)

# Visualize the crosstab table using a bar chart.
gender_counts.plot(kind='bar', color=['red', 'green'])
plt.xlabel('Gender')
plt.ylabel('Number of Patients')
plt.title('Comparison of Overall Patients wrt Gender')
plt.show()
```

```
target    0   1
sex
female   24  72
male    114  93
```




```
gender_counts1=pd.crosstab(data['sex'],data['target'])
```

```
gender_counts1
```

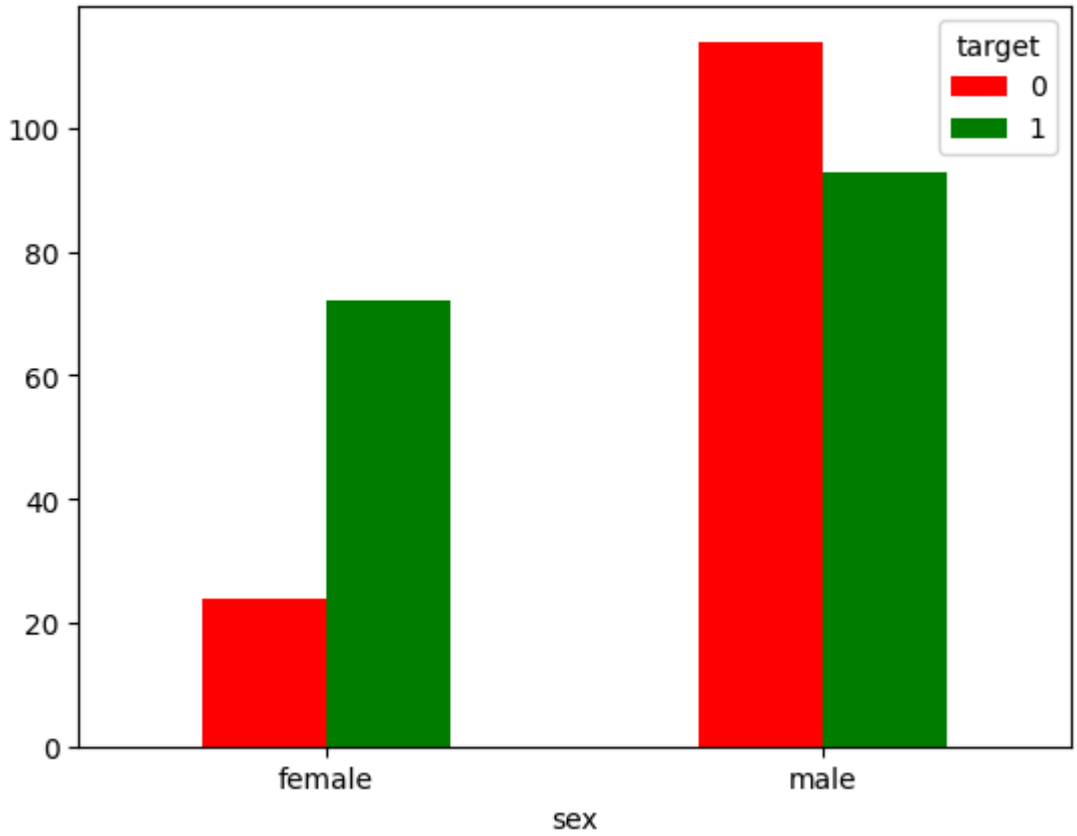
target	0	1
sex		
female	24	72
male	114	93

```
print(gender_counts1)
```

```
target    0   1
sex
female   24  72
male    114  93
```

```
gender_counts1.plot(kind='bar',color=['red','green'])
plt.xticks(rotation = 0)
```

```
(array([0, 1]), [Text(0, 0, 'female'), Text(1, 0, 'male')])
```



Double-click (or enter) to edit

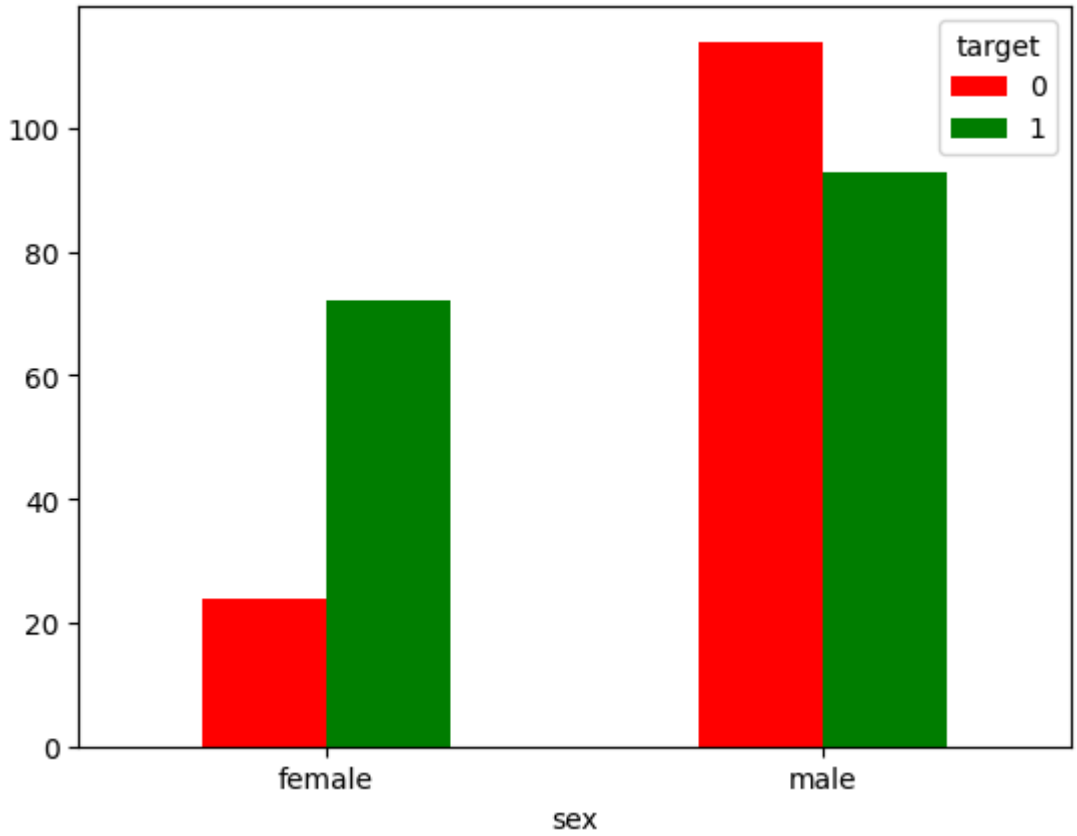
```
gender_counts2=data.groupby('sex')['target'].value_counts()
```

```
gender_counts2
```

```
sex     target
female  1          72
        0          24
male    0         114
        1          93
Name: count, dtype: int64
```

```
gender_counts2.unstack().plot(kind='bar',color=['red','green'])
plt.xticks(rotation=0)
```

```
(array([0, 1]), [Text(0, 0, 'female'), Text(1, 0, 'male')])
```

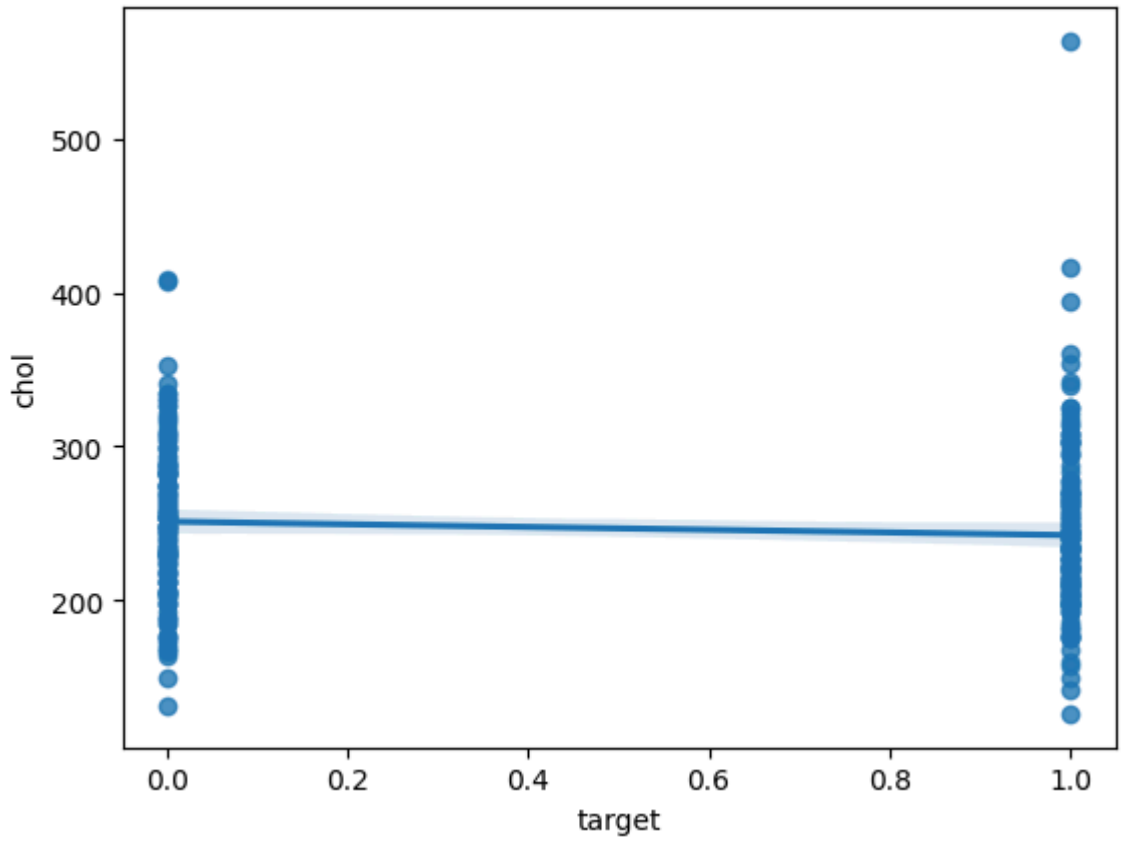



```
data1.head()
```

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

```
sns.regplot(x='target',y='chol',data=data1)
data[['target','chol']].corr()
```

	target	chol
target	1.000000	-0.085239
chol	-0.085239	1.000000



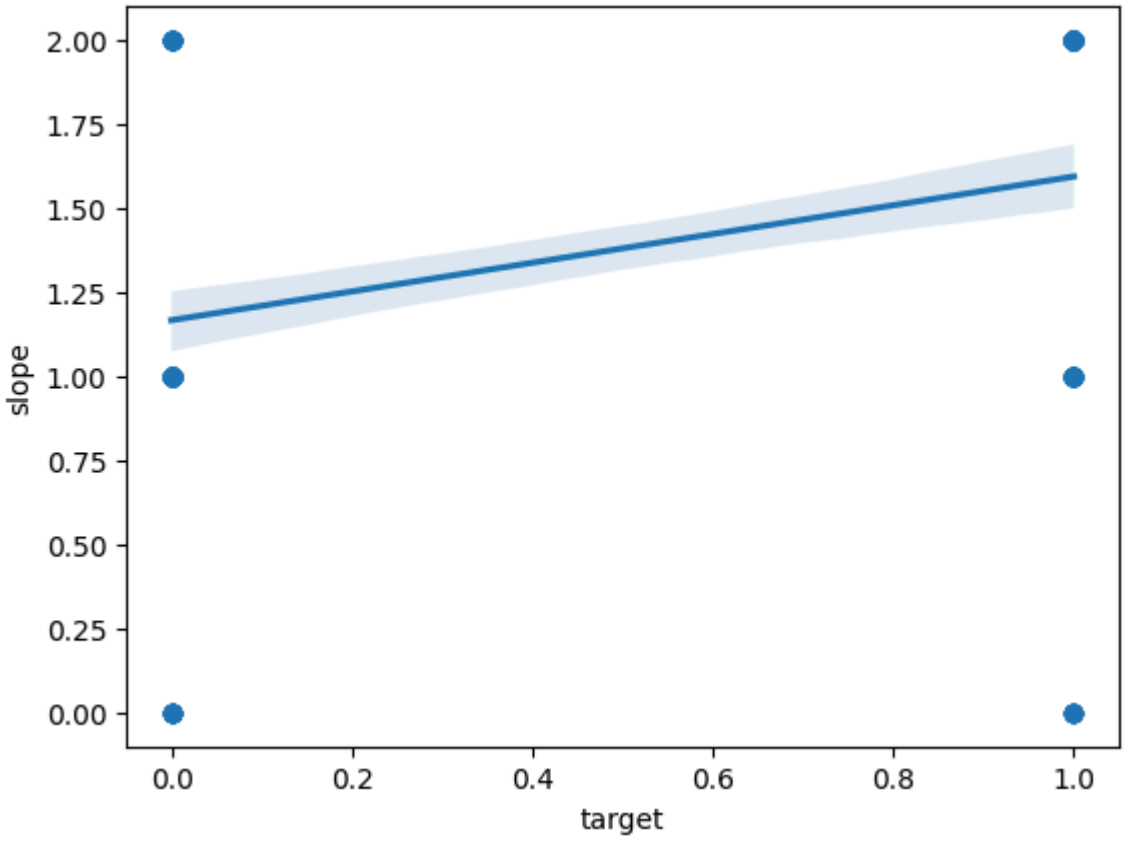

```
# prompt: describe relation between cholestrol level and target variable

The provided code snippet analyzes a dataset containing patient information and performs

1. **Correlation Analysis:**

The code snippet includes the following line:
```

```
data1.head()
```

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

```
sns.regplot(x='target',y='slope',data=data1)
data1[['target','slope']].corr()
```

	target	slope
target	1.000000	0.345877
slope	0.345877	1.000000

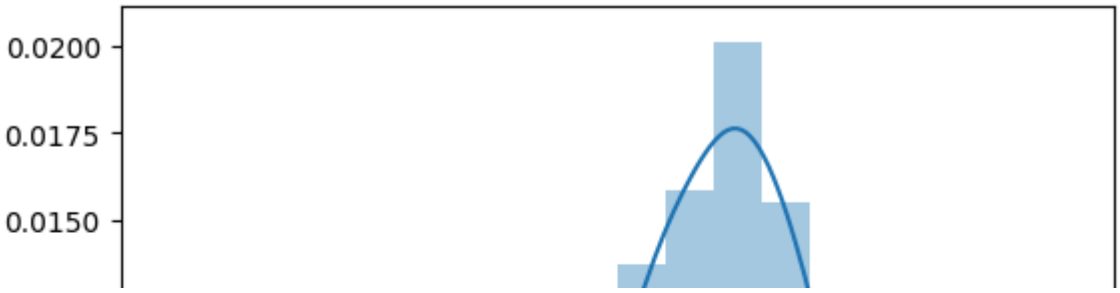



```
sns.pairplot(data1)
```

```
<seaborn.axisgrid.PairGrid at 0x7d68f6700b20>
```


```
sns.distplot(data1['thalach'])
```

```
<ipython-input-78-c579f341f382>:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(data1['thalach'])
<Axes: xlabel='thalach', ylabel='Density'>
```




```
data1.head()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	tha
0	63	1	3	145	233	1	0	150	0	2.3	0	0	
1	37	1	2	130	250	0	1	187	0	3.5	0	0	
2	41	0	1	130	204	0	0	172	0	1.4	2	0	