In [2]:

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
print('All library imported')
```

```
All library imported
```

In [3]:

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

In [5]:

```
data.head()
```

Out[5]:

	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 [6]:

```
data.shape
```

Out[6]:

```
(303, 14)
```

In [7]:

```
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
```

In [8]:

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

Out[8]:

```
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
```

In [9]:

```python
data.describe()
```

Out[9]:

	age	sex	cp	trestbps	chol	fbs	restecg	tha
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.00
mean	54.366337	0.683168	0.966997	131.623762	246.264026	0.148515	0.528053	149.64
std	9.082101	0.466011	1.032052	17.538143	51.830751	0.356198	0.525860	22.90
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.00
25%	47.500000	0.000000	0.000000	120.000000	211.000000	0.000000	0.000000	133.50
50%	55.000000	1.000000	1.000000	130.000000	240.000000	0.000000	1.000000	153.00
75%	61.000000	1.000000	2.000000	140.000000	274.500000	0.000000	1.000000	166.00
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000	202.00

In [10]:

```python
data.nunique()
```

Out[10]:

```
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
```

In [12]:

```python
duplicated_data=data[data.duplicated(keep='last')]
duplicated_data
```

Out[12]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
163	38	1	2	138	175	0	1	173	0	0.0	2	4	2	1

In [14]:

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

Out[14]:

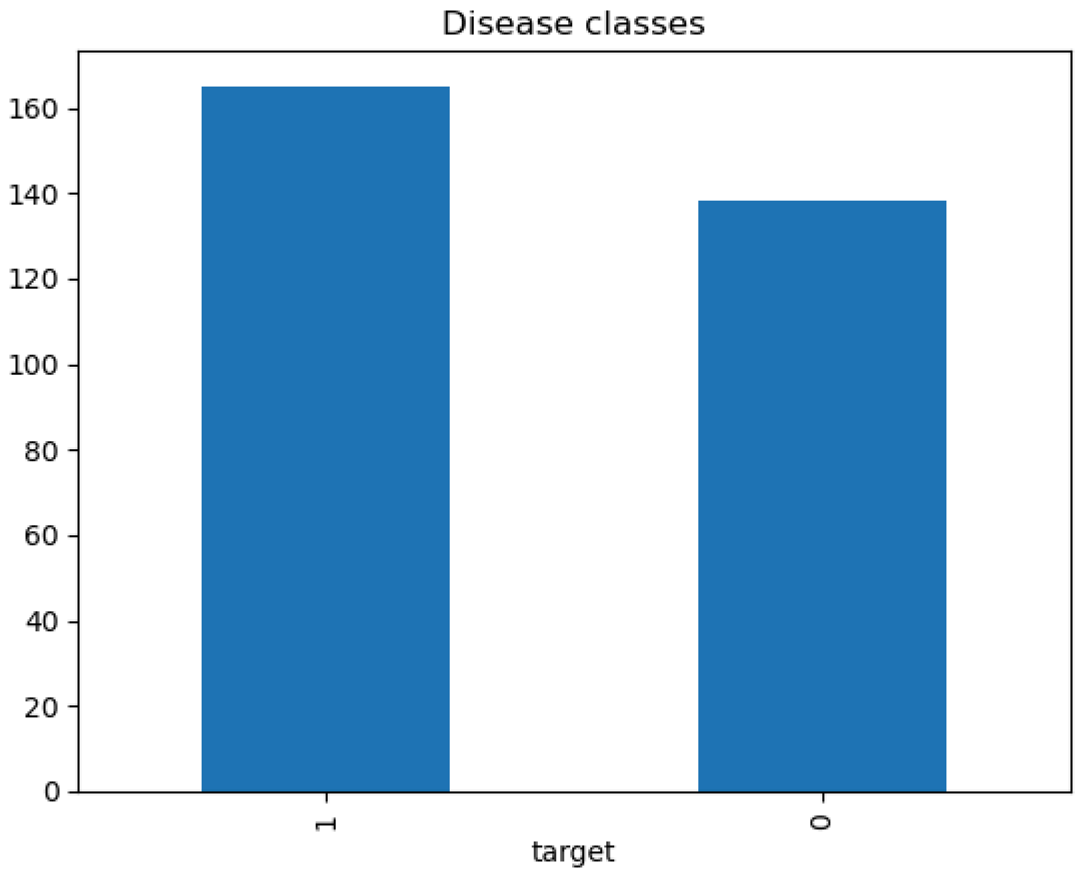
```
target
1    165
0    138
Name: count, dtype: int64
```

In [15]:

```
#Exploratory Data Analysis on Heart diseases
data['target'].value_counts().plot(kind='bar')
plt.title('Disease classes')
```

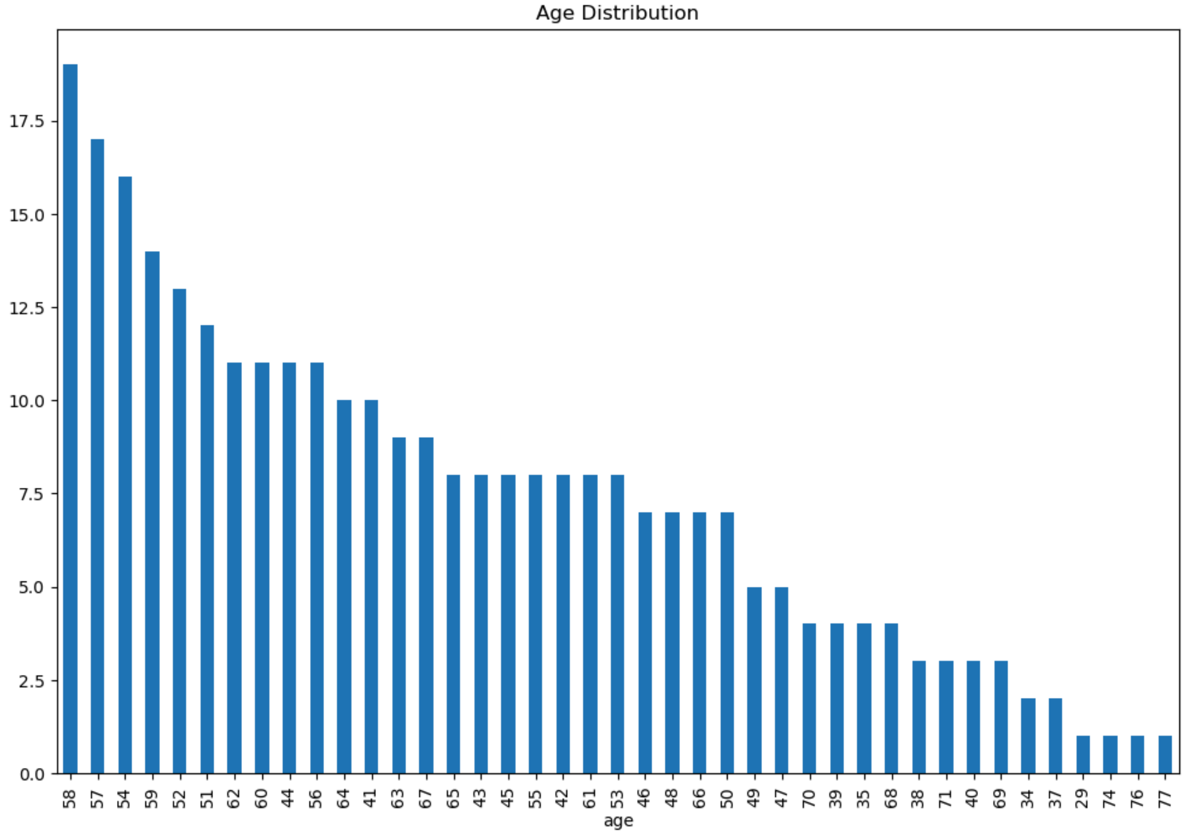
Out[15]:

```
Text(0.5, 1.0, 'Disease classes')
```



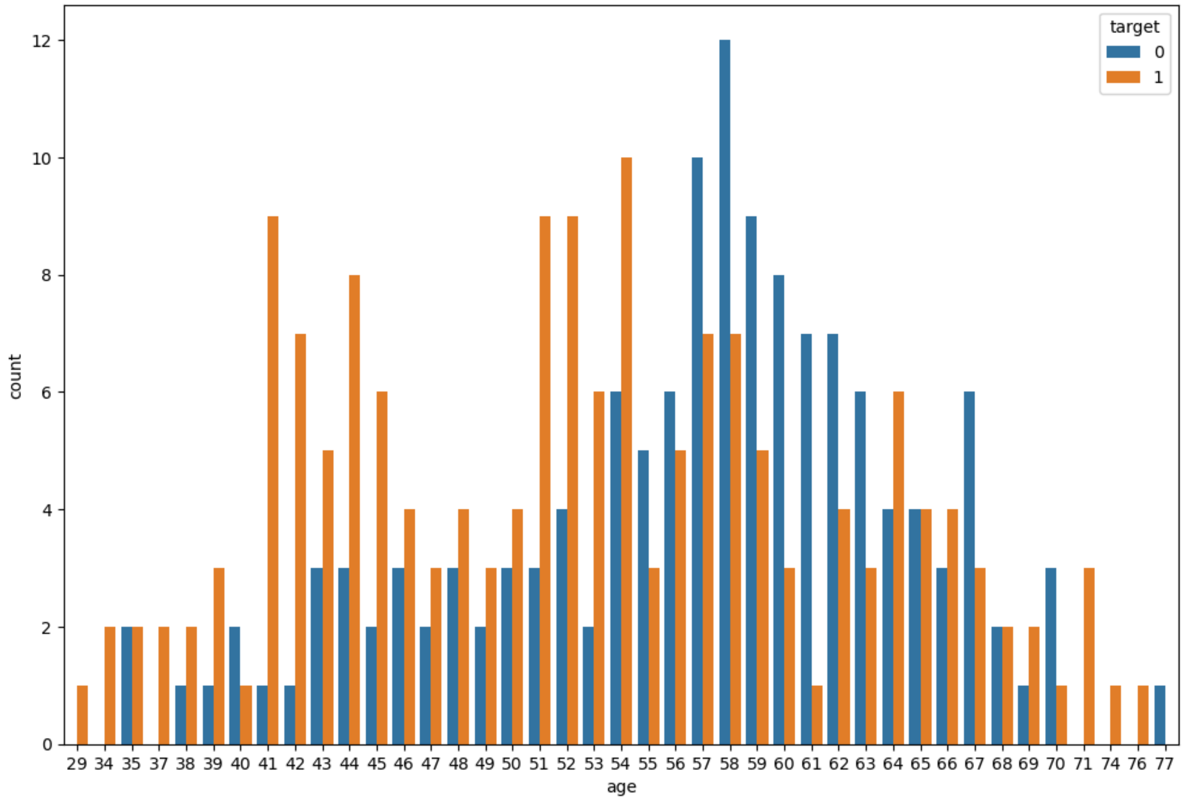
In [16]:

```
#distribution of age
plt.figure(figsize=(12,8))
data['age'].value_counts().plot(kind='bar')
plt.title('Age Distribution')
plt.show()
```



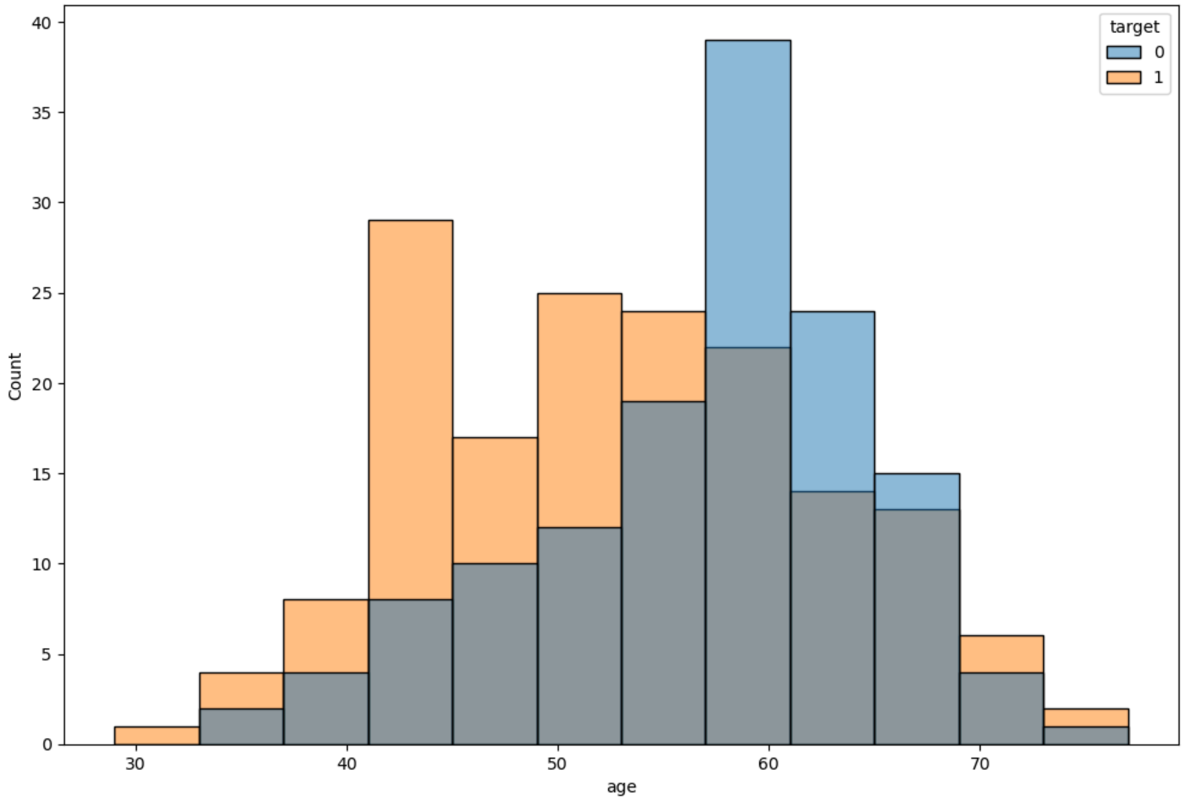
In [18]:

```
plt.figure(figsize=(12,8))
sns.countplot(x=data['age'],hue='target',data=data)
plt.show()
```



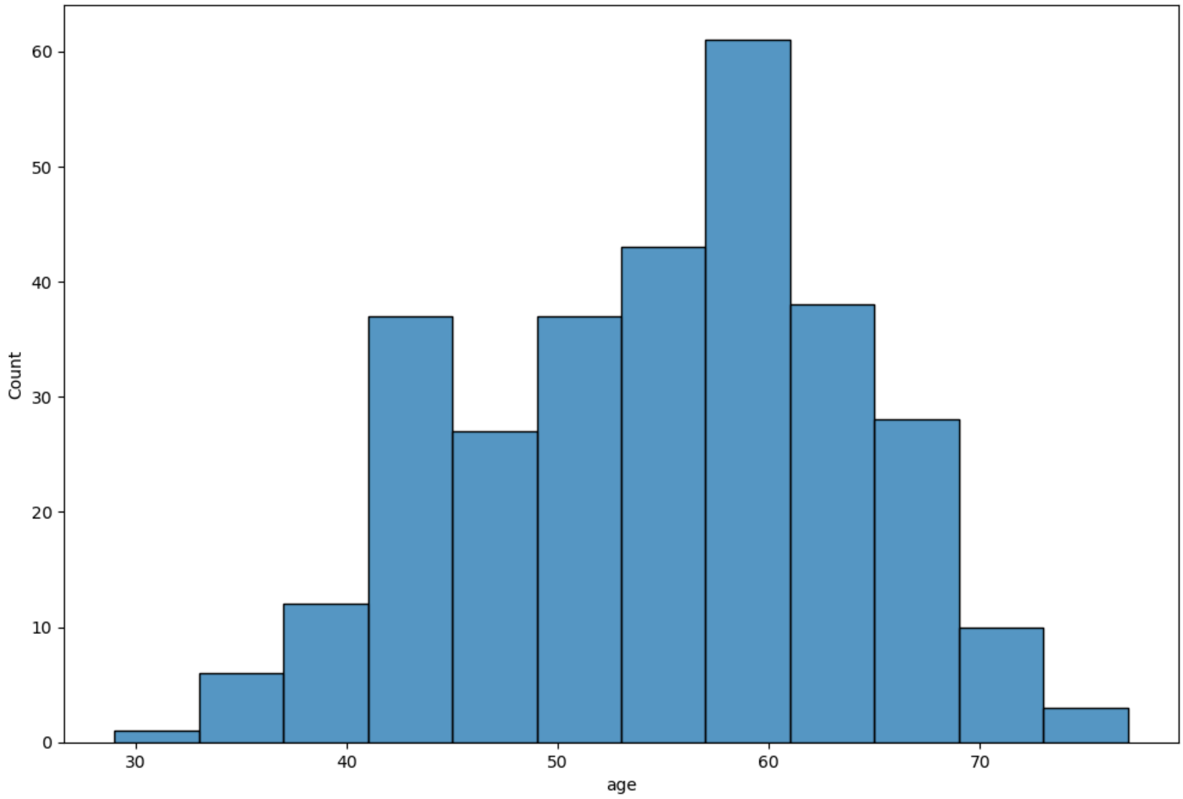
In [19]:

```
plt.figure(figsize=(12,8))
sns.histplot(x=data['age'],hue='target',data=data)
plt.show()
```



In [21]:

```
plt.figure(figsize=(12,8))
sns.histplot(x=data['age'],data=data)
plt.show()
```



In [22]:

```
#gender
data.columns
```

Out[22]:

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

In [23]:

```
print(data['sex'].value_counts())
```

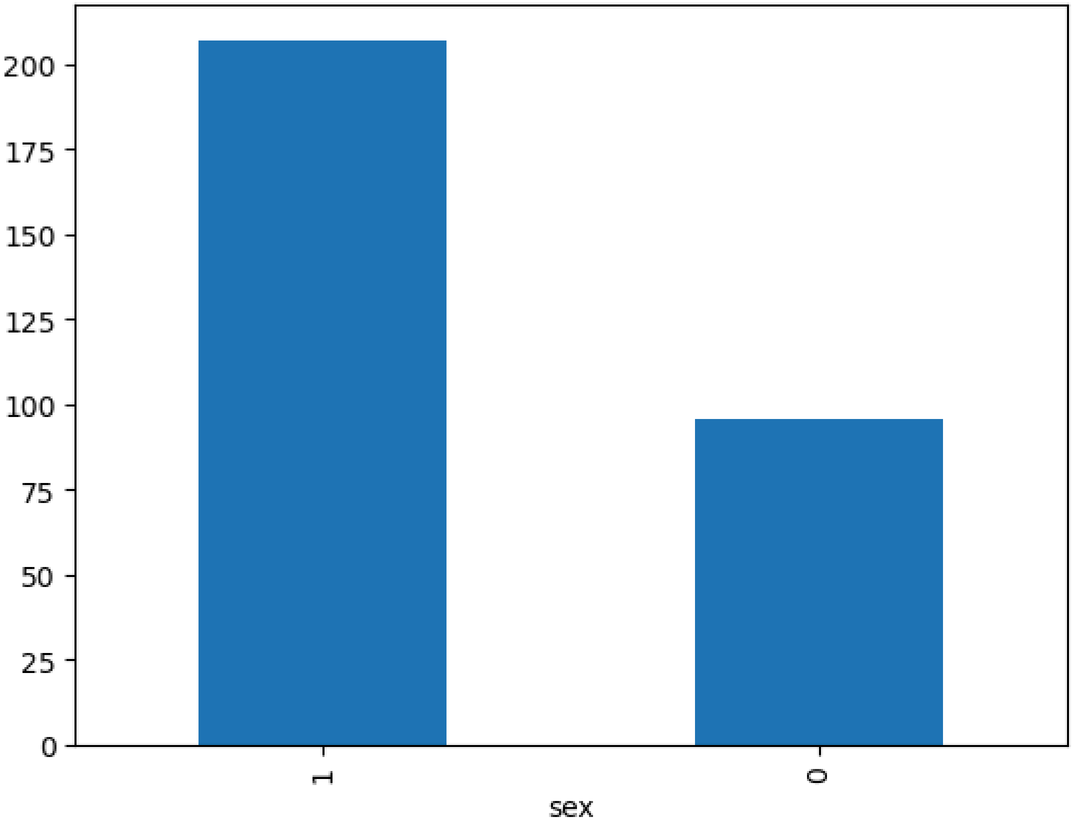
```
sex
1    207
0     96
Name: count, dtype: int64
```

In [24]:

```
#gender Analysis
data['sex'].value_counts().plot(kind='bar')
```

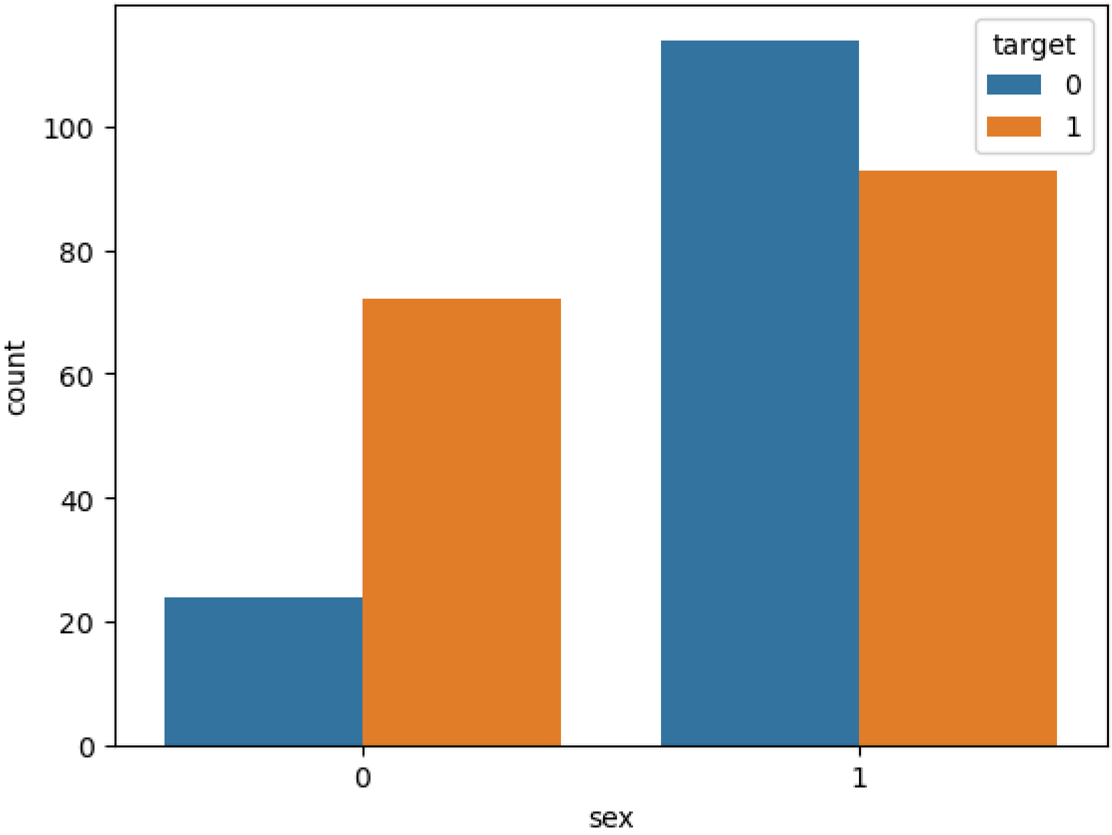
Out[24]:

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



In [27]:

```
#plt.figure(figsize=(12,8))
sns.countplot(x=data['sex'],hue='target',data=data)
plt.show()
```

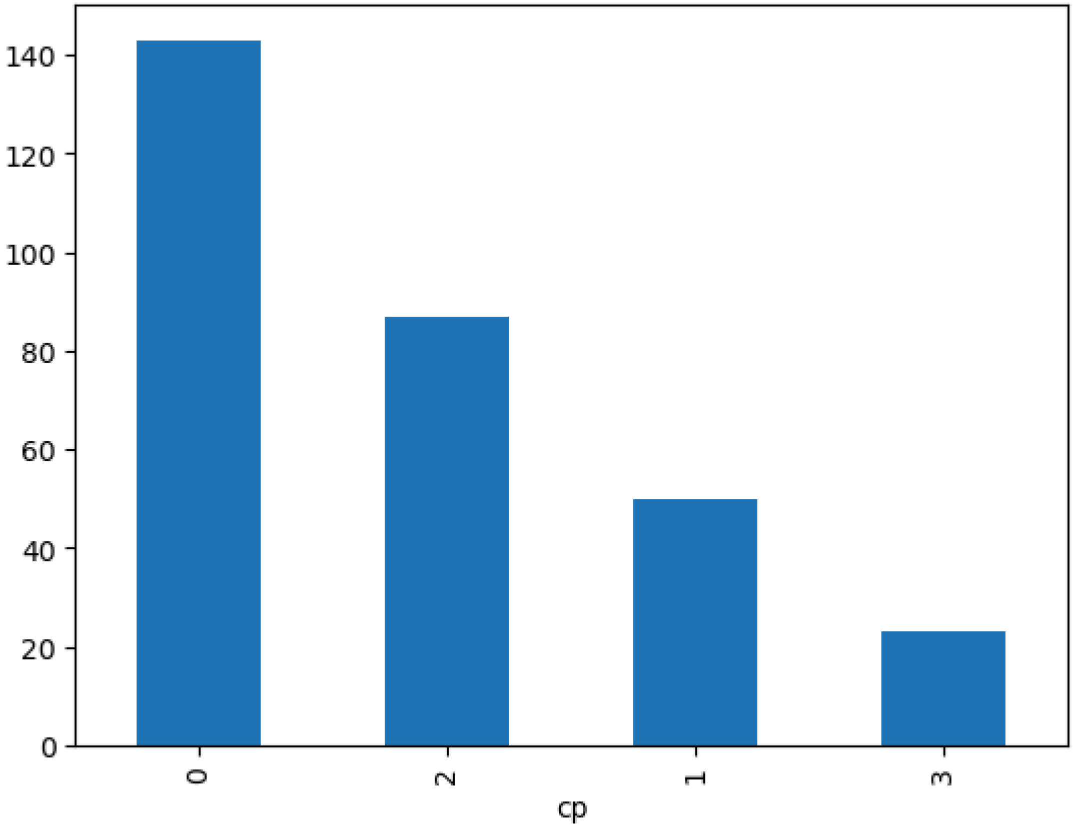


```
In [4]: #cp type
        data['cp'].value_counts()
```

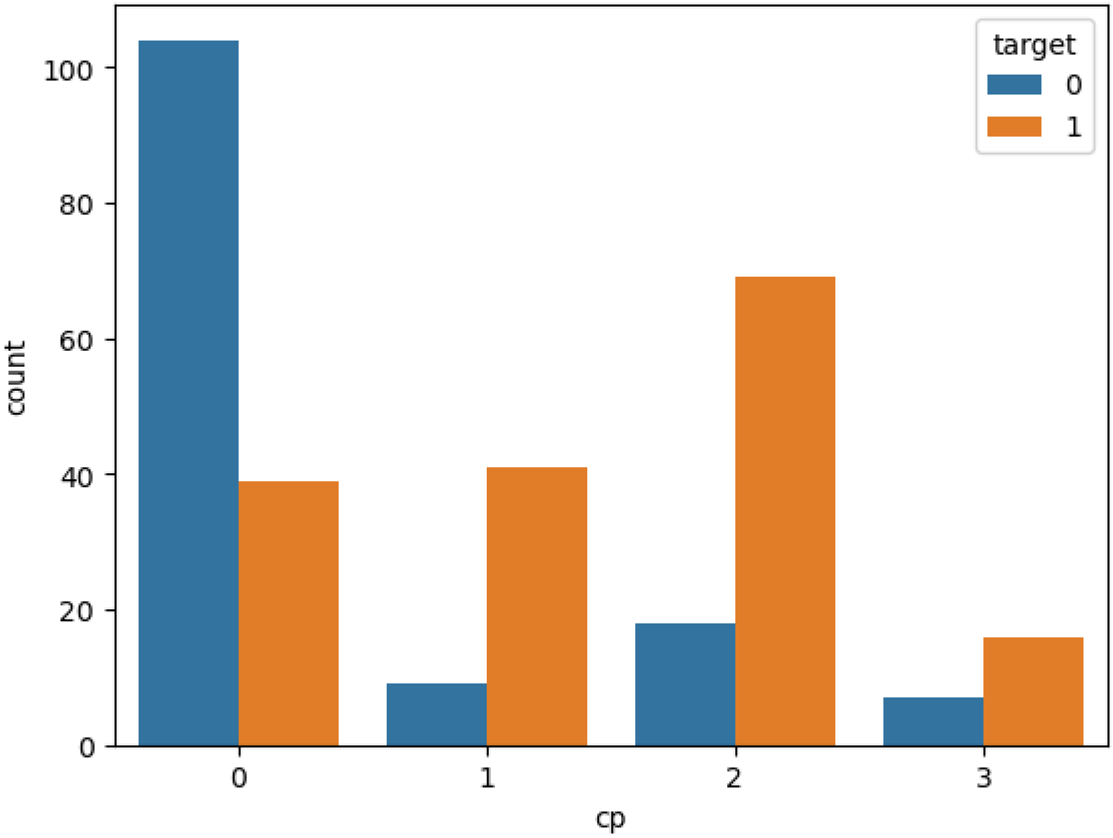
```
Out[4]: cp
        0    143
        2     87
        1     50
        3     23
        Name: count, dtype: int64
```

```
In [5]: data['cp'].value_counts().plot(kind='bar')
```

```
Out[5]: <Axes: xlabel='cp'>
```



In [6]:
```
sns.countplot(x=data['cp'],hue='target',data=data)
plt.show()
```



In [7]:
```
data.columns
```

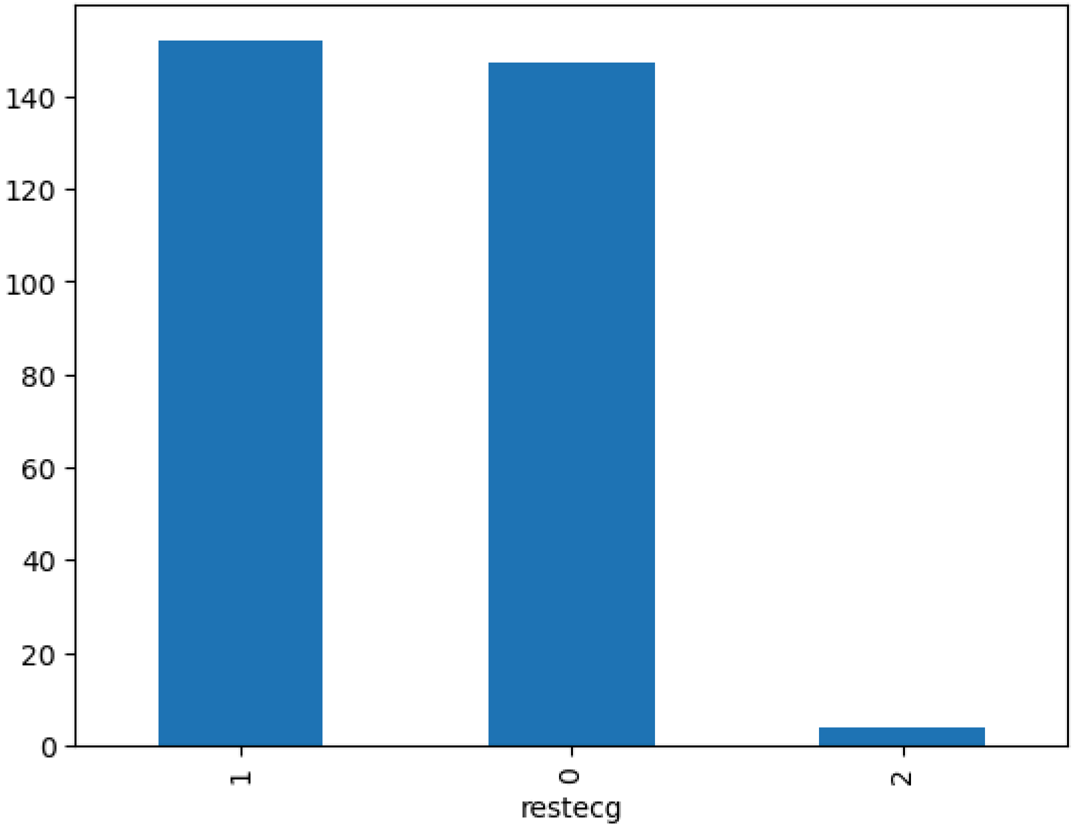
Out[7]:
```
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
      dtype='object')
```

In [8]:
```
#ecg
data['restecg'].value_counts()
```

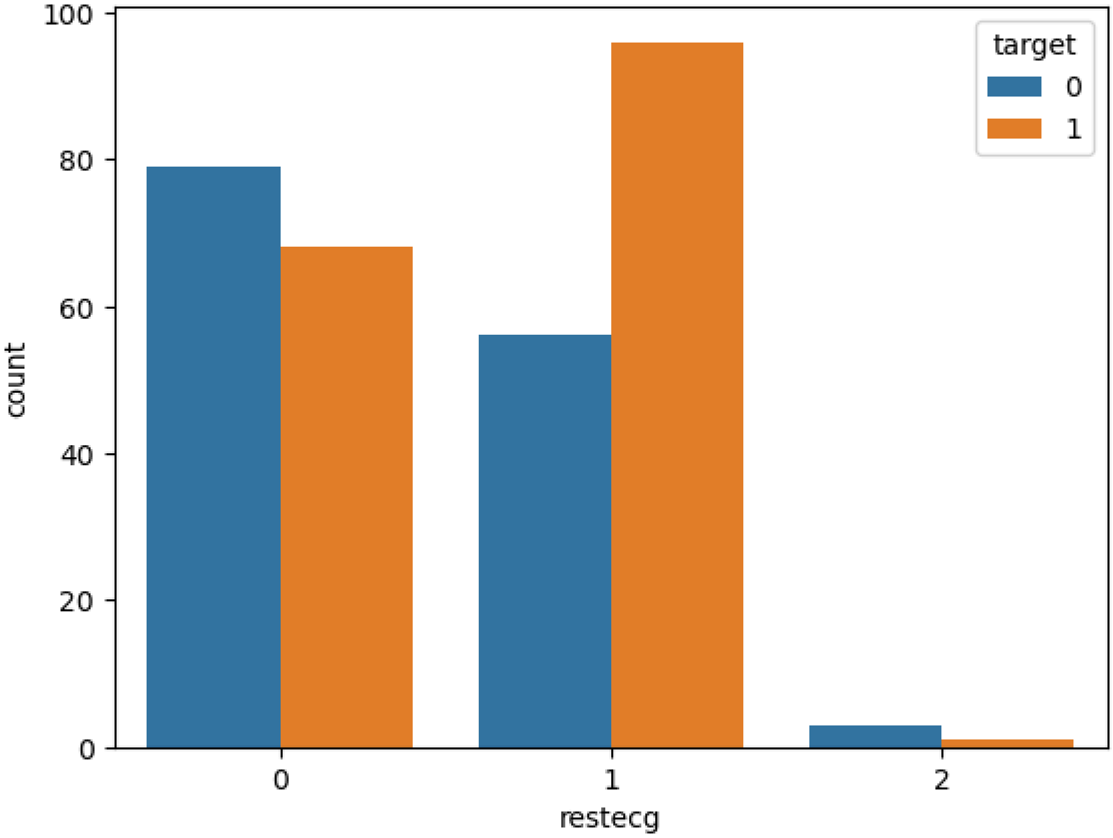
Out[8]:
```
restecg
1    152
0    147
2      4
Name: count, dtype: int64
```

In [81]:
```
data['restecg'].value_counts().plot(kind='bar')
```

Out[81]:
```
<Axes: xlabel='restecg'>
```



In [83]:
```
sns.countplot(x=data['restecg'],hue='target',data=data)
plt.show()
```

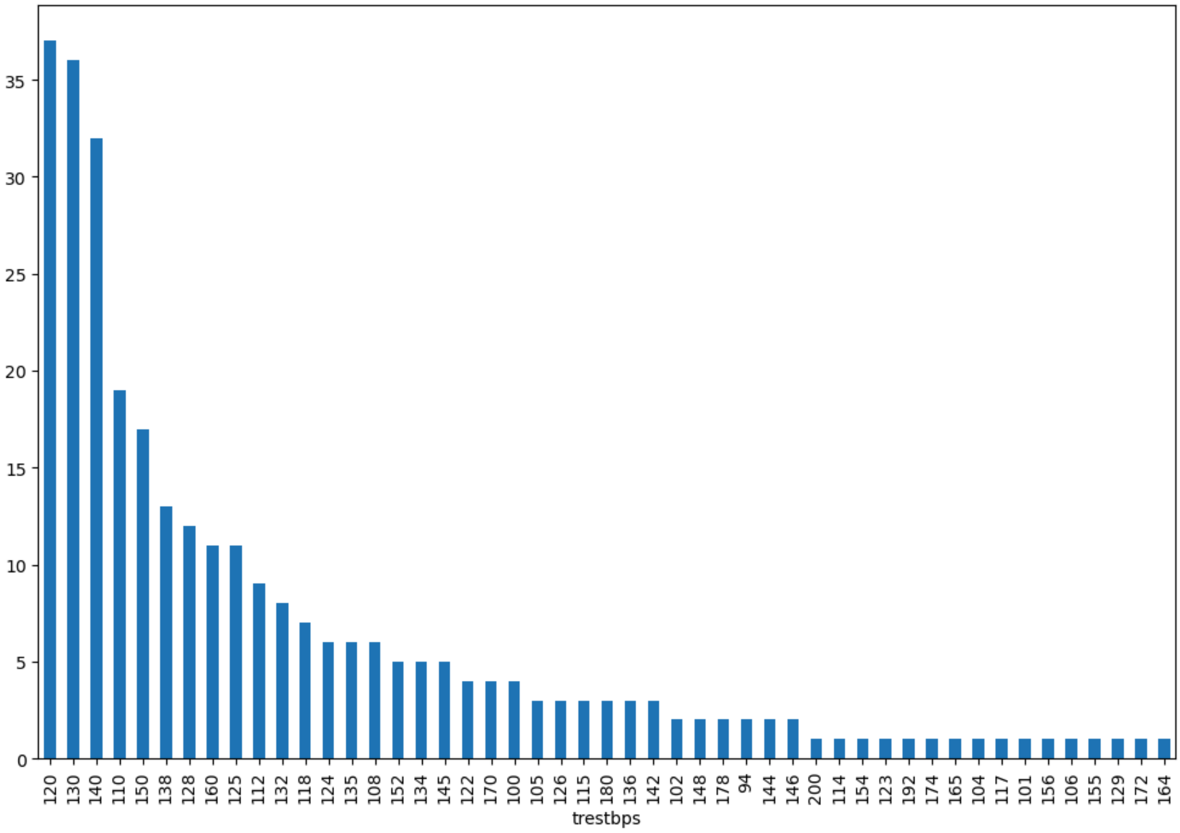


In [13]:

```
#bp
plt.figure(figsize=(12,8))
data['trestbps'].value_counts().plot(kind='bar')
```

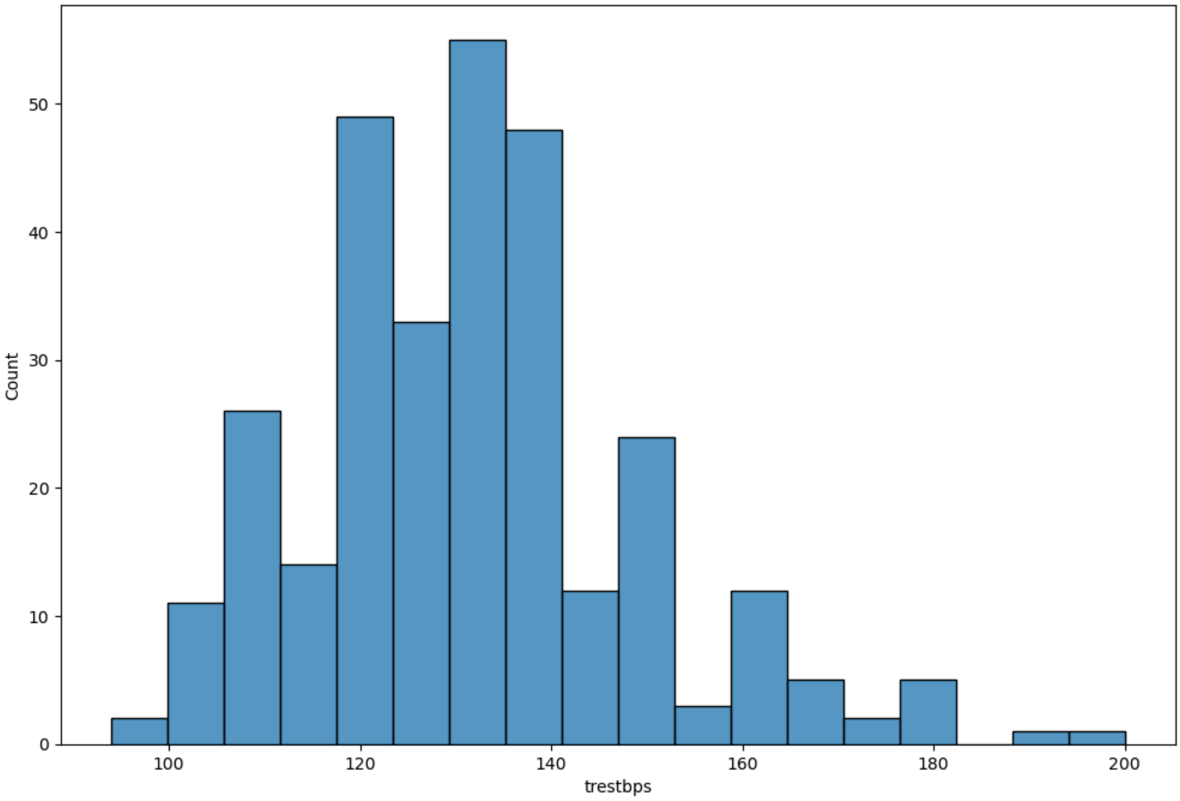
Out[13]:

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



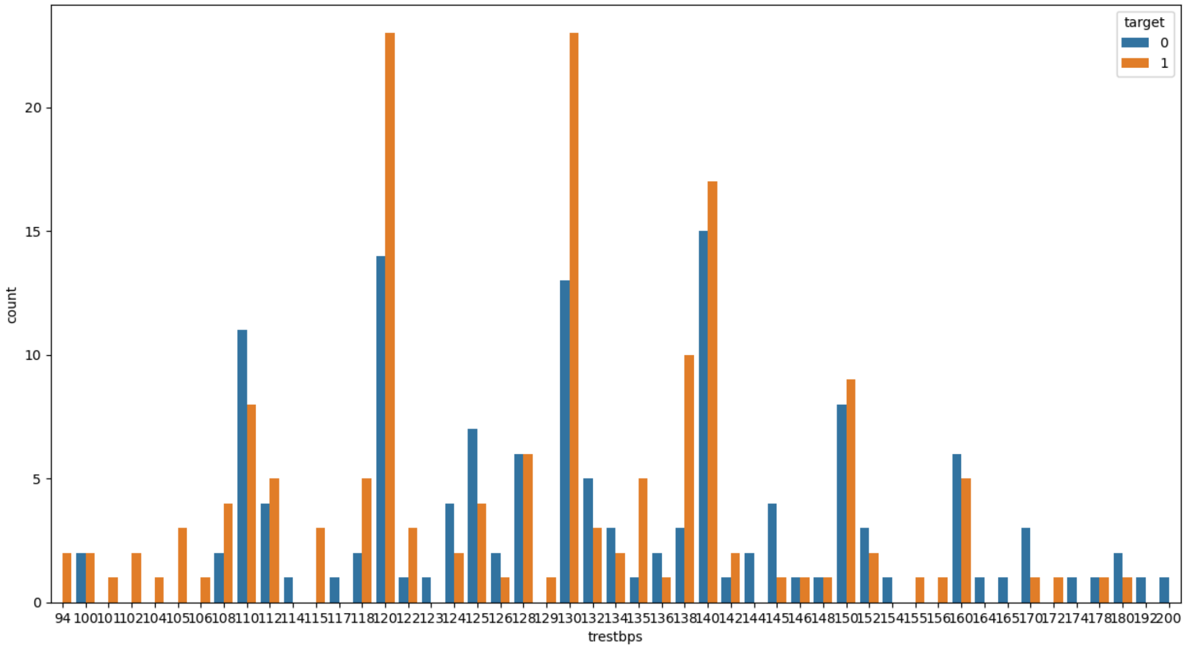
In [14]:

```
plt.figure(figsize=(12,8))
sns.histplot(x=data['trestbps'],data=data)
plt.show()
```



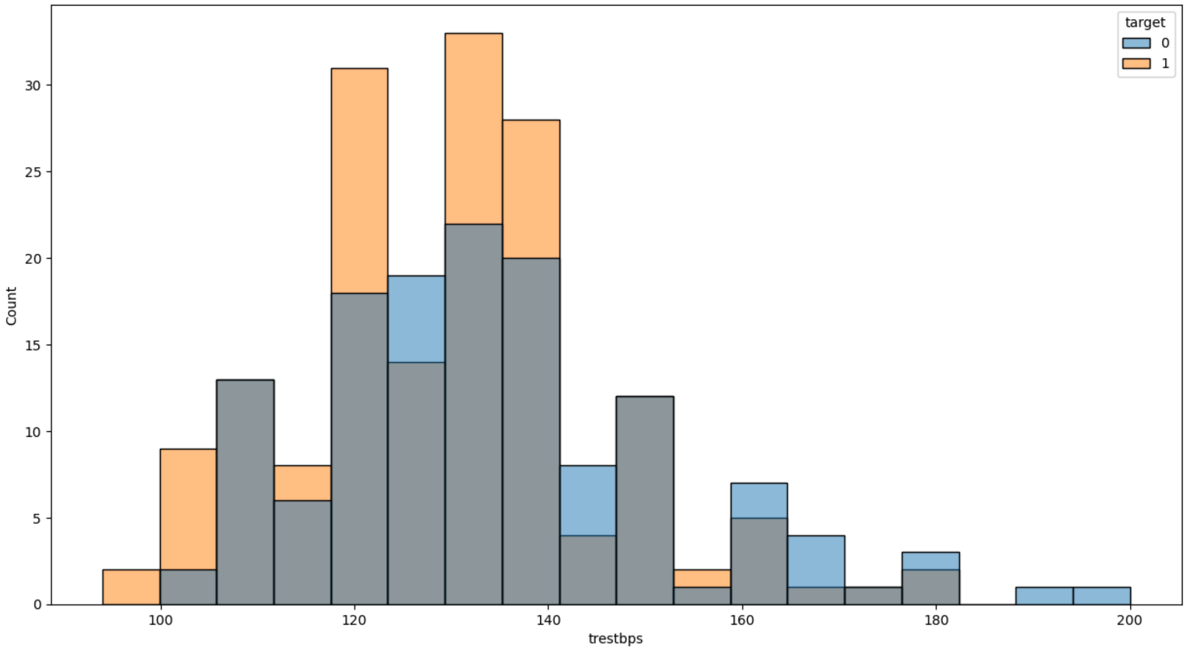
In [17]:

```
plt.figure(figsize=(15,8))
sns.countplot(x=data['trestbps'],hue='target',data=data)
plt.show()
```



In [16]:

```
plt.figure(figsize=(15,8))
sns.histplot(x=data['trestbps'],hue='target',data=data)
plt.show()
```



In [19]:
```
data.nunique()
```

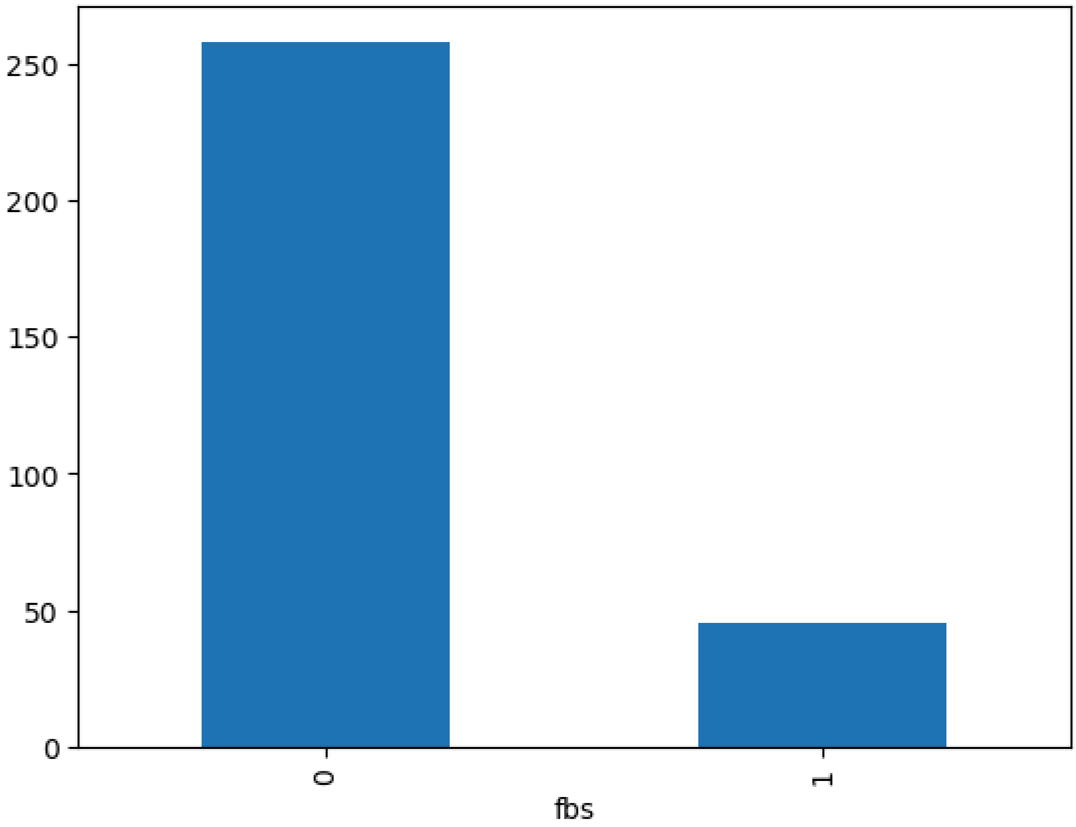
Out[19]:
```
age          41
sex           2
cp            4
trestbps     49
chol        152
fbs           2
restecg       3
thalach      91
exang         2
oldpeak      40
slope         3
ca            5
thal          4
target        2
dtype: int64
```

In [20]:
```
data['fbs'].value_counts()
```

Out[20]:
```
fbs
0    258
1     45
Name: count, dtype: int64
```

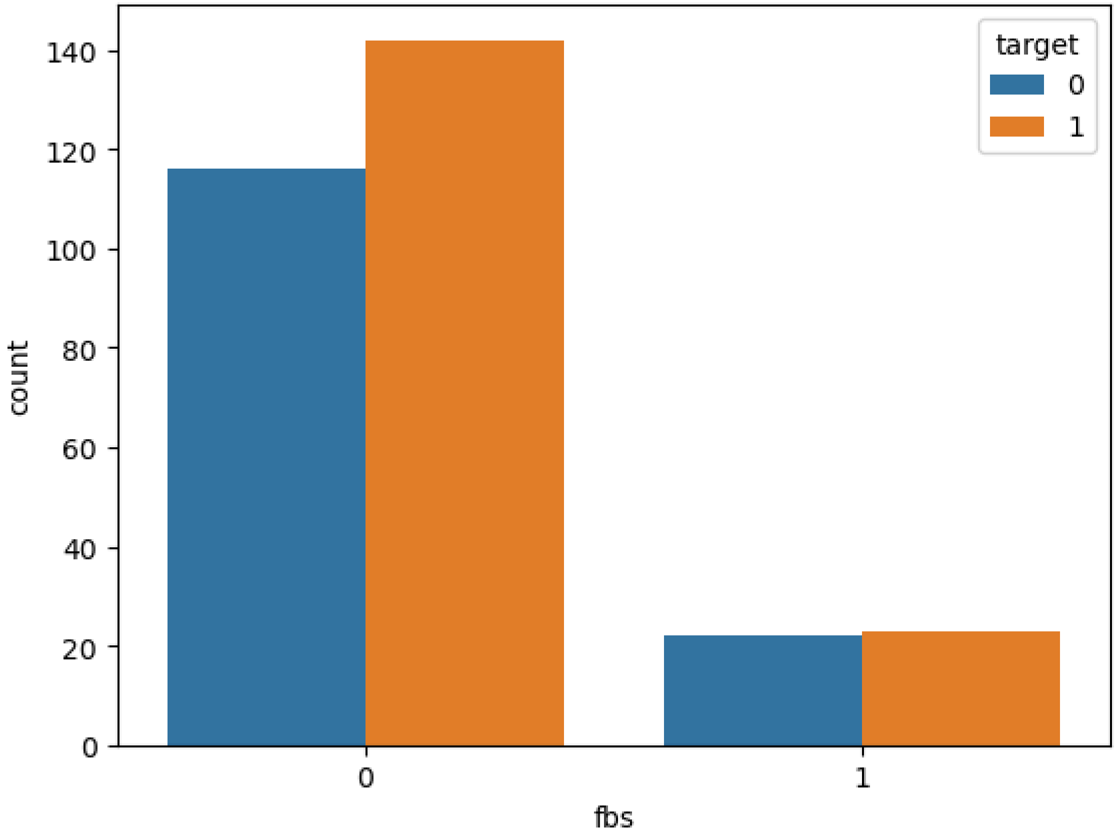
In [21]:
```
data['fbs'].value_counts().plot(kind='bar')
```

Out[21]:
```
<Axes: xlabel='fbs'>
```



In [22]:

```
sns.countplot(x=data['fbs'],hue='target',data=data)
plt.show()
```

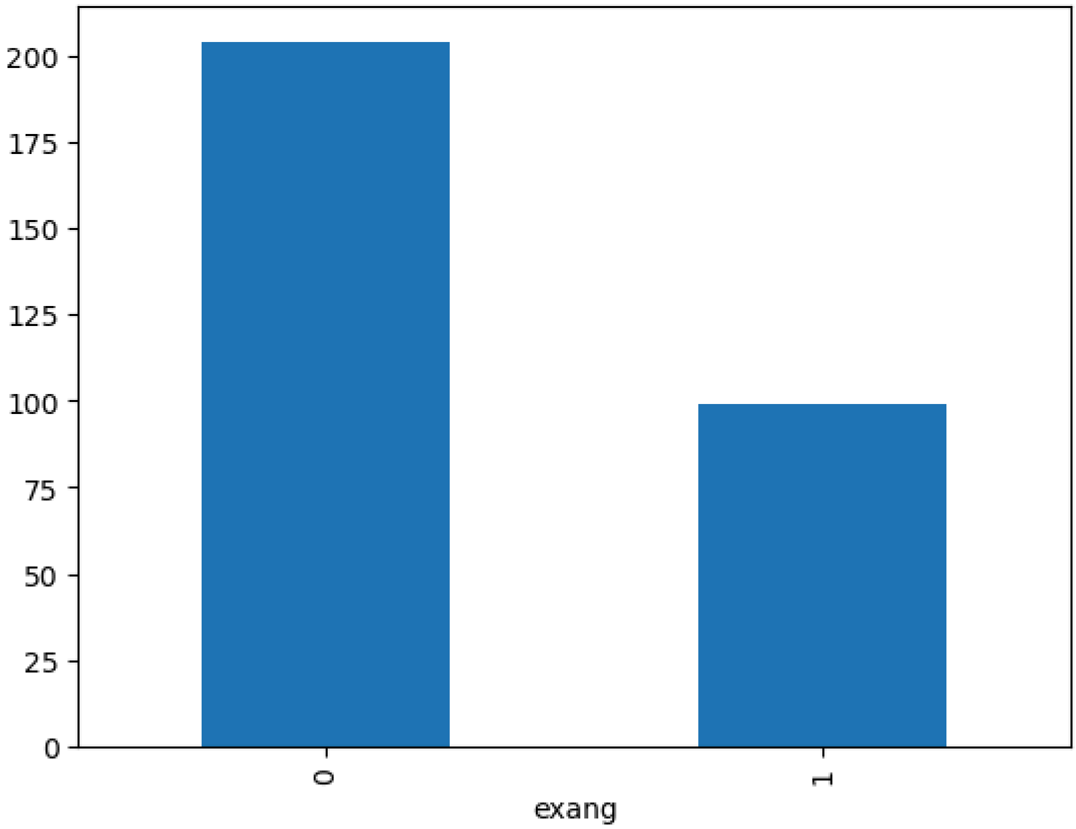


In [23]:

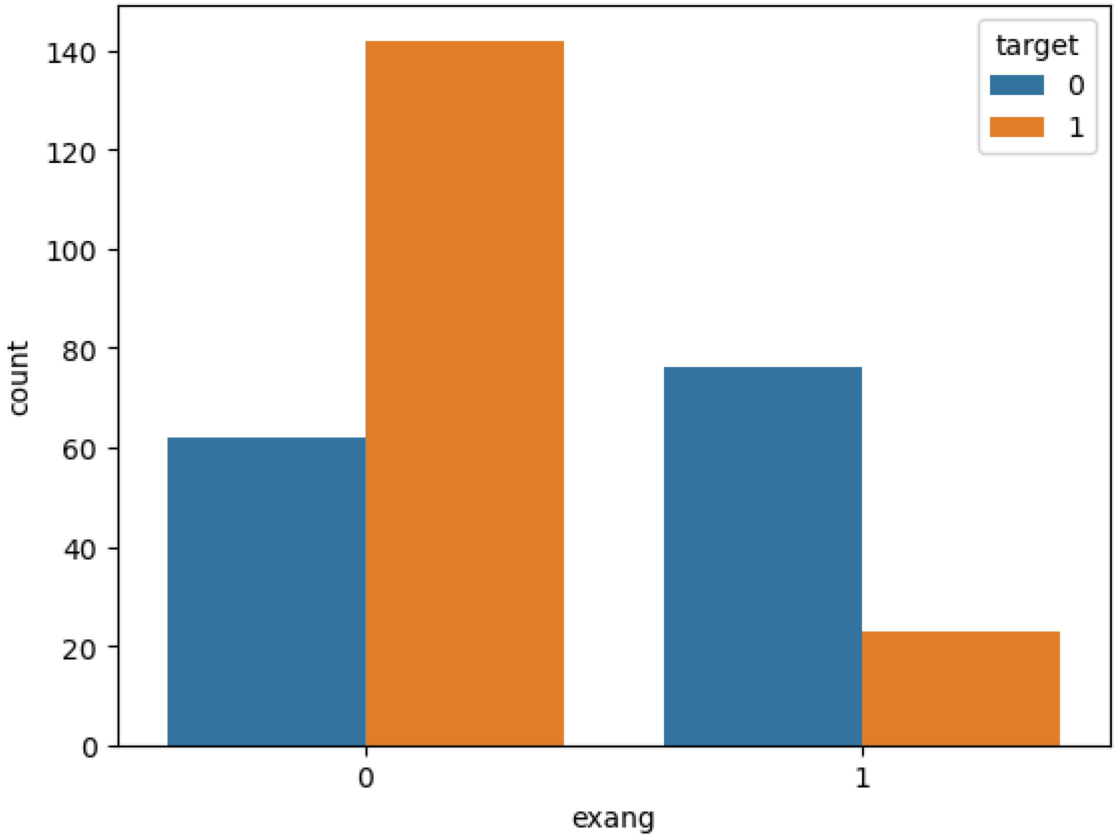
```
data['exang'].value_counts().plot(kind='bar')
```

Out[23]:

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

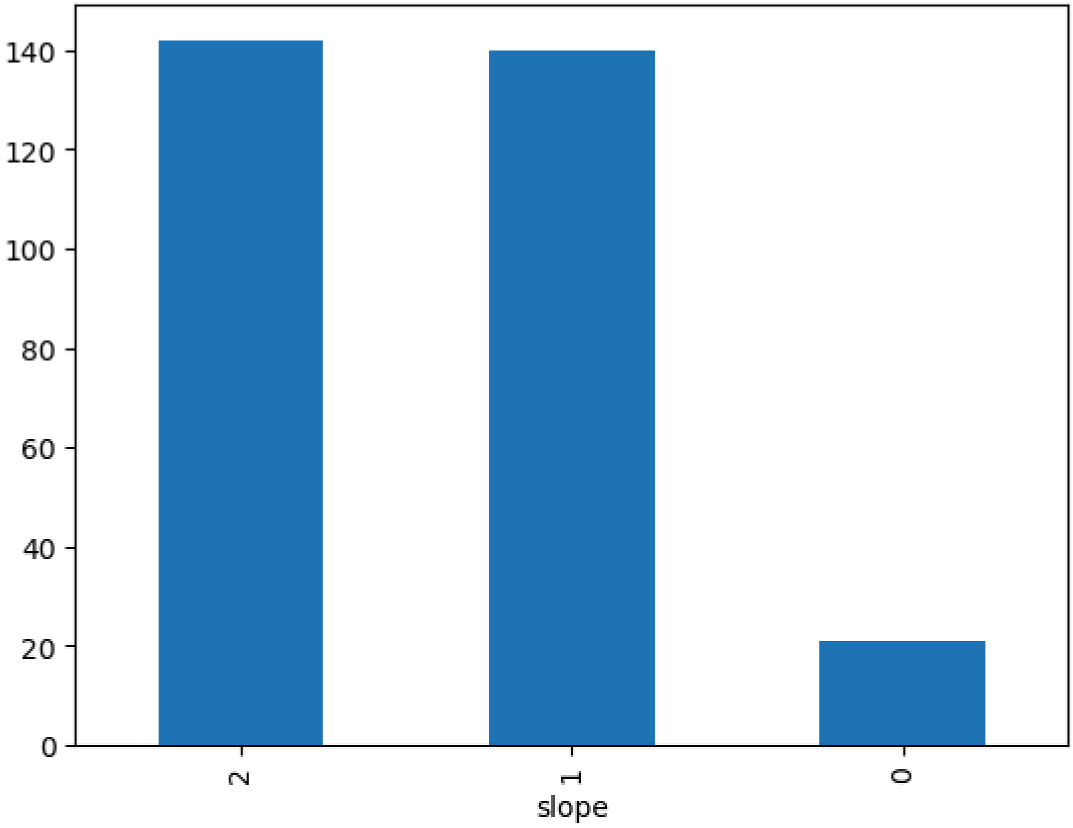


```
In [24]: sns.countplot(x=data['exang'],hue='target',data=data)
         plt.show()
```



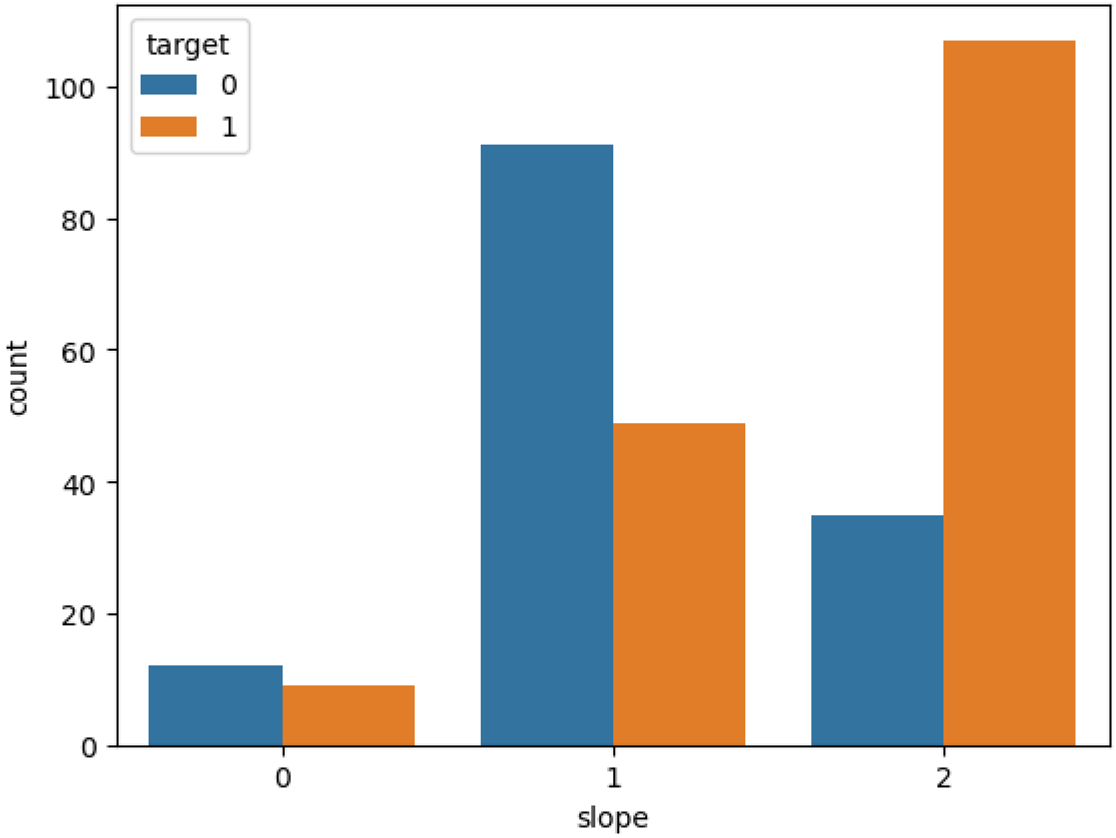
```
In [25]: data['slope'].value_counts().plot(kind='bar')
```

```
Out[25]: <Axes: xlabel='slope'>
```



In [26]:

```
sns.countplot(x=data['slope'],hue='target',data=data)
plt.show()
```

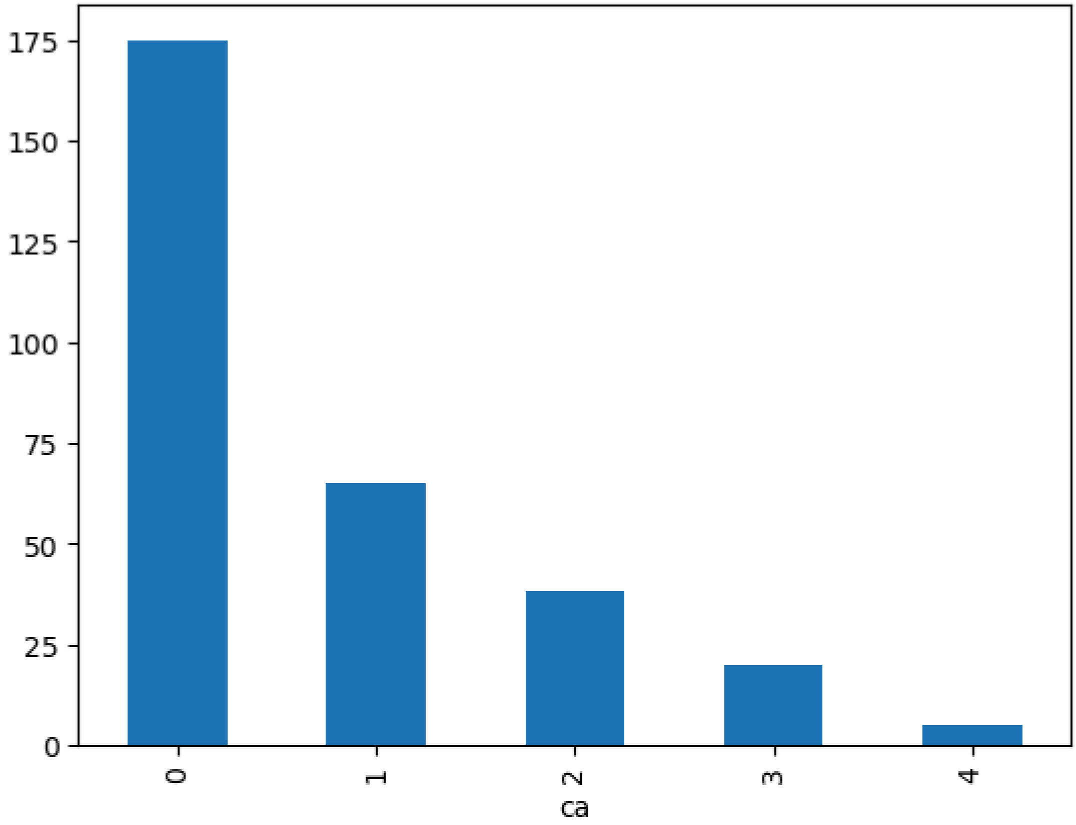


In [27]:

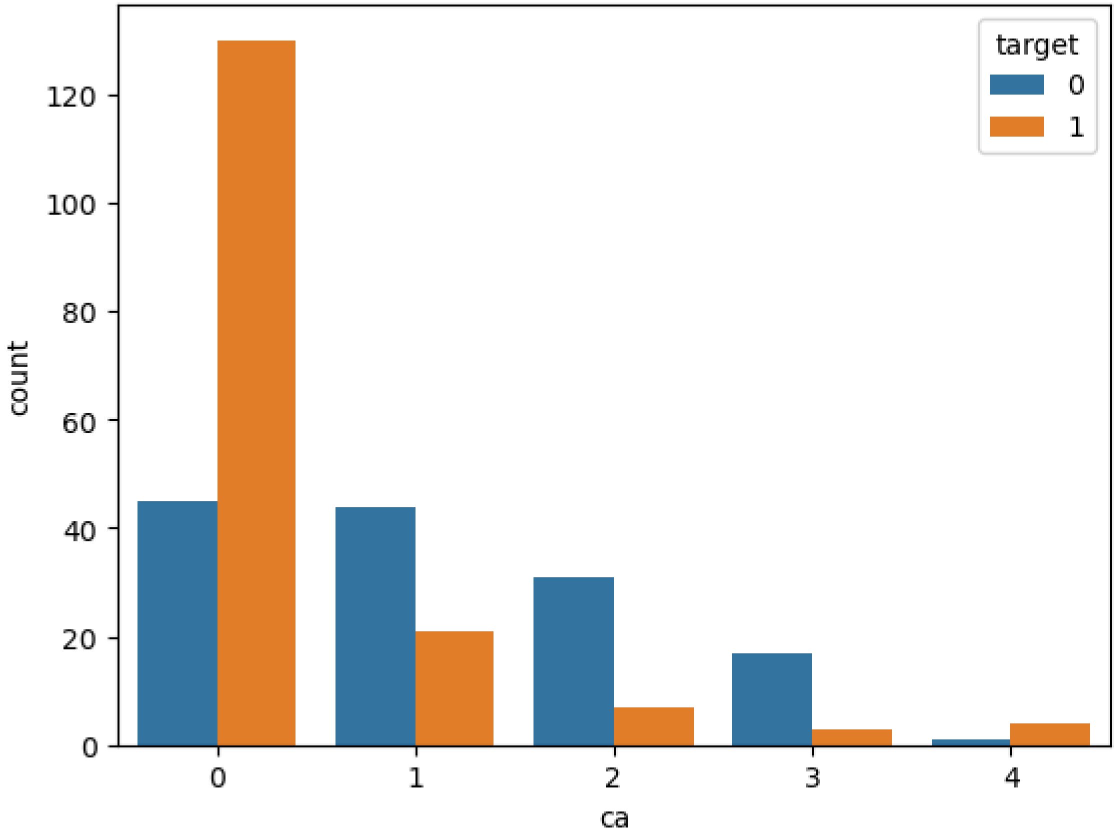
```
data['ca'].value_counts().plot(kind='bar')
```

Out[27]:

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

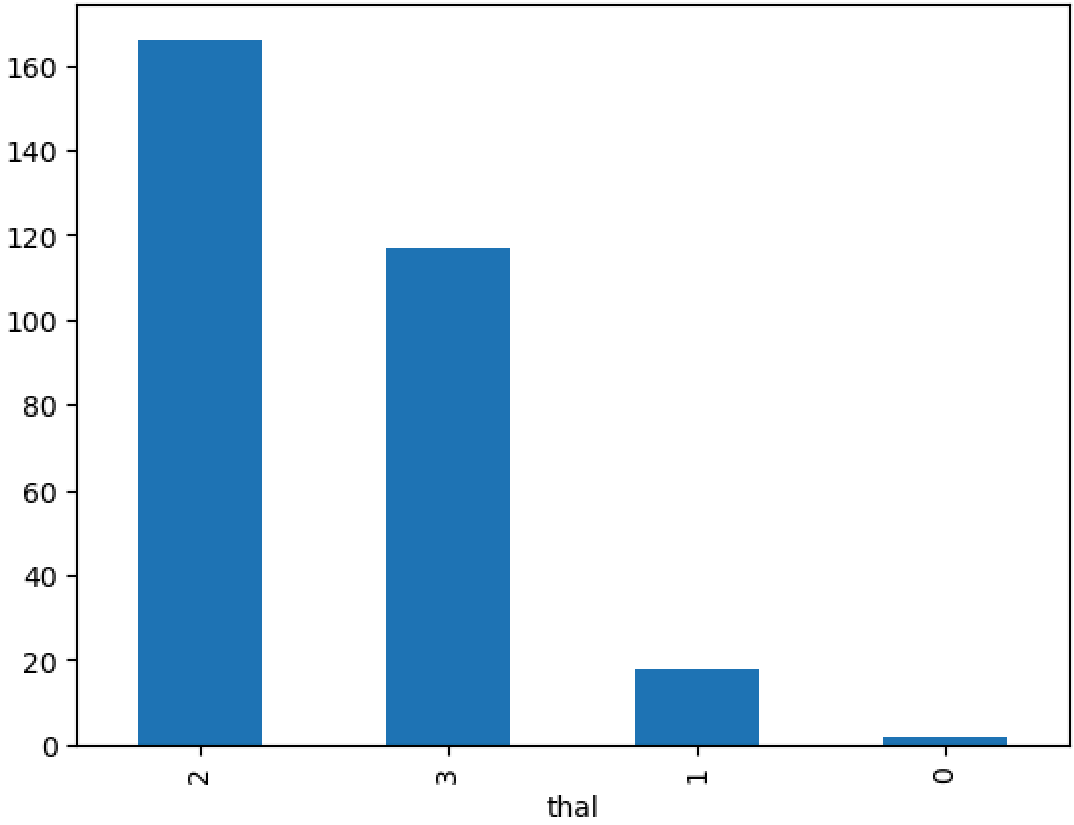


In [28]:
```
sns.countplot(x=data['ca'],hue='target',data=data)
plt.show()
```

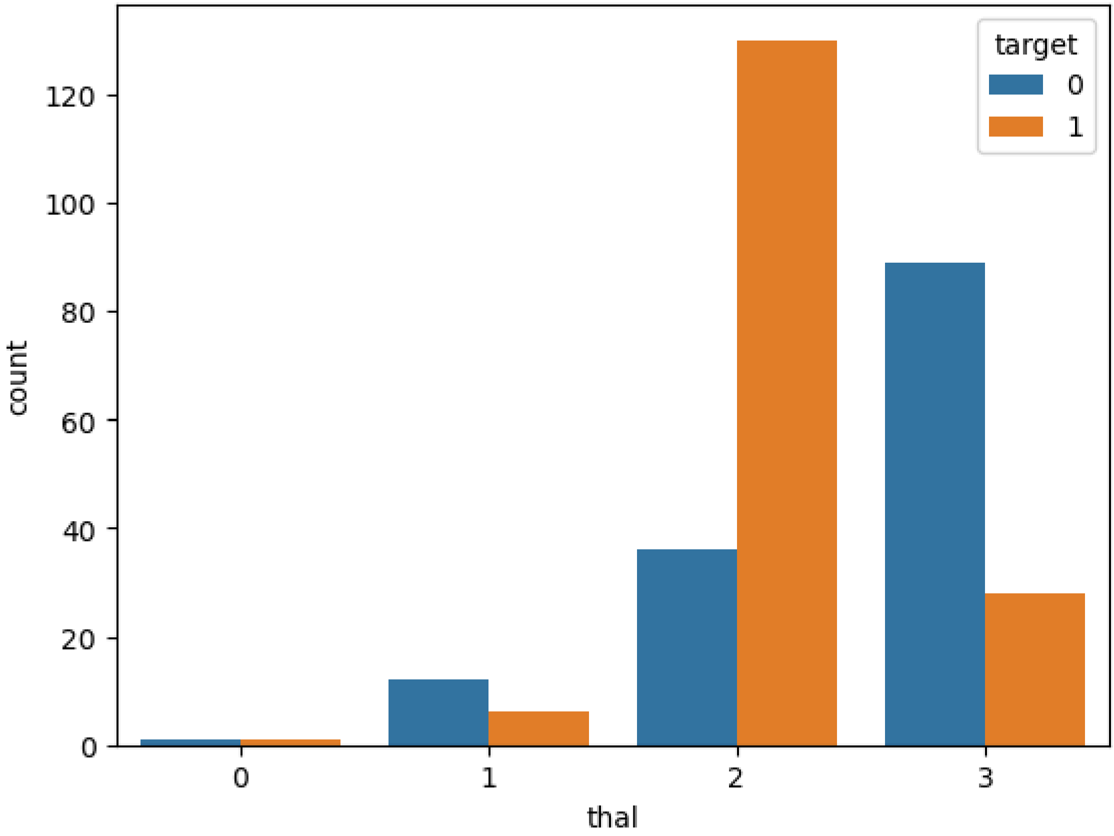


In [29]:
```
data['thal'].value_counts().plot(kind='bar')
```

Out[29]:
```
<Axes: xlabel='thal'>
```

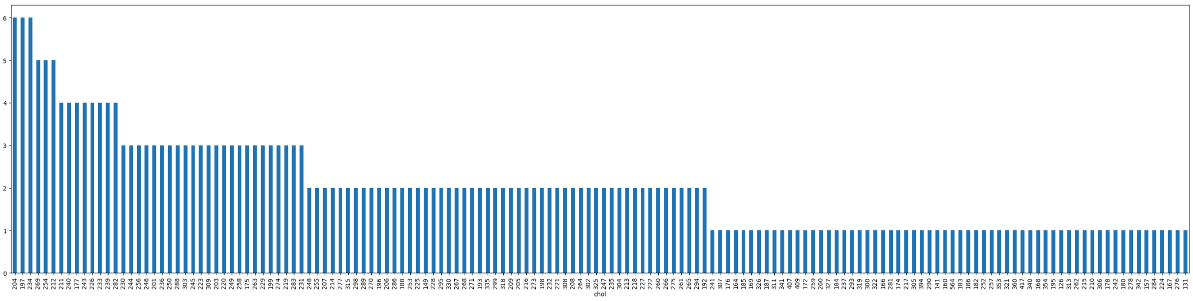


In [30]:
```
sns.countplot(x=data['thal'],hue='target',data=data)
plt.show()
```



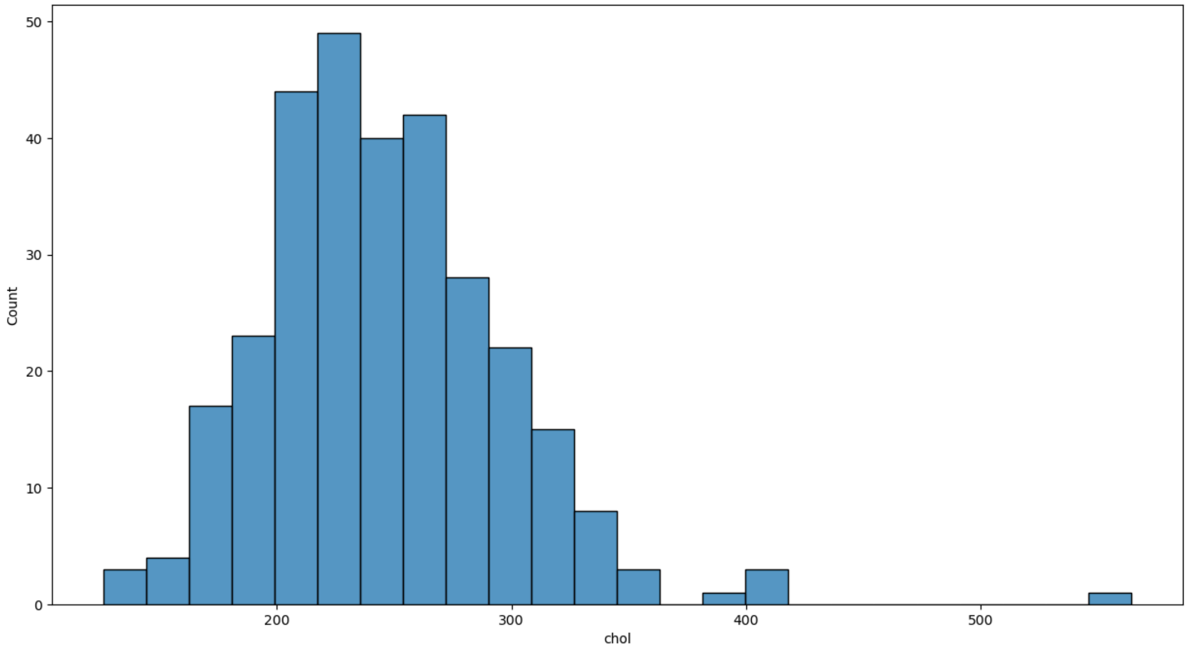
In [34]:
```
plt.figure(figsize=(35,8))
data['chol'].value_counts().plot(kind='bar')
```

Out[34]:
```
<Axes: xlabel='chol'>
```



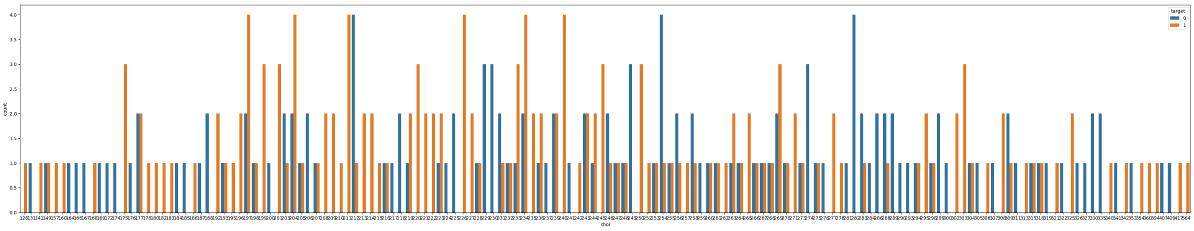
In [36]:

```
plt.figure(figsize=(15,8))
sns.histplot(x=data['chol'],data=data)
plt.show()
```



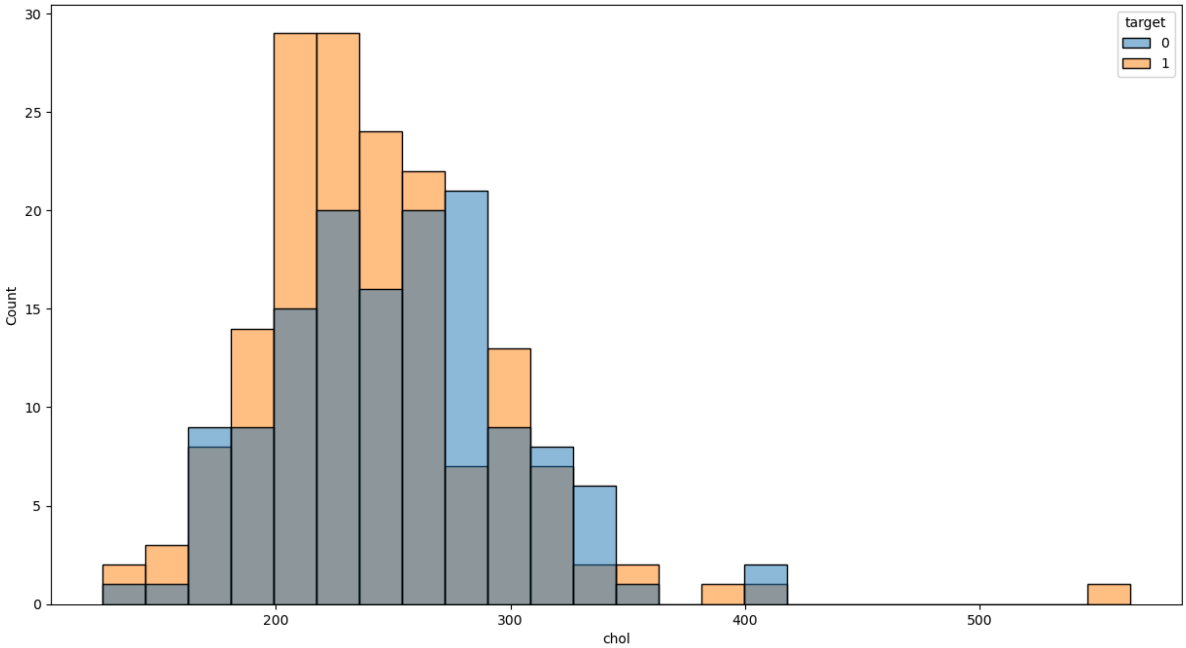
In [37]:

```
plt.figure(figsize=(45,8))
sns.countplot(x=data['chol'],hue='target',data=data)
plt.show()
```



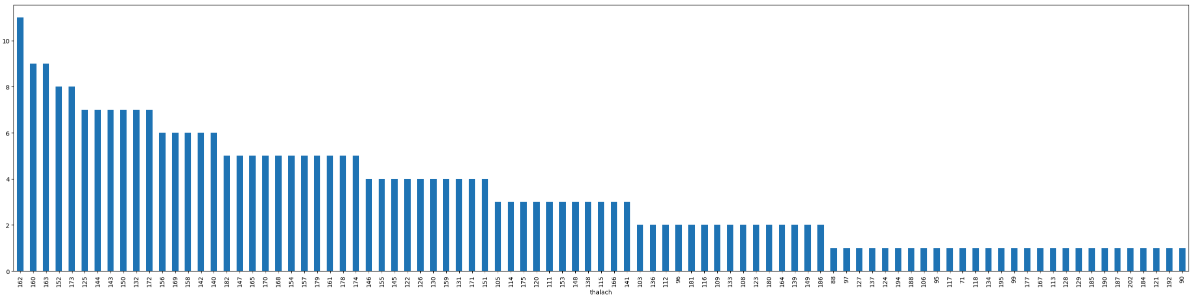
In [38]:

```
plt.figure(figsize=(15,8))
sns.histplot(x=data['chol'],hue='target',data=data)
plt.show()
```

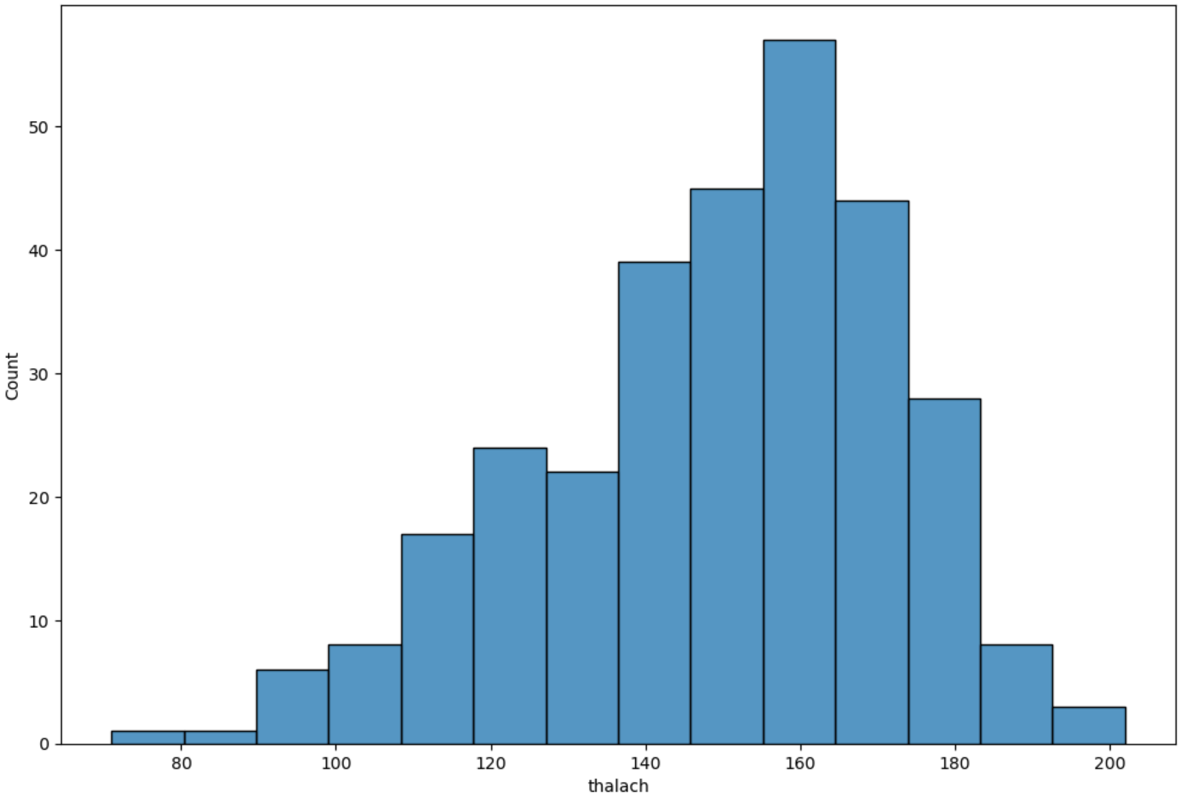


In [39]:
```
plt.figure(figsize=(35,8))
data['thalach'].value_counts().plot(kind='bar')
```

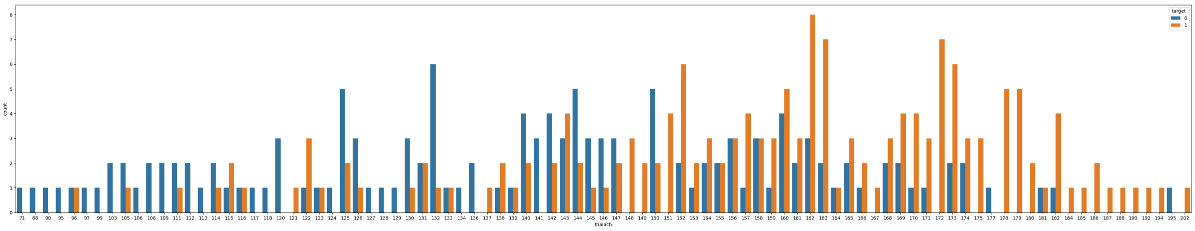
Out[39]:
```
<Axes: xlabel='thalach'>
```



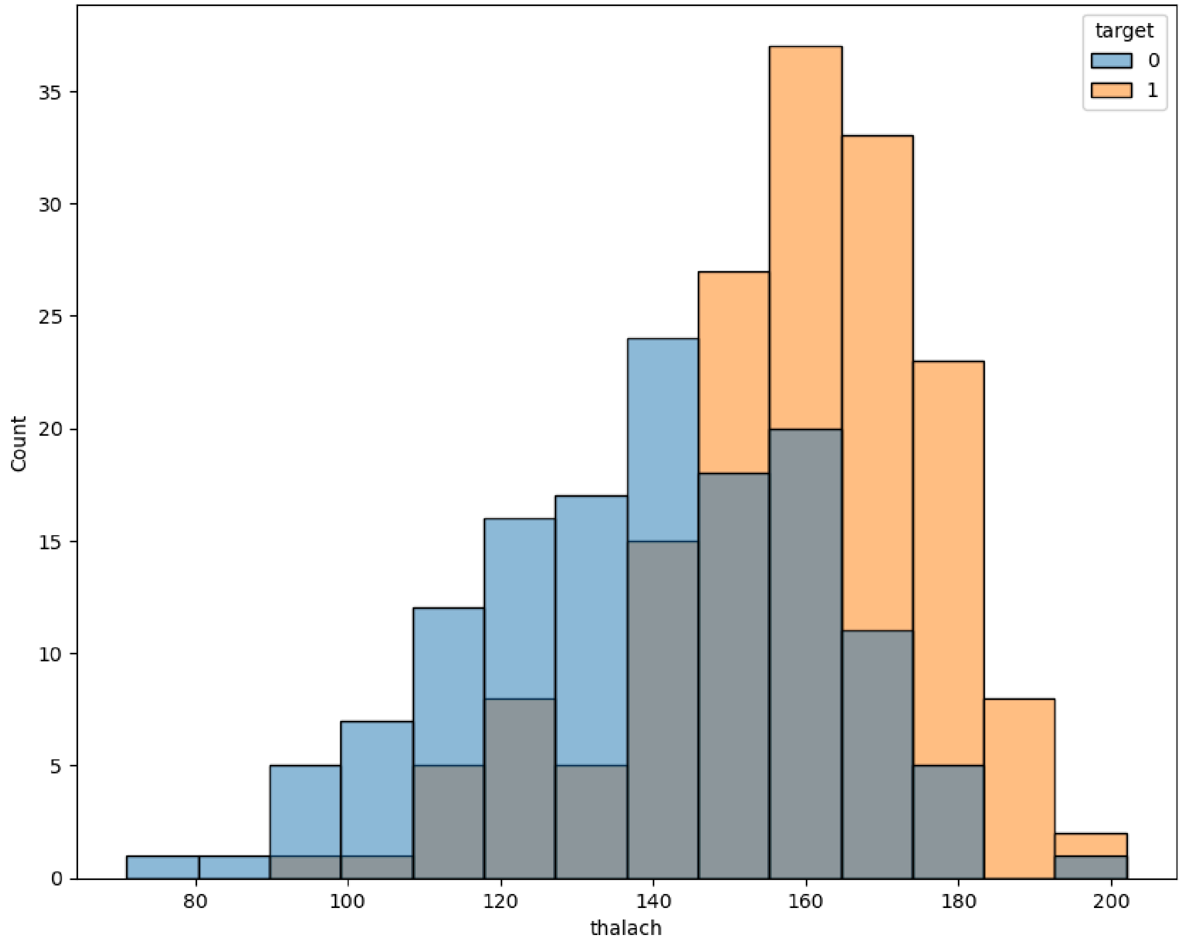
In [41]:
```
plt.figure(figsize=(12,8))
sns.histplot(x=data['thalach'],data=data)
plt.show()
```



In [42]:
```
plt.figure(figsize=(45,8))
sns.countplot(x=data['thalach'],hue='target',data=data)
plt.show()
```

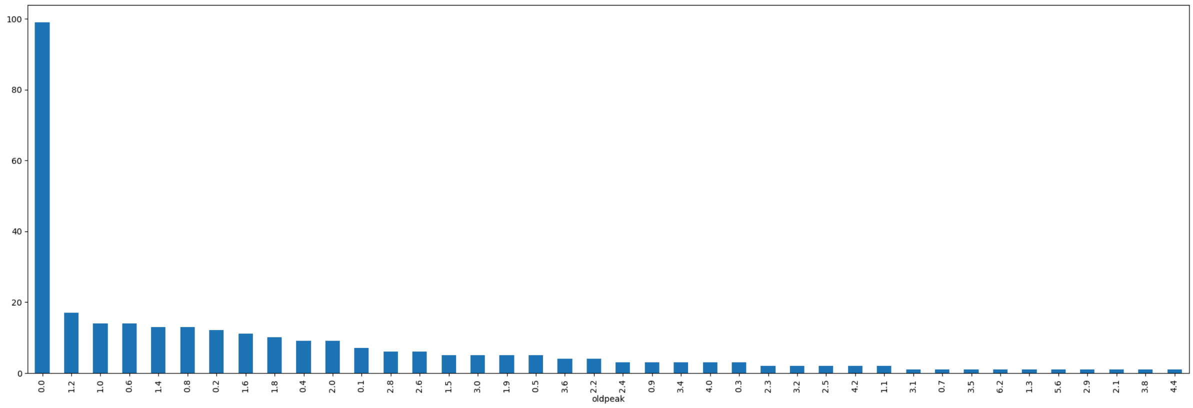


In [44]:
```
plt.figure(figsize=(10,8))
sns.histplot(x=data['thalach'],hue='target',data=data)
plt.show()
```

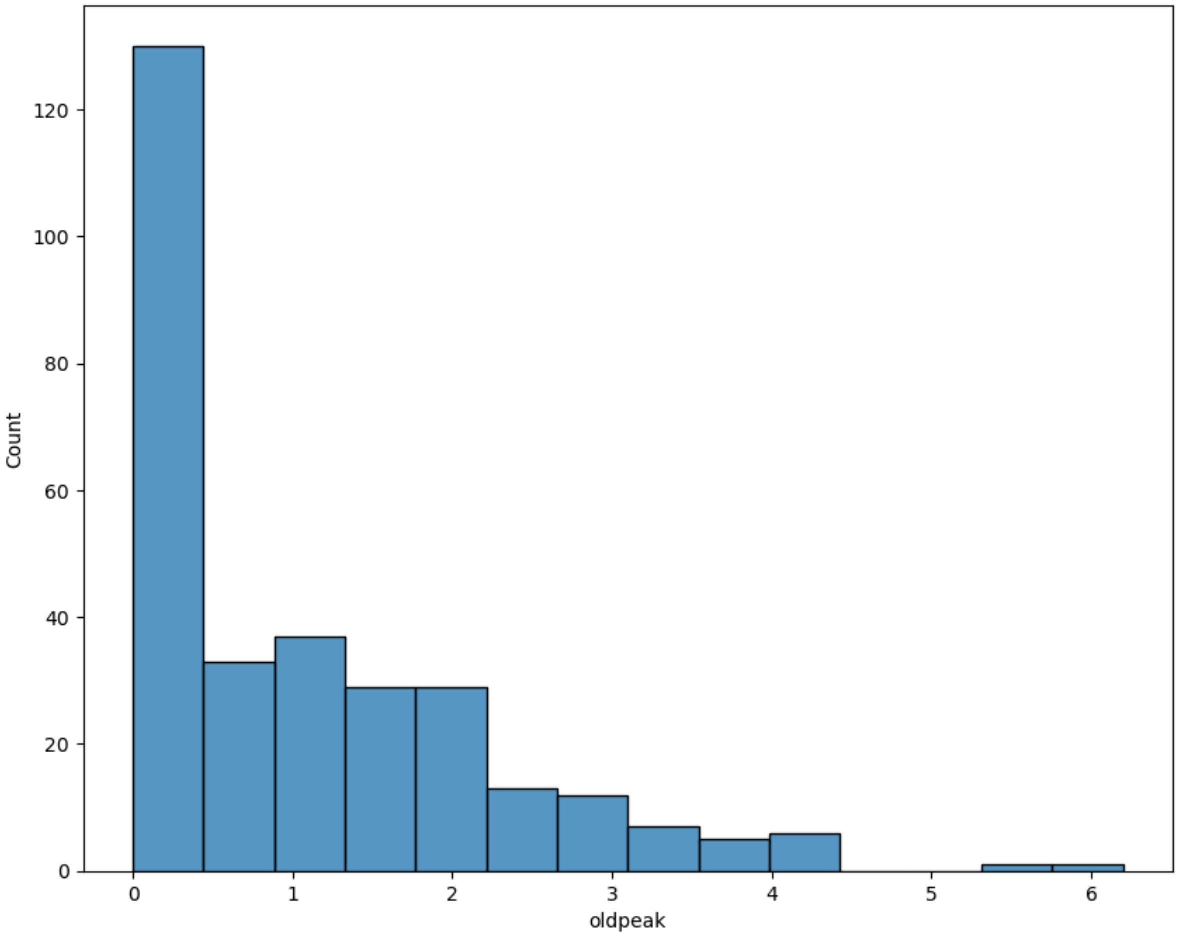


```
In [46]: #
         plt.figure(figsize=(25,8))
         data['oldpeak'].value_counts().plot(kind='bar')
```

```
Out[46]: <Axes: xlabel='oldpeak'>
```

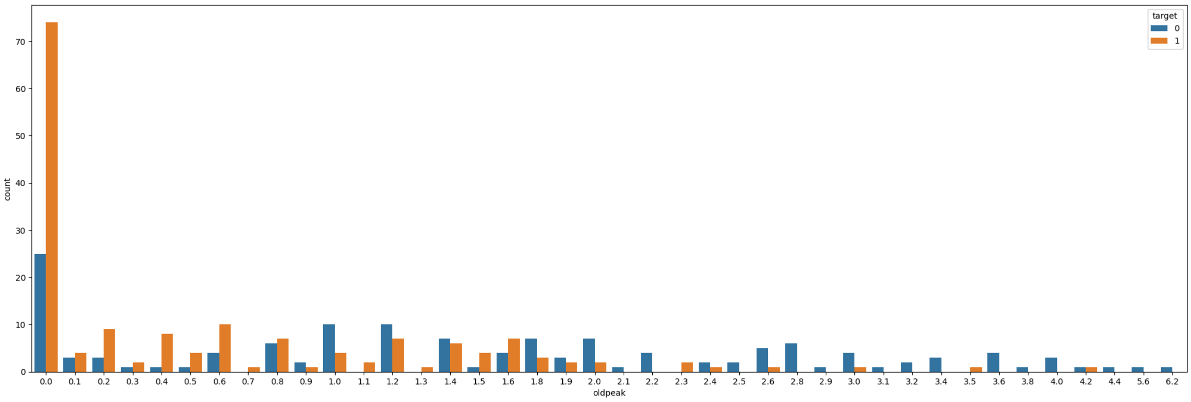


```
In [47]: plt.figure(figsize=(10,8))
         sns.histplot(x=data['oldpeak'],data=data)
         plt.show()
```



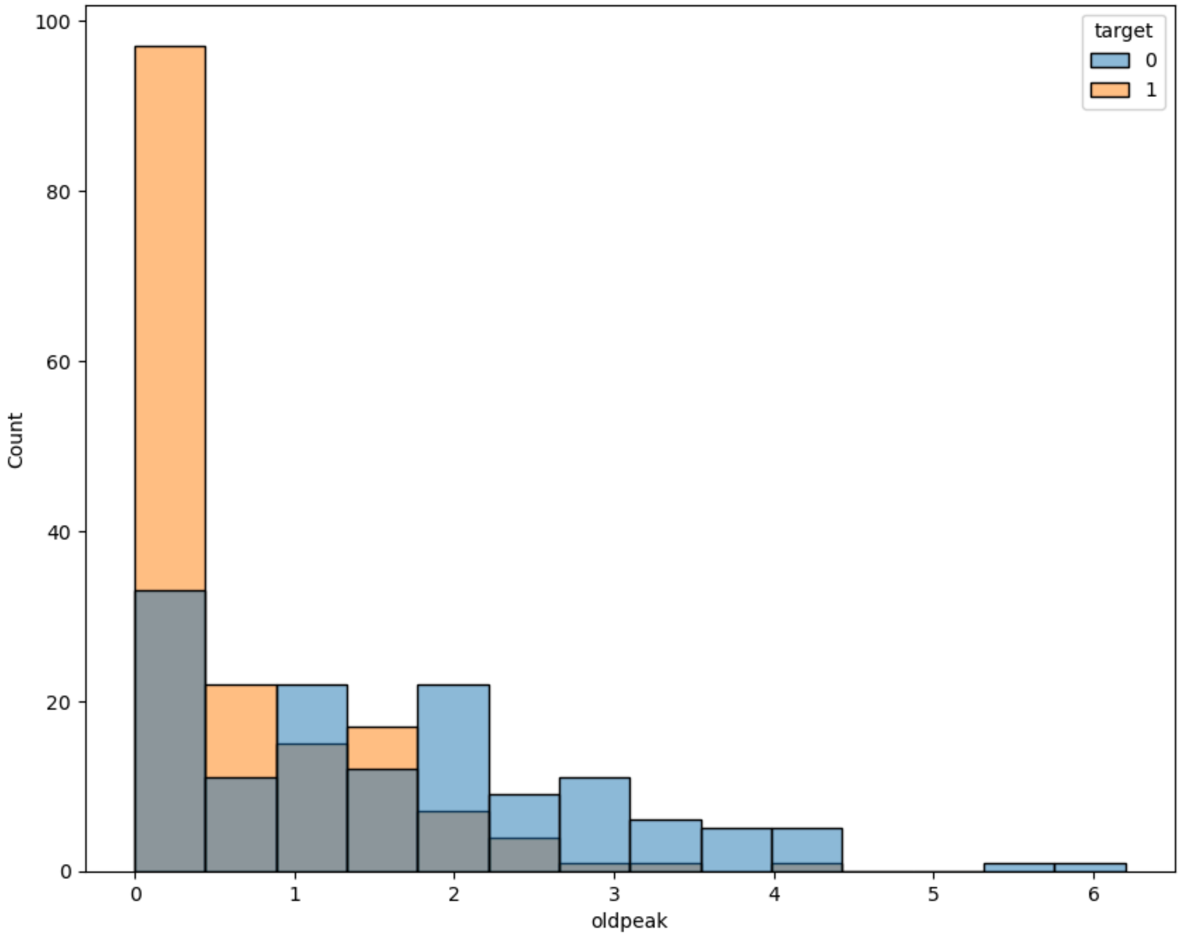
In [48]:

```
plt.figure(figsize=(25,8))
sns.countplot(x=data['oldpeak'],hue='target',data=data)
plt.show()
```



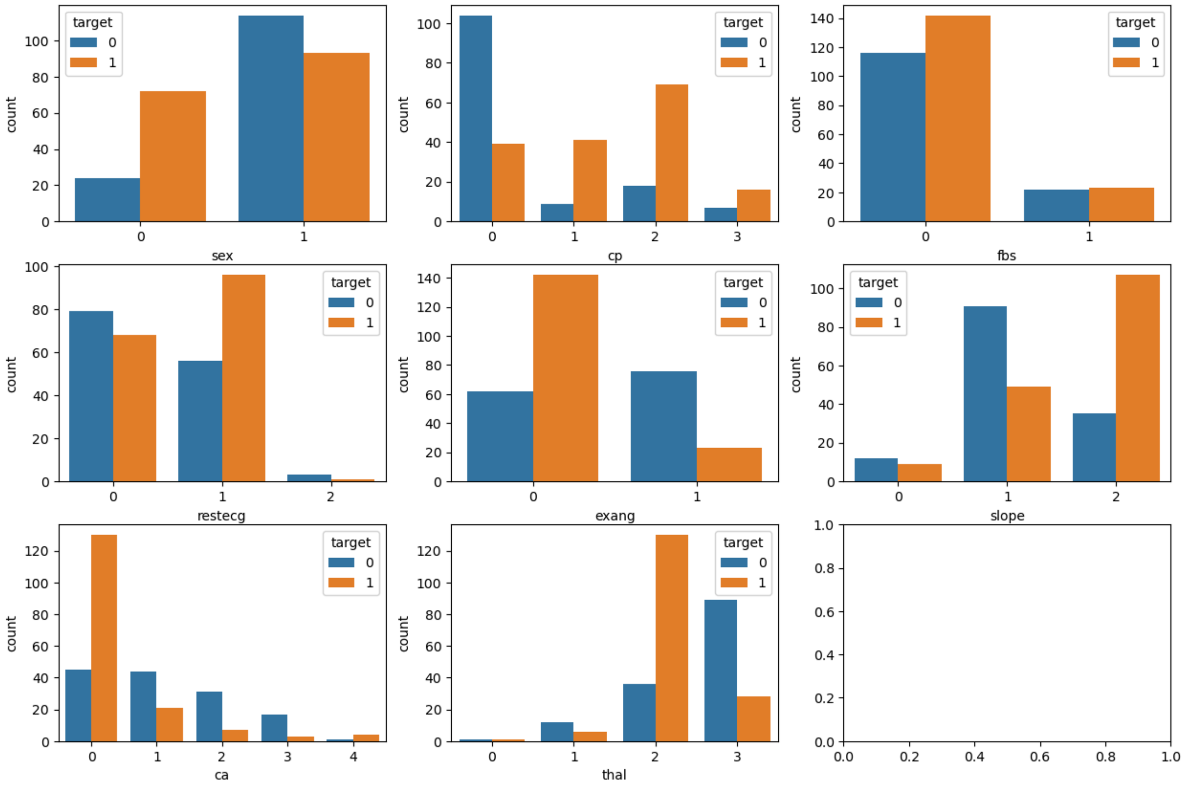
In [49]:

```
plt.figure(figsize=(10,8))
sns.histplot(x=data['oldpeak'],hue='target',data=data)
plt.show()
```



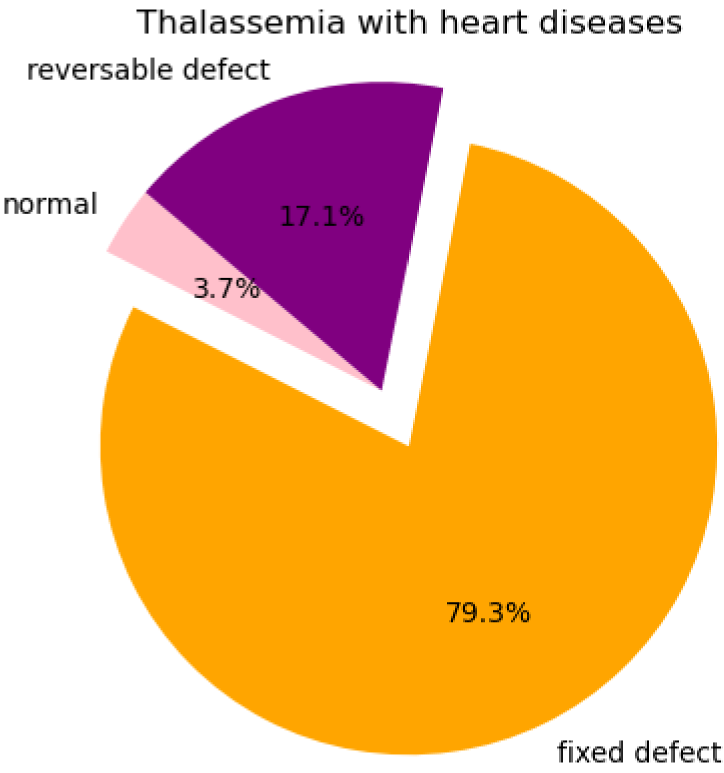
In [57]:

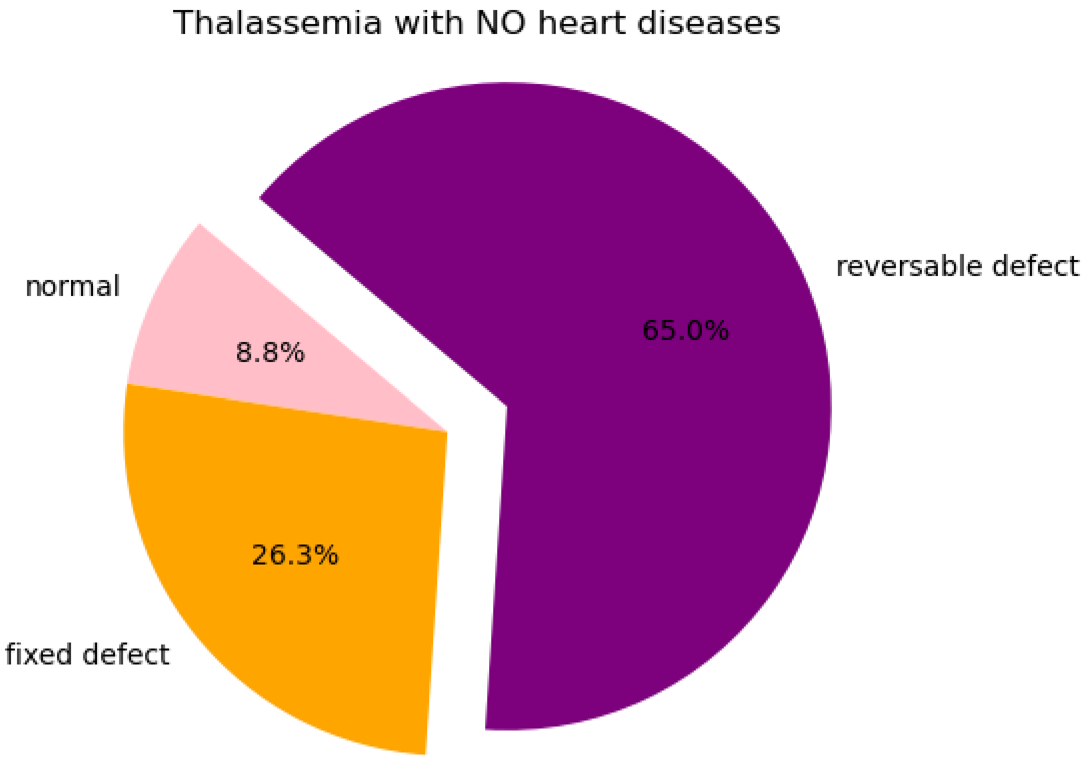
```
fig,axes=plt.subplots(nrows=3,ncols=3,figsize=(15,10))
cat_features=['sex','cp','fbs','restecg','exang','slope','ca','thal','target']
for idx,feature in enumerate(cat_features):
    if feature!='target':
        ax=axes[int(idx/3),idx%3]
        sns.countplot(x=feature,hue='target',ax=ax,data=data)
```



In [69]:

```
#create a pie chart
labels='normal','fixed defect','reversable defect'
sizes=[6,130,28]
explode=[0,0.2,0]
colors=['pink','orange','purple']
plt.pie(sizes,labels=labels,autopct='%.1f%%',explode=explode,colors=colors,startang
plt.axis('equal')
plt.title('Thalassemia with heart diseases')
plt.show()
labels='normal','fixed defect','reversable defect'
sizes=[12,36,89]
explode=[0,0,0.2]
colors=['pink','orange','purple']
plt.pie(sizes,labels=labels,autopct='%.1f%%',explode=explode,colors=colors,startang
plt.axis('equal')
plt.title('Thalassemia with NO heart diseases')
plt.show()
```





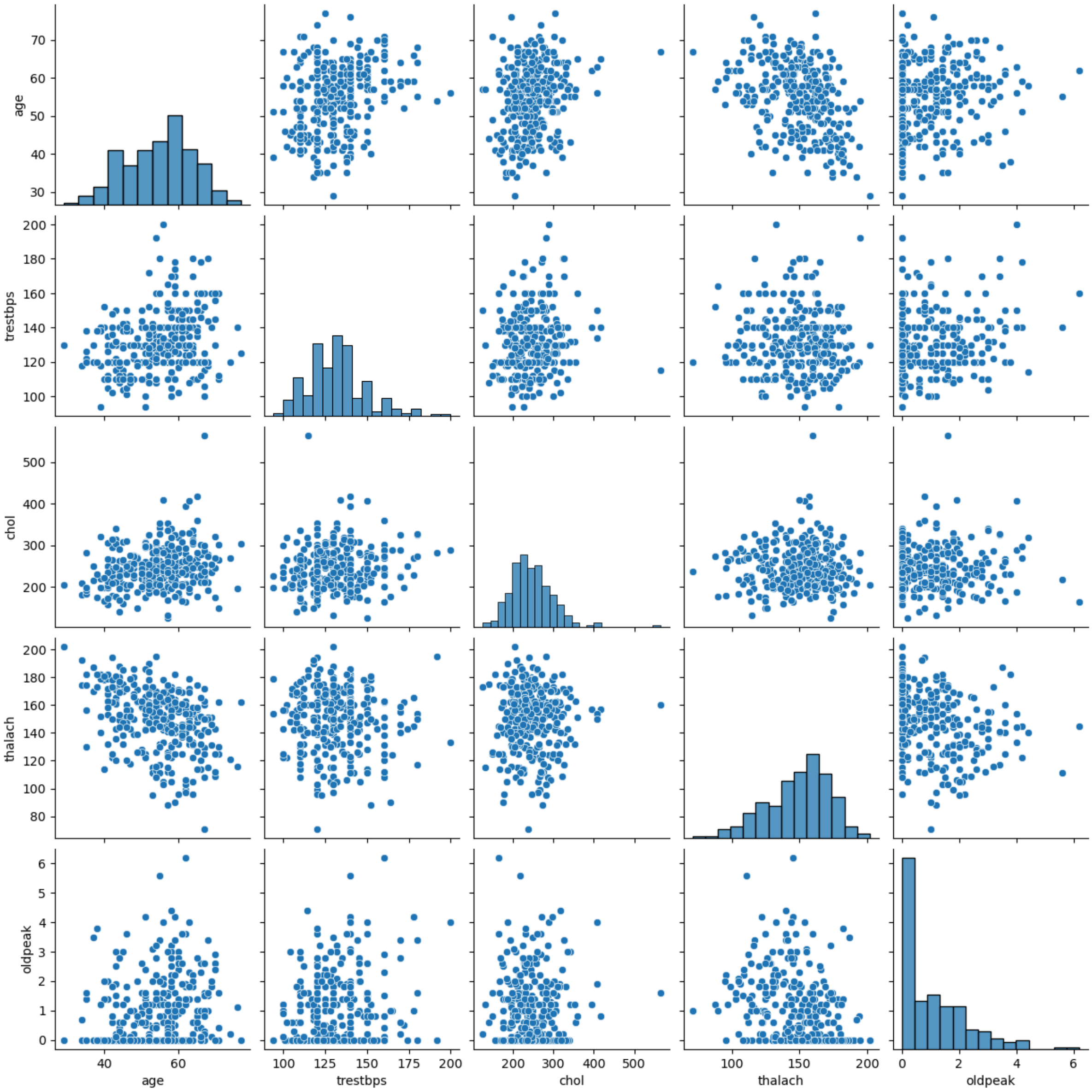
```
In [71]: #Distribution of continuous variable
         data.nunique()
```

```
Out[71]: 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
```

```
In [72]: num_var=['age','trestbps','chol','thalach','oldpeak']
         sns.pairplot(data[num_var])
```

```
C:\Users\SHONIMA\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning:
The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

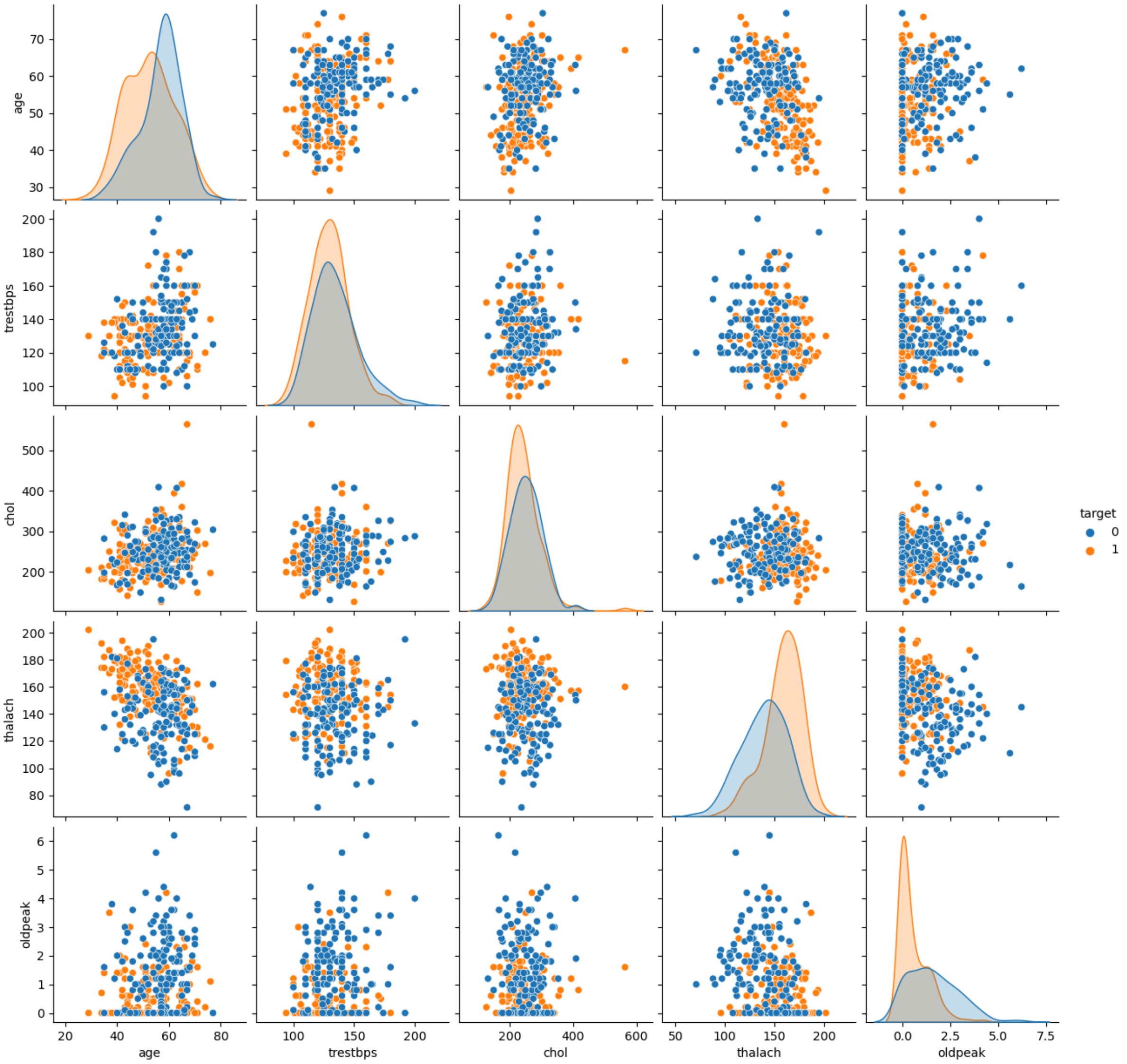
```
Out[72]: <seaborn.axisgrid.PairGrid at 0x1f6659b9190>
```



In [73]:

```
num_var=['age','trestbps','chol','thalach','oldpeak']
sns.pairplot(data[num_var+['target']],hue='target')
plt.show()
```

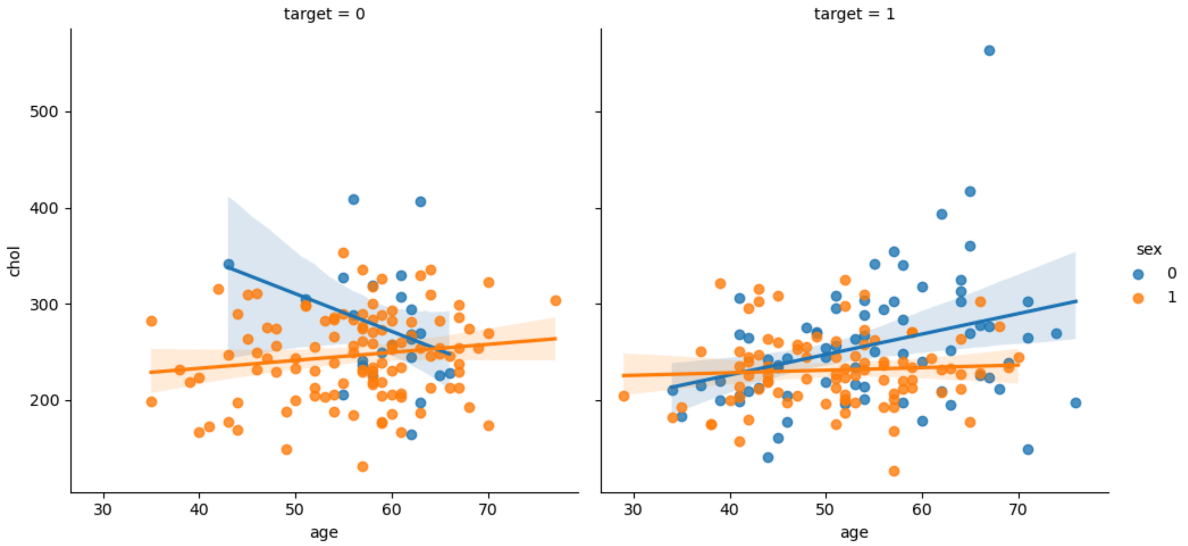
```
C:\Users\SHONIMA\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning:
The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```



In [74]:

```
#create a plot to understand relationship between age & chol acoording to target
sns.lmplot(x='age',y='chol',hue='sex',col='target',data=data)
plt.show()
```

```
C:\Users\SHONIMA\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning:
The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```



In [75]:

```
data.corr()
```

Out[75]:

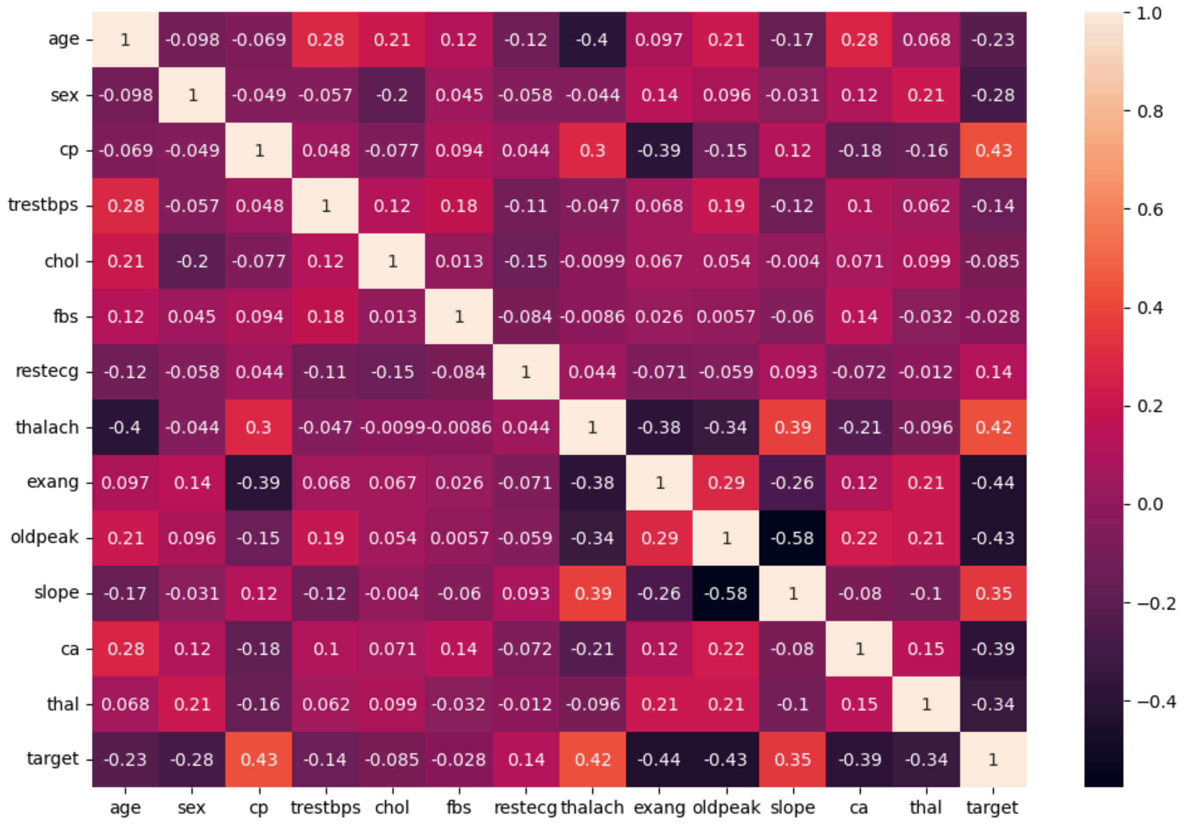
	age	sex	cp	trestbps	chol	fbs	restecg	thalach	
age	1.000000	-0.098447	-0.068653	0.279351	0.213678	0.121308	-0.116211	-0.398522	0.0
sex	-0.098447	1.000000	-0.049353	-0.056769	-0.197912	0.045032	-0.058196	-0.044020	0.1
cp	-0.068653	-0.049353	1.000000	0.047608	-0.076904	0.094444	0.044421	0.295762	-0.3
trestbps	0.279351	-0.056769	0.047608	1.000000	0.123174	0.177531	-0.114103	-0.046698	0.0
chol	0.213678	-0.197912	-0.076904	0.123174	1.000000	0.013294	-0.151040	-0.009940	0.0
fbs	0.121308	0.045032	0.094444	0.177531	0.013294	1.000000	-0.084189	-0.008567	0.0
restecg	-0.116211	-0.058196	0.044421	-0.114103	-0.151040	-0.084189	1.000000	0.044123	-0.0
thalach	-0.398522	-0.044020	0.295762	-0.046698	-0.009940	-0.008567	0.044123	1.000000	-0.3
exang	0.096801	0.141664	-0.394280	0.067616	0.067023	0.025665	-0.070733	-0.378812	1.0
oldpeak	0.210013	0.096093	-0.149230	0.193216	0.053952	0.005747	-0.058770	-0.344187	0.2
slope	-0.168814	-0.030711	0.119717	-0.121475	-0.004038	-0.059894	0.093045	0.386784	-0.2
ca	0.276326	0.118261	-0.181053	0.101389	0.070511	0.137979	-0.072042	-0.213177	0.1
thal	0.068001	0.210041	-0.161736	0.062210	0.098803	-0.032019	-0.011981	-0.096439	0.2
target	-0.225439	-0.280937	0.433798	-0.144931	-0.085239	-0.028046	0.137230	0.421741	-0.4

In [79]:

```
#Heatmap
plt.figure(figsize=(12,8))
sns.heatmap(data.corr(),annot=True)
```

Out[79]:

```
<Axes: >
```



In [86]:

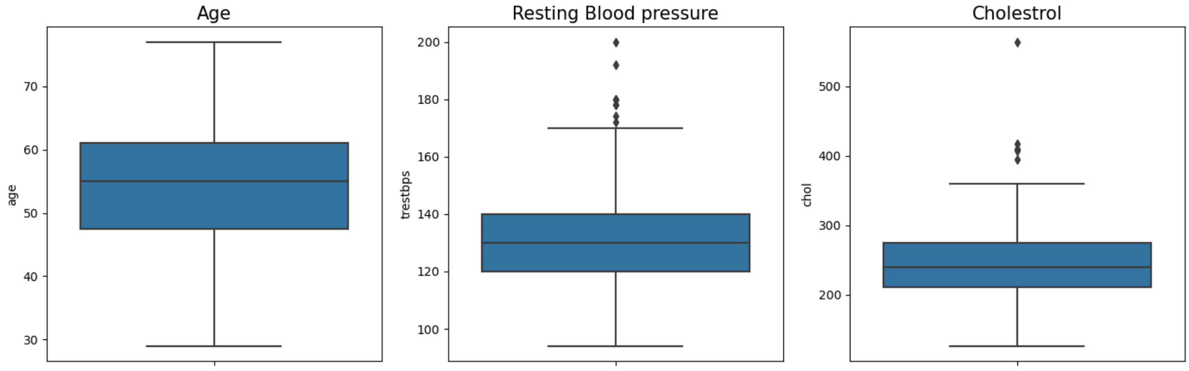
```
#outliers
plt.figure(figsize=(17,5))
plt.subplot(1,3,1)
```

```
sns.boxplot(y=data['age'])
plt.title('Age',fontsize=15)

plt.subplot(1,3,2)
sns.boxplot(y=data['trestbps'])
plt.title('Resting Blood pressure',fontsize=15)

plt.subplot(1,3,3)
sns.boxplot(y=data['chol'])
plt.title('Cholestrol',fontsize=15)


plt.show()
```

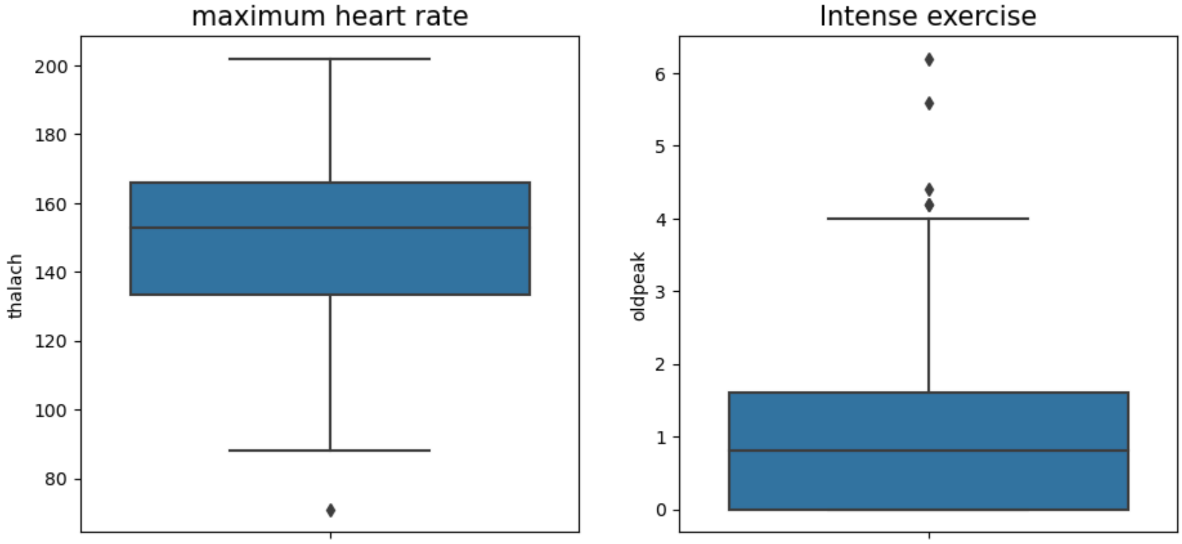


In [89]:

```
plt.figure(figsize=(17,5))
plt.subplot(1,3,1)
sns.boxplot(y=data['thalach'])
plt.title('maximum heart rate',fontsize=15)

plt.subplot(1,3,2)
sns.boxplot(y=data['oldpeak'])
plt.title('Intense exercise',fontsize=15)

plt.show()
```



low=Q1-1.5*IQR High=Q3+1.5*IQR

In [90]:

```
#machine learning
#seperate Independent & dependent Var
#create train set & test set
#Scaling
# Applying classification a;gorithm
#Evaluate the model
```

In [91]:

```
data.head()
```

Out[91]:

	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 [93]:

```
#seperate Independent & dependent Var
X=data.drop(['target'],axis=1)
y=data['target']
```

In [97]:

```
#create train set & test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.30, random_
```

In [98]:

```
X_train.shape

y_train
#y_test
```

Out[98]:

```
(212, 13)
```

In [99]:

```
X_test.shape
```

Out[99]:

```
(91, 13)
```

In [102…

```
X_train.head()
```

Out[102]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal
124	39	0	2	94	199	0	1	179	0	0.0	2	0	2
72	29	1	1	130	204	0	0	202	0	0.0	2	0	2
15	50	0	2	120	219	0	1	158	0	1.6	1	0	2
10	54	1	0	140	239	0	1	160	0	1.2	2	0	2
163	38	1	2	138	175	0	1	173	0	0.0	2	4	2

In [103…

```
#normalization/scale your data
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
```

In [104…

```
X_train=sc.fit_transform(X_train)
```

```
X_test=sc.transform(X_test)
```

In [106…

```
#create the model
from sklearn.linear_model import LogisticRegression
log_red=LogisticRegression()
```

In [107…

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

Out[107]:

```
▾ LogisticRegression
LogisticRegression()
```

In [108…

```
y_pred=log_red.predict(X_test)
```

In [109…

```
y_pred
```

Out[109]:

```
array([0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0,
       0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1,
       1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0,
       1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1,
       1, 0, 1], dtype=int64)
```

In [111…

```
#evaluate machine learning model
from sklearn.metrics import confusion_matrix

print(confusion_matrix(y_pred,y_test))
```

```
[[32  8]
 [ 9 42]]
```

In [113…

```
(32+42)/(32+8+9+42)
```

Out[113]:

```
0.8131868131868132
```

In [112…

```
#print model accuracy & classfication report
from sklearn.metrics import accuracy_score,classification_report
print(accuracy_score(y_pred,y_test))
```

```
0.8131868131868132
```

In [114…

```
print(classification_report(y_pred,y_test))
```

```
              precision    recall  f1-score   support

           0       0.78      0.80      0.79        40
           1       0.84      0.82      0.83        51

    accuracy                           0.81        91
   macro avg       0.81      0.81      0.81        91
weighted avg       0.81      0.81      0.81        91
```

pred actu 0 1 0 TN FP 1 FN TP precision= TP/TP+FP recall=TP/TP+FN

In [ ]: