Exploratory Data Analysis on Heart Disease Dataset.

Overview of Heart Disease

- **Heart disease** or **Cardiovascular disease (CVD)** is a class of diseases that involve the heart or blood vessels.
- Cardiovascular diseases are the leading cause of death globally. This is true in all areas of the world except Africa.
- CVD resulted in 17.9 million deaths (32.1%) in 2015.
- Deaths, at a given age, from CVD are more common and have been increasing in much of the developing world, while rates have declined in most of the developed world since the 1970s.

What we did in this project?

- In this project, I have conducted **Exploratory Data Analysis(EDA)** of the heart disease dataset.
- **Exploratory Data Analysis(EDA)** is a critical first step in analyzing a new dataset.
- The primary objective of EDA is to analyze the data for distribution, outliers and anomalies in the dataset.
- It enable us to direct specific testing of the hypothesis. It includes analysing the data to find the distribution of data, its main characteristics, identifying patterns and visualizations.
- It also provides tools for hypothesis generation by visualizing and understanding the data through graphical representation.

Q.What is difference between Anomaly and Outlier?

Ans---> "Outlier" and "anomaly" are related concepts but are not exactly the same.

An **'outlier'** is a data point that is significantly different from the other data points in a dataset. In statistics, outliers are often defined as values that fall outside a certain range or have extreme values that are distant from the mean (average) of the data. Outliers can be genuine data points that represent unusual or rare events, errors in data collection, or simply noise in the data.

An **anomaly**, on the other hand, is a broader term that refers to any observation or event that deviates from what is expected or considered normal. Anomalies can be outliers, but they can also be other types of unexpected or irregular events. Anomalies are often context-specific, meaning that what is considered an anomaly in one situation may not be an anomaly in another.**

1. Introduction to EDA

Several questions come to mind when we come across a new dataset. The below list shed light on some of these questions:-

• What is the distribution of the dataset?

• Are there any missing numerical values, outliers or anomalies in the dataset?

• What are the underlying assumptions in the dataset?

• Whether there exists relationships between variables in the dataset?

• How to be sure that our dataset is ready for input in a machine learning algorithm?

• How to select the most suitable algorithm for a given dataset?

So, how do we get answer to the above questions?

The answer is **Exploratory Data Analysis**. It enable us to answer all of the above questions.

2. Objective of EDA

The objectives of the EDA are as follows:-

i. To get an overview of the distribution of the dataset.

ii. Check for missing numerical values, outliers or other anomalies in the dataset.

iii.Discover patterns and relationships between variables in the dataset.

iv. Check the underlying assumptions in the dataset.

3. Types of EDA

EDA is generally cross-classified in two ways. First, each method is either non-graphical or graphical. Second, each method is either univariate or multivariate (usually bivariate). The non-graphical methods provide insight into the characteristics and the distribution of the variable(s) of interest. So, non-graphical methods involve calculation of summary statistics while graphical methods include summarizing the data diagrammatically.

There are four types of exploratory data analysis (EDA) based on the above cross-classification methods. Each of these types of EDA are described below:-

i. Univariate non-graphical EDA

The objective of the univariate non-graphical EDA is to understand the sample distribution and also to make some initial conclusions about population distributions. Outlier detection is also a part of this analysis.

ii. Multivariate non-graphical EDA

Multivariate non-graphical EDA techniques show the relationship between two or more variables in the form of either cross-tabulation or statistics.

iii. Univariate graphical EDA

In addition to finding the various sample statistics of univariate distribution (discussed above), we also look graphically at the distribution of the sample. The non-graphical methods are quantitative and objective. They do not give full picture of the data. Hence, we need graphical methods, which are more qualitative in nature and presents an overview of the data.

iv. Multivariate graphical EDA

There are several useful multivariate graphical EDA techniques, which are used to look at the distribution of multivariate data. These are as follows:-

- Side-by-Side Boxplots
- Scatterplots
- Heat Maps and 3-D Surface Plots

4. Import necessary libraries

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

import warnings
warnings.filterwarnings('ignore')

%matplotlib inline
```

5. Load the dataset

In [2]:
```
heart=pd.read_csv(r"D:\EDA\heart.csv")
heart
```

Out[2]:

	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
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
298	57	0	0	140	241	0	1	123	1	0.2	1	0	3	0
299	45	1	3	110	264	0	1	132	0	1.2	1	0	3	0
300	68	1	0	144	193	1	1	141	0	3.4	1	2	3	0
301	57	1	0	130	131	0	1	115	1	1.2	1	1	3	0
302	57	0	1	130	236	0	0	174	0	0.0	1	1	2	0

303 rows × 14 columns

Dataset Description

The dataset contains several columns which are as follows -

- age : age in years
- sex : (1 = male; 0 = female)
- cp : chest pain type
- trestbps : resting blood pressure (in mm Hg on admission to the hospital)
- chol : serum cholestoral in mg/dl
- fbs : (fasting blood sugar > 120 mg/dl) (1 = true; 0 = false)
- restecg : resting electrocardiographic results
- thalach : maximum heart rate achieved
- exang : exercise induced angina (1 = yes; 0 = no)
- oldpeak : ST depression induced by exercise relative to rest
- slope : the slope of the peak exercise ST segment
- ca : number of major vessels (0-3) colored by flourosopy
- thal : 3 = normal; 6 = fixed defect; 7 = reversable defect
- target : 1 or 0

6. Exploratory Data Analysis

Check the shape of dataset

In [3]:
```
1 heart.shape
```

Out[3]:
```
(303, 14)
```

There are 303 instances and 14 variables in our dataset.

Summary of dataset

In [4]:
```
1 heart.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
```

Important points about dataset

- `sex` is a character variable. Its data type should be object. But it is encoded as (1 = male; 0 = female). So, its data type is given as int64.
- Same is the case with several other variables - `fbs` , `exang` and `target` .
- `fbs (fasting blood sugar)` should be a character variable as it contains only 0 and 1 as values (1 = true; 0 = false). As it contains only 0 and 1 as values, so its data type is given as int64.
- `exang (exercise induced angina)` should also be a character variable as it contains only 0 and 1 as values (1 = yes; 0 = no). It also contains only 0 and 1 as values, so its data type is given as int64.

- `target` should also be a character variable. But, it also contains 0 and 1 as values. So, its data type is given as int64.

To check missing values in dataset

In [5]:
```
1 heart.isnull().sum()
```

Out[5]:
```
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
```

No null values in dataset

In [6]:
```
1 heart.columns
```

Out[6]:
```
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
      dtype='object')
```

Statistical properties of numerical dataset.

In [7]:
```
1 heart.describe()
```

Out[7]:

	age	sex	cp	trestbps	chol	fbs	restecg	t
count	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.
std	9.082101	0.466011	1.032052	17.538143	51.830751	0.356198	0.525860	22.
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.
25%	47.500000	0.000000	0.000000	120.000000	211.000000	0.000000	0.000000	133.
50%	55.000000	1.000000	1.000000	130.000000	240.000000	0.000000	1.000000	153.
75%	61.000000	1.000000	2.000000	140.000000	274.500000	0.000000	1.000000	166.
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000	202.

Statistical properties of character variables

- heart.describe(include='object')

Statistical properties of all variables

- heart.describe(include='all')

7. Univariate Analysis

- Our dependent variable is target i.e Patient has heart disease or not.

Analysis of `target` feature variable

- Our feature variable of interest is `target` .
- It refers to the presence of heart disease in the patient.
- It is integer valued as it contains two integers 0 and 1 - (0 stands for absence of heart disease and 1 for presence of heart disease).
- So, in this section, I will analyze the `target` variable.

First we check the number of unique values in target variable.

In [8]:
```
1 heart.target.nunique()
```

Out[8]:
```
2
```

In [9]:
```
1 heart.target.unique()
```

Out[9]:
```
array([1, 0], dtype=int64)
```

- There is 2 unique values 0 & 1
- Presence of heart disease is denoted by 1 & absence of heart disease is denoted by 0.

Frequency distribution of target variable.

In [10]:
```
1 heart.target.value_counts()
```

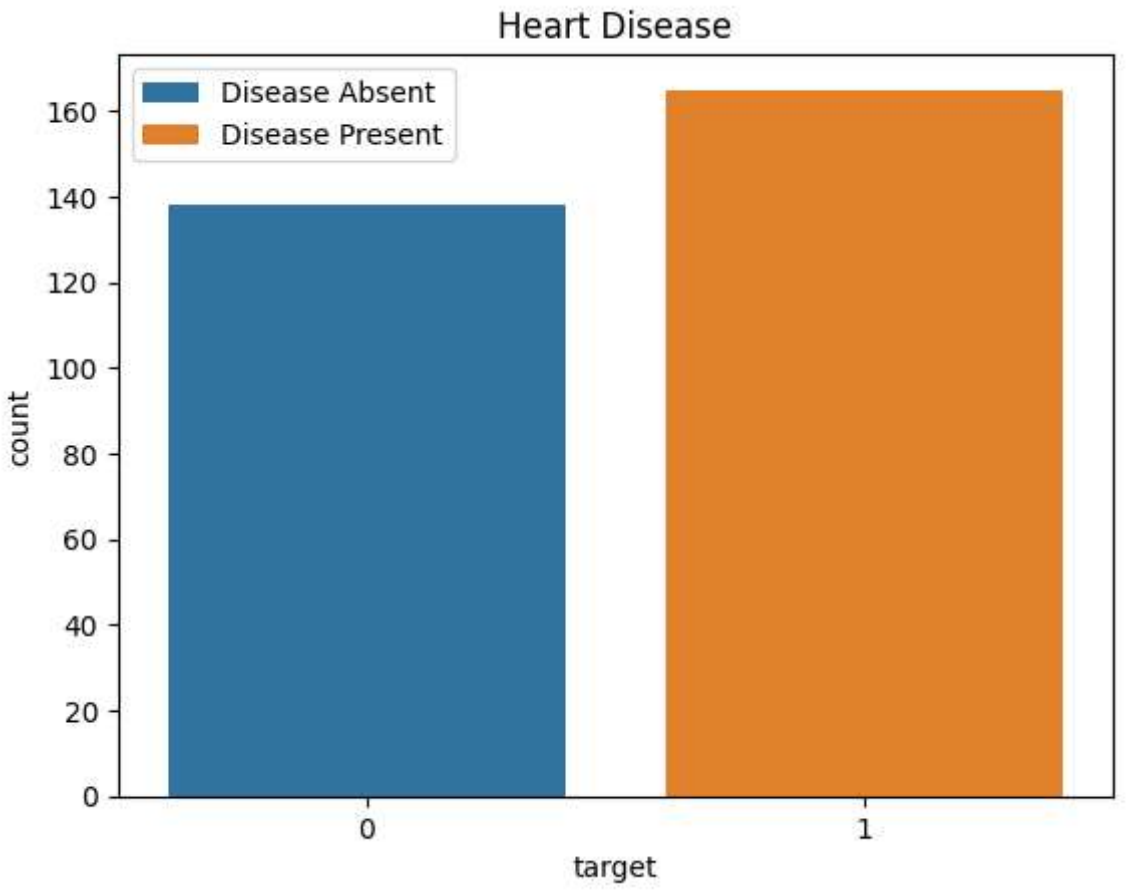
Out[10]:
```
target
1    165
0    138
Name: count, dtype: int64
```

Comment

- 1 is stands for presence of disease,there are 165 patients are suffering from heart disease.
- 0 is stands for absence of heart disease, there is 138 patient do not have heart disease.

Visualize frequency distribution of `target` variable

In [11]:
```
f=sns.countplot(data=heart,x='target',label=('Disease Absent','Disease Pre
f.set_title('Heart Disease')
f=plt.legend()
```



Interpretation

- The above plot confirms the findings that -
  - There are 165 patients suffering from heart disease, and
  - There are 138 patients who do not have any heart disease.

Frequency of target variable with respect gender of patient.

In [12]:
```
1 heart.sex.value_counts()
```

Out[12]:
```
sex
1    207
0     96
Name: count, dtype: int64
```

In [13]:
```
1 heart.groupby('sex')['target'].value_counts()
```

Out[13]:
```
sex  target
0    1          72
     0          24
1    0         114
     1          93
Name: count, dtype: int64
```

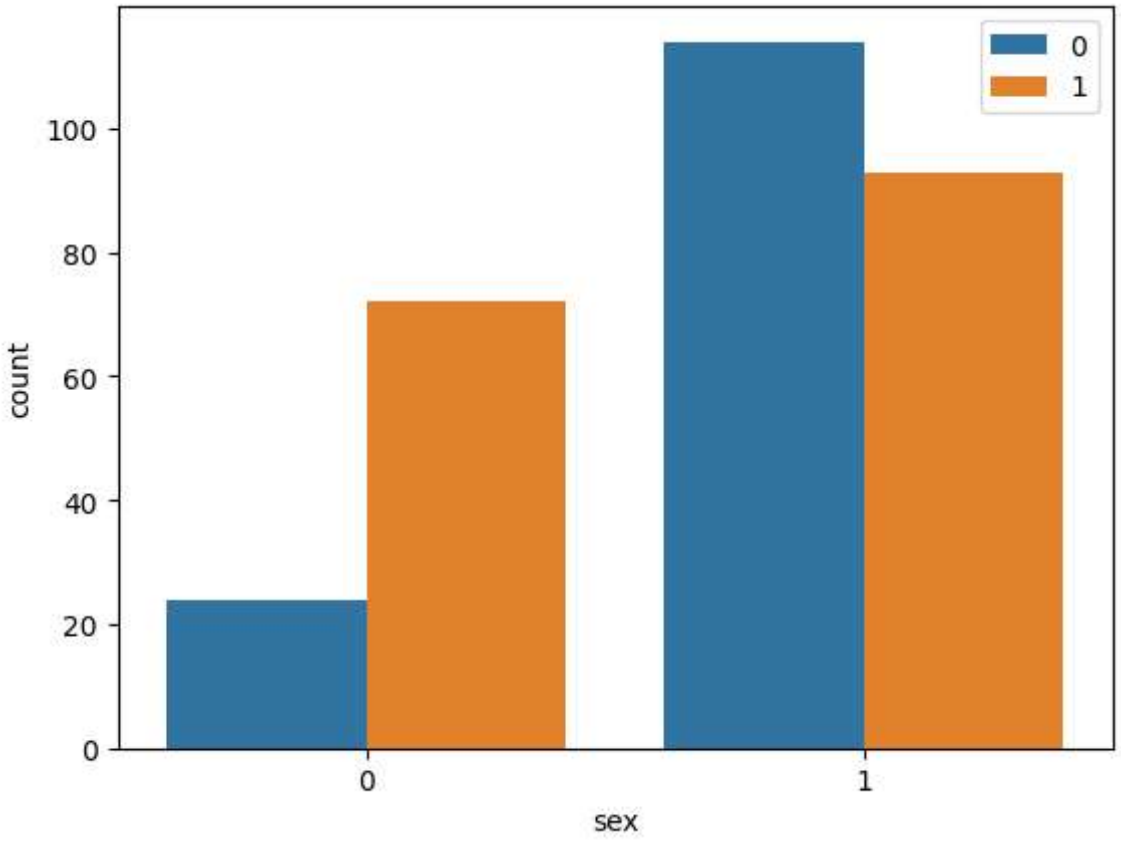
Comment

- `sex` variable contains two integer values 1 and 0 : (1 = male; 0 = female).
- `target` variable also contains two integer values 1 and 0 : (1 = Presence of heart disease; 0 = Absence of heart disease)
- So, out of 96 females - 72 have heart disease and 24 do not have heart disease.
- Similarly, out of 207 males - 93 have heart disease and 114 do not have heart disease.
- We can visualize this information below.

We can visualize the value counts of the `sex` variable wrt `target` as follows -

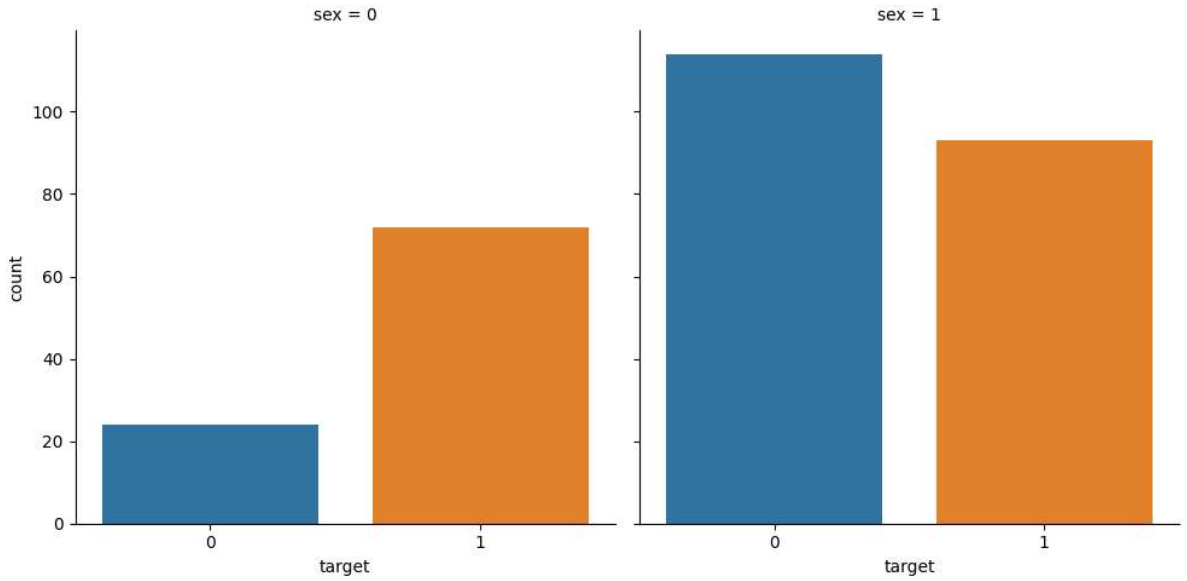
In [14]:
```
g=sns.countplot(data=heart,x='sex',hue='target',orient='h')
plt.legend()
```

Out[14]: <matplotlib.legend.Legend at 0x167949f4b50>



- Total 96 females are in dataset in that 72 females having heart disease remaining 24 do not have heart disease.
- Total 207 males are in dataset in that 93 males having heart disease and remaining 114 do not have heart disease.

In [15]:
```
1 f=sns.catplot(data=heart,x='target',col='sex',kind='count',height=5,aspect
```

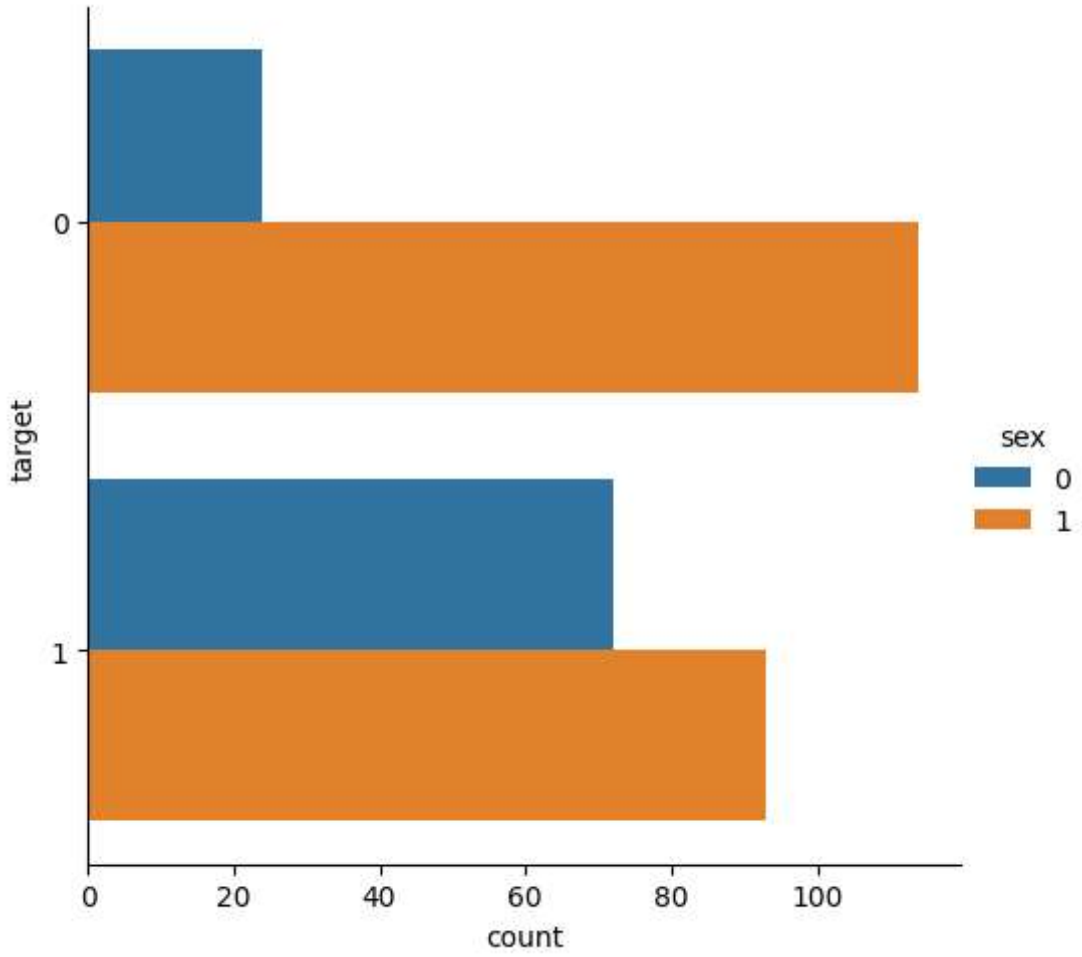


Interpretation

- The above plot segregate the values of `target` variable and plot on two different columns labelled as (sex = 0, sex = 1).
- I think it is more convinient way of interpret the plots.

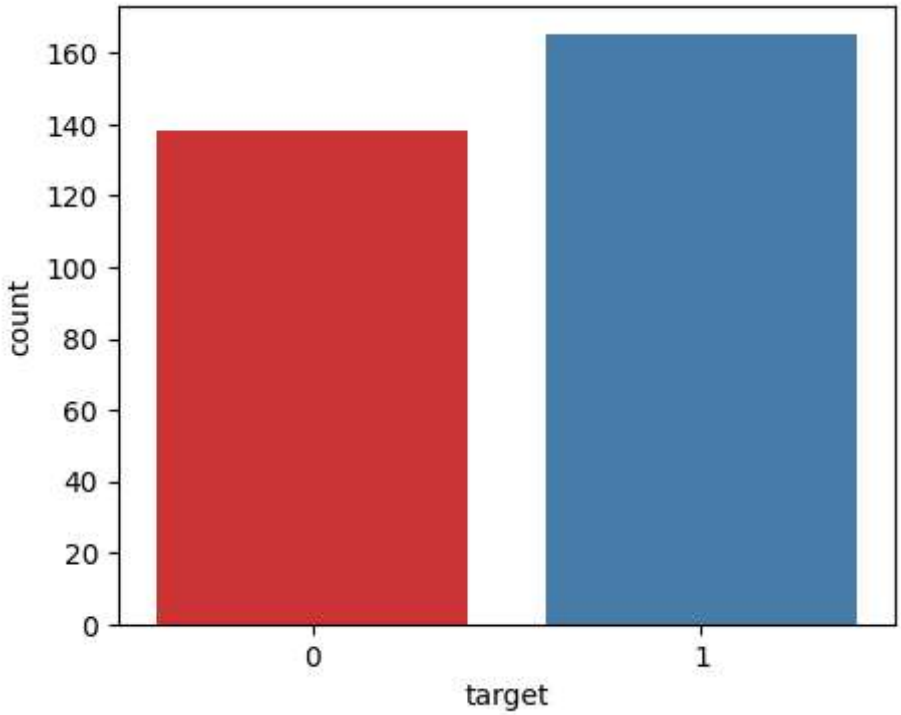
We can plot the bars horizontally as follows

In [16]:
```
1 f=sns.catplot(data=heart,y='target',hue='sex',kind='count',height=5,aspect
```

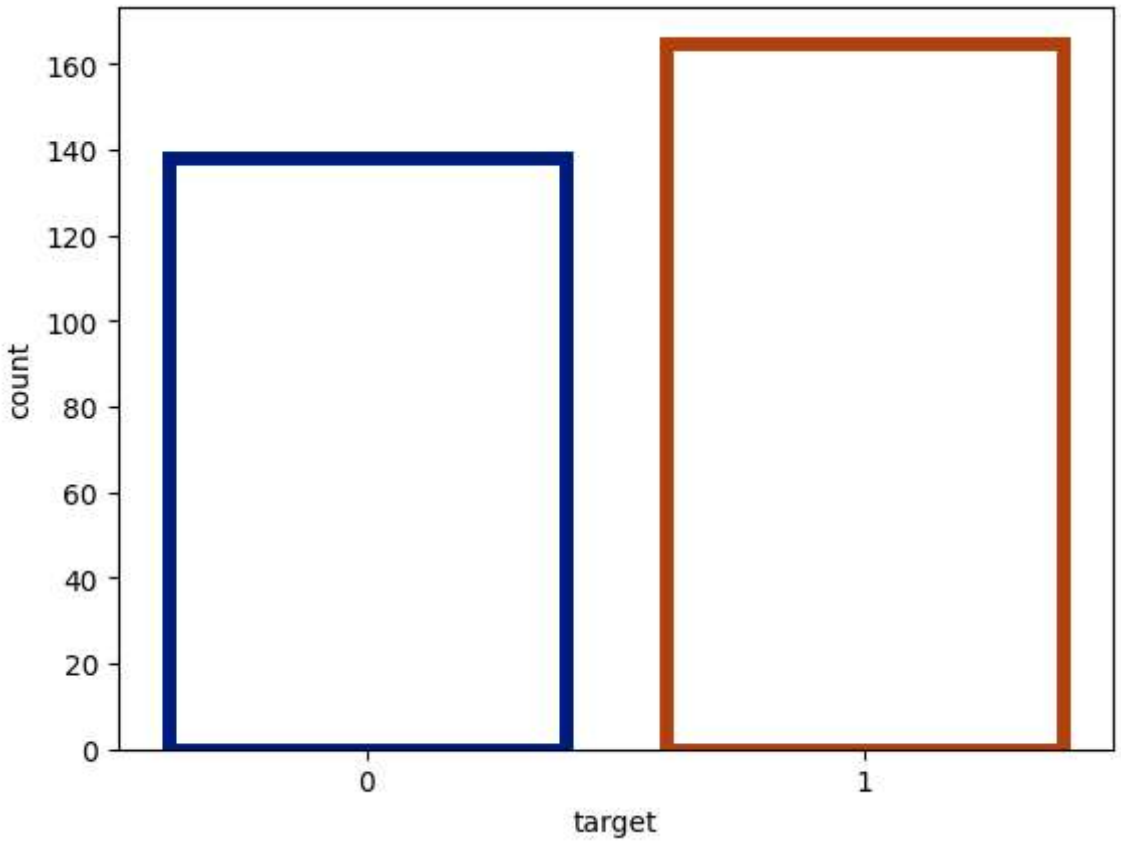


We can use a different color palette as follows

In [17]:
```
1 f,ax=plt.subplots(figsize=(5,4))
2 s=sns.countplot(data=heart,x='target',palette='Set1')  ## palette- To Chan
```



In [18]:
```
1 s=sns.countplot(data=heart,x='target',palette='Set1',facecolor=(0,0,0,0),l
```

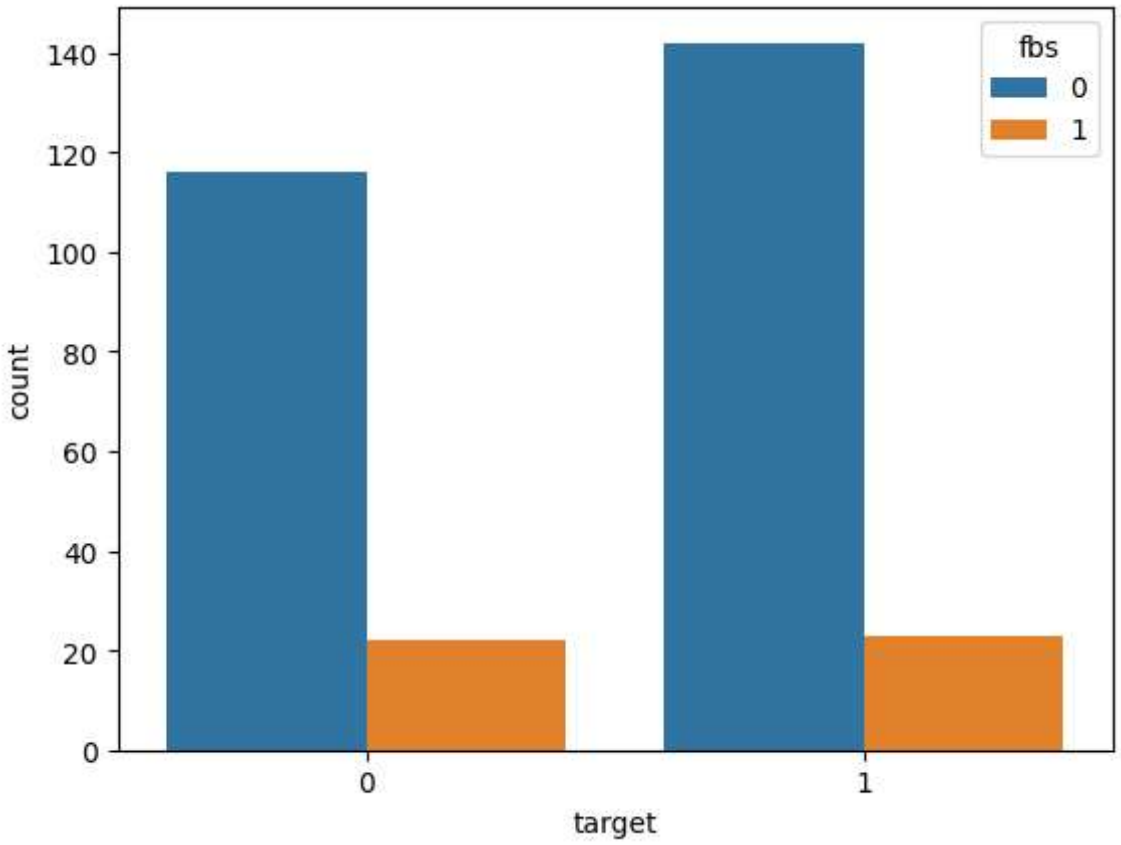


Comment

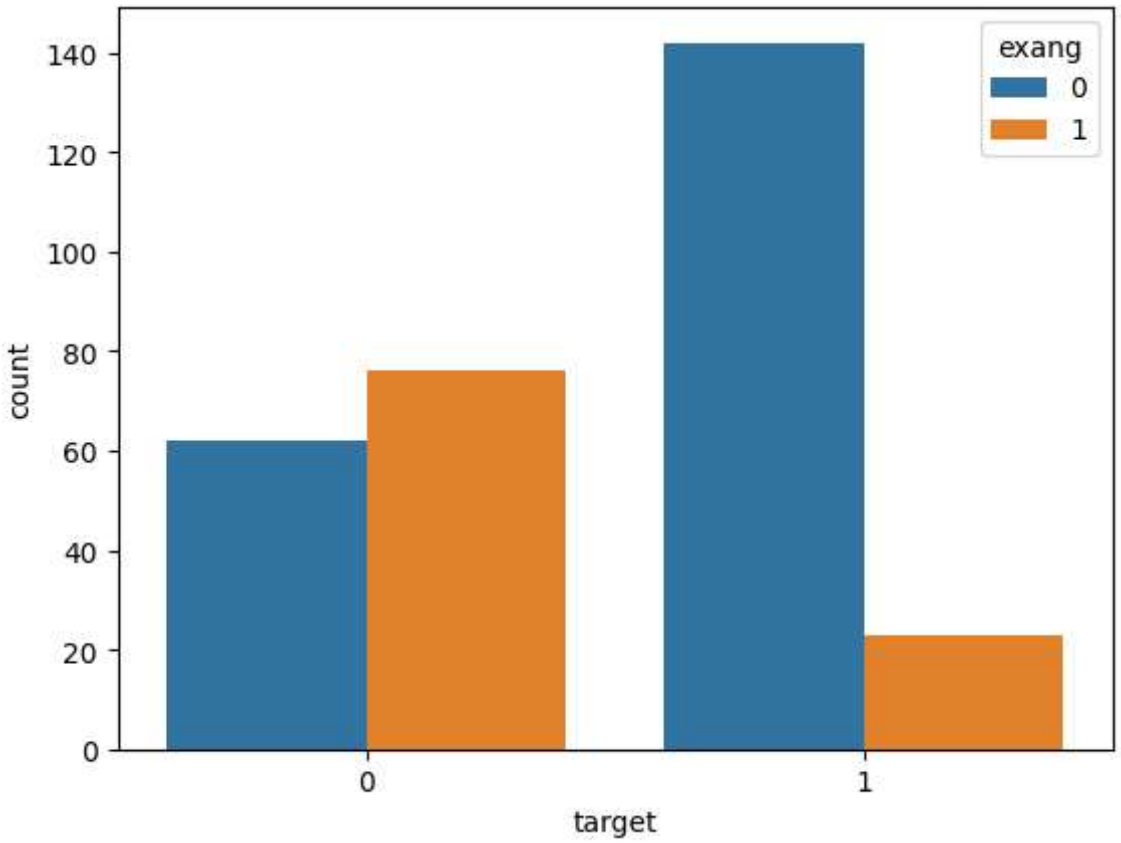
- I have visualize the `target` values distribution wrt `sex` .
- We can follow the same principles and visualize the `target` values distribution wrt `fbs (fasting blood sugar)` and `exang (exercise induced angina)` .

In [19]:

```
## For fbs-fasting blood sugar
ax=sns.countplot(data=heart,x='target',hue='fbs')
```



In [20]:
```
## for exang-exercise induced angina
ax=sns.countplot(data=heart,x='target',hue='exang')
```



Findings of Univariate Analysis

- Our feature variable of interest is `target` .
- It refers to the presence of heart disease in the patient.
- It is integer valued as it contains two integers 0 and 1 - (0 stands for absence of heart disease and 1 for presence of heart disease).
- `1` stands for presence of heart disease. So, there are 165 patients suffering from heart disease.
- Similarly, `0` stands for absence of heart disease. So, there are 138 patients who do not have any heart disease.
- There are 165 patients suffering from heart disease, and
- There are 138 patients who do not have any heart disease.
- Out of 96 females - 72 have heart disease and 24 do not have heart disease.
- Similarly, out of 207 males - 93 have heart disease and 114 do not have heart disease.

8. Bivariate Analysis

Estimate correlation coefficient

Our dataset is very small. So, I will compute the standard correlation coefficient (also called Pearson's r) between every pair of attributes. I will compute it using the `df.corr()` method as

In [21]:
```
correlation=heart.corr()
correlation
```

Out[21]:

	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
sex	-0.098447	1.000000	-0.049353	-0.056769	-0.197912	0.045032	-0.058196	-0.044020	0
cp	-0.068653	-0.049353	1.000000	0.047608	-0.076904	0.094444	0.044421	0.295762	-0
trestbps	0.279351	-0.056769	0.047608	1.000000	0.123174	0.177531	-0.114103	-0.046698	0
chol	0.213678	-0.197912	-0.076904	0.123174	1.000000	0.013294	-0.151040	-0.009940	0
fbs	0.121308	0.045032	0.094444	0.177531	0.013294	1.000000	-0.084189	-0.008567	0
restecg	-0.116211	-0.058196	0.044421	-0.114103	-0.151040	-0.084189	1.000000	0.044123	-0
thalach	-0.398522	-0.044020	0.295762	-0.046698	-0.009940	-0.008567	0.044123	1.000000	-0
exang	0.096801	0.141664	-0.394280	0.067616	0.067023	0.025665	-0.070733	-0.378812	1
oldpeak	0.210013	0.096093	-0.149230	0.193216	0.053952	0.005747	-0.058770	-0.344187	0
slope	-0.168814	-0.030711	0.119717	-0.121475	-0.004038	-0.059894	0.093045	0.386784	-0
ca	0.276326	0.118261	-0.181053	0.101389	0.070511	0.137979	-0.072042	-0.213177	0
thal	0.068001	0.210041	-0.161736	0.062210	0.098803	-0.032019	-0.011981	-0.096439	0
target	-0.225439	-0.280937	0.433798	-0.144931	-0.085239	-0.028046	0.137230	0.421741	-0

Correlation of target variable w.r.t other variables

In [22]:
```
correlation.target.sort_values(ascending=False)
```

Out[22]:
```
target      1.000000
cp          0.433798
thalach     0.421741
slope       0.345877
restecg     0.137230
fbs        -0.028046
chol       -0.085239
trestbps   -0.144931
age        -0.225439
sex        -0.280937
thal       -0.344029
ca         -0.391724
oldpeak    -0.430696
exang      -0.436757
Name: target, dtype: float64
```

Interpretation of correlation coefficient

- The correlation coefficient ranges from -1 to +1.
- When it is close to +1, this signifies that there is a strong positive correlation. So, we can see that there is no variable which has strong positive correlation with `target` variable.
- When it is clsoe to -1, it means that there is a strong negative correlation. So, we can see that there is no variable which has strong negative correlation with `target` variable.
- When it is close to 0, it means that there is no correlation. So, there is no correlation between `target` and `fbs` .
- We can see that the `cp` and `thalach` variables are mildly positively correlated with `target` variable. So, I will analyze the interaction between these features and `target` variable.

Analysis of target and cp (chest pain) variable

Explore `cp` variable

- `cp` stands for chest pain type.
- First, I will check number of unique values in `cp` variable.

In [23]:
```
1 heart.cp.nunique()
```

Out[23]:
```
4
```

So, there are 4 unique values in `cp` variable. Hence, it is a categorical variable.

In [24]:
```
1 heart.cp.unique()
```

Out[24]:
```
array([3, 2, 1, 0], dtype=int64)
```

In [25]:
```
1 heart.cp.value_counts() ## To count the values
```

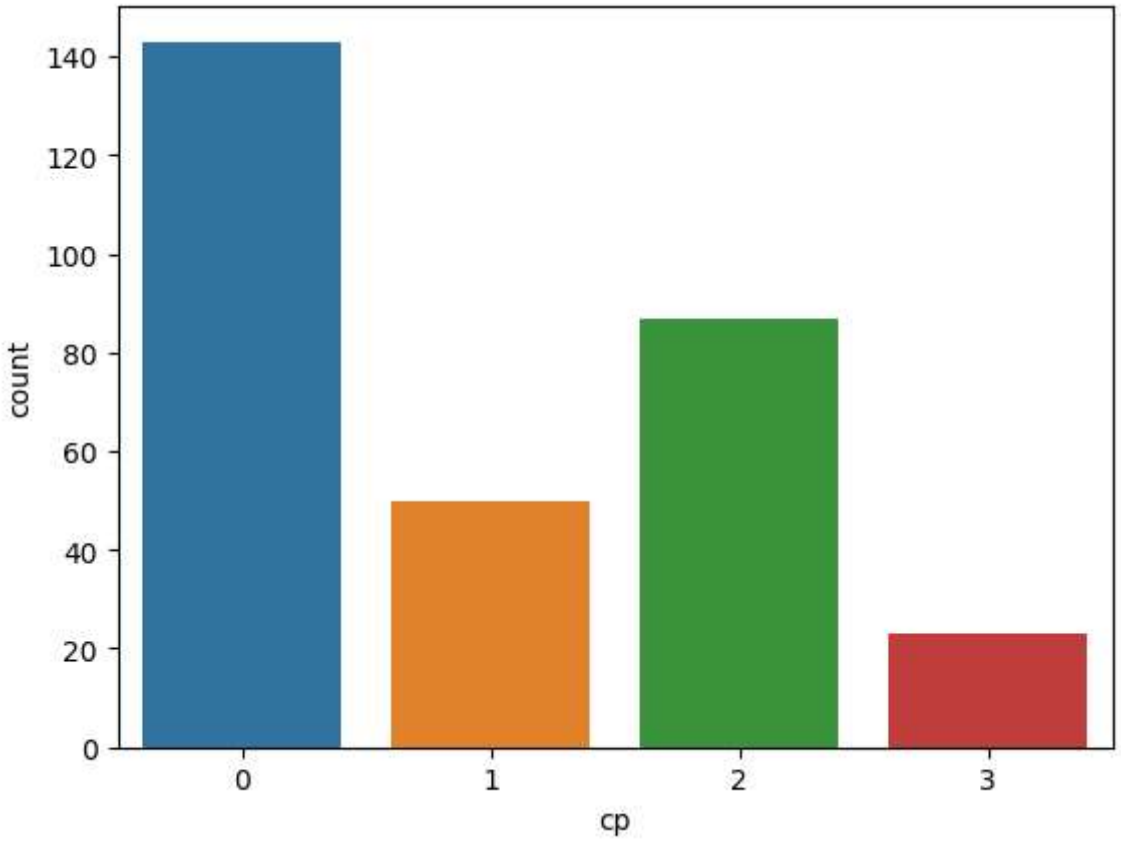
Out[25]:
```
cp
0    143
2     87
1     50
3     23
Name: count, dtype: int64
```

Comment

- It can be seen that `cp` is a categorical variable and it contains 4 types of values - 0, 1, 2 and 3.

Visualize the frequency distribution of cp variable

In [26]:
```
1 cp=sns.countplot(data=heart,x='cp')
```



Frequency distribution of target variable w.r.t cp

In [27]:
```
1 heart.groupby('cp').target.value_counts()
```

Out[27]:
```
cp  target
0   0         104
    1          39
1   1          41
    0           9
2   1          69
    0          18
3   1          16
    0           7
Name: count, dtype: int64
```

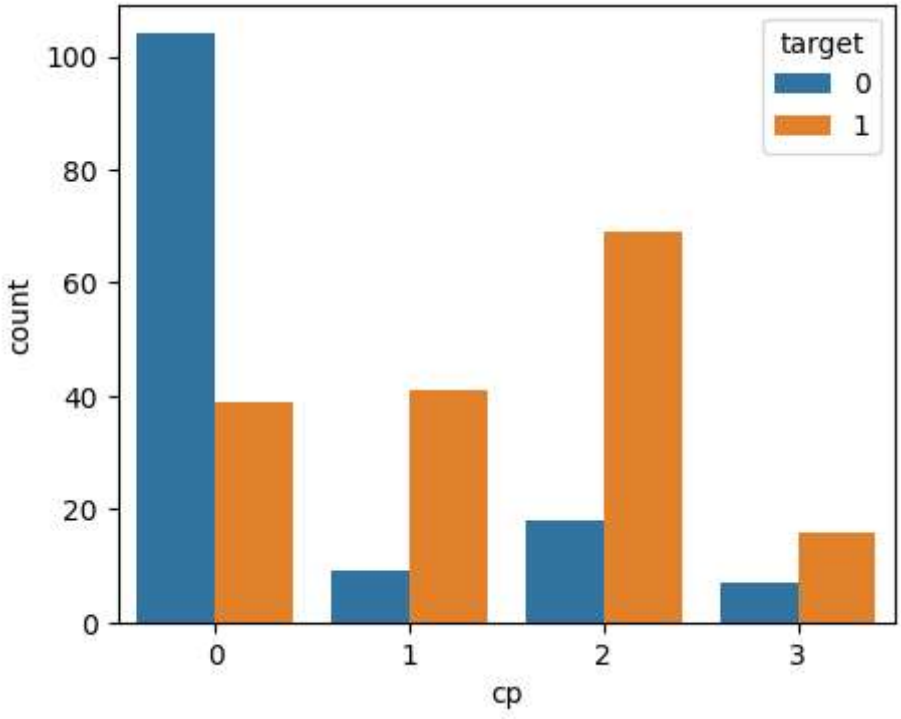
Comment

- `cp` variable contains four integer values 0, 1, 2 and 3.
- `target` variable contains two integer values 1 and 0 : (1 = Presence of heart disease; 0 = Absence of heart disease)

- So, the above analysis gives `target` variable values categorized into presence and absence of heart disease and groupby `cp` variable values.
- We can visualize this information below.

We can visualize the value counts of the `cp` variable wrt `target` as follows -

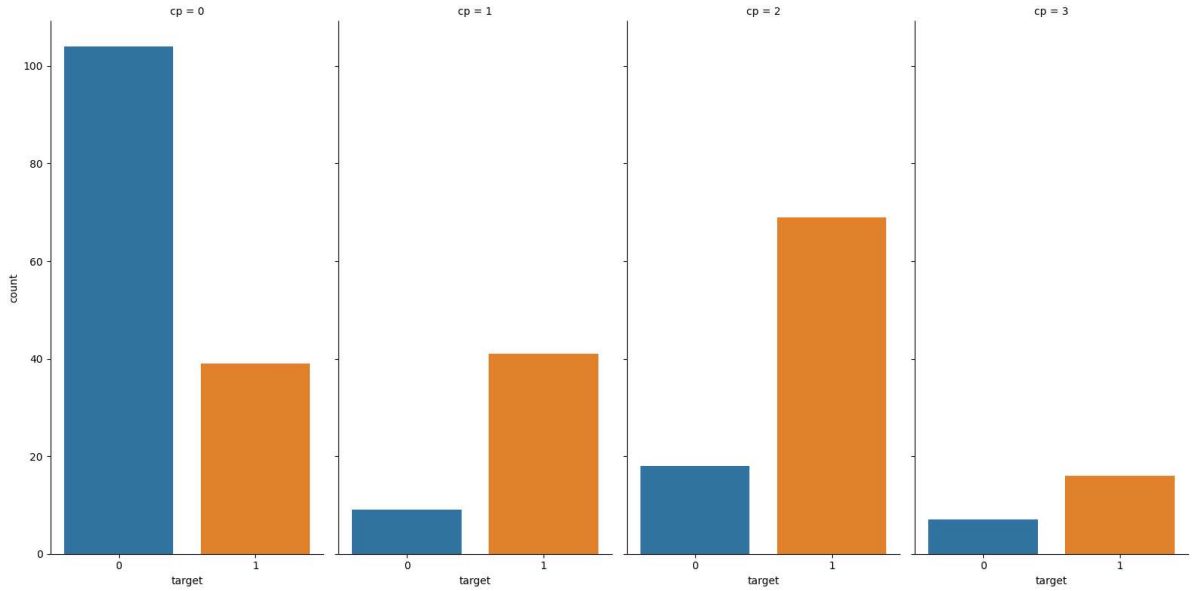
In [28]:
```
f,ax=plt.subplots(figsize=(5,4))

ax=sns.countplot(data=heart,x='cp',hue='target')
```



- When chest pain type is 0 then heart disease not present is not present.
- when chest pain type is 2 the presence rate of heart disease is very high.

Alternatively, we can visualize the same information as follows :

In [29]:
```
1 ax=sns.catplot(data=heart,x='target',col='cp',kind='count',height=8,aspect
```



Analysis of target and thalach

Explore `thalach` variable

- `thalach` stands for maximum heart rate achieved.
- I will check number of unique values in `thalach` variable as follows :

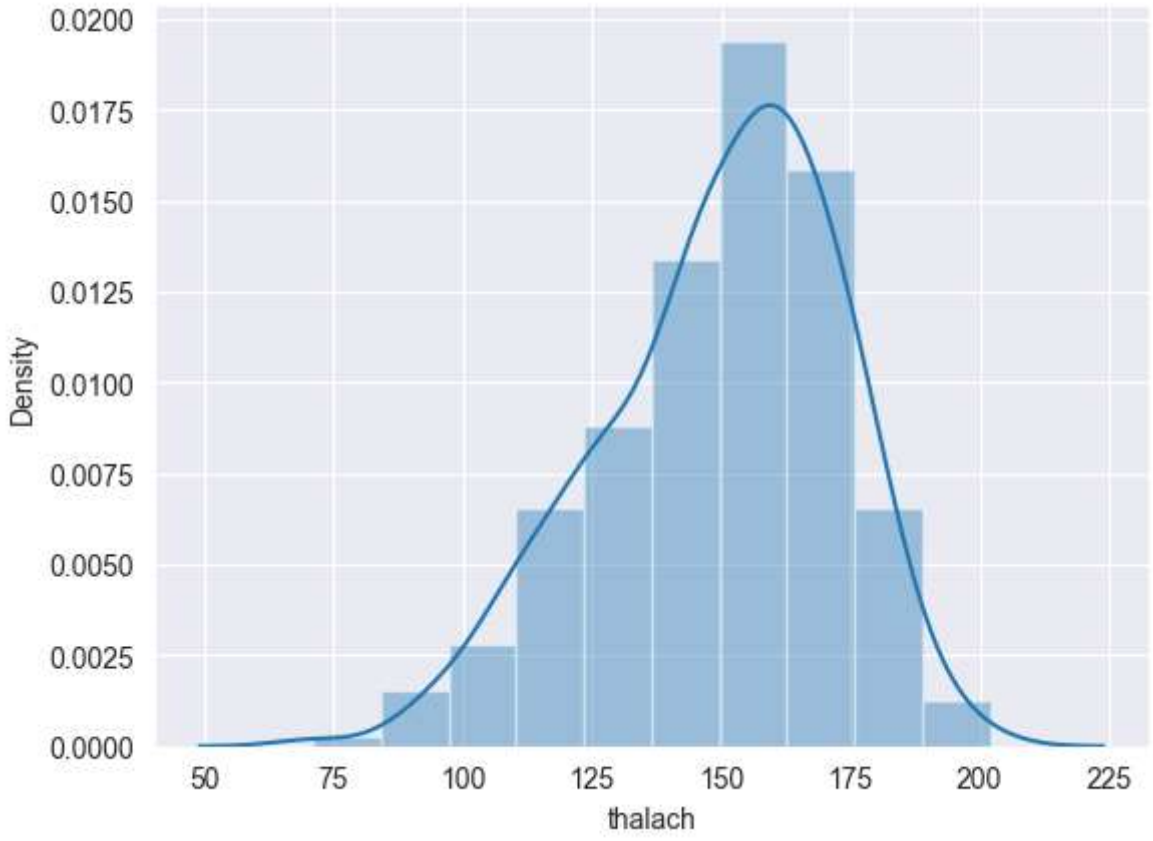
In [30]:
```
1 heart.thalach.nunique()
```

Out[30]: 91

- Number of unique values in thalach variable is 91. Hence it is numerical variable

Visualize the frequency distribution of thalach variable

In [31]:
```
sns.set_style('darkgrid')
x=heart.thalach
d=sns.distplot(x,bins=10)
```

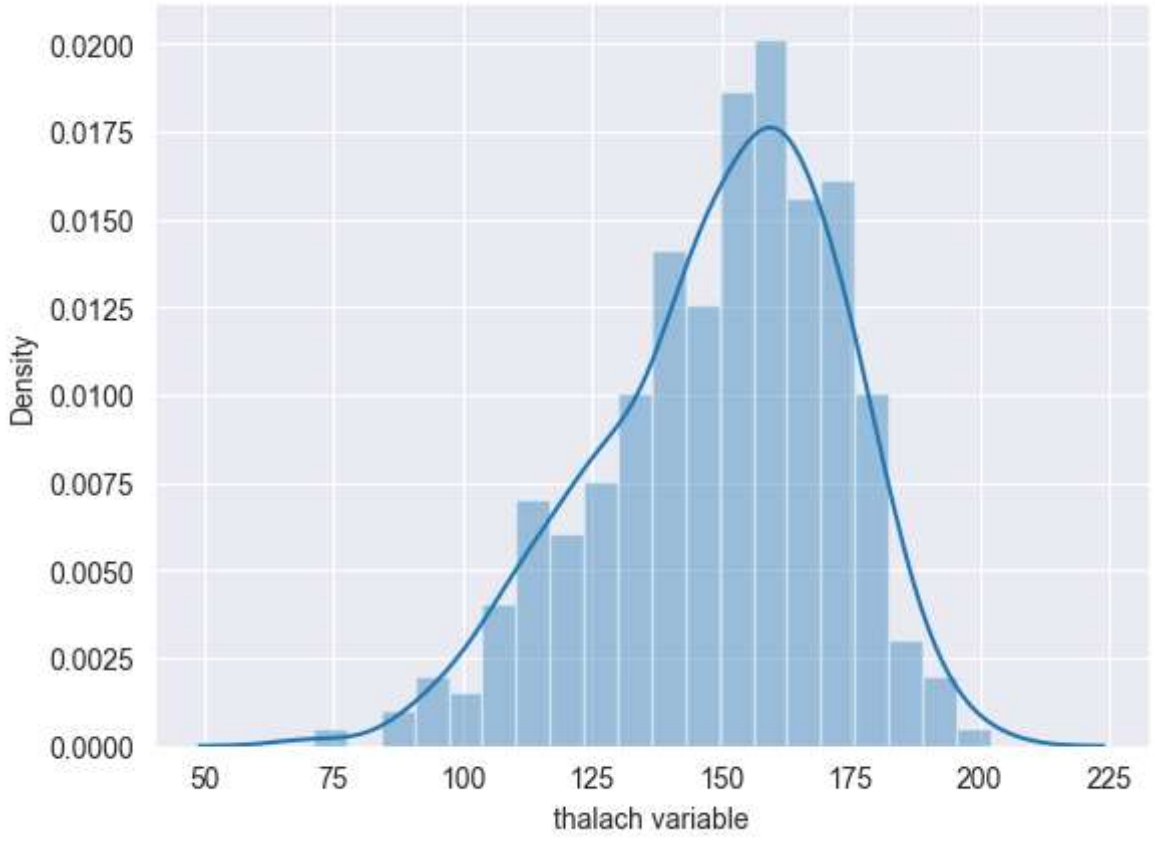


- we can see that thalach variable is slightly negatively skewed

We can use pandas series object to get an informative axis label as follows

In [32]:

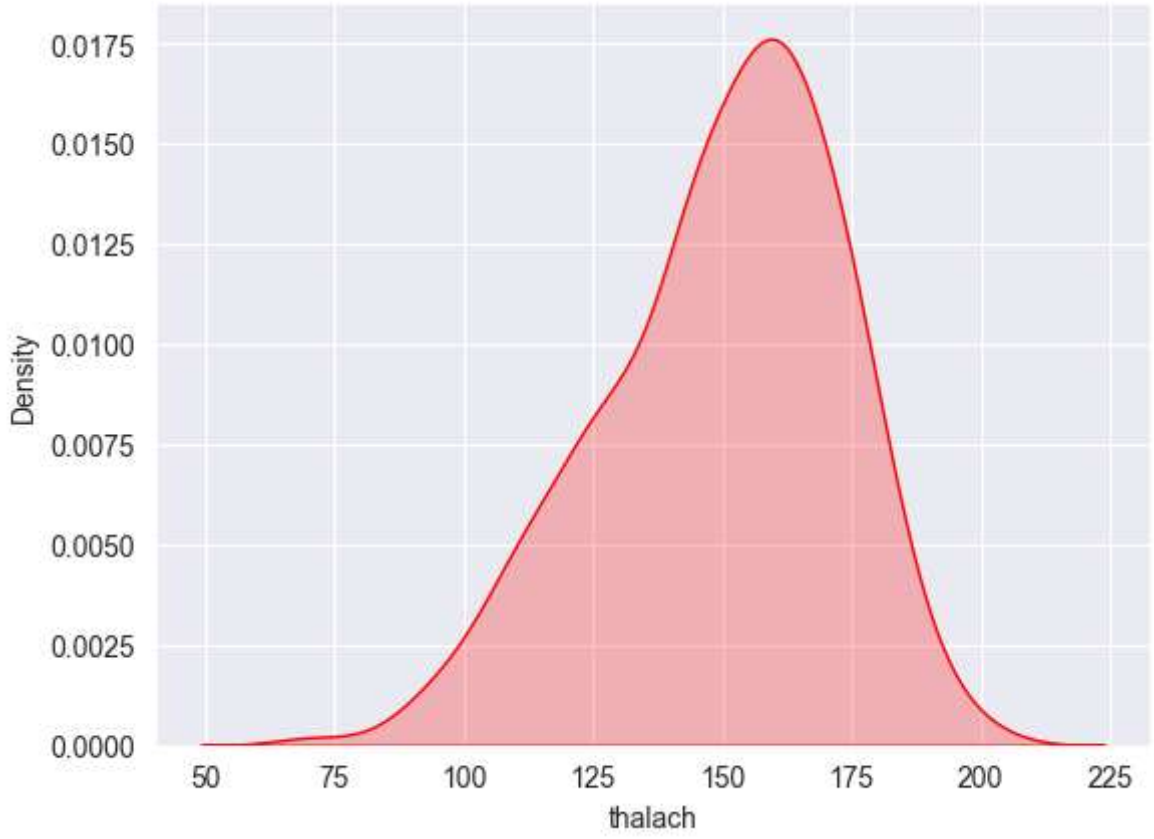
```
1 x=heart.thalach
2
3 x=pd.Series(x,name='thalach variable')
4 ax=sns.distplot(x,bins=20)
```



Kernel Density Estimation(KDE) Plot

In [33]:
```
1 sns.kdeplot(heart.thalach,color='r',shade=True)
```

Out[33]: <Axes: xlabel='thalach', ylabel='Density'>

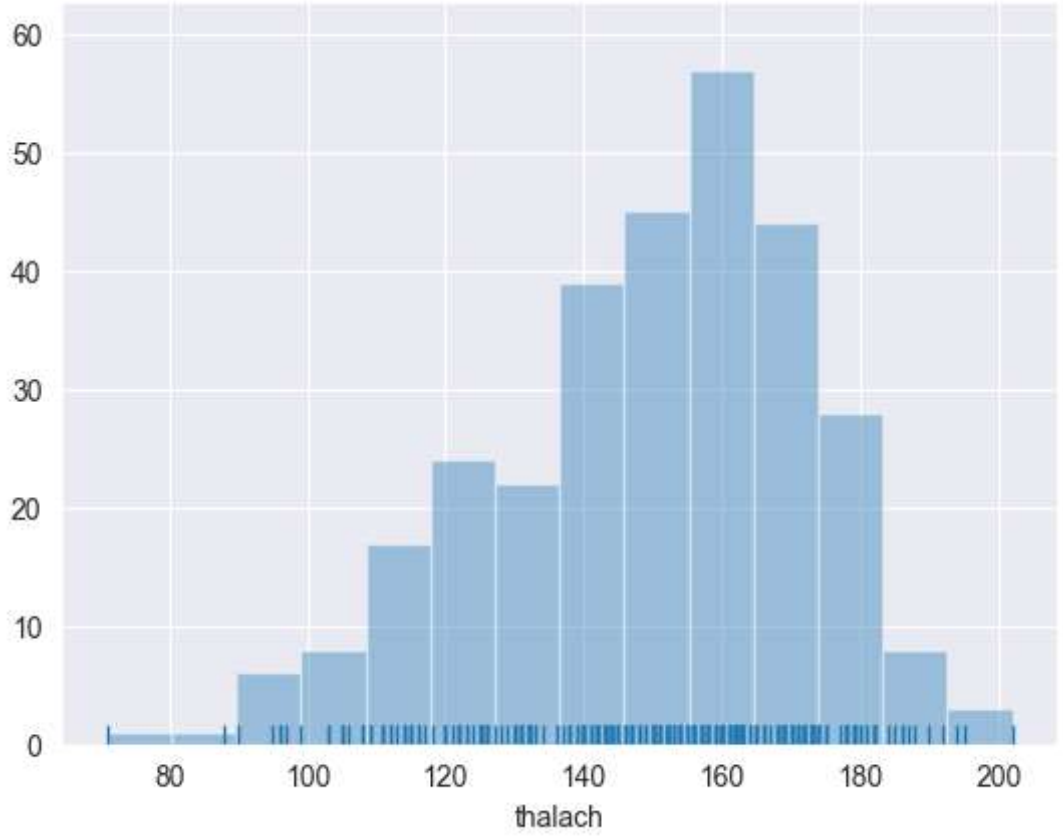


Seaborn Kernel Density Estimation (KDE) Plot

- The kernel density estimate (KDE) plot is a useful tool for plotting the shape of a distribution.
- The KDE plot plots the density of observations on one axis with height along the other axis.
- We can plot a KDE plot as follows :

Distribution plot

In [34]:
```
1 ax=sns.distplot(heart.thalach,kde=False,rug=True)
```

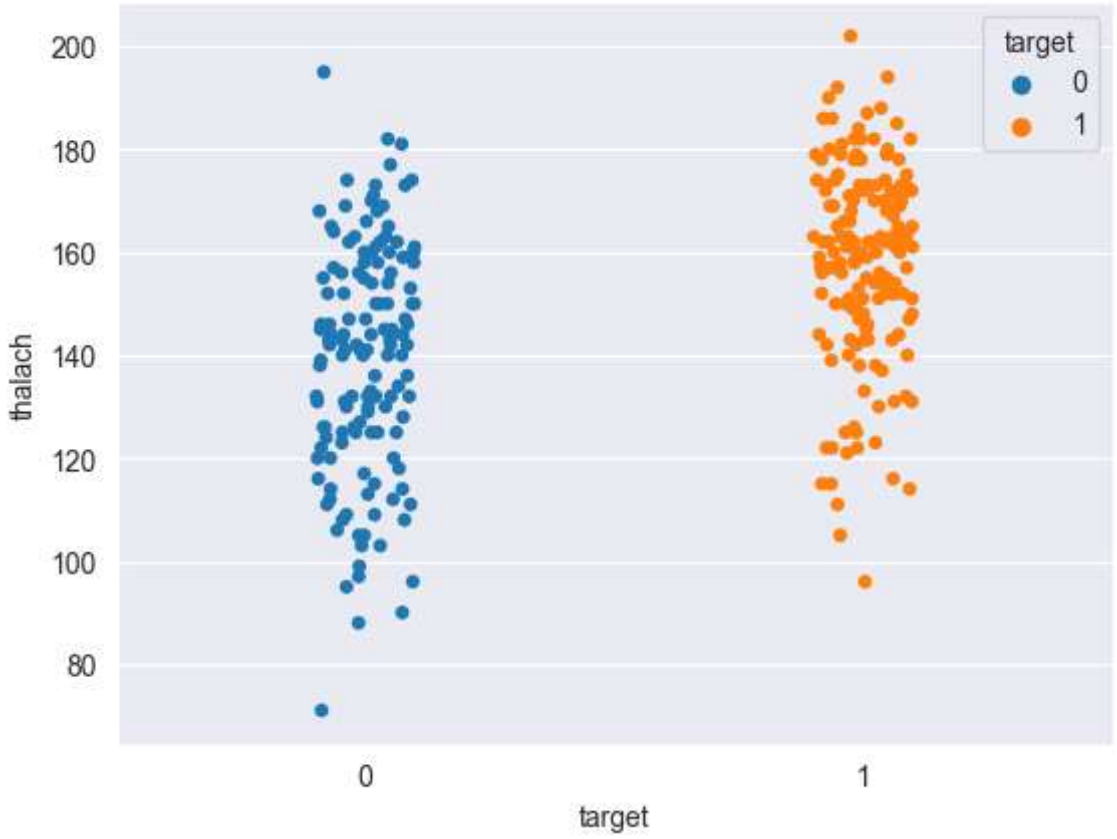


Histogram

- A histogram represents the distribution of data by forming bins along the range of the data and then drawing bars to show the number of observations that fall in each bin.
- We can plot a histogram as follows :

Visualize frequency distribution of thalach variable w.r.t target.

In [35]:
```
1 ax=sns.stripplot(data=heart,x='target',y='thalach',hue='target')
```

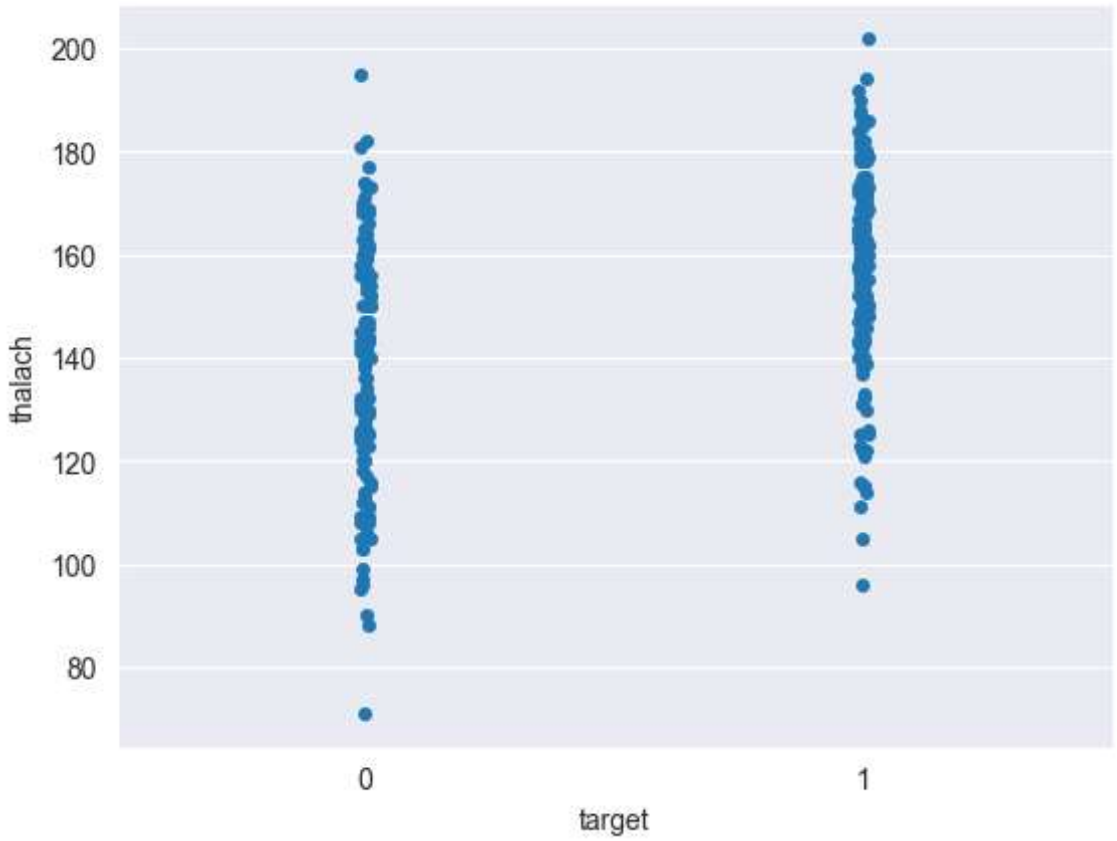


Interpretation

- we can see that those people suffering from heart disease(target=1) have relatively higher heart rate(thalach) as compared to people who are not suffering from heart disease(target=0)

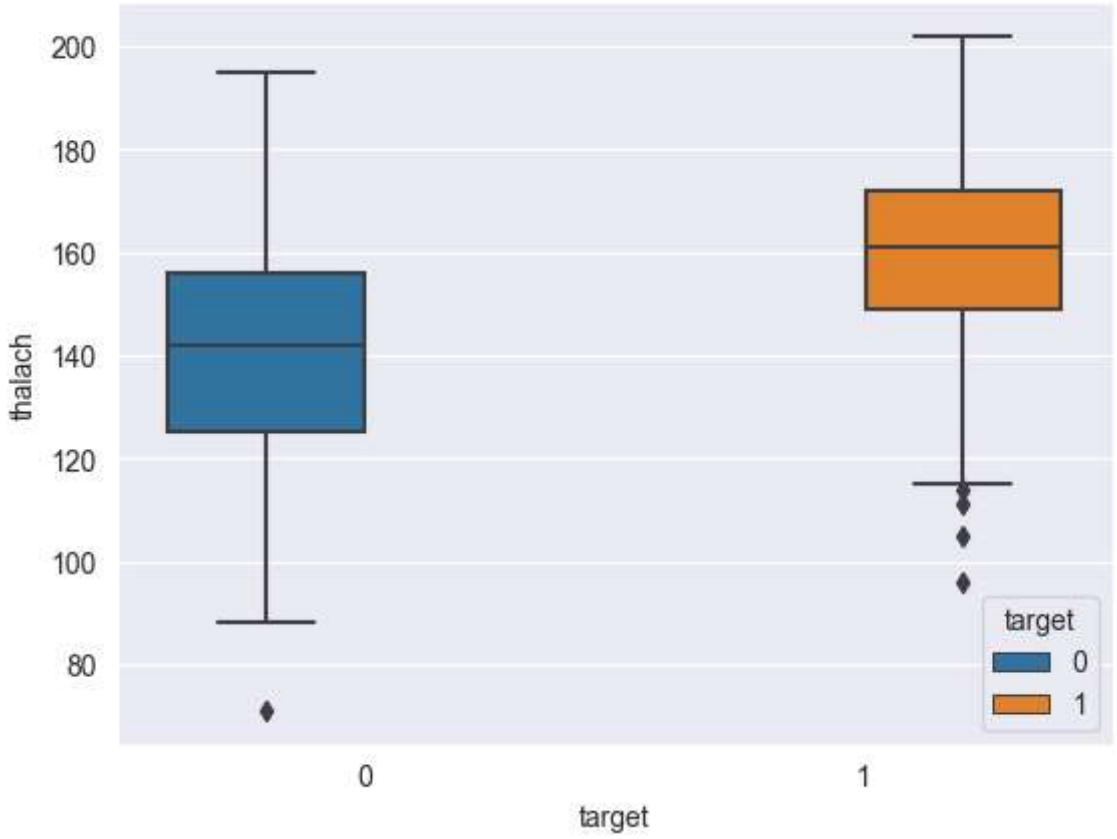
In [36]:
```
# We can add jitter to bring out the distribution of values as folows
sns.stripplot(data=heart,x='target',y='thalach',jitter=0.01)
```

Out[36]: <Axes: xlabel='target', ylabel='thalach'>



Visualize distribution of thalach variable w.r.t target with boxplot

In [37]:
```
1 b=sns.boxplot(data=heart,x='target',y='thalach',hue='target')
```



Interpretation

- The above boxplot confirms our finding that people suffering from heart disease (target = 1) have relatively higher heart rate (thalach) as compared to people who are not suffering from heart disease (target = 0).

Bivariate analysis interpretation

- There is no variable which has strong positive correlation with target variable.
- There is no variable which has strong negative correlation with target variable.
- There is no correlation between target and fbs.
- The cp and thalach variables are mildly positively correlated with target variable.
- We can see that the thalach variable is slightly negatively skewed.
- The people suffering from heart disease (target = 1) have relatively higher heart rate (thalach) as compared to people who are not suffering from heart disease (target = 0).
- The people suffering from heart disease (target = 1) have relatively higher heart rate (thalach) as compared to people who are not suffering from heart disease (target = 0).

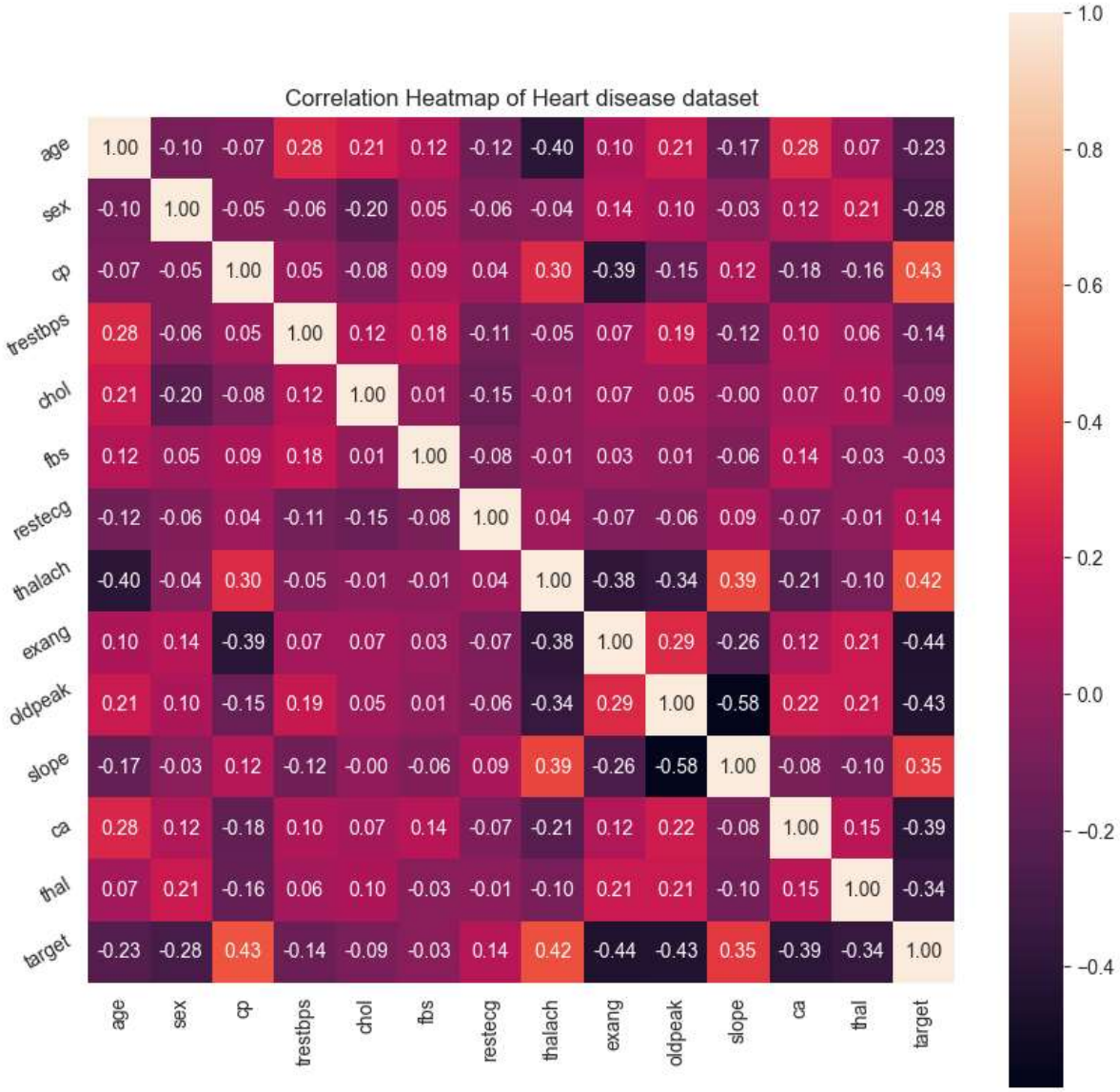
9. Multivariate Analysis

Discover patterns and relationships

- An important step in EDA is to discover patterns and relationships between variables in the dataset.
- I will use `heat map` and `pair plot` to discover the patterns and relationships in the dataset.

Heat Map

In [38]:
```
plt.figure(figsize=(10,10))
h=sns.heatmap(correlation,annot=True,square=True,fmt='.2f',linecolor='whit

plt.title('Correlation Heatmap of Heart disease dataset')

h.set_yticklabels(h.get_yticklabels(),rotation=30);
```



Interpretation

From the above correlation heat map, we can conclude that :-

- target and cp variable are mildly positively correlated (correlation coefficient = 0.43).
- target and thalach variable are also mildly positively correlated (correlation coefficient = 0.42).
- target and slope variable are weakly positively correlated (correlation coefficient = 0.35).

- target and exang variable are mildly negatively correlated (correlation coefficient = -0.44).
- target and oldpeak variable are also mildly negatively correlated (correlation coefficient = -0.43).
- target and ca variable are weakly negatively correlated (correlation coefficient = -0.39).
- target and thal variable are also waekly negatively correlated (correlation coefficient = -0.34).

Pair Plot

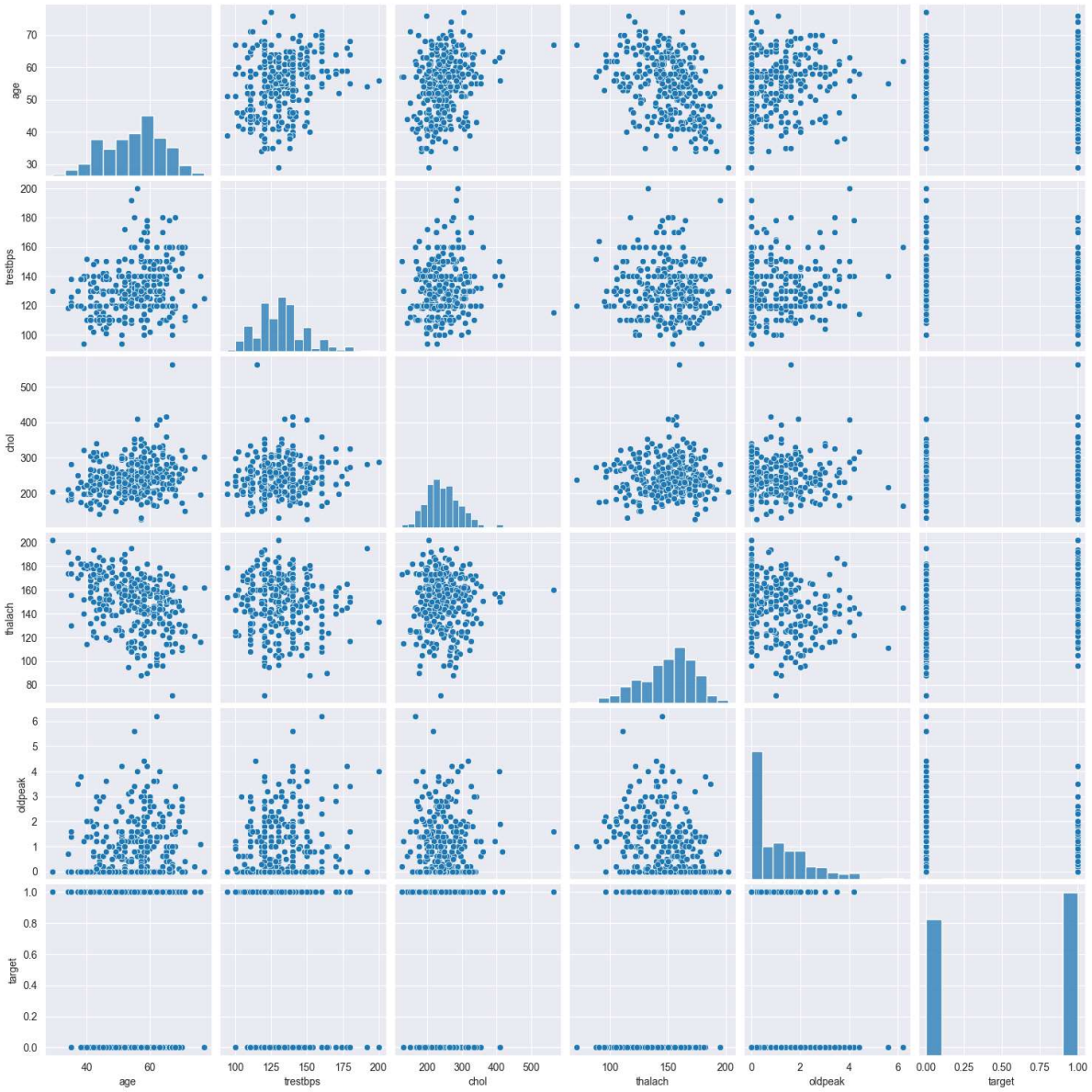
In [39]:
```
1 heart.columns
```

Out[39]:
```
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
      dtype='object')
```

To check the relationship between target and other numerical variables

In [40]:
```
numerical_var=['age','trestbps','chol','thalach','oldpeak','target']
sns.pairplot(heart[numerical_var],kind='scatter')
```

Out[40]: <seaborn.axisgrid.PairGrid at 0x1679b15c590>



Comment

- I have defined a variable `num_var` . Here `age` , `trestbps` , `chol`, `thalach` and `oldpeak` are numerical variables and `target` is the categorical variable.
- So, I wll check relationships between these variables.

Analysis of age variable

Check the number of unique values in age variable

In [41]:
```
1 heart.age.nunique()
```

Out[41]:
```
41
```

In [42]:
```
1 heart.age.unique()
```

Out[42]:
```
array([63, 37, 41, 56, 57, 44, 52, 54, 48, 49, 64, 58, 50, 66, 43, 69, 59,
       42, 61, 40, 71, 51, 65, 53, 46, 45, 39, 47, 62, 34, 35, 29, 55, 60,
       67, 68, 74, 76, 70, 38, 77], dtype=int64)
```

View statistical summary of age variable

In [43]:
```
1 heart.age.describe()
```

Out[43]:
```
count    303.000000
mean      54.366337
std        9.082101
min       29.000000
25%       47.500000
50%       55.000000
75%       61.000000
max       77.000000
Name: age, dtype: float64
```

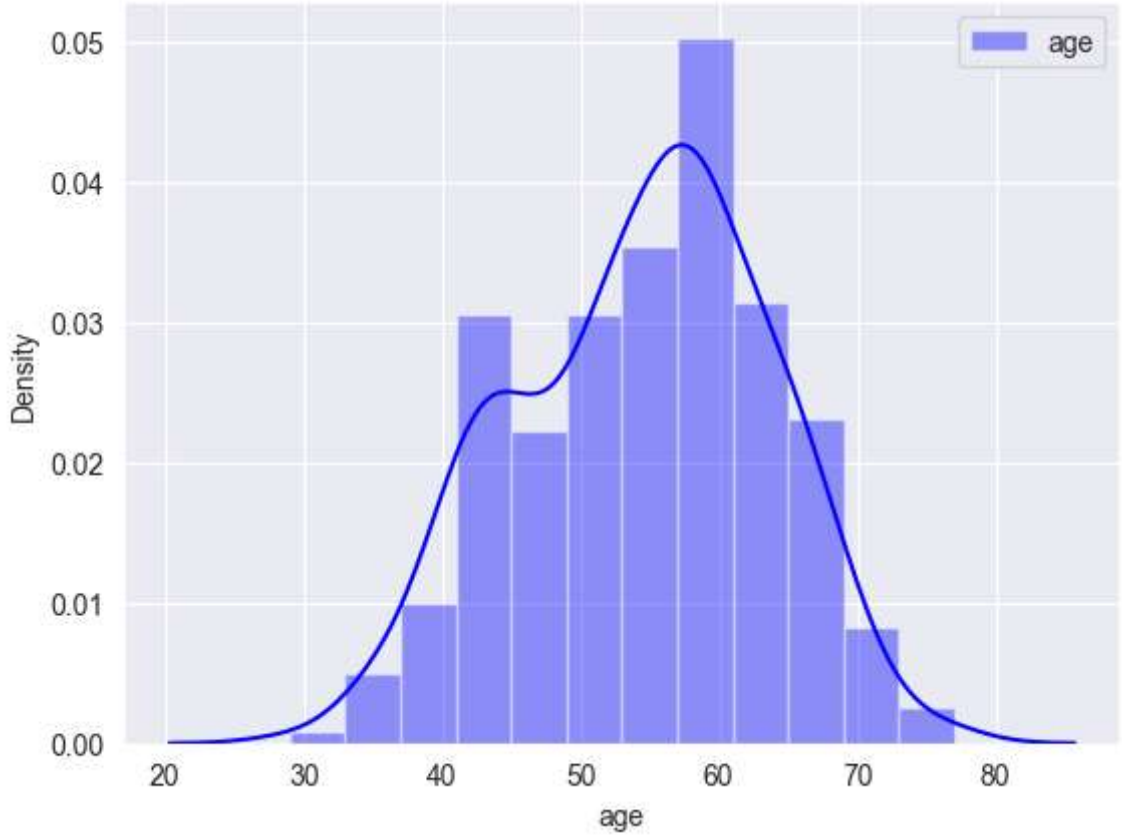
Interpretation

- Then min age is 29 and maximum age is 77
- The mean value of age is 54 years.

Plot the distribution of age variable

In [44]:
```
1 age=heart.age
2 a=sns.distplot(age,color='b',label='age')
3 plt.legend()
```

Out[44]: <matplotlib.legend.Legend at 0x1679c0d6c10>

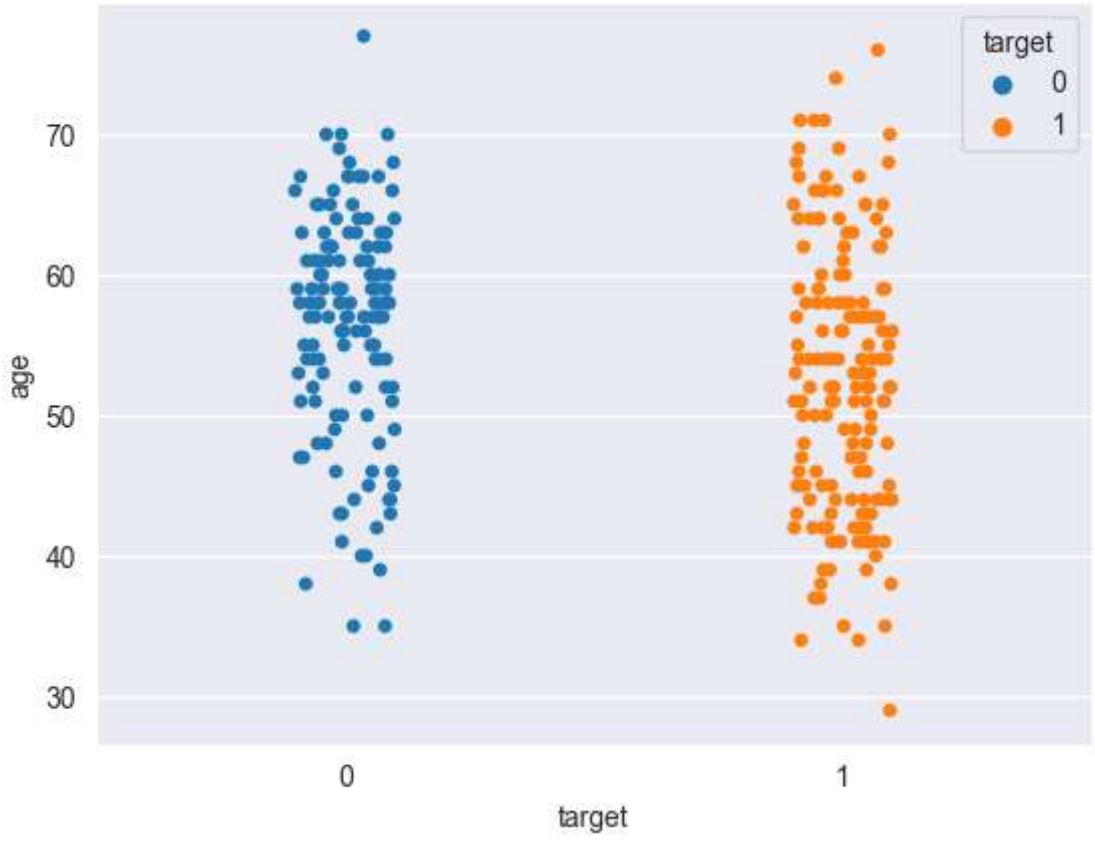


Interpretation

- The age variables distribution is approximately normal.

Analyze age and target variable

In [45]:
```
1 vis=sns.stripplot(data=heart,x='target',y='age',hue='target')
```



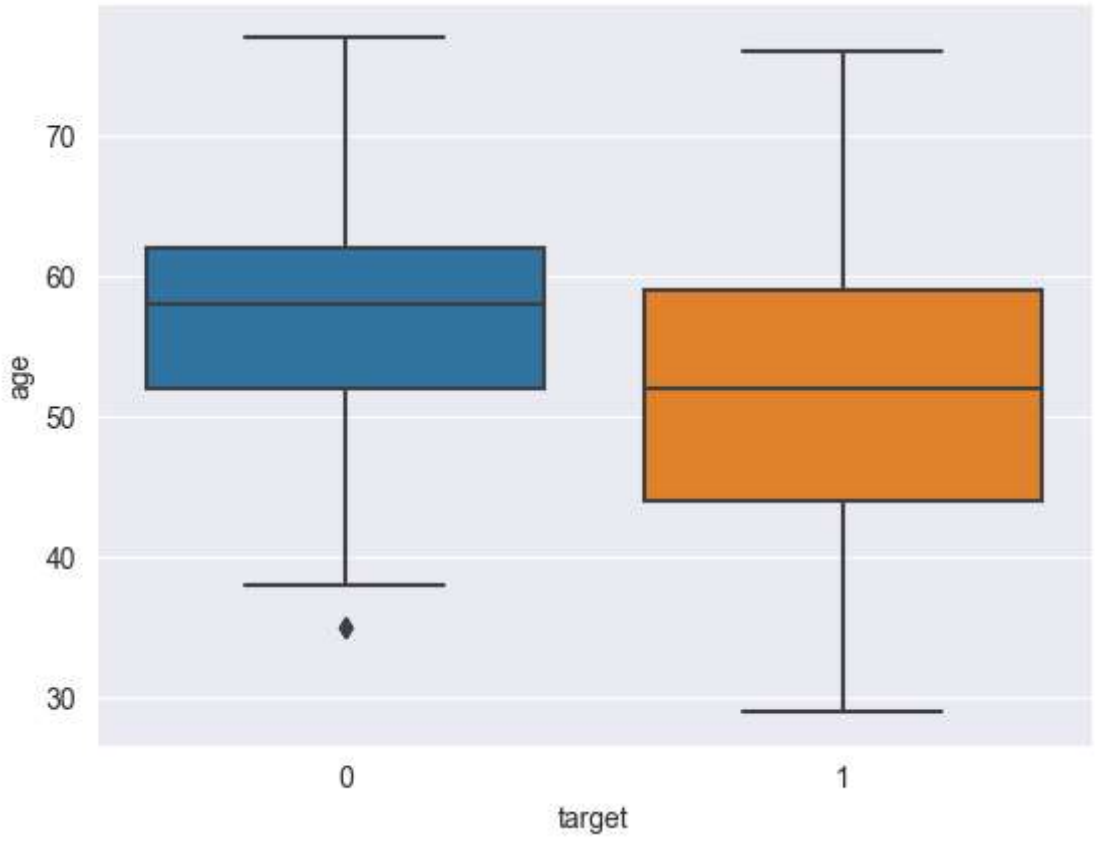
Interpretation

- We can see that the people suffering from heart disease (target = 1) and people who are not suffering from heart disease (target = 0) have comparable ages.

Visualize distribution of `age` variable wrt `target` with boxplot

In [46]:

```
# Boxplot
b=sns.boxplot(data=heart,x='target',y='age')
```



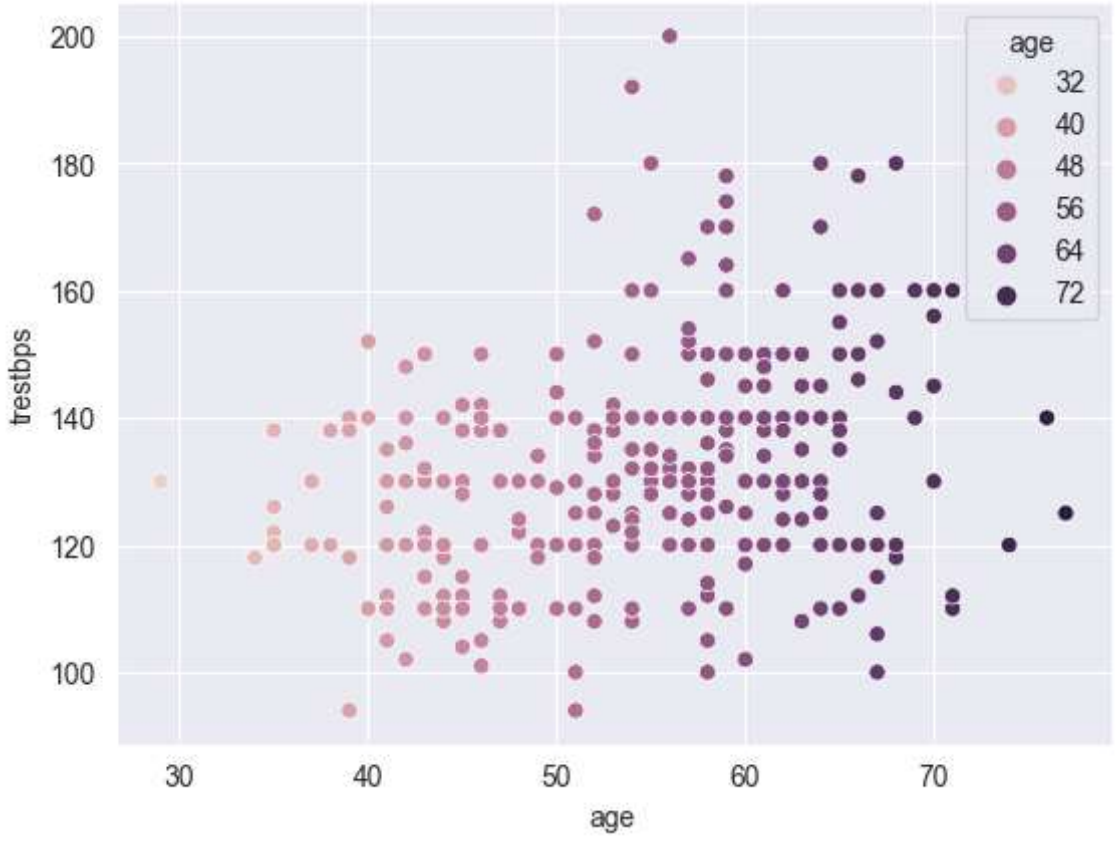
Interpretation

The above boxplot tells two different things :

- The mean age of the people who have heart disease is less than the mean age of the people who do not have heart disease.
- The dispersion or spread of age of the people who have heart disease is greater than the dispersion or spread of age of the people who do not have heart disease.

Analyse age and trestbps variable

In [47]:
```
1 s=sns.scatterplot(data=heart,x='age',y='trestbps',hue='age')
```

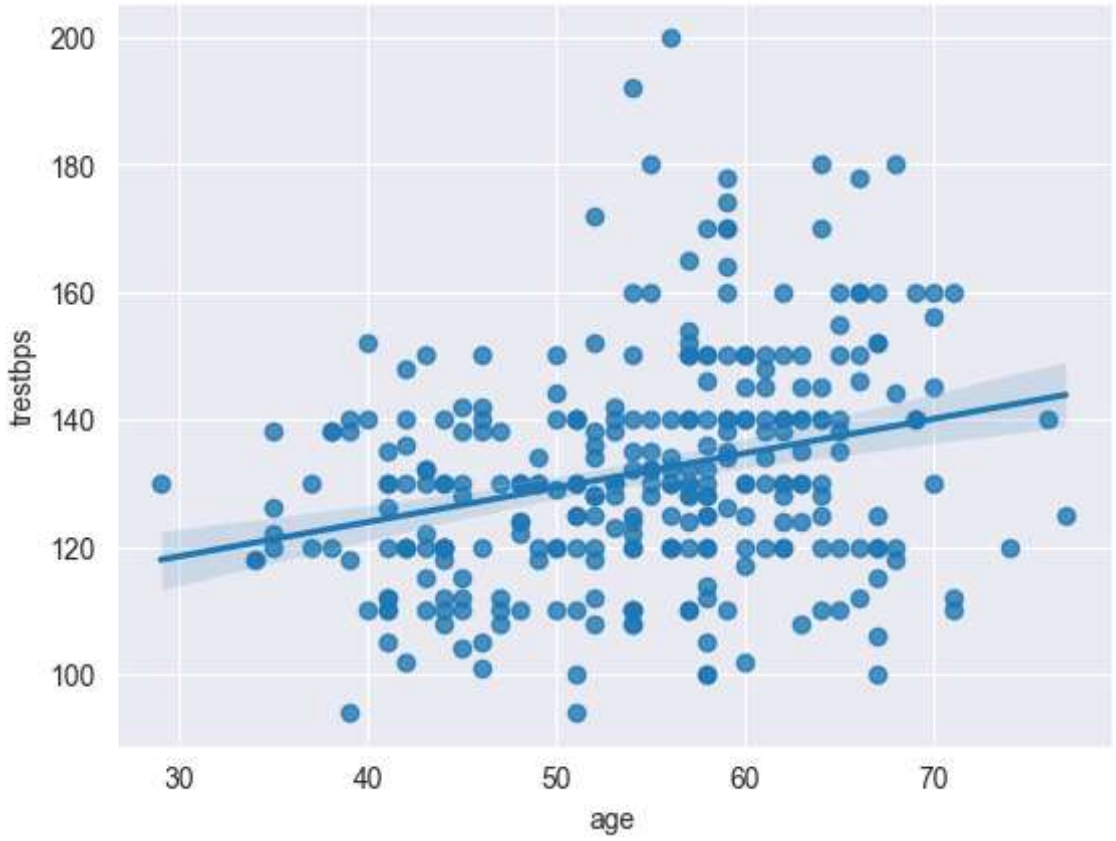


Interpretation

- There is no correlation between age and trestbps variable

In [48]:

```
## Regression Plot
r=sns.regplot(data=heart,x='age',y='trestbps')
```

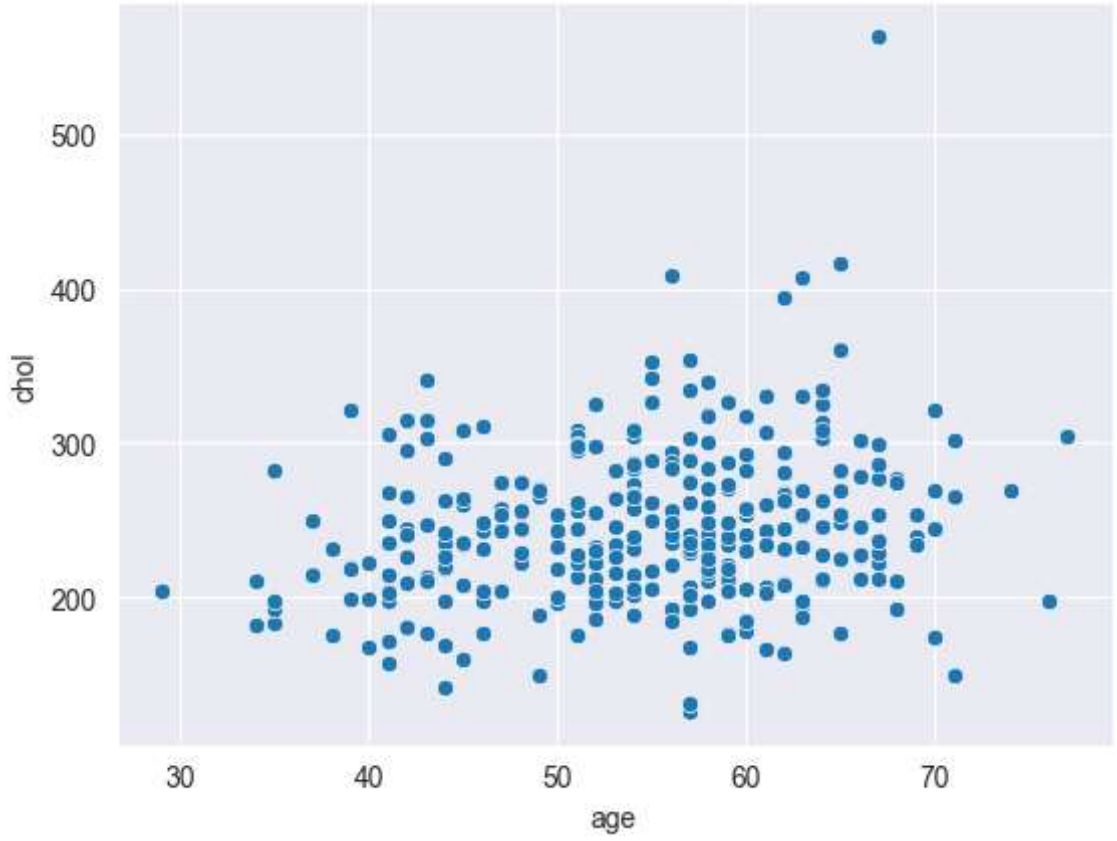


Interpretation

- The above line shows that linear regression model is not good fit to the data.

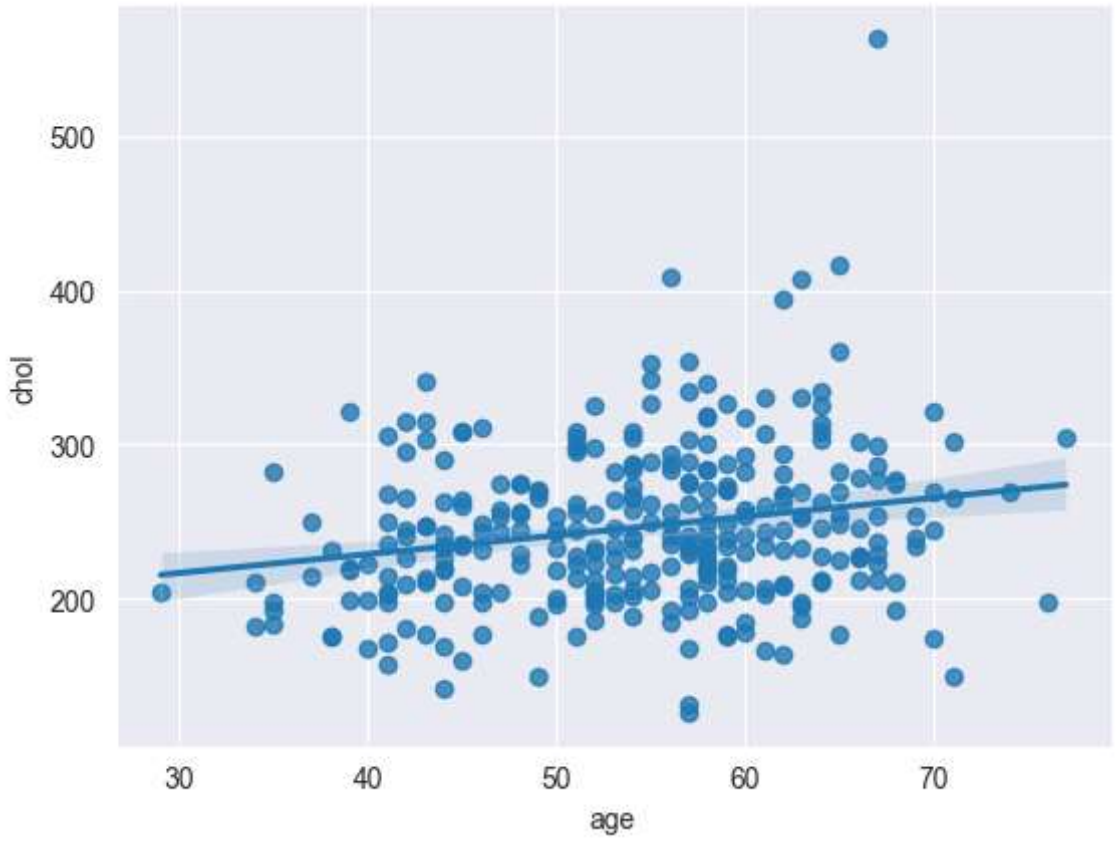
Analyze age and chol variable

In [49]:
```
1 ax=sns.scatterplot(data=heart,x='age',y='chol')
```



In [50]:
```
1 sns.regplot(data=heart,x='age',y='chol')
```

Out[50]: <Axes: xlabel='age', ylabel='chol'>

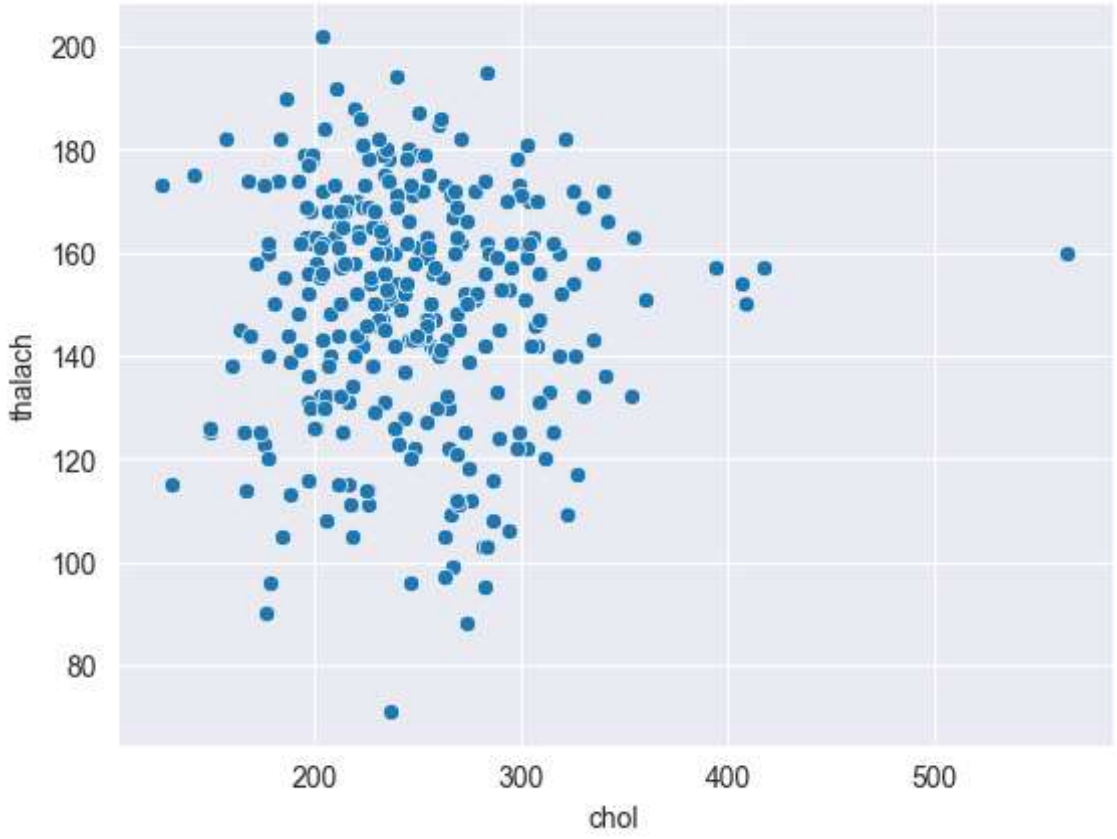


Interpretation

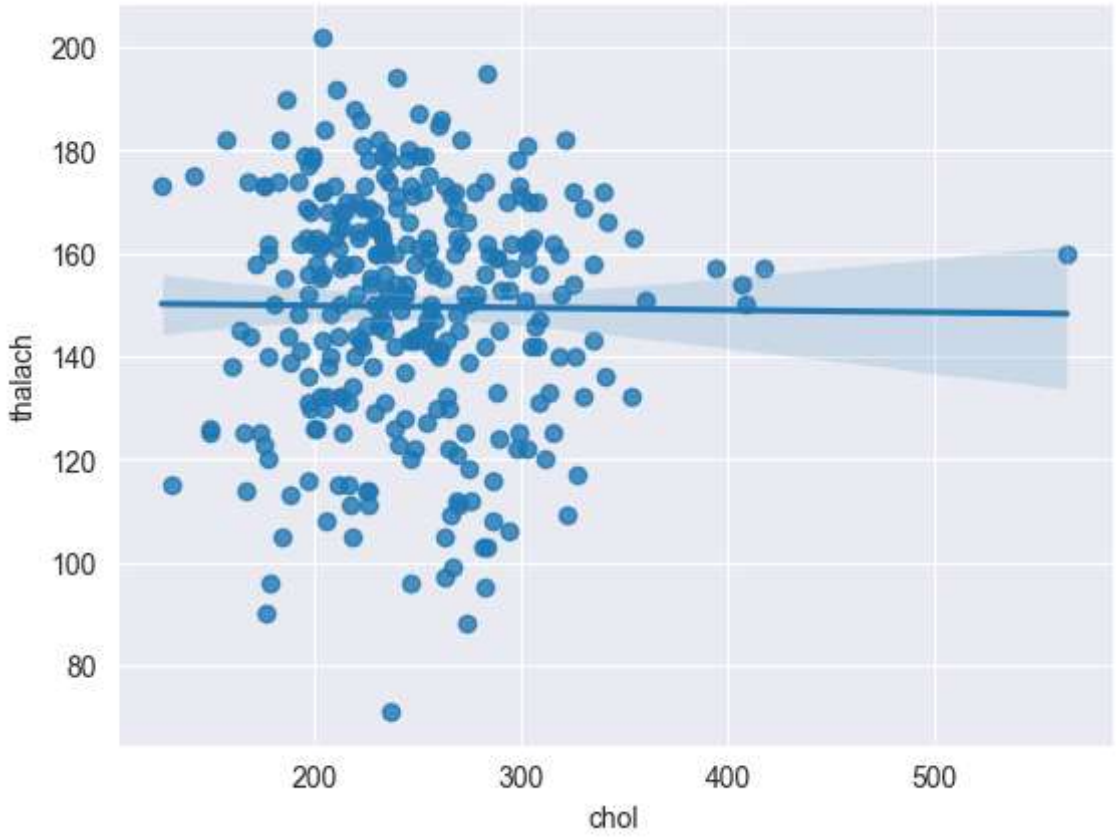
- The above plot confirms that there is a slightly correlation between age and chol variables.

Analyze chol and thalach variable

In [51]:
```
1 ax=sns.scatterplot(data=heart,x='chol',y='thalach')
```



In [52]:
```
1 ax=sns.regplot(data=heart,x='chol',y='thalach')
```



Interpretation

- The above plot shows that there is no correlation between chol and thalach variable

10. Dealing with missing values

- In Pandas missing data is represented by two values:
- **None**: None is a Python singleton object that is often used for missing data in Python code.
- **NaN** : NaN (an acronym for Not a Number), is a special floating-point value recognized by all systems that use the standard IEEE floating-point representation.
- There are different methods in place on how to detect missing values.

Pandas isnull() and notnull() functions

- Pandas offers two functions to test for missing data - `isnull()` and `notnull()` . These are simple functions that return a boolean value indicating whether the passed in argument value is in fact missing data.
- Below, I will list some useful commands to deal with missing values.

Useful commands to detect missing values

- `**df.isnull()**`

The above command checks whether each cell in a dataframe contains missing values or not. If the cell contains missing value, it returns True otherwise it returns False.

- `**df.isnull().sum()**`

The above command returns total number of missing values in each column in the dataframe.

- `**df.isnull().sum().sum()**`

It returns total number of missing values in the dataframe.

- `**df.isnull().mean()**`

It returns percentage of missing values in each column in the dataframe.

- `**df.isnull().any()**`

It checks which column has null values and which has not. The columns which has null values returns TRUE and FALSE otherwise.

- `**df.isnull().any().any()**`

It returns a boolean value indicating whether the dataframe has missing values or not. If dataframe contains missing values it returns TRUE and FALSE otherwise.

- `**df.isnull().values.any()**`

It checks whether a particular column has missing values or not. If the column contains missing values, then it returns TRUE otherwise FALSE.

- `**df.isnull().values.sum()**`

It returns the total number of missing values in the dataframe.

11.Check with ASSERT Statement

- We must confirm that our dataset has no missing values.
- We can write an **assert statement** to verify this.
- We can use an assert statement to programmatically check that no missing, unexpected 0 or negative values are present.
- This gives us confidence that our code is running properly.
- **Assert statement** will return nothing if the value being tested is true and will throw an AssertionError if the value is false.
- **Asserts**
  - assert 1 == 1 (return Nothing if the value is True)
  - assert 1 == 2 (return AssertionError if the value is False)

In [53]:
```
## assert that there are no missing values in the dataset
assert pd.notnull(heart).all().all()
```

In [54]:
```
## assert all values are greater than or equal to 0
assert (heart>=0).all().all()
```

Interpretation

- The above command do not show any error. Hence it is confirmed that there are no missing or negative values in the dataset.
- All the values are greater than or equal to zero.

12. Outlier Detection

I will make boxplots to visualise outliers in the continuous numerical variables : -

`age` , `trestbps` , `chol` , `thalach` and `oldpeak` variables.

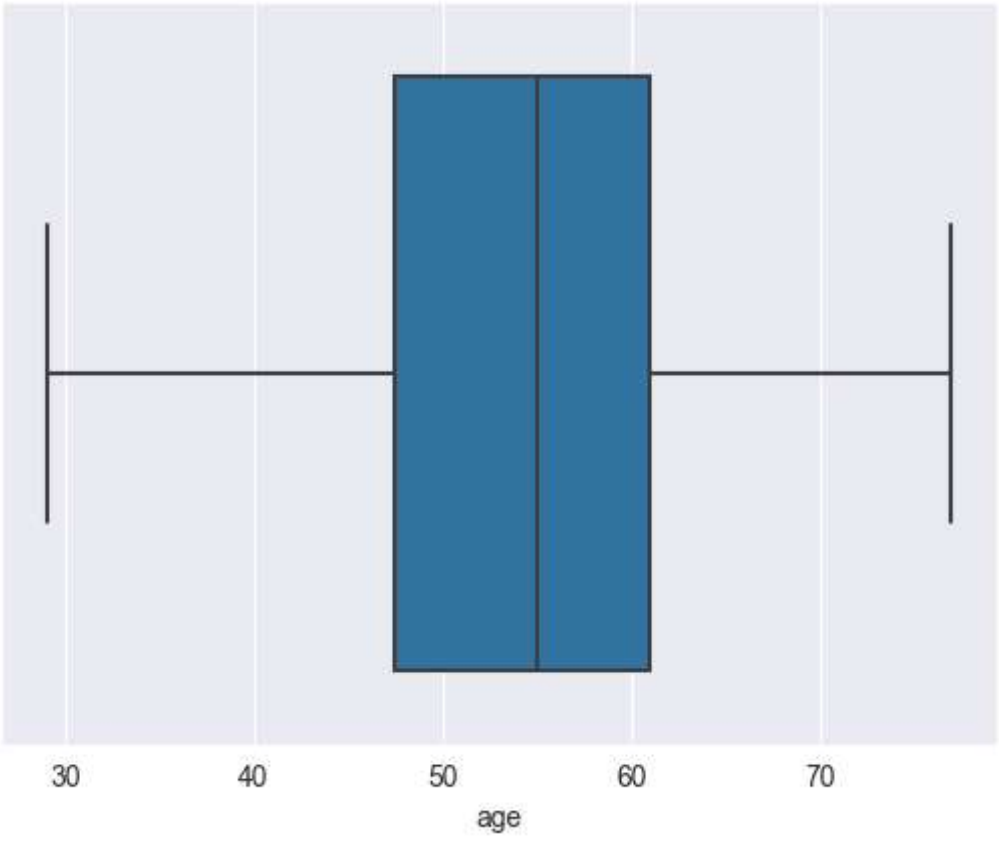
Age

In [55]:
```
heart.age.describe()
```

Out[55]:
```
count    303.000000
mean      54.366337
std        9.082101
min       29.000000
25%       47.500000
50%       55.000000
75%       61.000000
max       77.000000
Name: age, dtype: float64
```

Boxplot of AGE variable

In [56]:
```
1 a=sns.boxplot(x=heart.age)
```



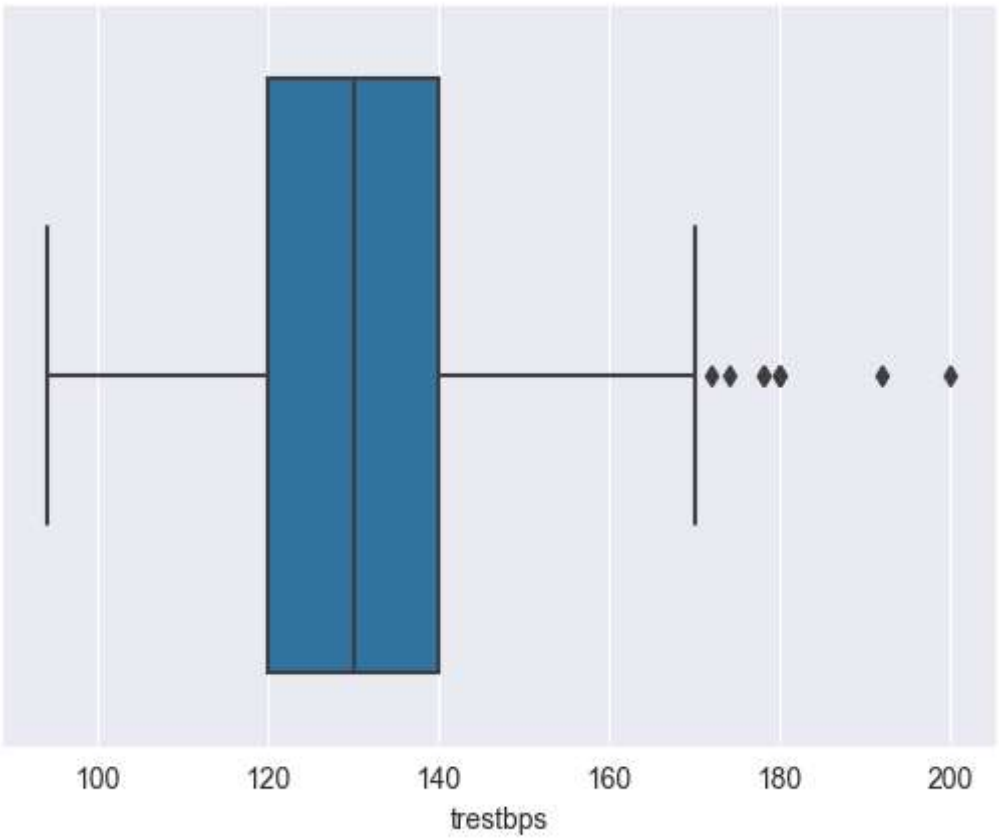
trestbps variable

In [57]:
```
1 heart.trestbps.describe()
```

Out[57]:
```
count    303.000000
mean     131.623762
std       17.538143
min       94.000000
25%      120.000000
50%      130.000000
75%      140.000000
max      200.000000
Name: trestbps, dtype: float64
```

In [58]:
```
1 sns.boxplot(x=heart.trestbps)
```

Out[58]:
```
<Axes: xlabel='trestbps'>
```



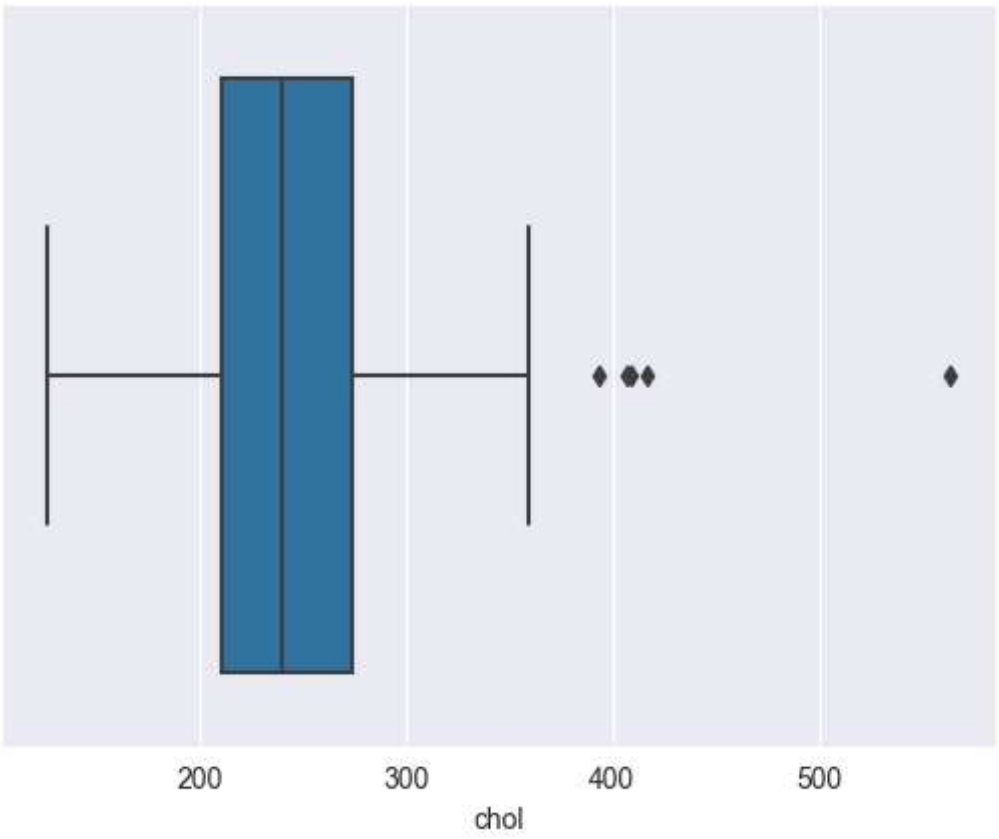
Chol variable

In [59]:
```
1 heart.chol.describe()
```

Out[59]:
```
count    303.000000
mean     246.264026
std       51.830751
min      126.000000
25%      211.000000
50%      240.000000
75%      274.500000
max      564.000000
Name: chol, dtype: float64
```

In [60]:
```
1 sns.boxplot(x=heart.chol)
```

Out[60]:
```
<Axes: xlabel='chol'>
```



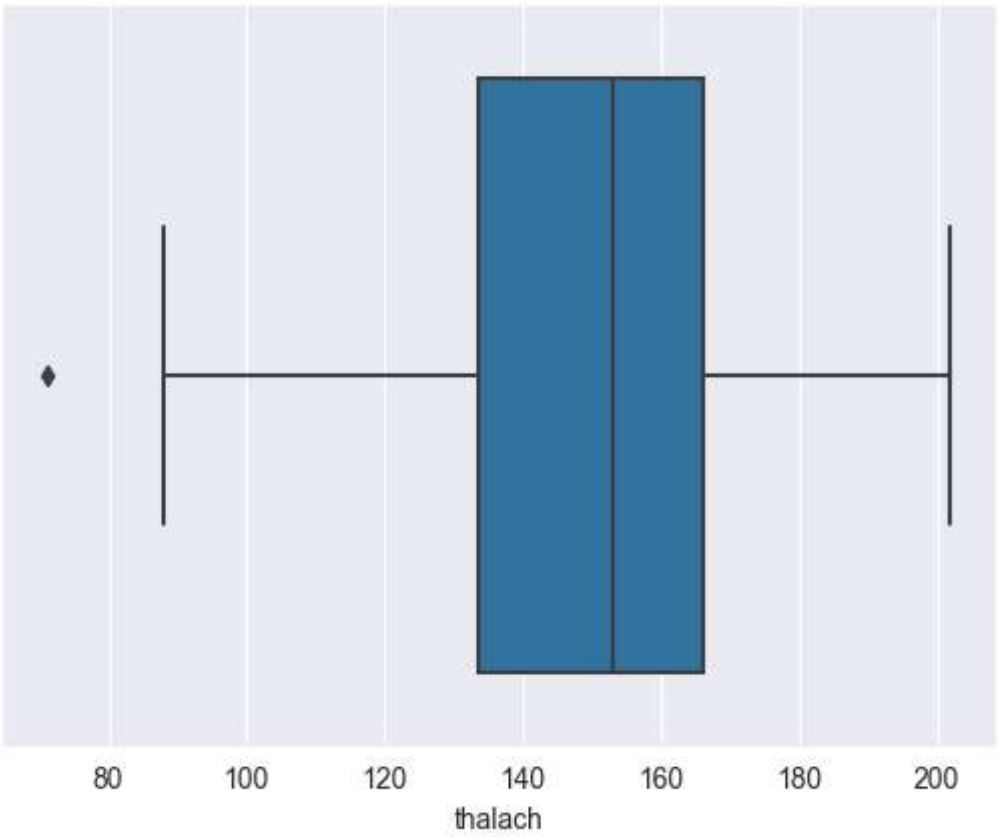
thalach variable

In [61]:
```
1 heart.thalach.describe()
```

Out[61]:
```
count    303.000000
mean     149.646865
std       22.905161
min       71.000000
25%      133.500000
50%      153.000000
75%      166.000000
max      202.000000
Name: thalach, dtype: float64
```

In [62]:
```
1 sns.boxplot(x=heart.thalach)
```

Out[62]:
```
<Axes: xlabel='thalach'>
```



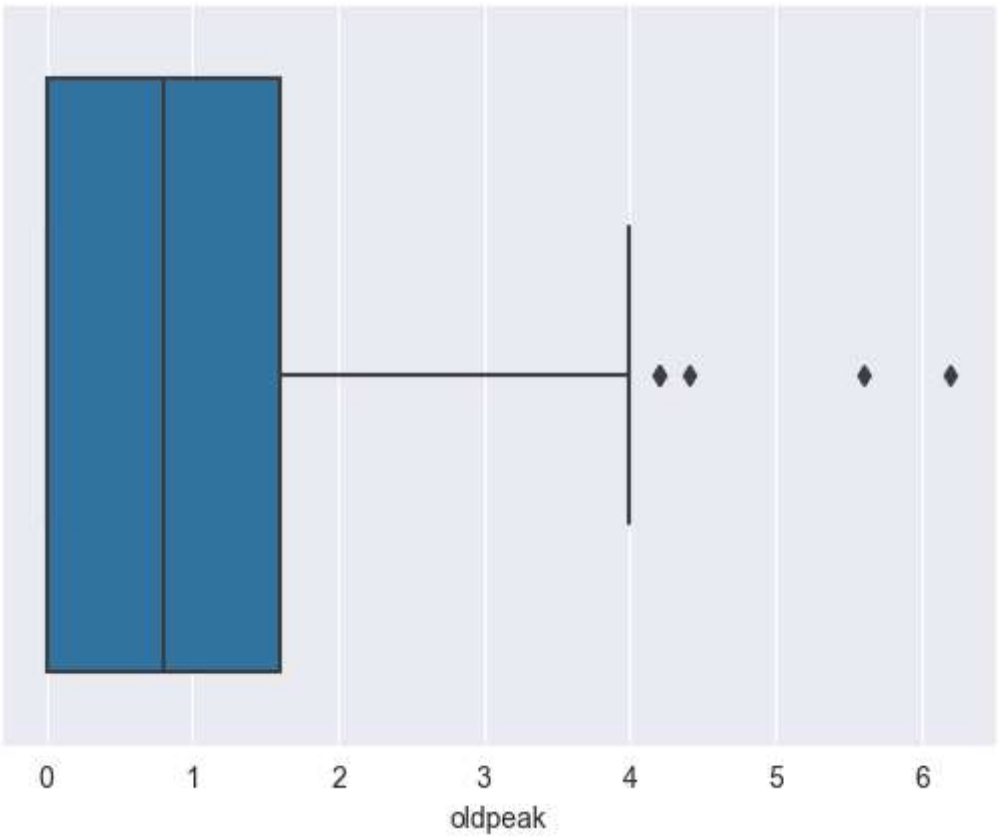
oldpeak

In [63]:
```
1 heart.oldpeak.describe()
```

Out[63]:
```
count    303.000000
mean       1.039604
std        1.161075
min        0.000000
25%        0.000000
50%        0.800000
75%        1.600000
max        6.200000
Name: oldpeak, dtype: float64
```

In [64]:
```
1 sns.boxplot(x=heart.oldpeak)
```

Out[64]: <Axes: xlabel='oldpeak'>



Interpretation

- The age variable does not contain any outlier.
- trestbps variable contains outliers to the right side.
- chol variable also contains outliers to the right side.
- thalach variable contains a single outlier to the left side.
- oldpeak variable contains outliers to the right side.
- Those variables containing outliers needs further investigation.

Conclusion

- In ths project, we have explored the heart disease dataset.
- The feature variable of interest is target variable. We have analyzed it alone and check its interaction with other variables.
- We have also discussed how to detect missing data and outliers.
- In this project, we have implemented many of the strategies presented in the book **Think Stats- Exploratory Data Analysis in Python by Allen B Downey**.

Thank You!!!