In [1]:

```
# EXPLORATORY DATA ANALYSIS
```

INTRODUCTION TO EDA(EXPLORATORY DATA ANALYSIS)

In [3]:

```
import seaborn as sns
import matplotlib.pyplot as plt
import scipy.stats as st
%matplotlib inline
```

In [4]:

```
# ignore warnings
import warnings
warnings.filterwarnings('ignore')
```

In [5]:

```
import pandas as pd
```

In [6]:

```
# import dataset
df = pd.read_excel(r"C:\Users\Vansh\OneDrive\Documents\HEART ANALYSIS.xlsx")
df
```

Out[6]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	t
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	
...	...	...	...	...	...	...	...	...	...	...	...	...	
298	57	0	0	140	241	0	1	123	1	0.2	1	0	
299	45	1	3	110	264	0	1	132	0	1.2	1	0	
300	68	1	0	144	193	1	1	141	0	3.4	1	2	
301	57	1	0	130	131	0	1	115	1	1.2	1	1	
302	57	0	1	130	236	0	0	174	0	0.0	1	1	

303 rows × 14 columns

In [7]:

```
# print the shape
print('The shape of the dataset : ', df.shape)
```

```
The shape of the dataset :  (303, 14)
```

In [8]:

```
#now we can see the dataset contains 303 instances and 14 variables
```

In [9]:

```
# preview the dataset
```

In [10]:
```
df.head()
```

Out[10]:

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

In [11]:
```
# summary of dataset
df.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 [12]:
```
df.dtypes
```

Out[12]:
```
age           int64
sex           int64
cp            int64
trestbps      int64
chol          int64
fbs           int64
restecg       int64
thalach       int64
exang         int64
oldpeak     float64
slope         int64
ca            int64
thal          int64
target        int64
dtype: object
```

In [13]:
```
# stastical proprties of dataset
```

```
df.describe()
```

Out[13]:

	age	sex	cp	trestbps	chol	fbs	reste
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.0000
mean	54.366337	0.683168	0.966997	131.623762	246.264026	0.148515	0.5280
std	9.082101	0.466011	1.032052	17.538143	51.830751	0.356198	0.525
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.0000
25%	47.500000	0.000000	0.000000	120.000000	211.000000	0.000000	0.0000
50%	55.000000	1.000000	1.000000	130.000000	240.000000	0.000000	1.0000
75%	61.000000	1.000000	2.000000	140.000000	274.500000	0.000000	1.0000
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	2.0000

In [14]:

```
# view column names
```

In [15]:

```
df.columns
```

Out[15]:

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

UNIVARIATE ANALYSIS

In [17]:

```
df['target'].nunique()
```

Out[17]:

```
2
```

In [18]:

```
df['target'].unique()
```

Out[18]:

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

frequency distribution of target variable

In [20]:

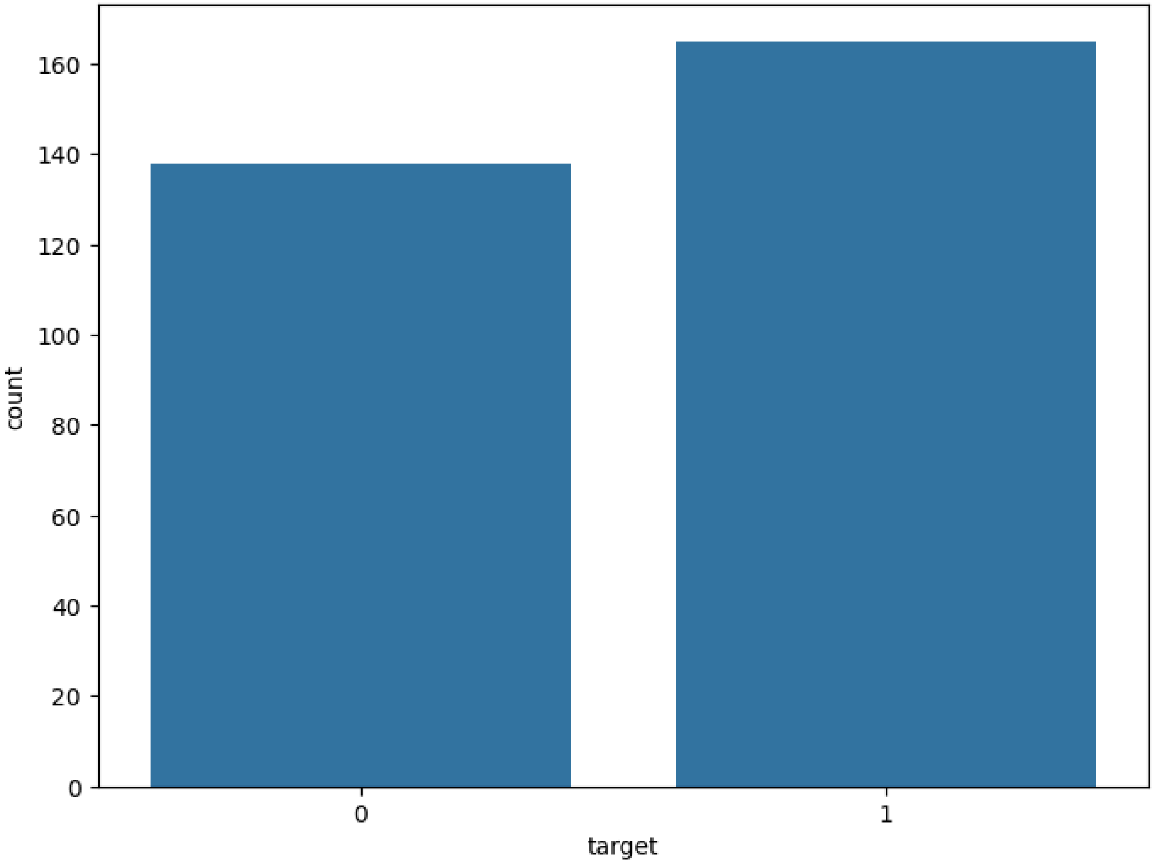
```
df['target'].value_counts()
```

Out[20]:

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

In [21]:

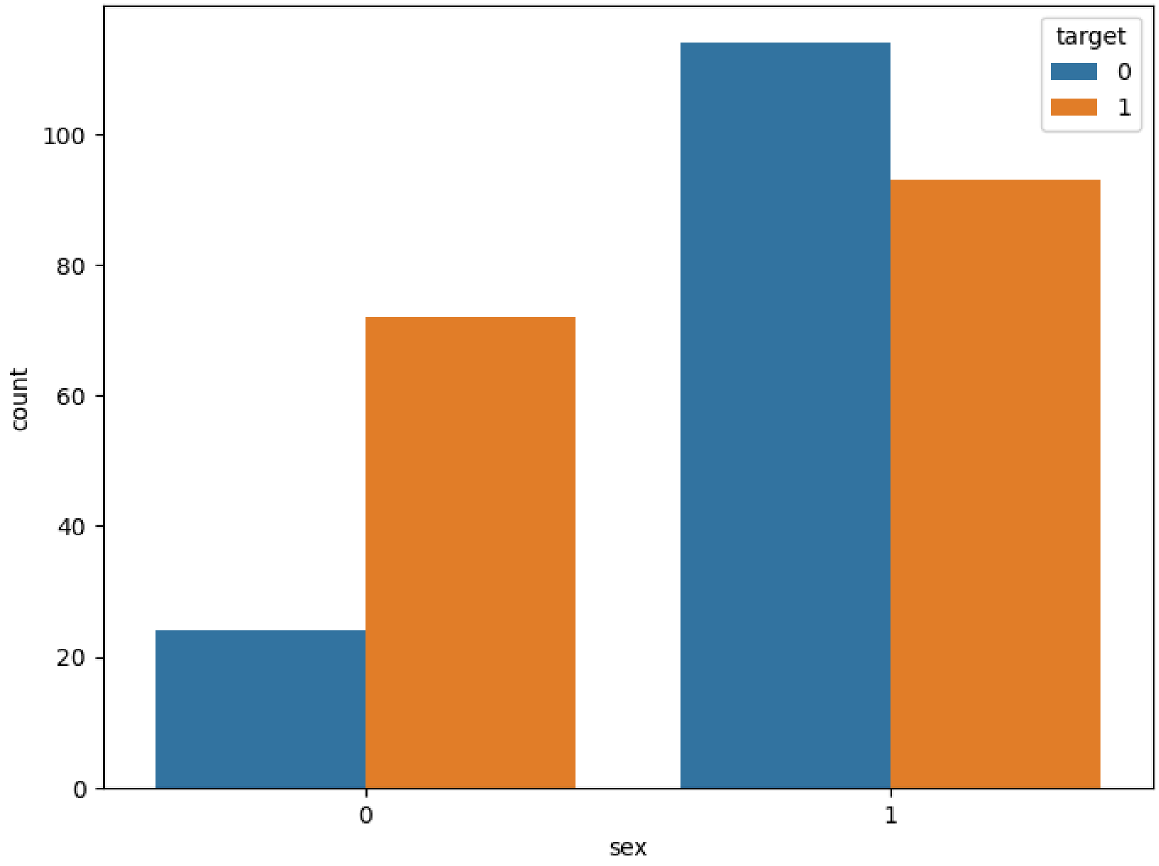
```
f,ax = plt.subplots(figsize=(8,6))
ax = sns.countplot(x="target",data=df)
plt.show()
```



```
In [22]: df.groupby('sex')['target'].value_counts()
```

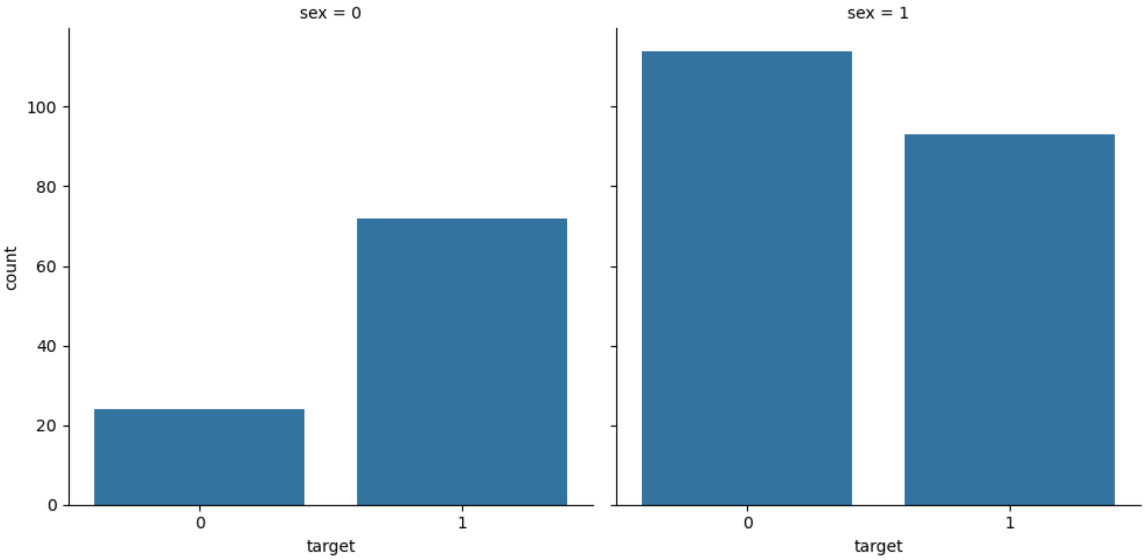
```
Out[22]: sex  target
         0    1          72
              0          24
         1    0         114
              1          93
         Name: count, dtype: int64
```

```
In [23]: f, ax = plt.subplots(figsize=(8, 6))
         ax = sns.countplot(x="sex", hue="target", data=df)
         plt.show()
```



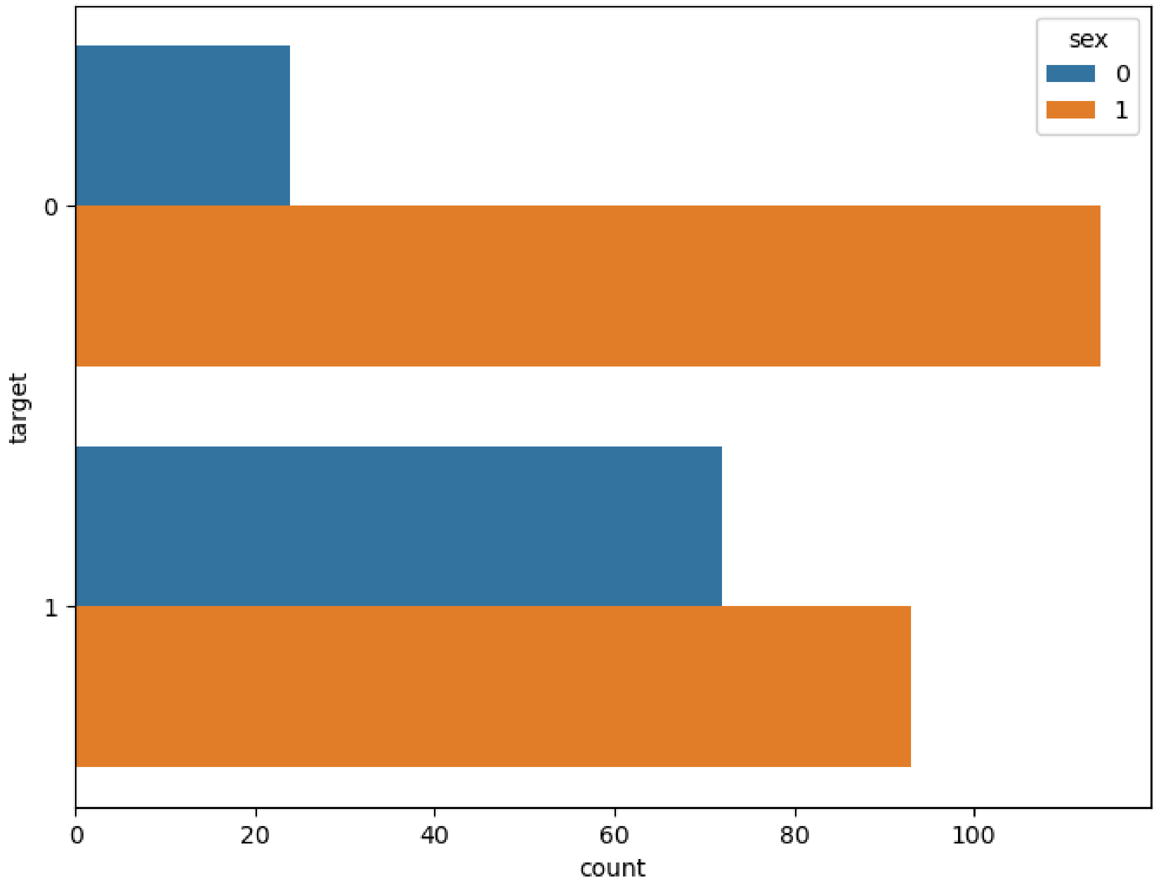
In [24]:

```python
ax = sns.catplot(x="target", col="sex", data=df, kind="count", height=5, aspect=
plt.show()
```



In [25]:

```python
f, ax = plt.subplots(figsize=(8, 6))
ax = sns.countplot(y="target", hue="sex", data=df)
plt.show()
```

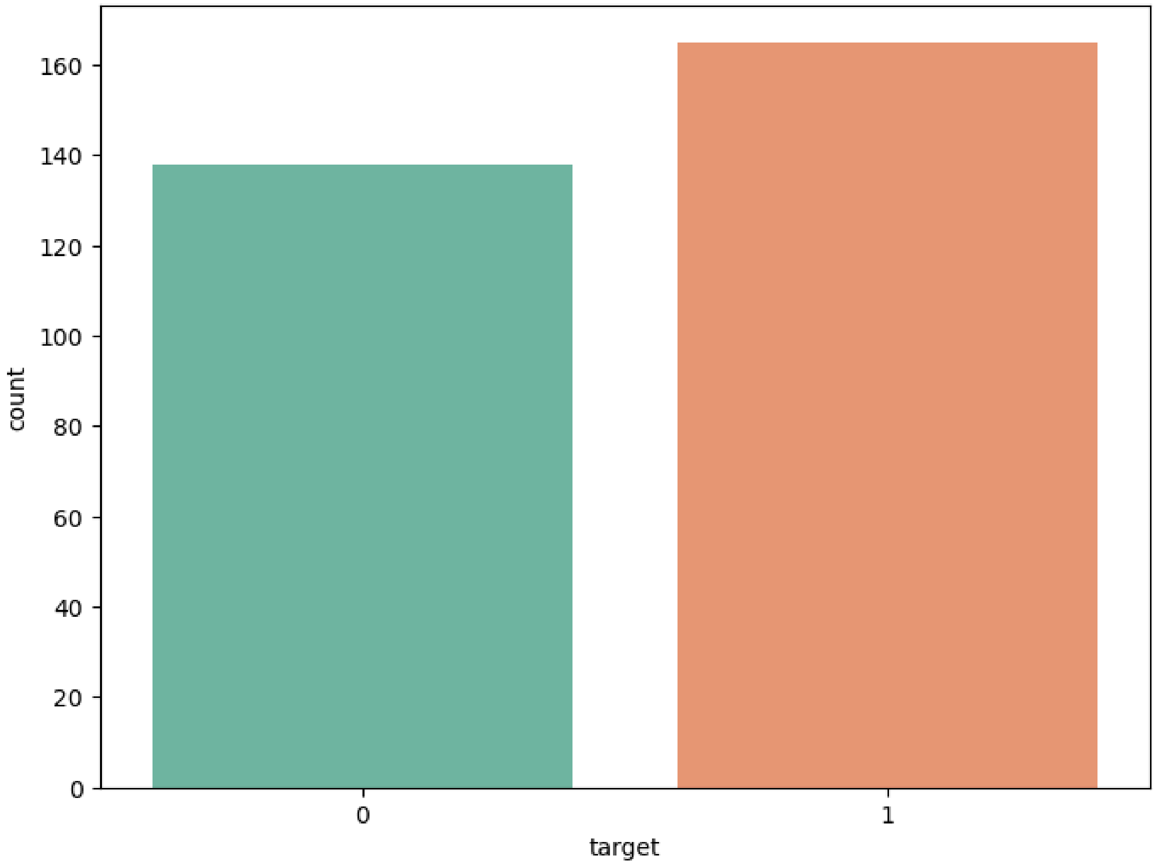


In [26]:

```
# we can use a different color palatte
```

In [27]:

```
f, ax = plt.subplots(figsize=(8,6))
ax = sns.countplot(x="target",data=df,palette= "Set2")
plt.show()
```

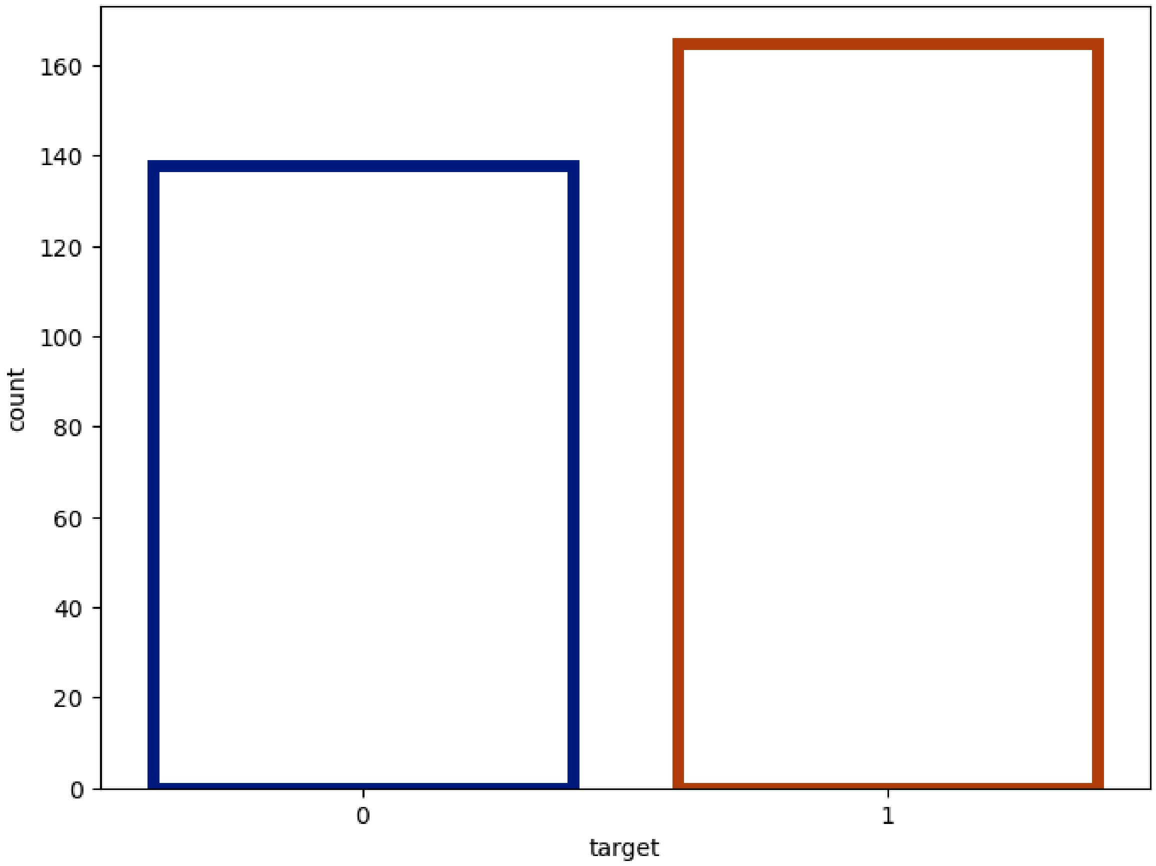


In [28]:

```
# we can use plt.bar
```

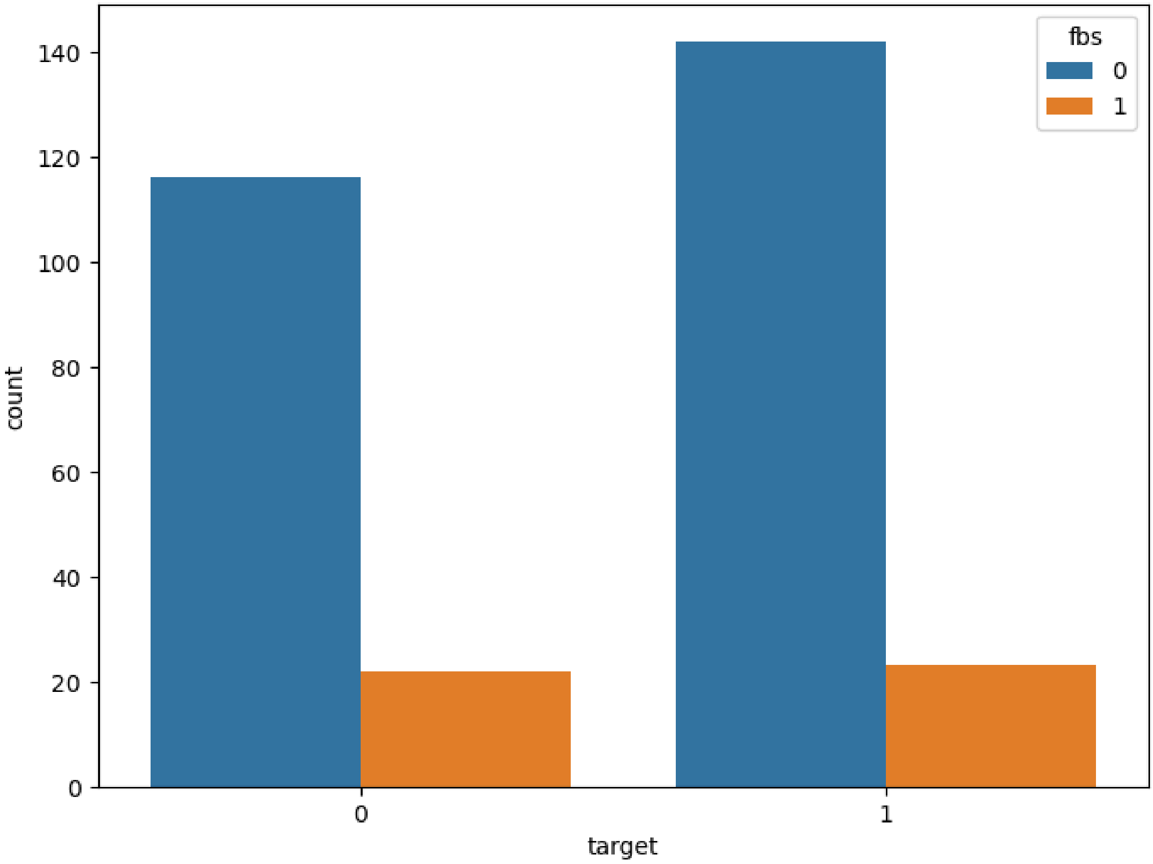
In [29]:

```
f, ax = plt.subplots(figsize=(8, 6))
ax = sns.countplot(x="target", data=df, facecolor=(0, 0, 0, 0), linewidth=5, edg
plt.show()
```



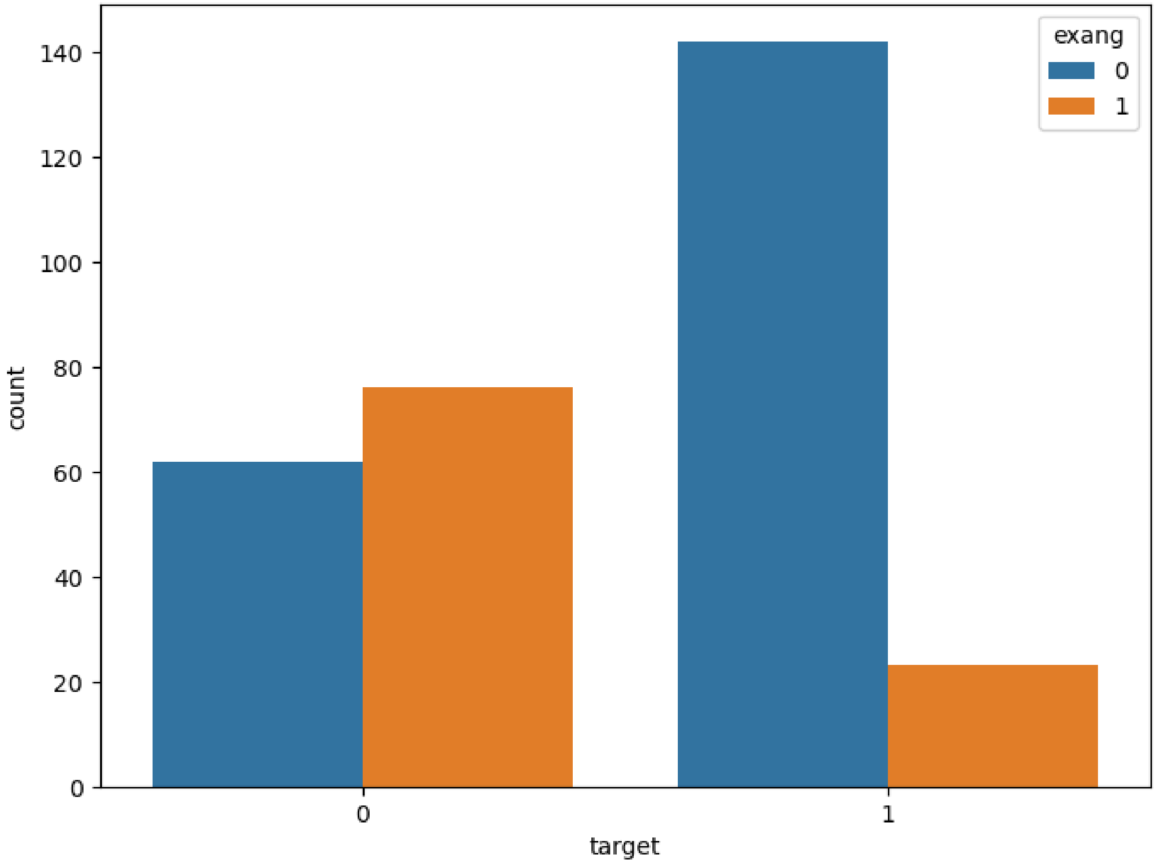
In [30]:

```
f,ax = plt.subplots(figsize=(8,6))
ax = sns.countplot(x="target",hue="fbs",data=df)
plt.show()
```



In [31]:

```
f, ax = plt.subplots(figsize=(8, 6))
ax = sns.countplot(x="target", hue="exang", data=df)
plt.show()
```



bivariate analysis

```
In [33]: df.corr()
```

Out[33]:

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

In [34]:

```
correlation = df.corr()
```

In [35]:

```
correlation['target'].sort_values(ascending=False)
```

Out[35]:

```
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
```

ANALYSIS OF TARGET AND CP VARIABLE

In [37]:

```
df['cp'].nunique()
```

Out[37]:

```
4
```

In [38]:

```
df['cp'].value_counts()
```

Out[38]:

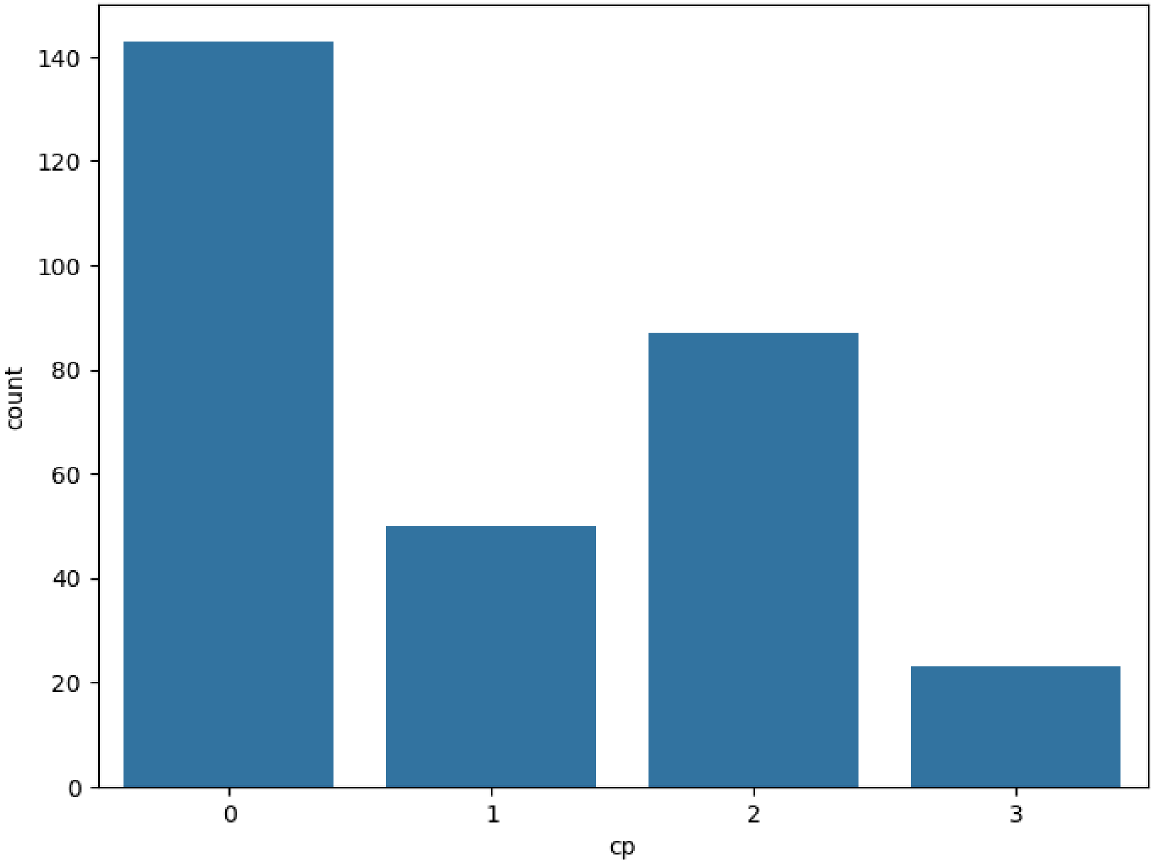
```
cp
0    143
2     87
1     50
3     23
Name: count, dtype: int64
```

In [39]:

```
#visualize the frequency distribution of cp variable
```

In [40]:

```
f,ax = plt.subplots(figsize=(8,6))
ax= sns.countplot(x="cp",data=df)
plt.show()
```



In [41]:

```
# frequency  distribution of target variable wrt cp
```

In [42]:

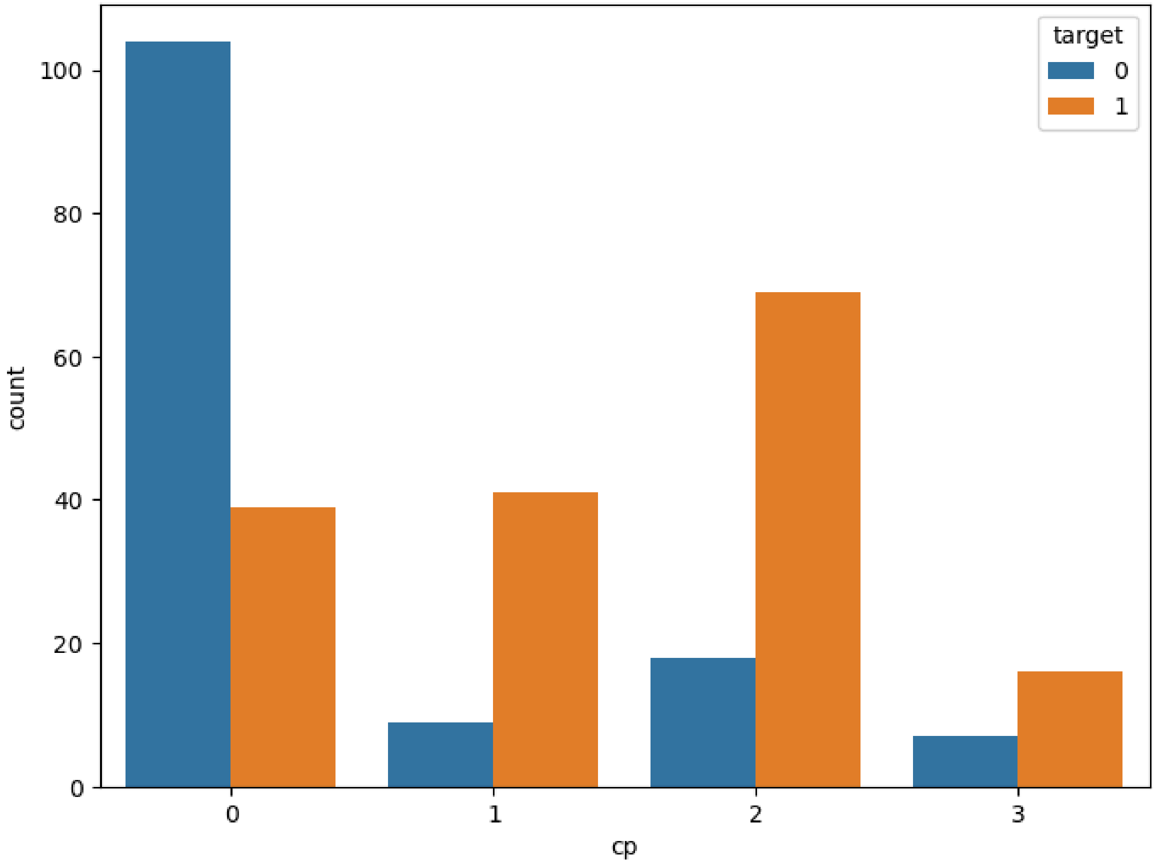
```
df.groupby('cp')['target'].value_counts()
```

Out[42]:

```
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
```

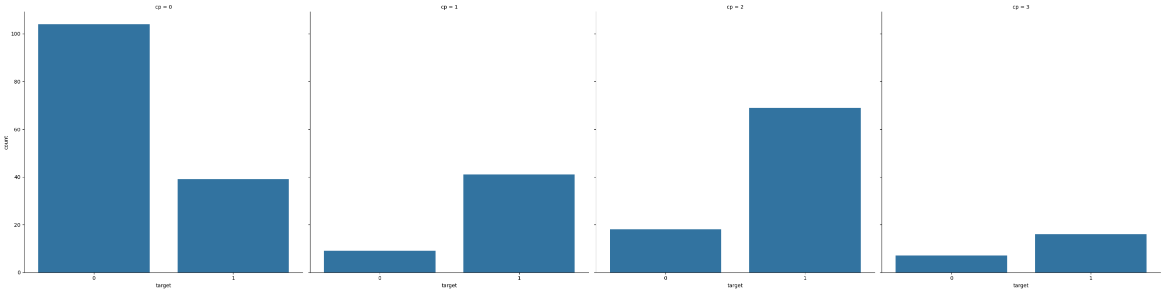
In [43]:

```
f,ax = plt.subplots(figsize=(8,6))
ax = sns.countplot(x="cp",hue="target",data=df)
plt.show()
```



In [44]:
```
# INTERPRETATION
```

In [45]:
```
ax = sns.catplot(x="target",col="cp",data=df,kind="count",height=8,aspect=1)
plt.show()
```



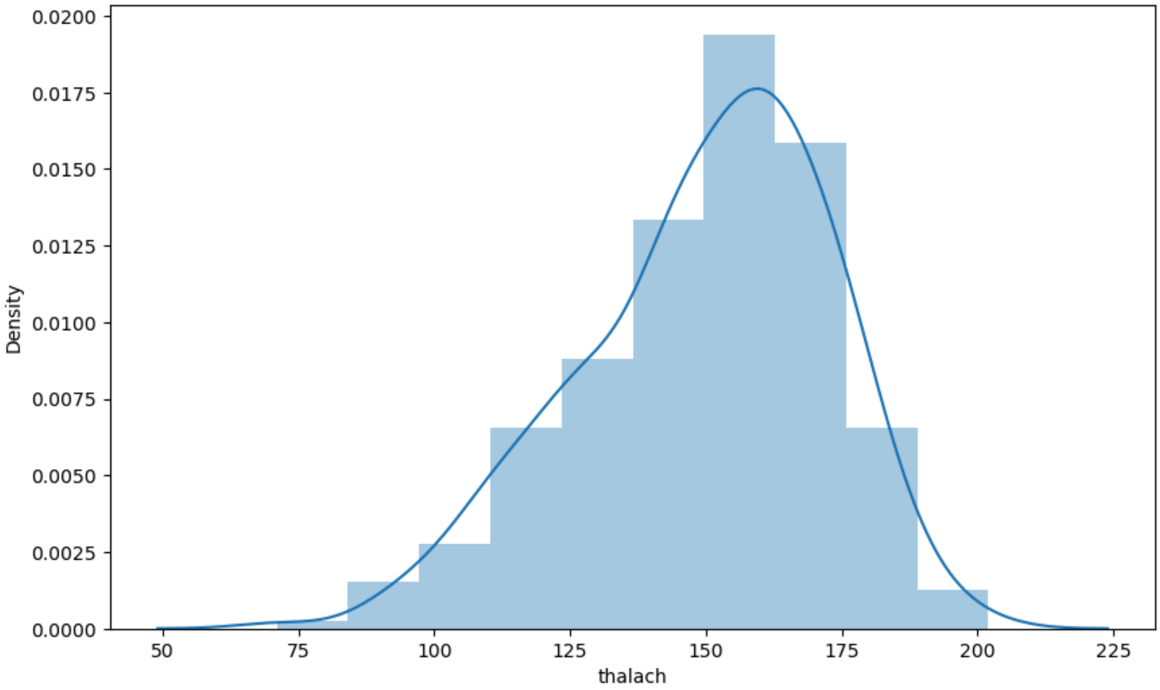
In [46]:
```
# ANALYSIS OF TARGET AND THALACH VARIABLE
```

In [47]:
```
df['thalach'].nunique()
```

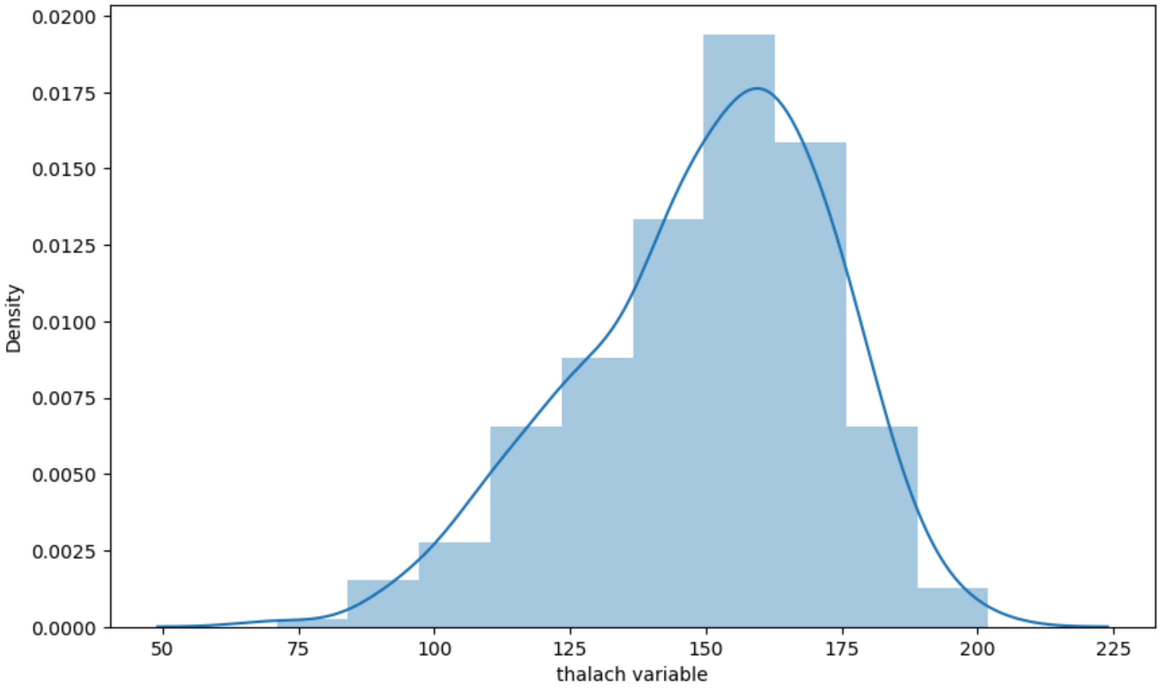
Out[47]:
```
91
```

In [48]:
```
#visualize the frequency distribution of thalach variable
```

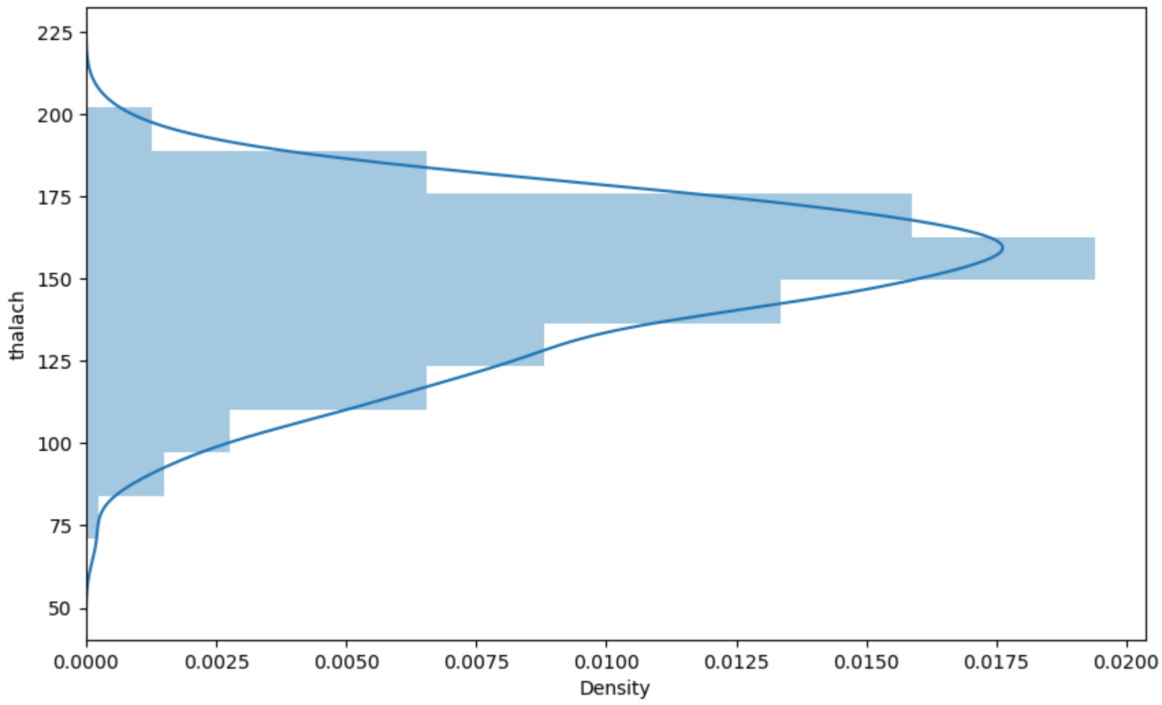
In [49]:
```
f,ax = plt.subplots(figsize=(10,6))
x = df['thalach']
ax = sns.distplot(x,bins=10)
plt.show()
```



In [50]:
```
f,ax = plt.subplots(figsize=(10,6))
x=df['thalach']
x = pd.Series(x,name="thalach variable")
ax=sns.distplot(x,bins=10)
plt.show()
```

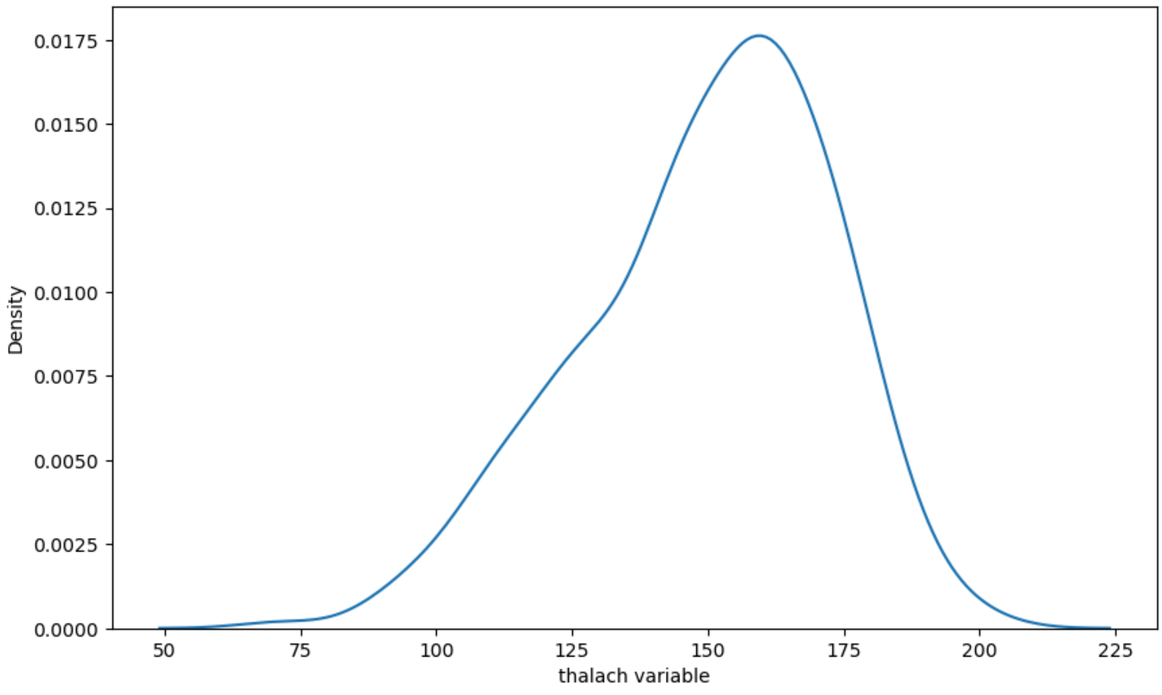


In [51]:
```
f,ax = plt.subplots(figsize=(10,6))
x= df['thalach']
ax = sns.distplot(x,bins=10,vertical=True)
plt.show()
```



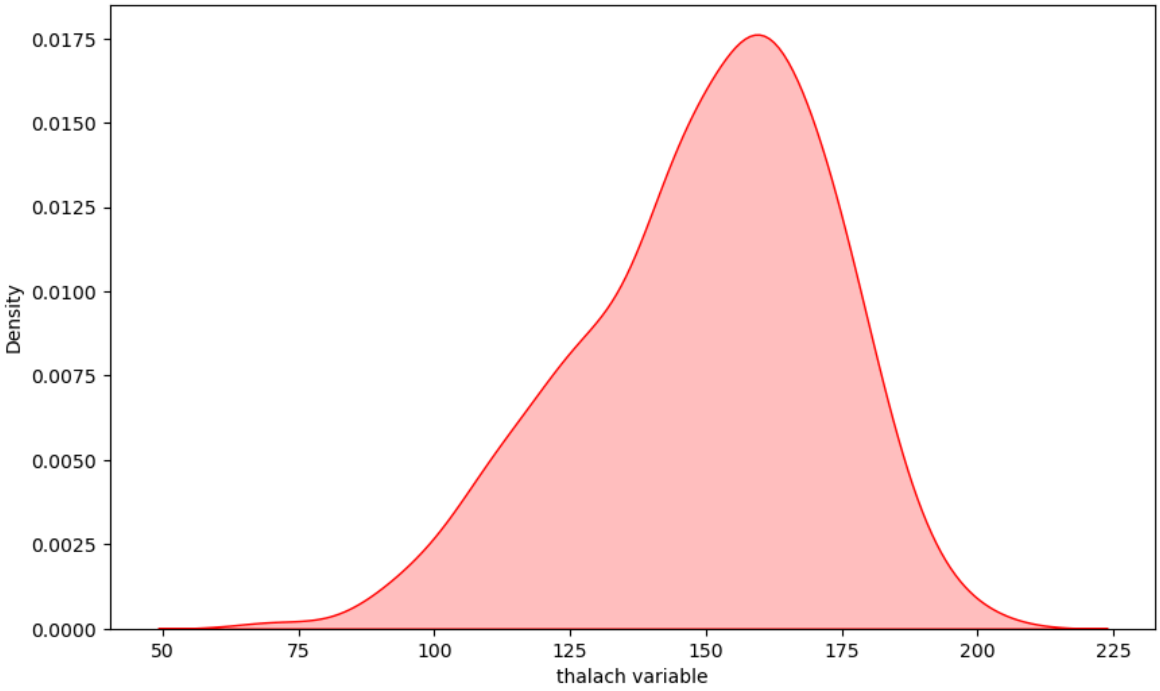
SEABORN KERNEL DENSITY ESTIMATION (KDE) PLOT

In [53]:
```
f,ax = plt.subplots(figsize=(10,6))
x = df['thalach']
x = pd.Series(x,name="thalach variable")
ax = sns.kdeplot(x)
plt.show()
```



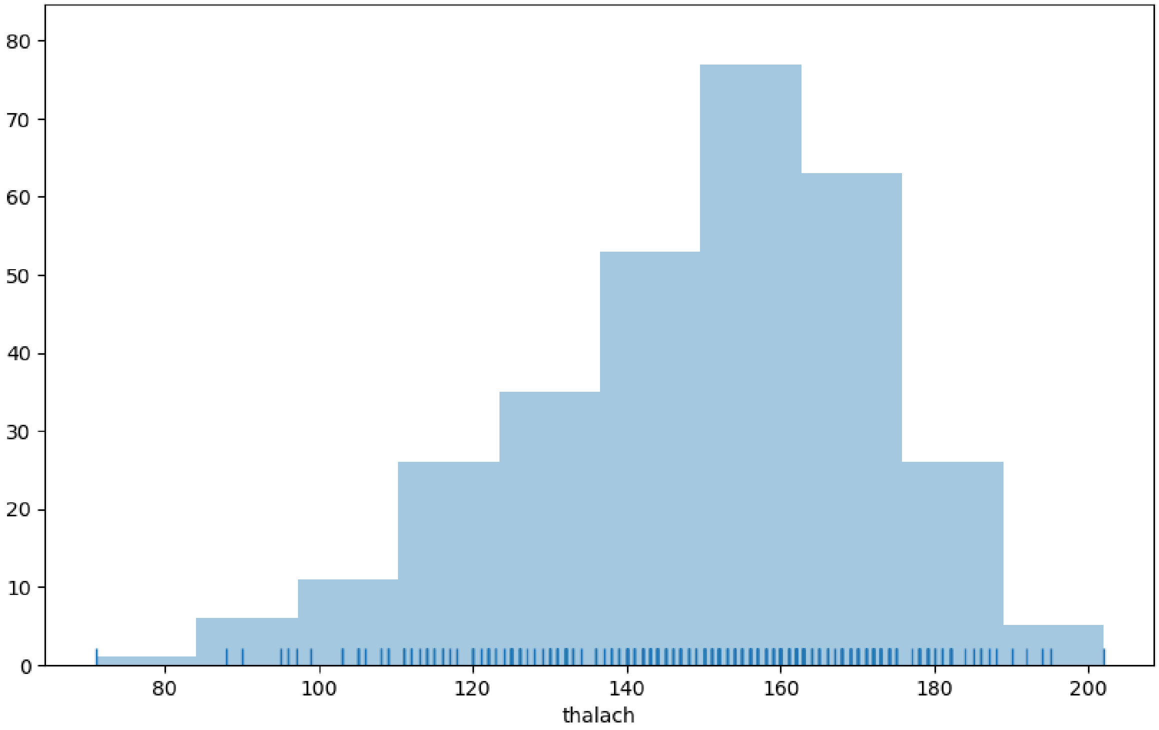
In [54]:
```
f, ax = plt.subplots(figsize=(10,6))
x = df['thalach']
x = pd.Series(x, name="thalach variable")
```

```
ax = sns.kdeplot(x, shade=True, color='r')
plt.show()
```



HISTOGRAM

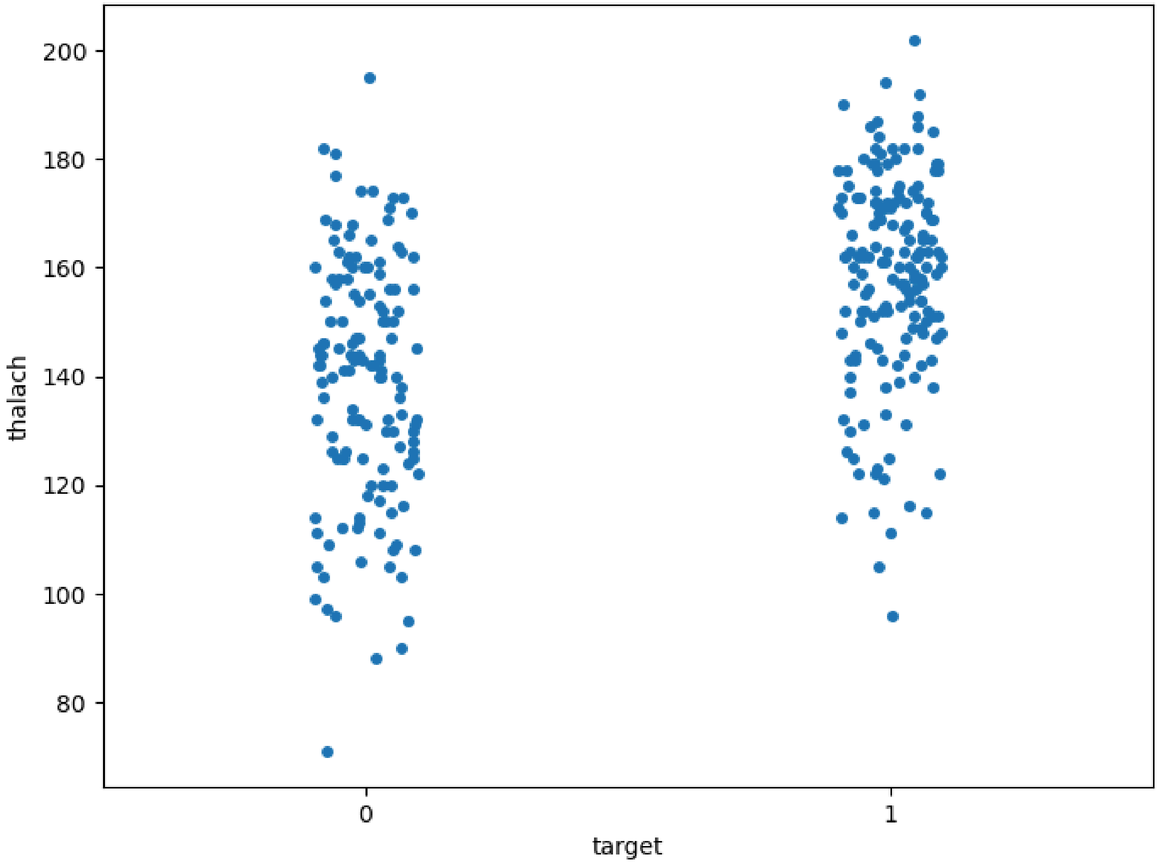
In [56]:
```
f,ax = plt.subplots(figsize=(10,6))
x = df['thalach']
ax = sns.distplot(x,kde= False,rug=True,bins = 10)
plt.show()
```



In [57]:
```
# visualize frequency distribution of thalach variable wrt target
```

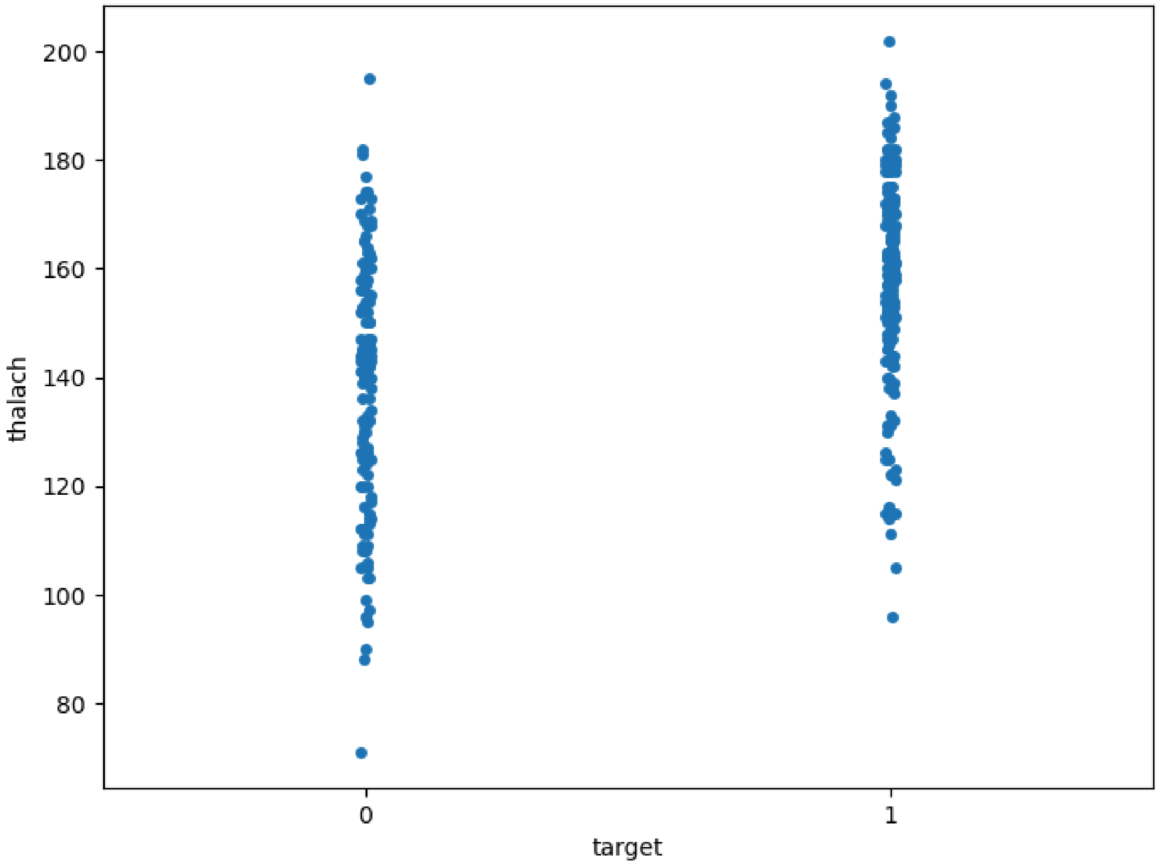
In [58]:

```
f,ax = plt.subplots(figsize=(8,6))
sns.stripplot(x ="target", y="thalach", data=df)
plt.show()
```



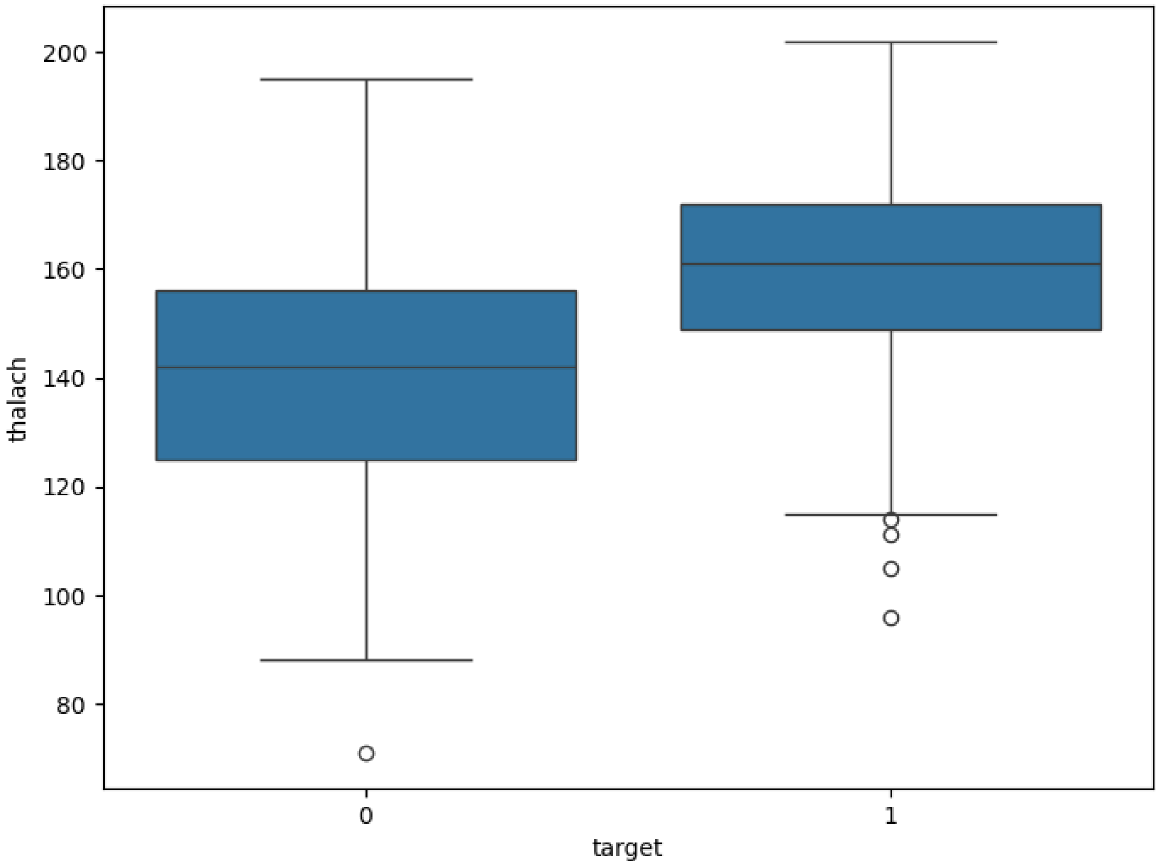
In [59]:

```
f,ax =  plt.subplots(figsize=(8,6))
sns.stripplot( x="target" , y = "thalach",data=df,jitter = 0.01)
plt.show()
```



```
In [60]: # VISUALIZE DISTRIBUTION OF THALACH VARIABLE WRT TARGET WITH BOXPLOT
```

```
In [61]: f,ax = plt.subplots(figsize=(8,6))
         sns.boxplot(x ="target", y="thalach",data=df)
         plt.show()
```



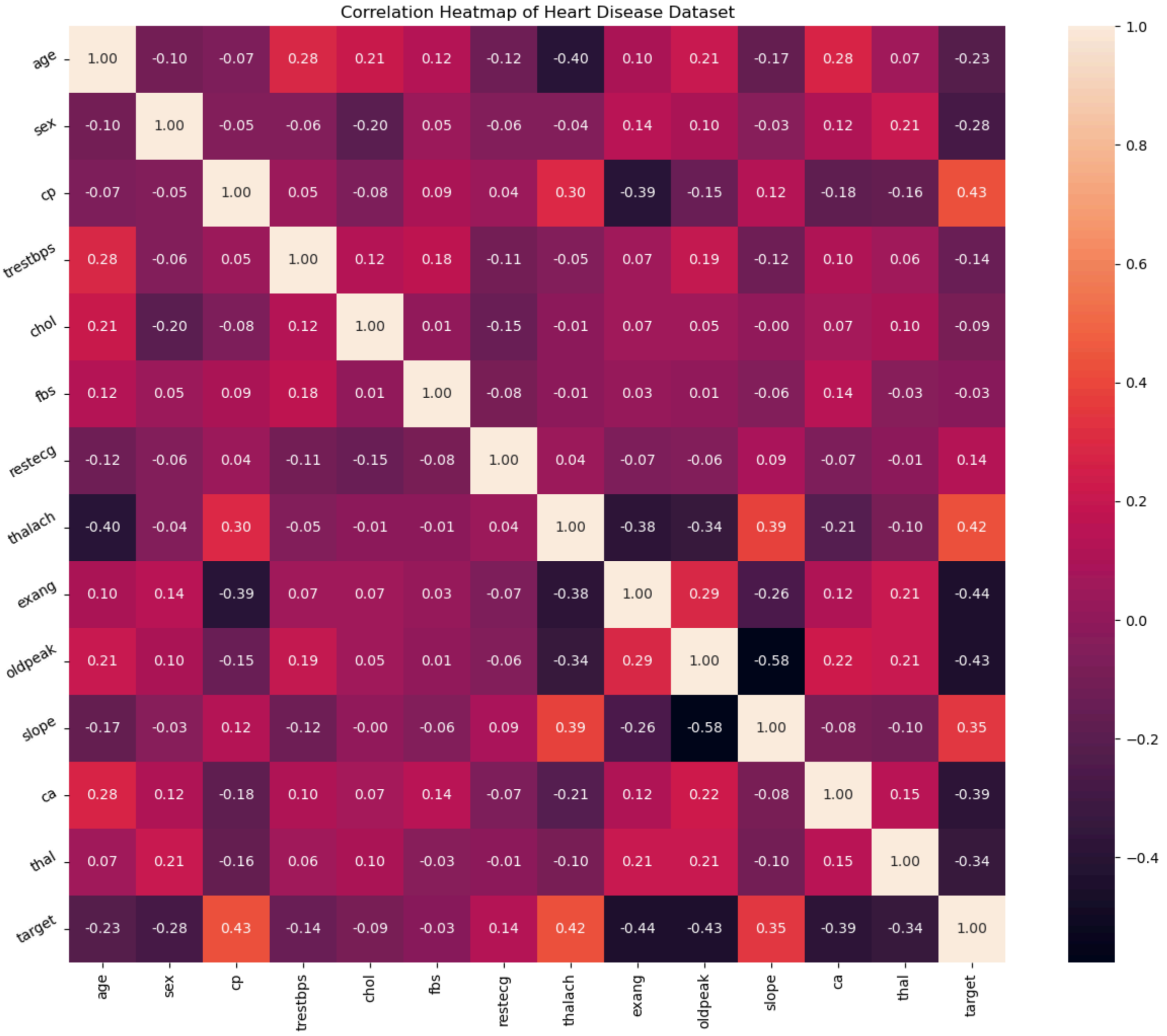
In [62]:

```
# interpretation
```

HEAT MAP

In [64]:

```
plt.figure(figsize=(16,12))
plt.title('Correlation Heatmap of Heart Disease Dataset')
a = sns.heatmap(correlation, square=True, annot=True, fmt='.2f', linecolor='whit
a.set_xticklabels(a.get_xticklabels(), rotation=90)
a.set_yticklabels(a.get_yticklabels(), rotation=30)
plt.show()
```

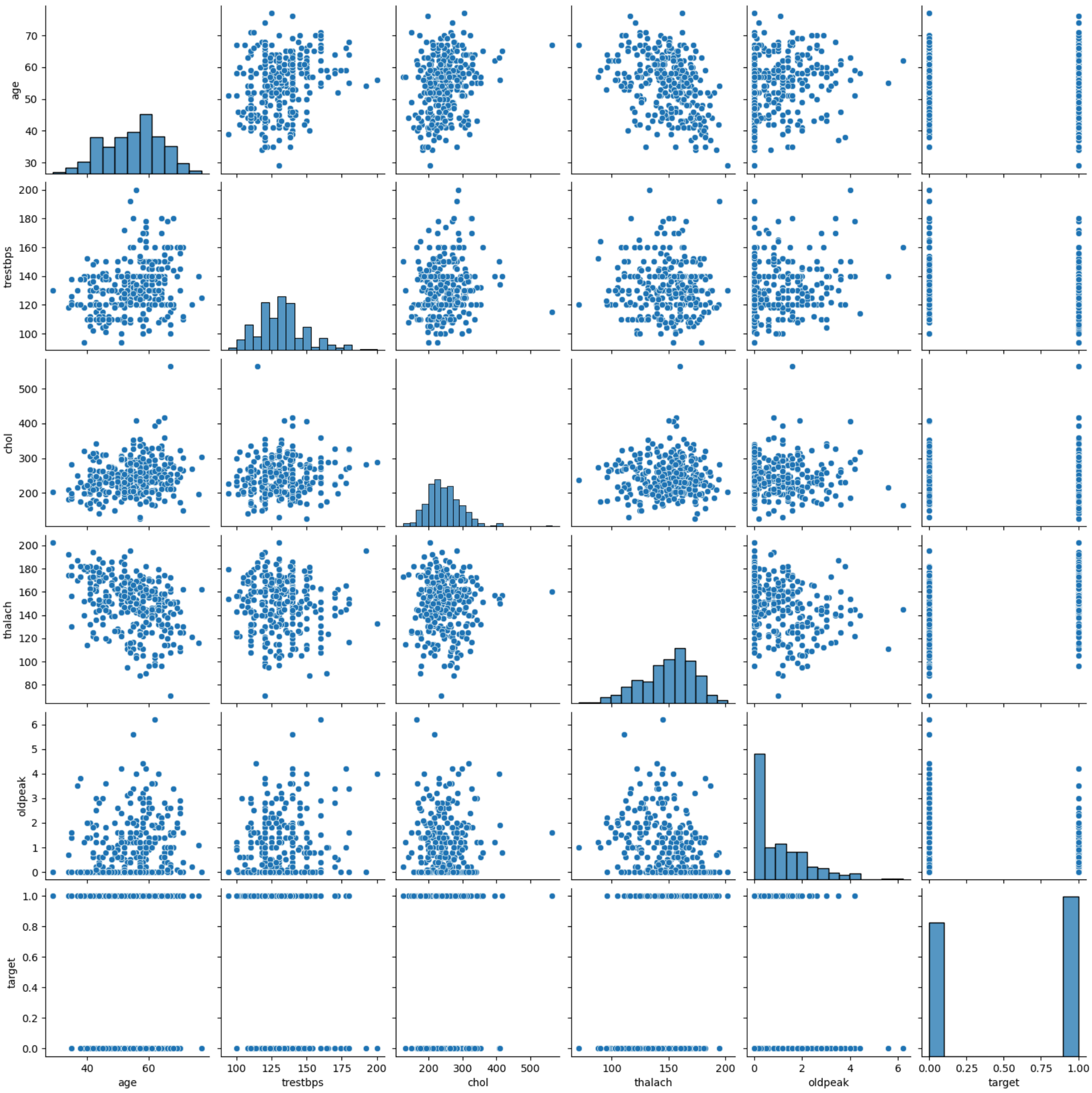


	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
age	1.00	-0.10	-0.07	0.28	0.21	0.12	-0.12	-0.40	0.10	0.21	-0.17	0.28	0.07	-0.23
sex	-0.10	1.00	-0.05	-0.06	-0.20	0.05	-0.06	-0.04	0.14	0.10	-0.03	0.12	0.21	-0.28
cp	-0.07	-0.05	1.00	0.05	-0.08	0.09	0.04	0.30	-0.39	-0.15	0.12	-0.18	-0.16	0.43
trestbps	0.28	-0.06	0.05	1.00	0.12	0.18	-0.11	-0.05	0.07	0.19	-0.12	0.10	0.06	-0.14
chol	0.21	-0.20	-0.08	0.12	1.00	0.01	-0.15	-0.01	0.07	0.05	-0.00	0.07	0.10	-0.09
fbs	0.12	0.05	0.09	0.18	0.01	1.00	-0.08	-0.01	0.03	0.01	-0.06	0.14	-0.03	-0.03
restecg	-0.12	-0.06	0.04	-0.11	-0.15	-0.08	1.00	0.04	-0.07	-0.06	0.09	-0.07	-0.01	0.14
thalach	-0.40	-0.04	0.30	-0.05	-0.01	-0.01	0.04	1.00	-0.38	-0.34	0.39	-0.21	-0.10	0.42
exang	0.10	0.14	-0.39	0.07	0.07	0.03	-0.07	-0.38	1.00	0.29	-0.26	0.12	0.21	-0.44
oldpeak	0.21	0.10	-0.15	0.19	0.05	0.01	-0.06	-0.34	0.29	1.00	-0.58	0.22	0.21	-0.43
slope	-0.17	-0.03	0.12	-0.12	-0.00	-0.06	0.09	0.39	-0.26	-0.58	1.00	-0.08	-0.10	0.35
ca	0.28	0.12	-0.18	0.10	0.07	0.14	-0.07	-0.21	0.12	0.22	-0.08	1.00	0.15	-0.39
thal	0.07	0.21	-0.16	0.06	0.10	-0.03	-0.01	-0.10	0.21	0.21	-0.10	0.15	1.00	-0.34
target	-0.23	-0.28	0.43	-0.14	-0.09	-0.03	0.14	0.42	-0.44	-0.43	0.35	-0.39	-0.34	1.00

PAIR PLOT

In [132…

```
num_var = ['age','trestbps','chol','thalach','oldpeak','target']
sns.pairplot(df[num_var],kind= 'scatter',diag_kind = 'hist')
plt.show()
```



In [134…

```
df['age'].nunique()
```

Out[134…

```
41
```

In [136…

```
df['age'].nunique()
```

Out[136…

```
41
```

In [138…

```
df['age'].describe()
```

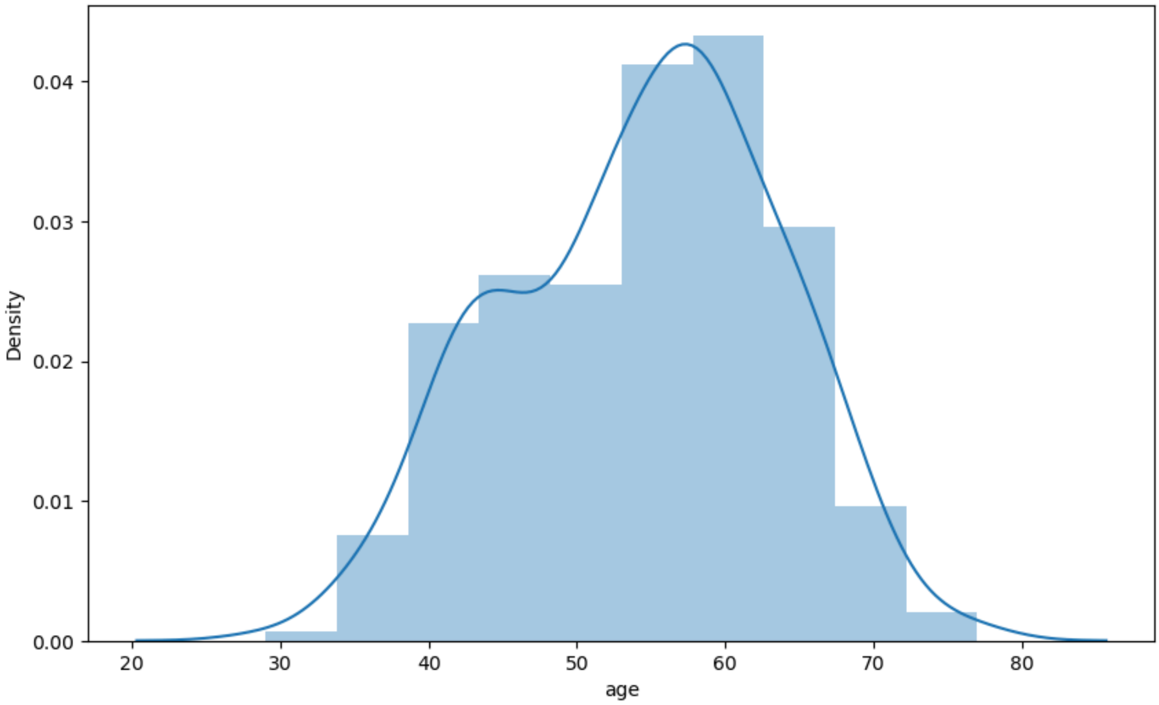
Out[138…

```
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
```

In [140…

```
f,ax= plt.subplots(figsize=(10,6))
x = df['age']
```

```
ax = sns.distplot(x,bins = 10)
plt.show()
```

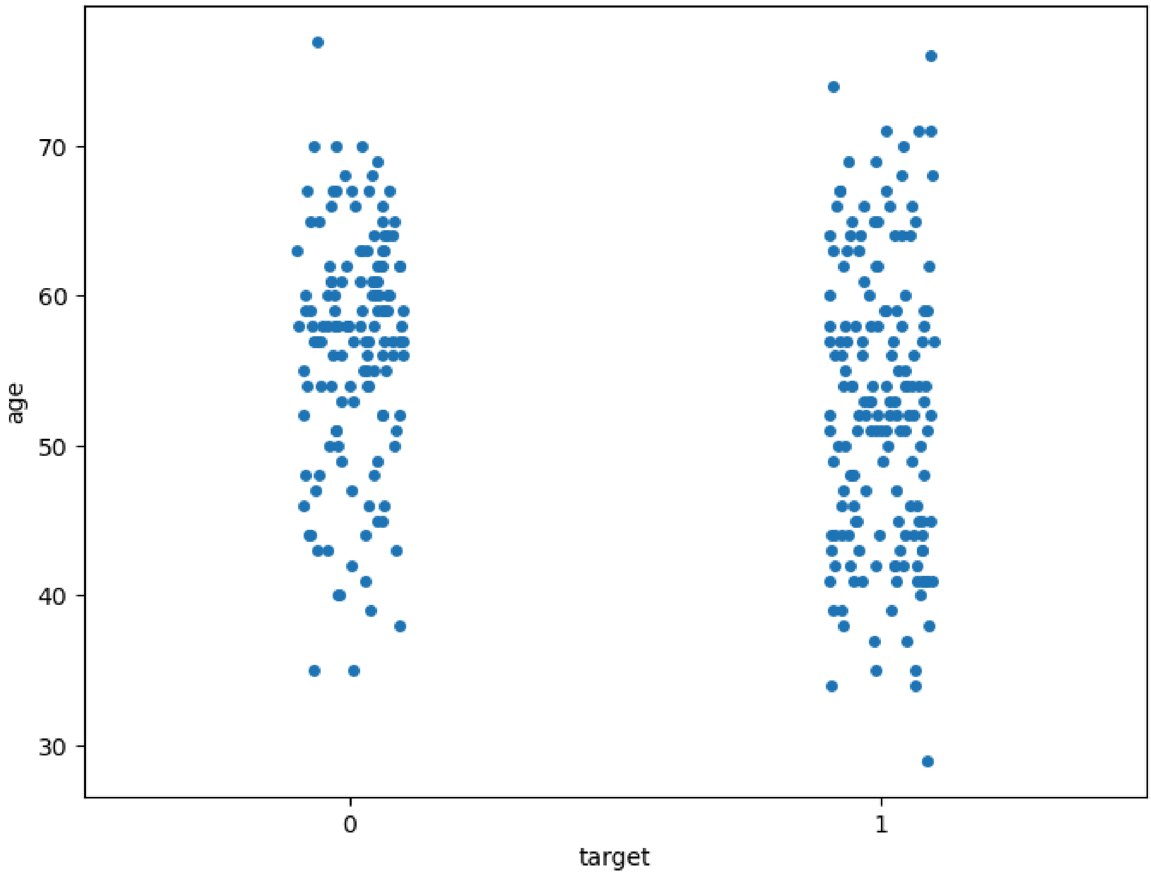


analyze the age and target variable

VISUALIZE FREQUENCY OF AGE VARIABLE WRT TARGET

In [144…

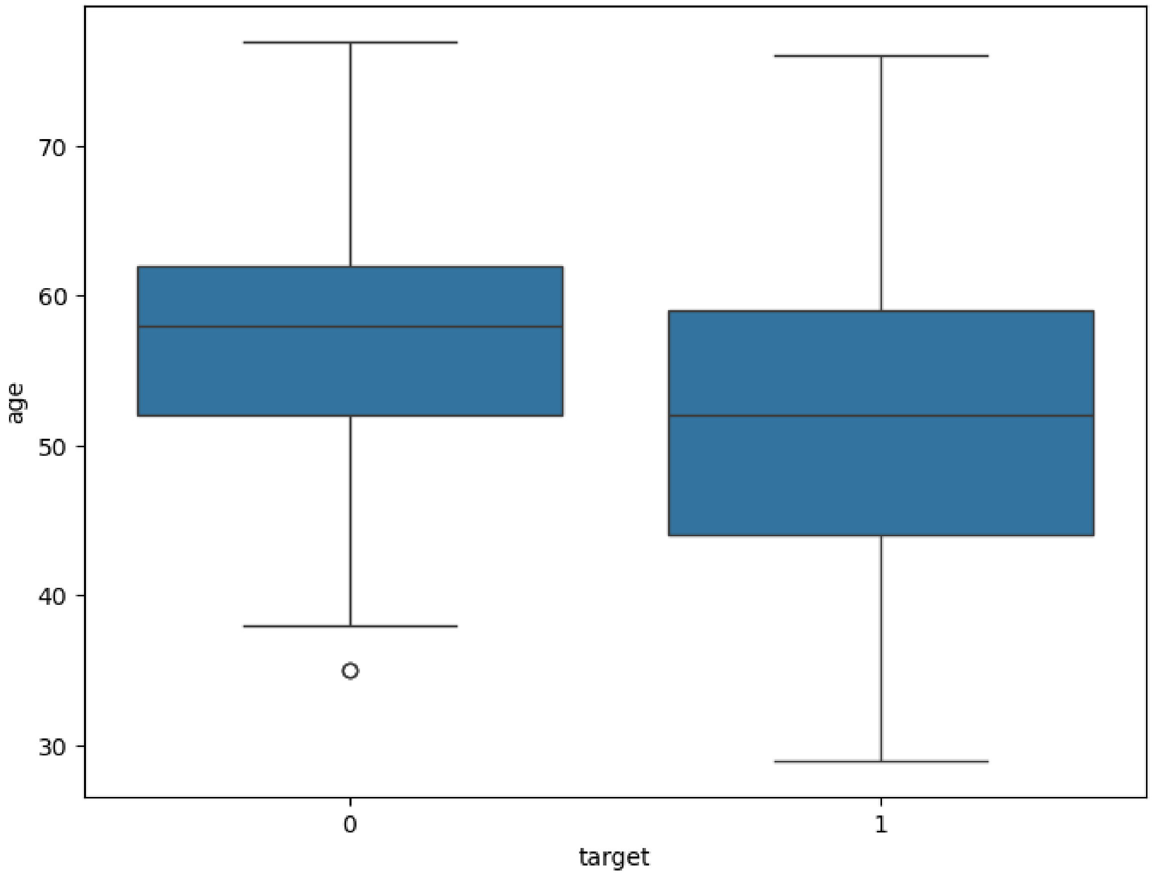
```
f, ax = plt.subplots(figsize=(8,6))
sns.stripplot(x="target",y = "age",data=df)
plt.show()
```



visualize the distribution of age variable wrt target with boxplot

In [146…

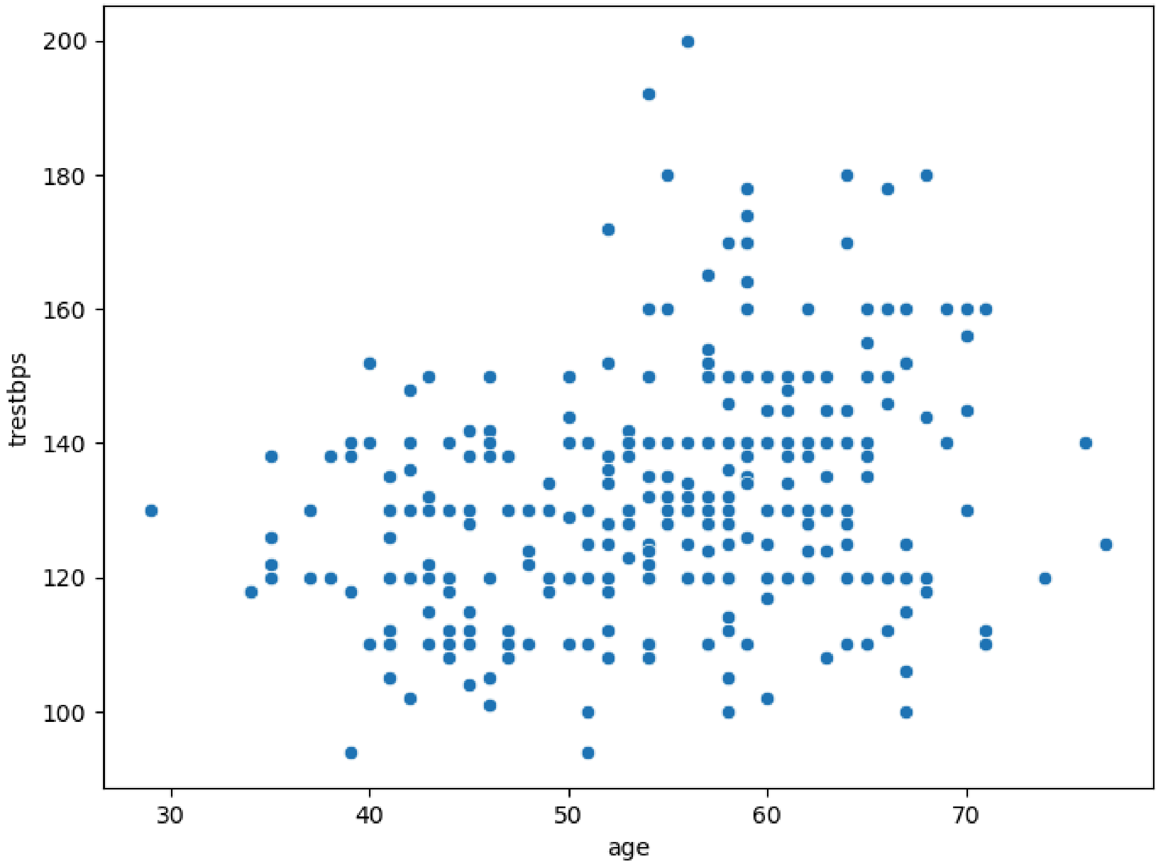
```
f, ax = plt.subplots(figsize=(8, 6))
sns.boxplot(x="target", y="age", data=df)
plt.show()
```



analyze age and trestbps variable

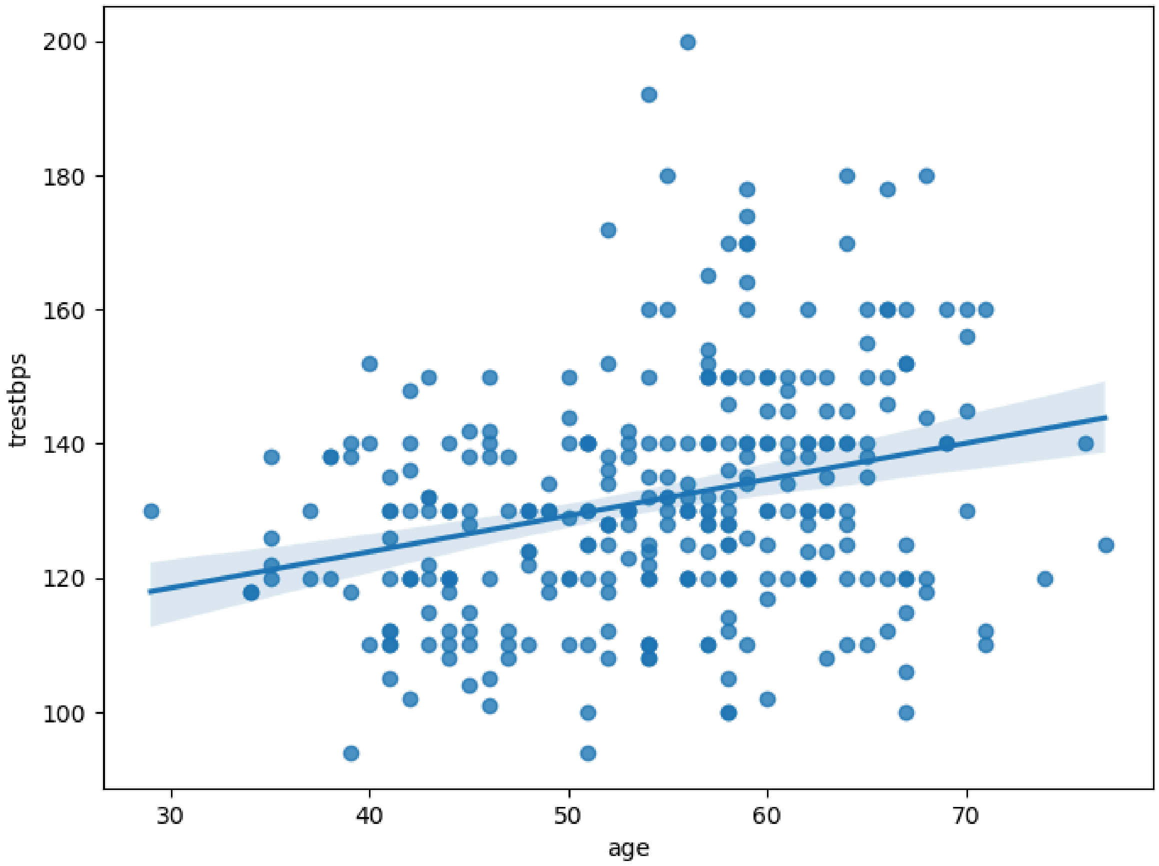
In [150…

```
f, ax = plt.subplots(figsize=(8, 6))
ax = sns.scatterplot(x="age", y="trestbps", data=df)
plt.show()
```



In [152…

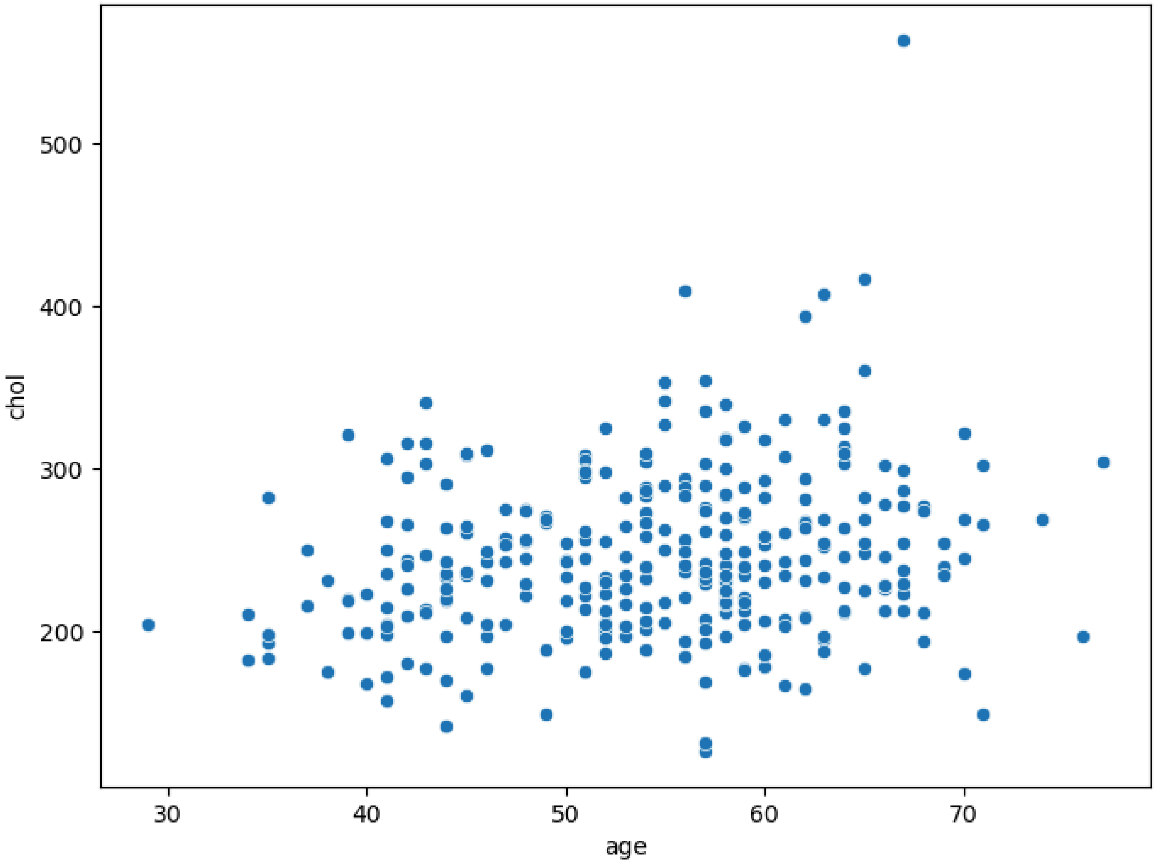
```
f,ax = plt.subplots(figsize=(8,6))
ax = sns.regplot(x= "age",y = "trestbps",data=df)
plt.show()
```



ANALYZE AGE AND CHOL VARIABLE

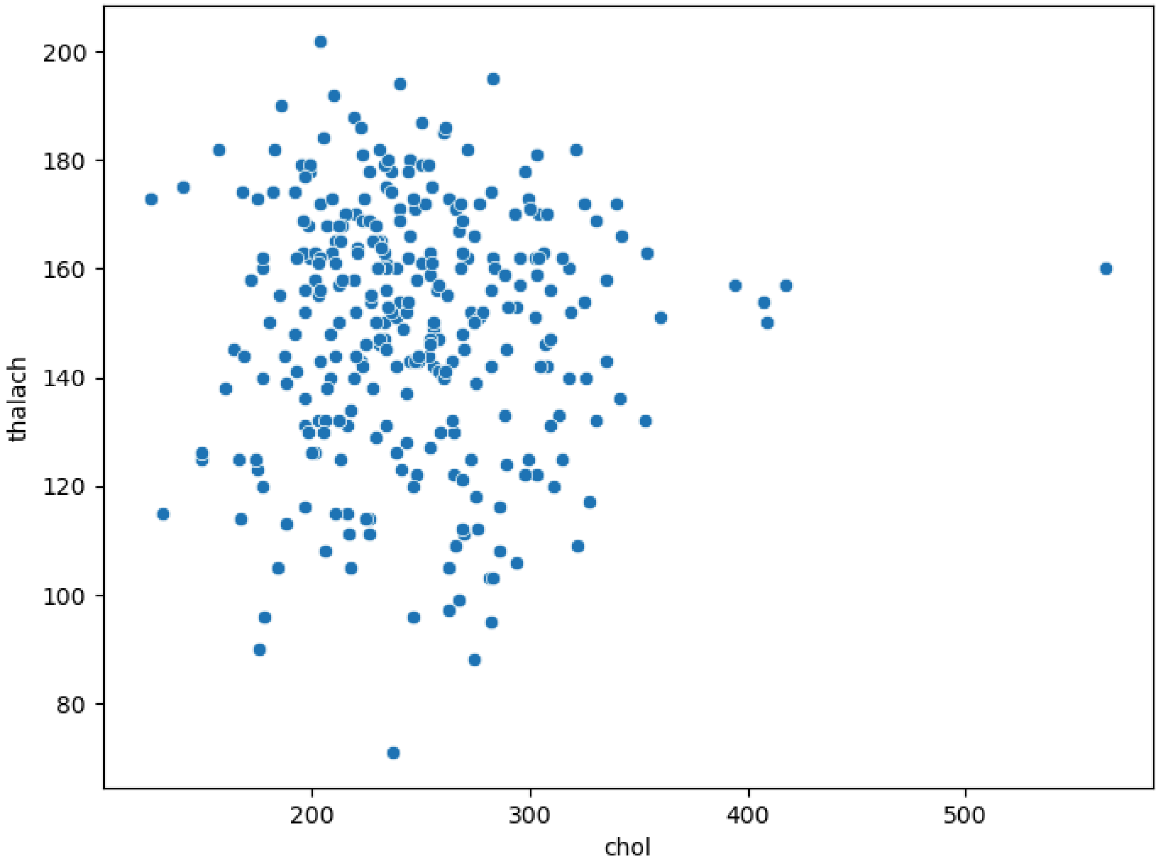
In [155…

```
f, ax = plt.subplots(figsize=(8, 6))
ax = sns.scatterplot(x="age", y="chol", data=df)
plt.show()
```



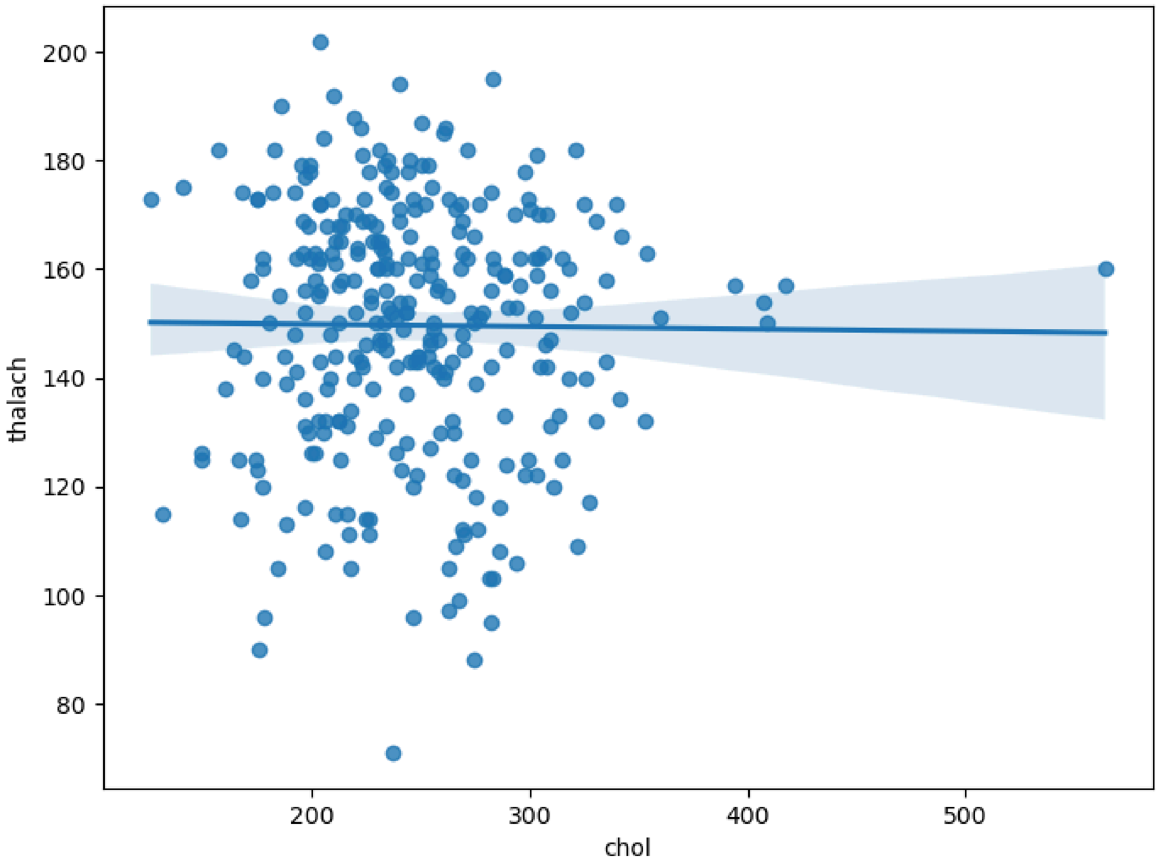
In [157…

```
f, ax = plt.subplots(figsize=(8, 6))
ax = sns.scatterplot(x="chol", y = "thalach", data=df)
plt.show()
```



In [159…

```
f, ax = plt.subplots(figsize=(8, 6))
ax = sns.regplot(x="chol", y="thalach", data=df)
plt.show()
```



DEALING WITH MISSING VALUES

In [162…

```
df.isnull().sum()
```

Out[162…

```
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
```

CHECK WITH ASSERT STATEMENT

In [165…

```
# ASSERT THAT THERE ARE NO MISSING VALUES IN THE DATAFRAMES
```

In [169…

```
# assert that there are no missing values in the dataframes

assert pd.notnull(df).all().all()
```

In [173…

```
# assert that all values are greater than or equal to 0
assert (df>=0).all().all()
```

OUTLIER DETECTION

In [176…

```
df['age'].describe()
```

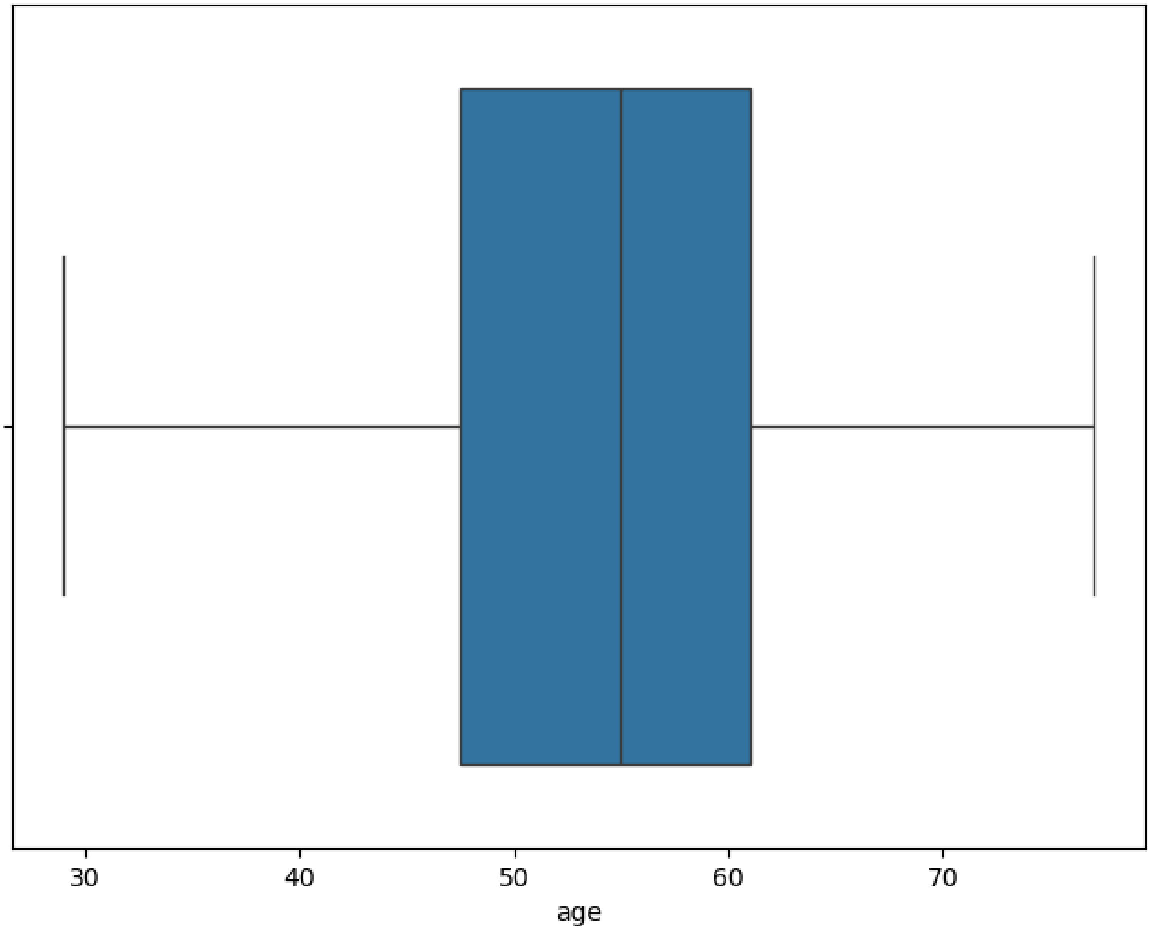
Out[176…

```
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
```

BOX PLOT OF AGE VARIABLE

In [179…

```
f,ax = plt.subplots(figsize=(8,6))
sns.boxplot(x=df["age"])
plt.show()
```



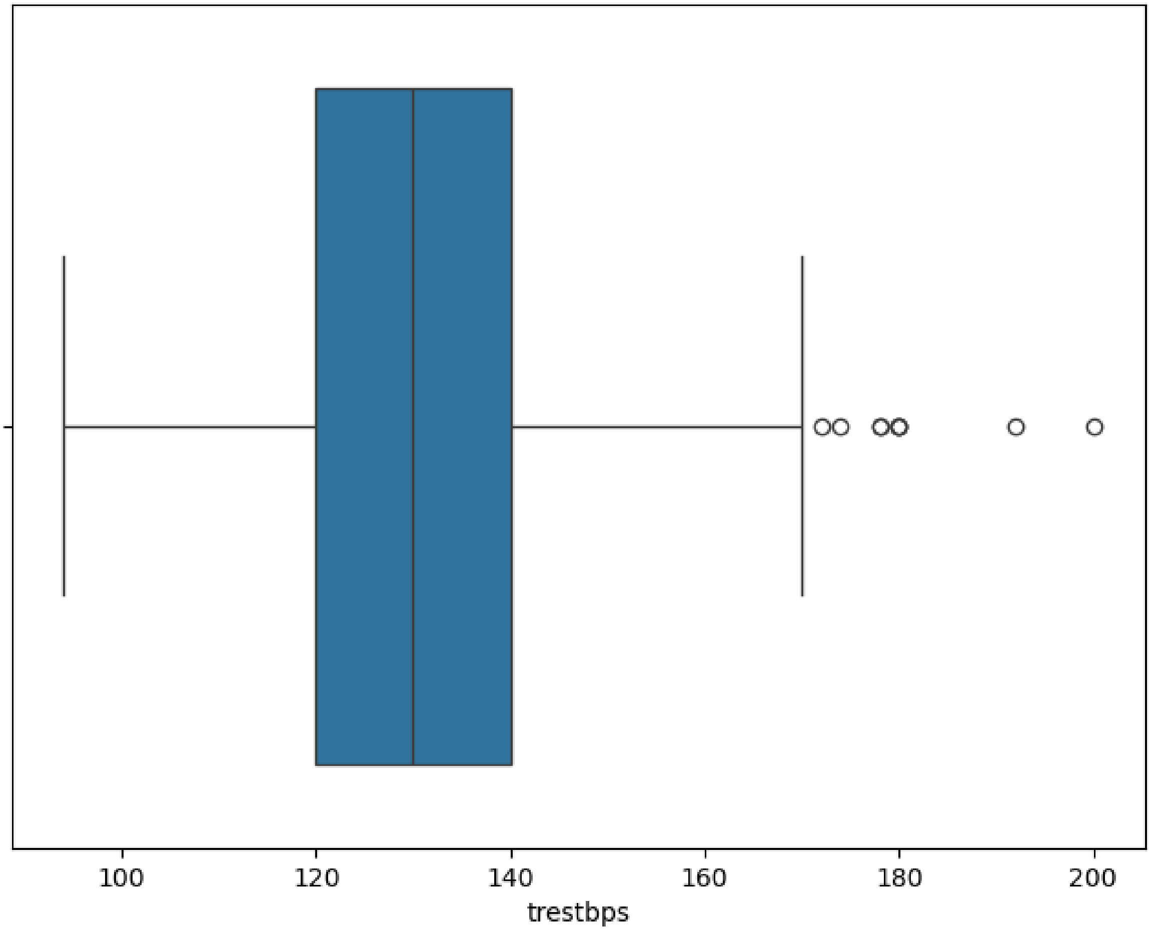
TRESTBPS VARIABLE

In [183…
```
df['trestbps'].describe()
```

Out[183…
```
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
```

BOXPLOTS OF TRESTSBPS VARIABLE

In [186…
```
f, ax = plt.subplots(figsize=(8, 6))
sns.boxplot(x=df["trestbps"])
plt.show()
```



CHOL VARIABLE

In [189…

```
df['chol'].describe()
```

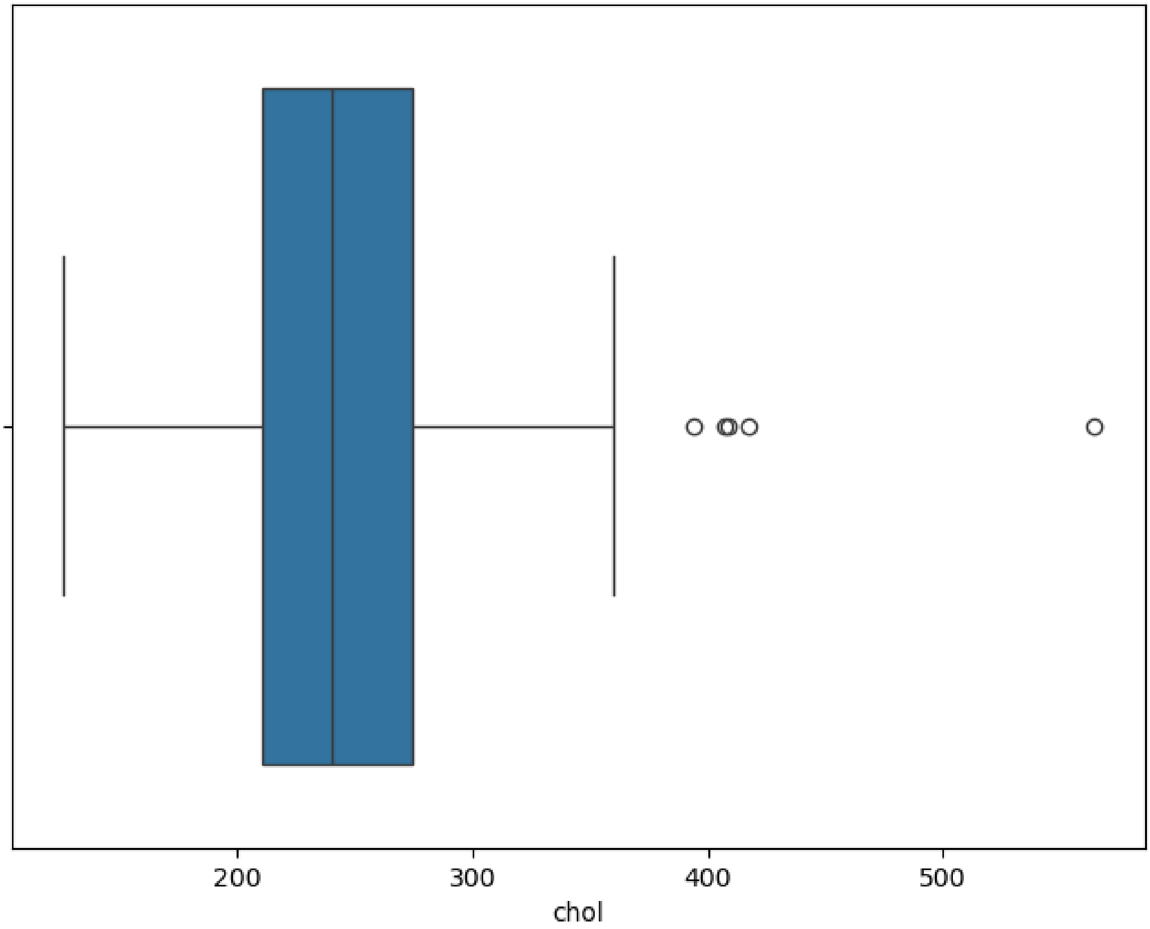
Out[189…

```
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
```

BOXPLOT OF CHOL VARIABLE

In [192…

```
f, ax = plt.subplots(figsize=(8, 6))
sns.boxplot(x=df["chol"])
plt.show()
```



THALACH VARIABLE

In [195…

```
df['thalach'].describe()
```

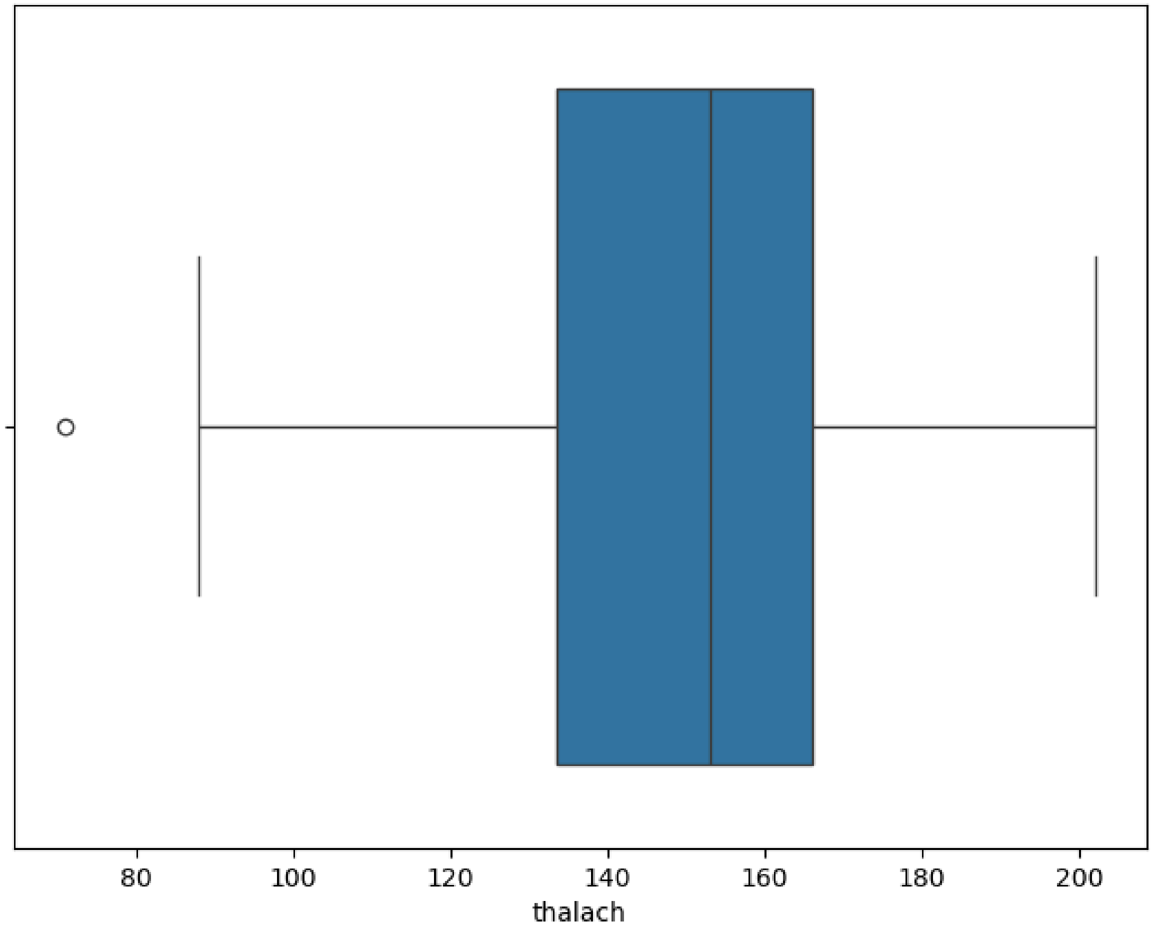
Out[195…

```
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
```

BOXPLOT OF THALACH VARIABLE

In [198…

```
f, ax = plt.subplots(figsize=(8, 6))
sns.boxplot(x=df["thalach"])
plt.show()
```



OLDPEAK VARIABLE

In [201…

```
df['oldpeak'].describe()
```

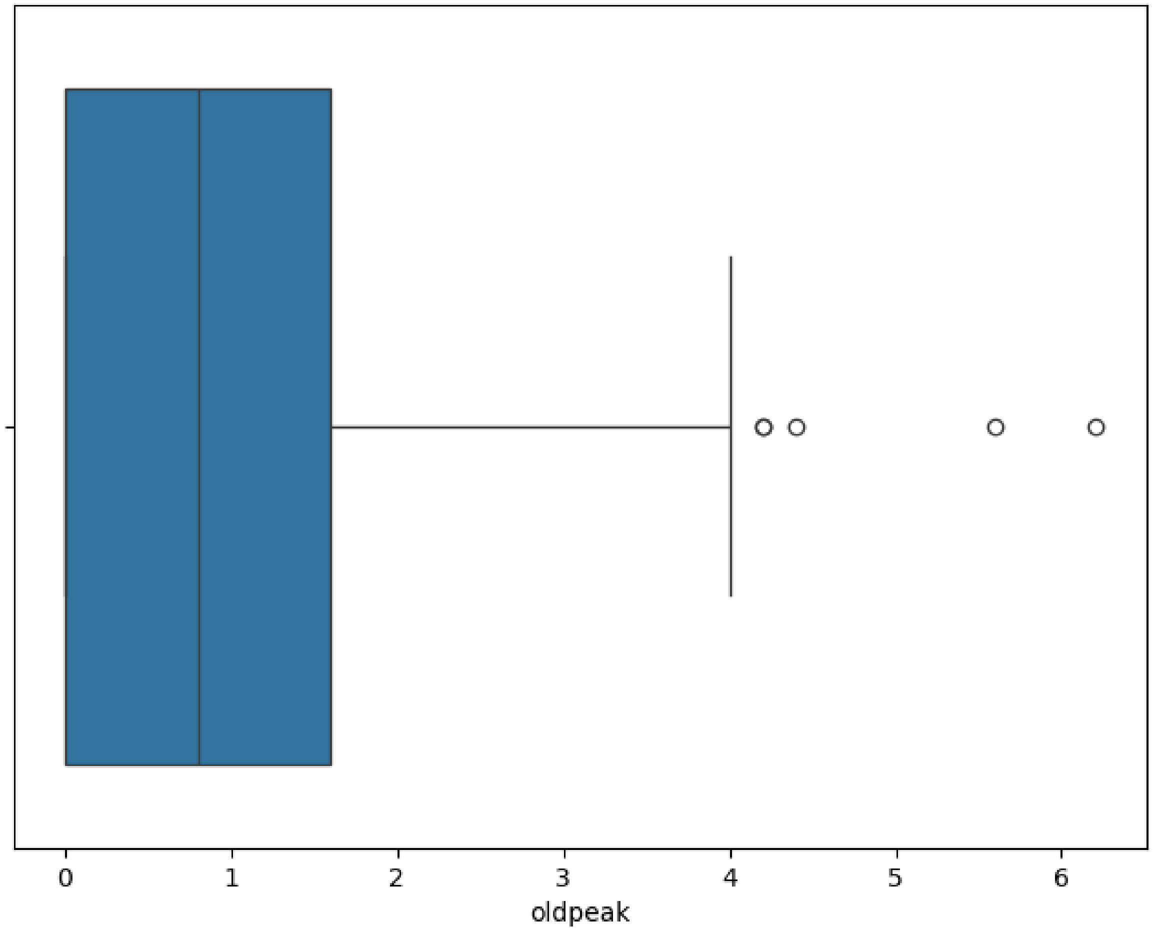
Out[201…

```
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
```

BOX-PLOT OF OLDPEAK VARIABLE

In [206…

```
f, ax = plt.subplots(figsize=(8, 6))
sns.boxplot(x=df["oldpeak"])
plt.show()
```



In [ ]:

```
# conclusion :
```