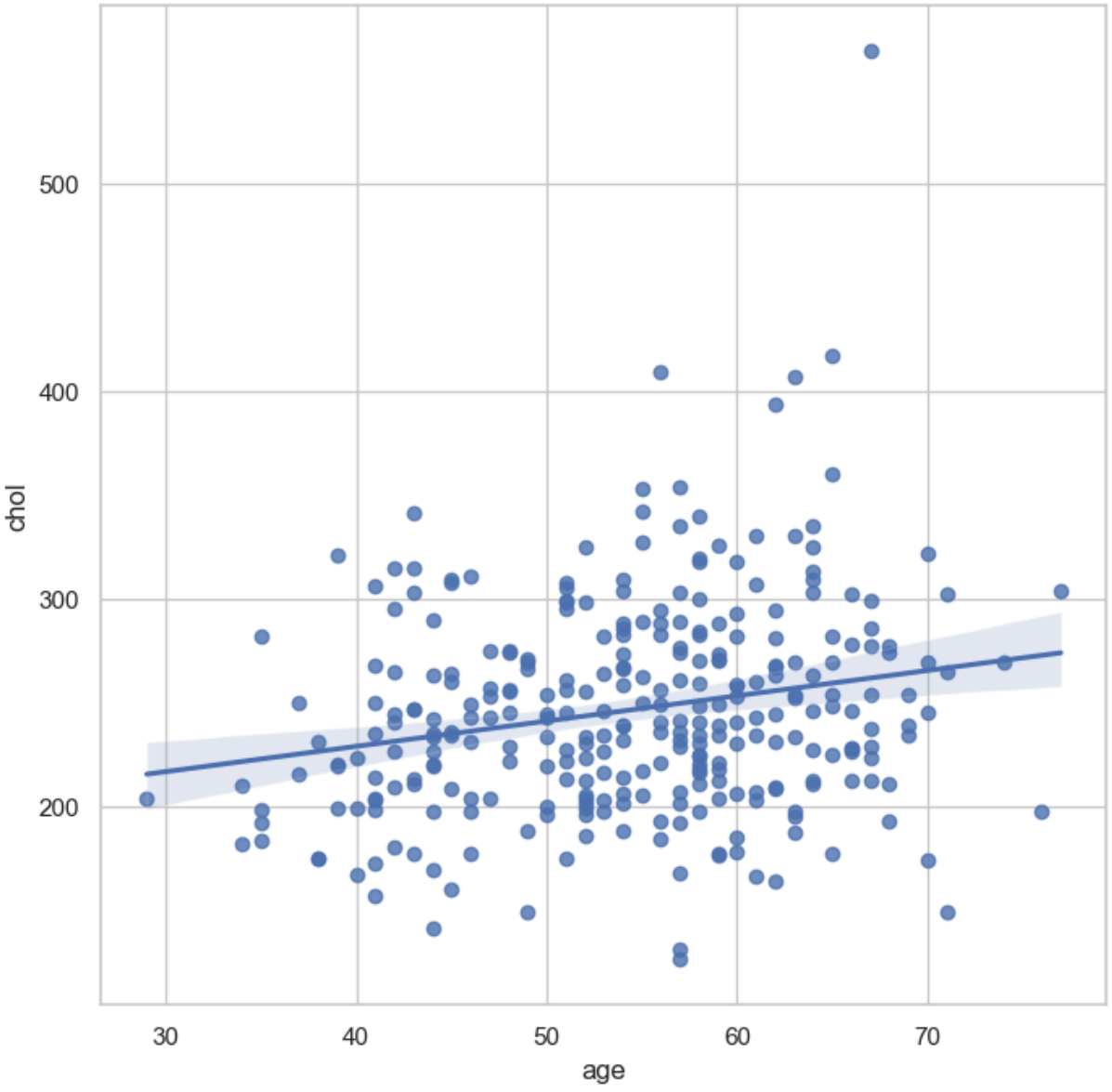
```
plt.show()
```



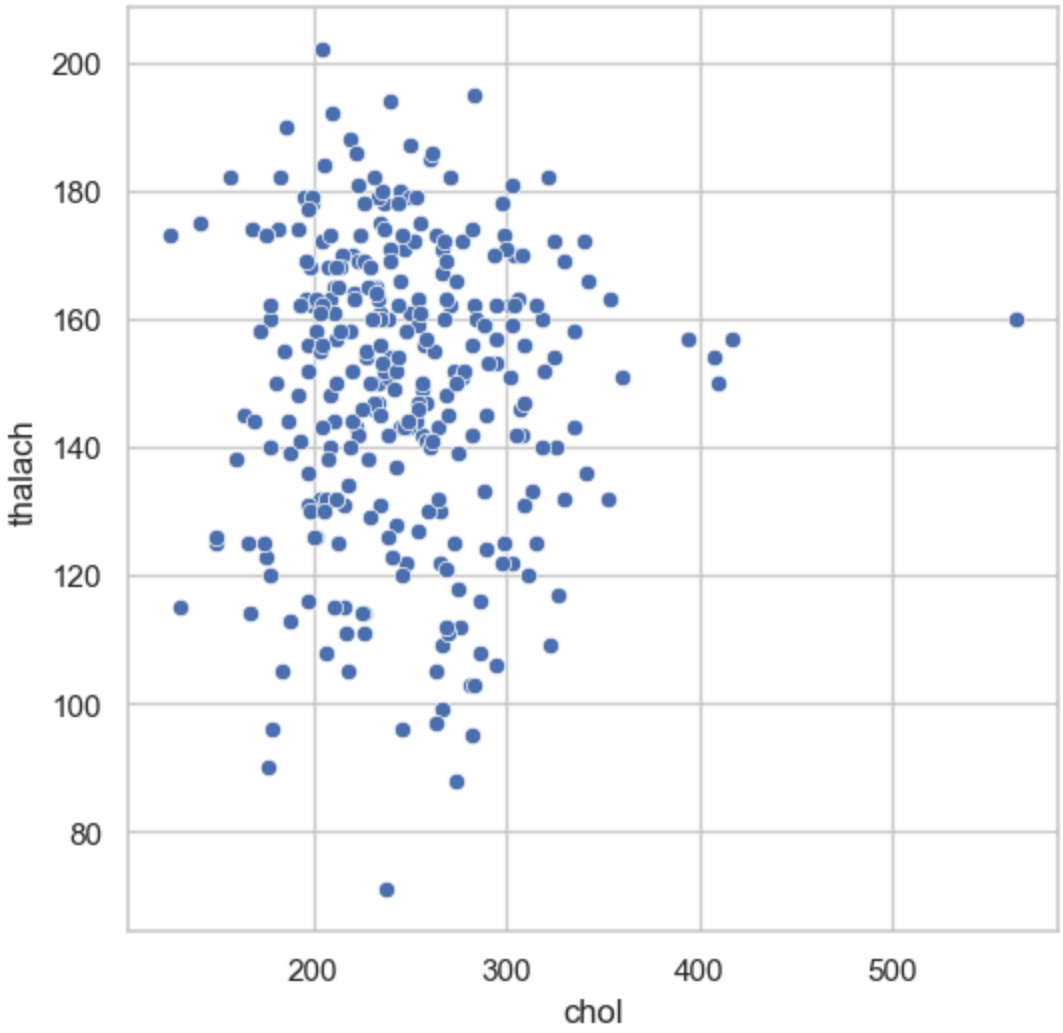
Interpretation

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

Analyze `chol` and `thalach` variable

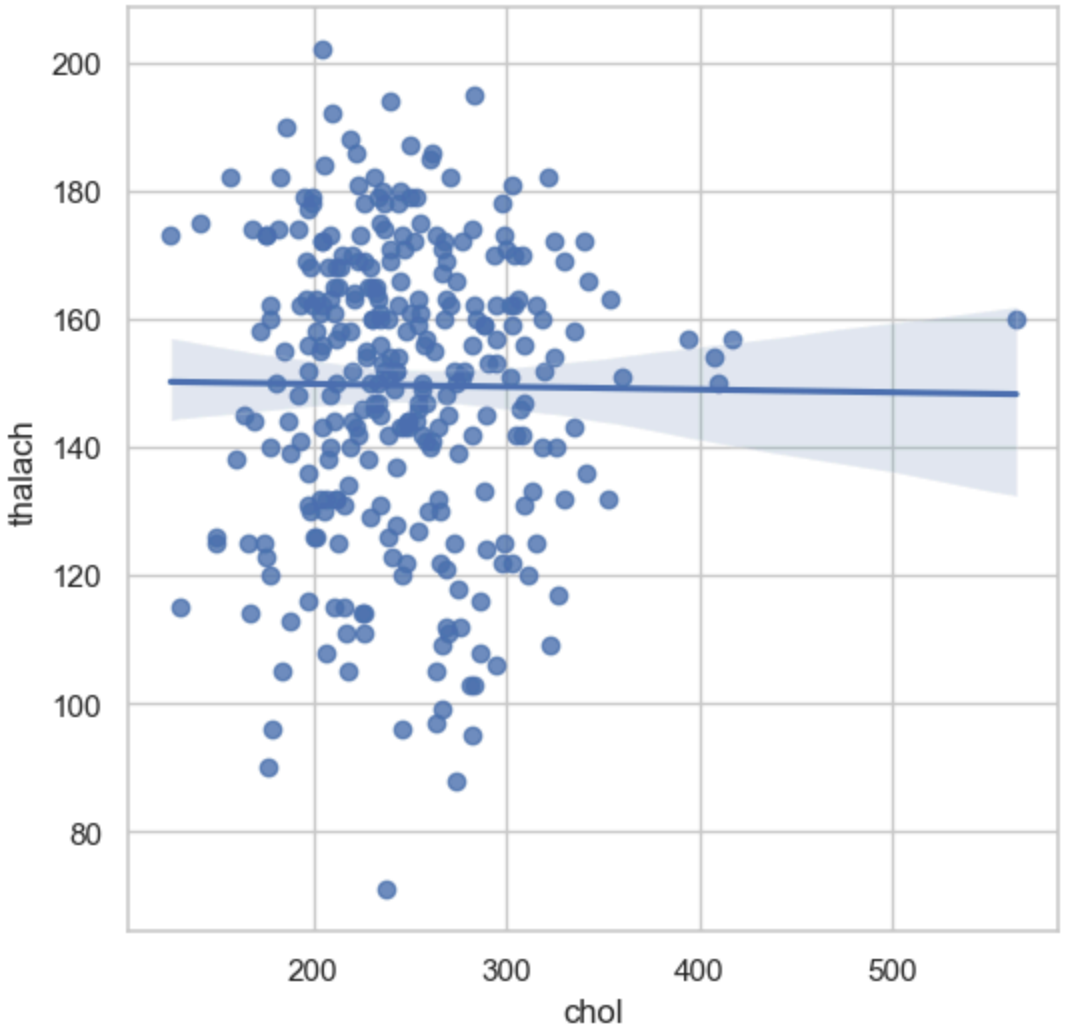
In [157…

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



In [230…

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



Interpretation

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

#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.

In [243…
```
#check missing values
df.isnull().sum()
```

Out[243…
```
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
```

Interpretation

We can see that there are no missing values in the dataset.

#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 [250…
```
#assert that there are no missing values in the dataframe
assert pd.notnull(df).all().all()
```

In [252…
```
#assert all values are greater than or equal to 0
assert (df >= 0).all().all()
```

Interpretation

- The above two commands do not throw 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.

#Outlier detection

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

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

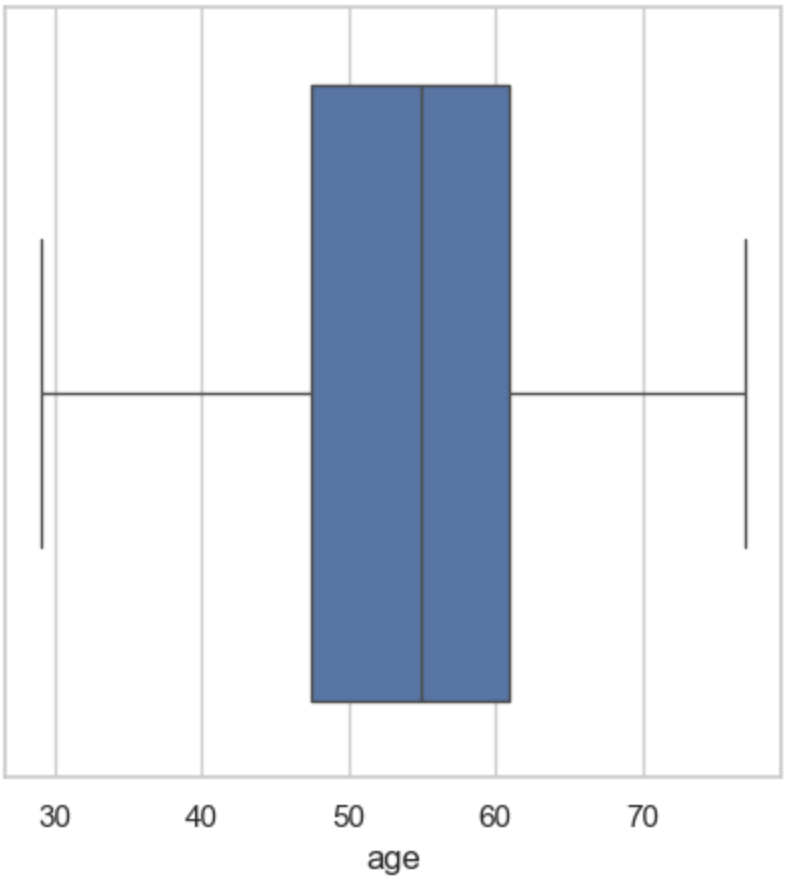
age veriable

In [260…
```
df['age'].describe()
```

Out[260…
```
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 [269…
```
f,ax = plt.subplots(figsize=(5,5))
sns.boxplot(x=df['age'])
plt.show()
```



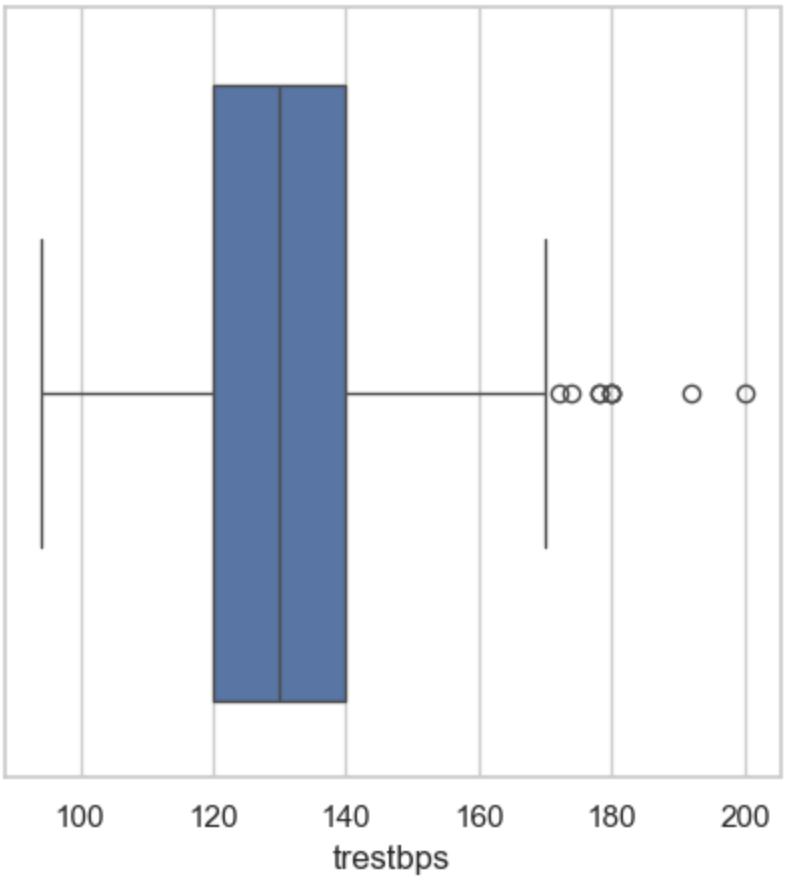
`trestbps` variable

In [272…
```
df['trestbps'].describe()
```

Out[272…
```
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
```

Box-plot of `trestbps` variable

In [279…
```
f, ax = plt.subplots(figsize=(5,5))
sns.boxplot(x=df['trestbps'])
plt.show()
```



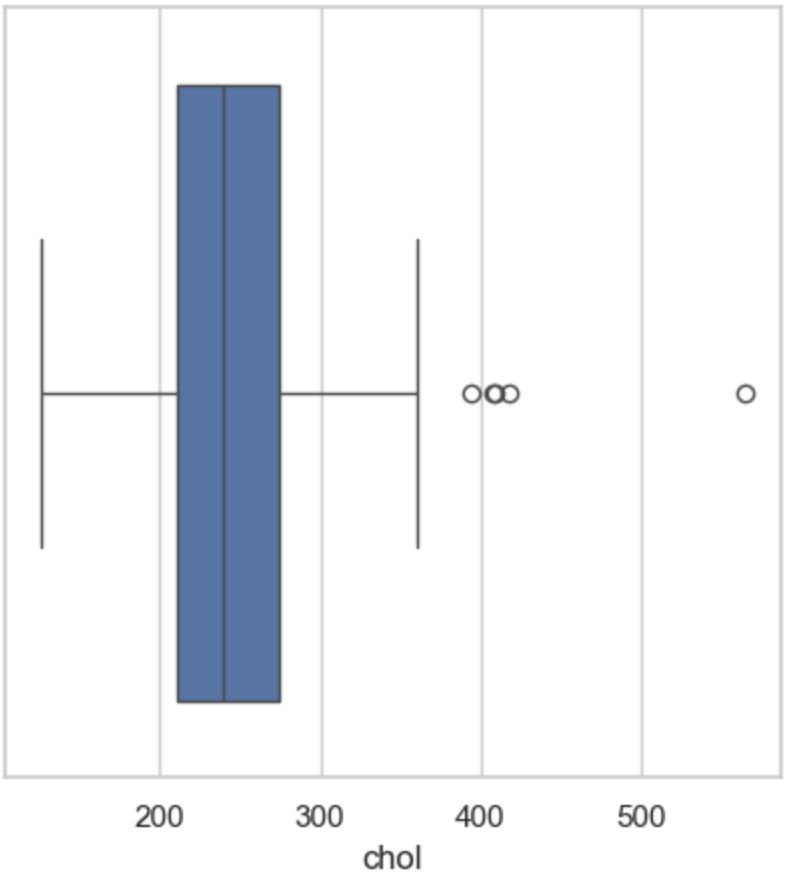
`chol` veriable

In [282…
```
df['chol'].describe()
```

Out[282…
```
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
```

Box-plot of `chol` variable

In [296…
```
f,ax = plt.subplots(figsize=(5,5))
sns.boxplot(x=df['chol'])
plt.show()
```

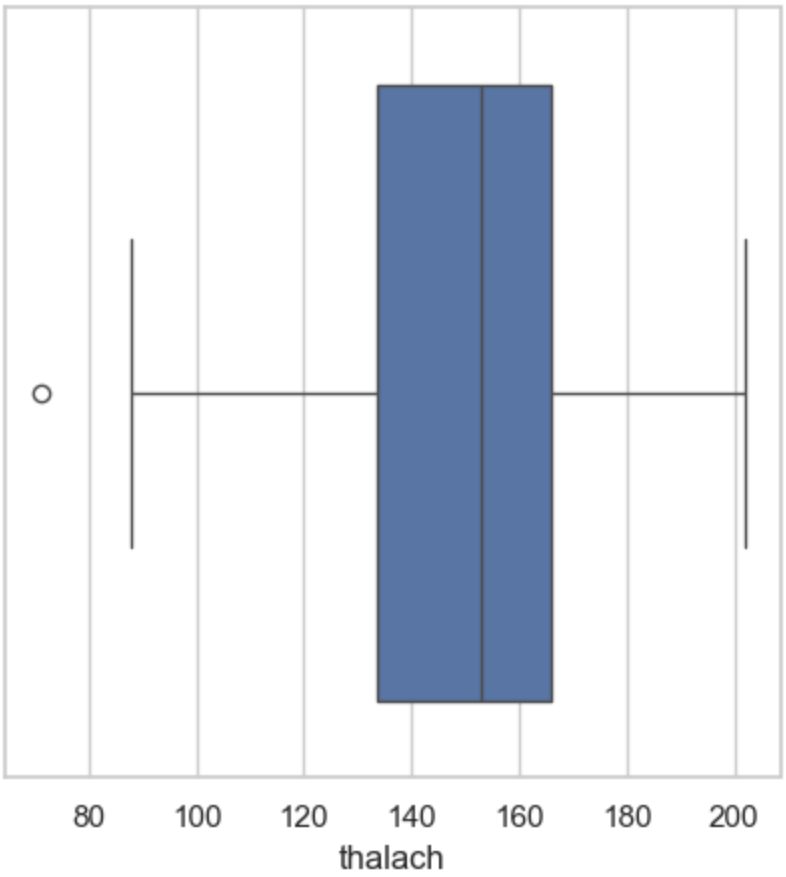


Thalach veriable

In [299…
```
df['thalach'].describe()
```

Out[299…
```
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 [301…
```
f,ax = plt.subplots(figsize=(5,5))
sns.boxplot(x=df['thalach'])
plt.show()
```

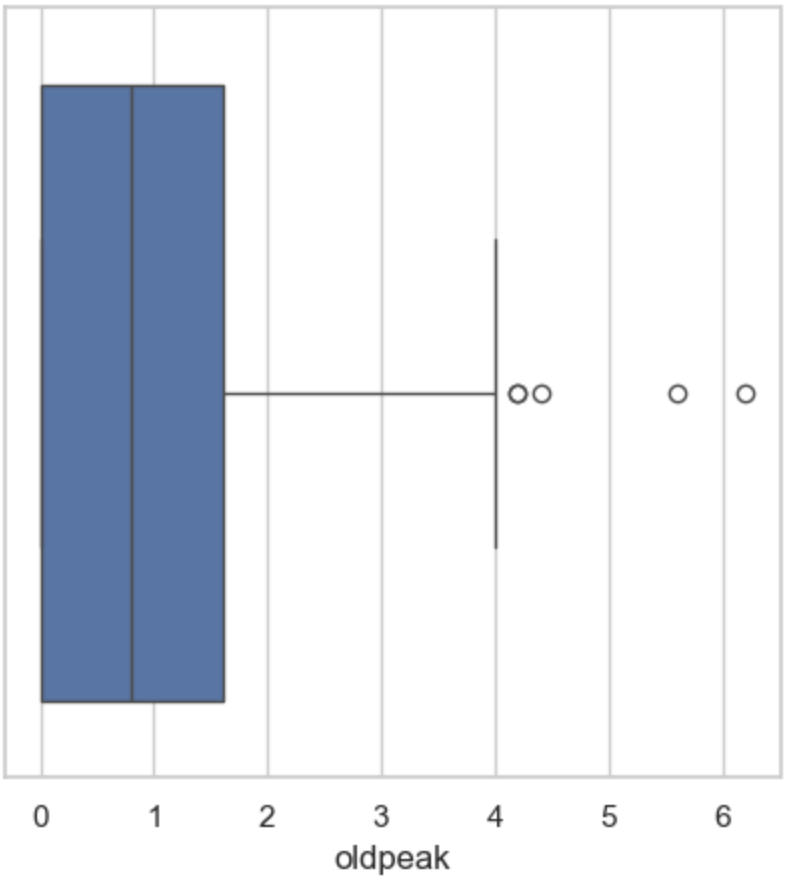


`oldpeak` variable

In [304…
```
df['oldpeak'].describe()
```

Out[304…
```
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 [308…
```
f, ax = plt.subplots(figsize=(5,5))
sns.boxplot(x=df['oldpeak'])
plt.show()
```



Findings

- 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.

In [ ]: