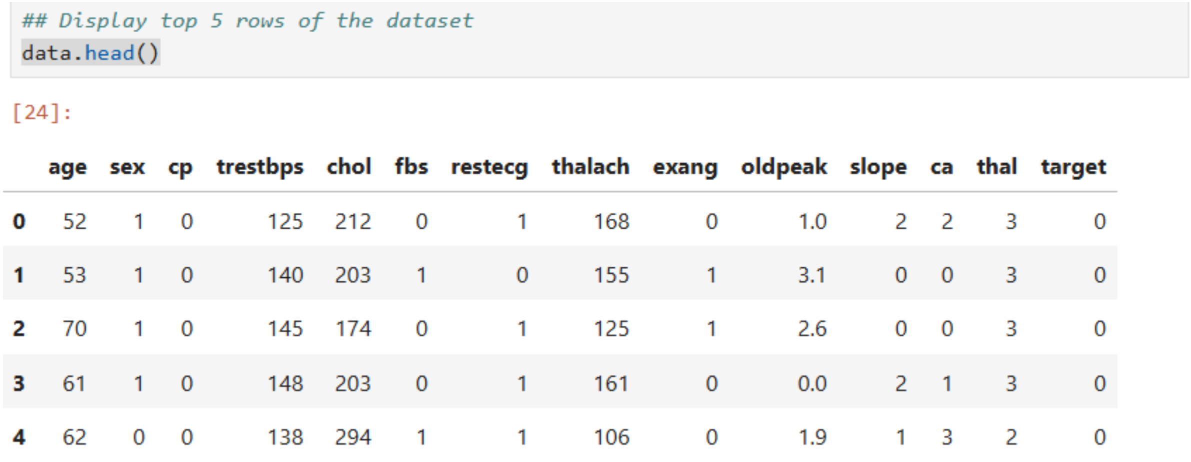
SATISH M – AF0378136

TITLE – HEART DISEASE

1. Import the libraries and dataset.

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
data = pd.read_csv('heart.csv')

data.head()



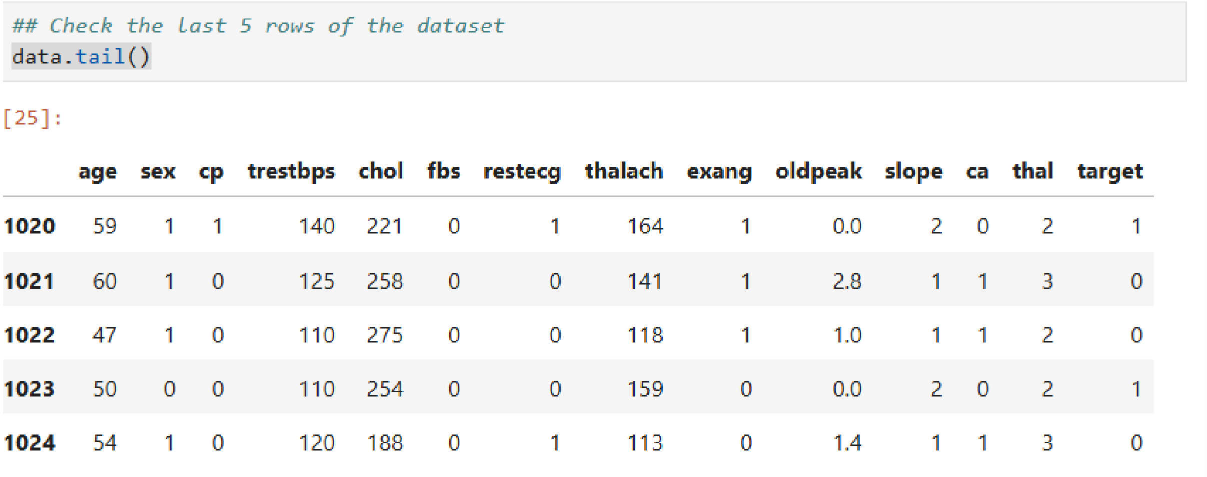
```
## Display top 5 rows of the dataset
data.head()
```

[24]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0

2. Check the last 5 rows of the dataset.

data.tail()

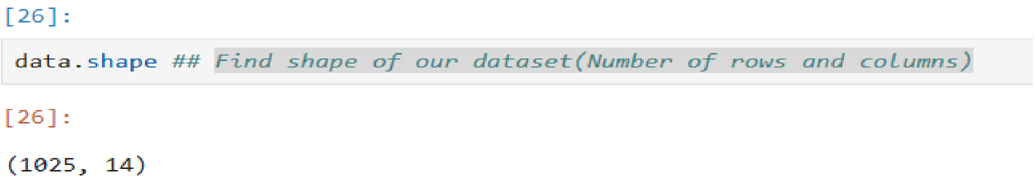


```
## Check the last 5 rows of the dataset
data.tail()
```

[25]:

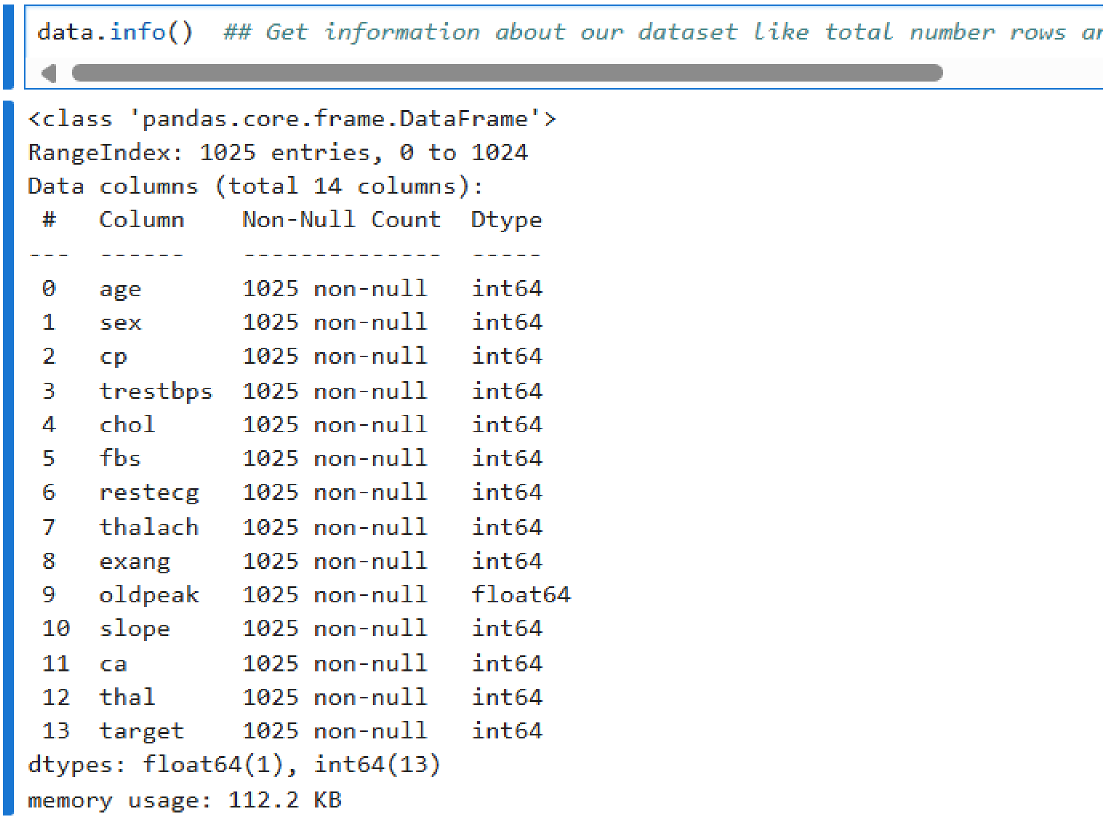
	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
1020	59	1	1	140	221	0	1	164	1	0.0	2	0	2	1
1021	60	1	0	125	258	0	0	141	1	2.8	1	1	3	0
1022	47	1	0	110	275	0	0	118	1	1.0	1	1	2	0
1023	50	0	0	110	254	0	0	159	0	0.0	2	0	2	1
1024	54	1	0	120	188	0	1	113	0	1.4	1	1	3	0

3. Find shape of our dataset(Number of rows and columns)



```
[26]:
data.shape ## Find shape of our dataset(Number of rows and columns)
[26]:
(1025, 14)
```

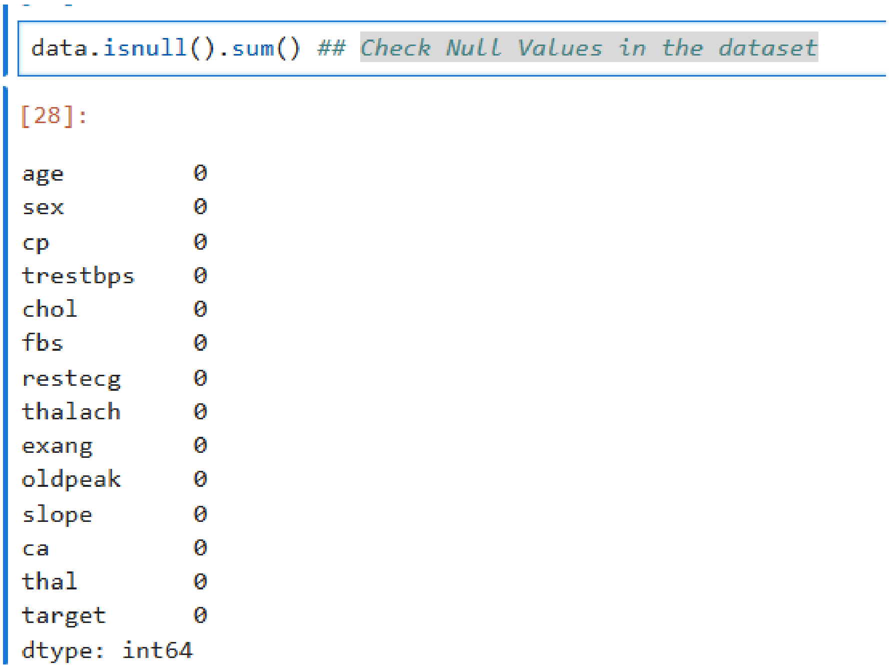
4. **Get information about our dataset like total number rows and columns, data type of each column and memory requirement.**



```
data.info()  ## Get information about our dataset like total number rows ar
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1025 entries, 0 to 1024
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       1025 non-null   int64  
 1   sex       1025 non-null   int64  
 2   cp        1025 non-null   int64  
 3   trestbps  1025 non-null   int64  
 4   chol      1025 non-null   int64  
 5   fbs       1025 non-null   int64  
 6   restecg   1025 non-null   int64  
 7   thalach   1025 non-null   int64  
 8   exang     1025 non-null   int64  
 9   oldpeak   1025 non-null   float64
 10  slope     1025 non-null   int64  
 11  ca        1025 non-null   int64  
 12  thal      1025 non-null   int64  
 13  target    1025 non-null   int64  
dtypes: float64(1), int64(13)
memory usage: 112.2 KB
```

5. **Check Null Values in the dataset.**



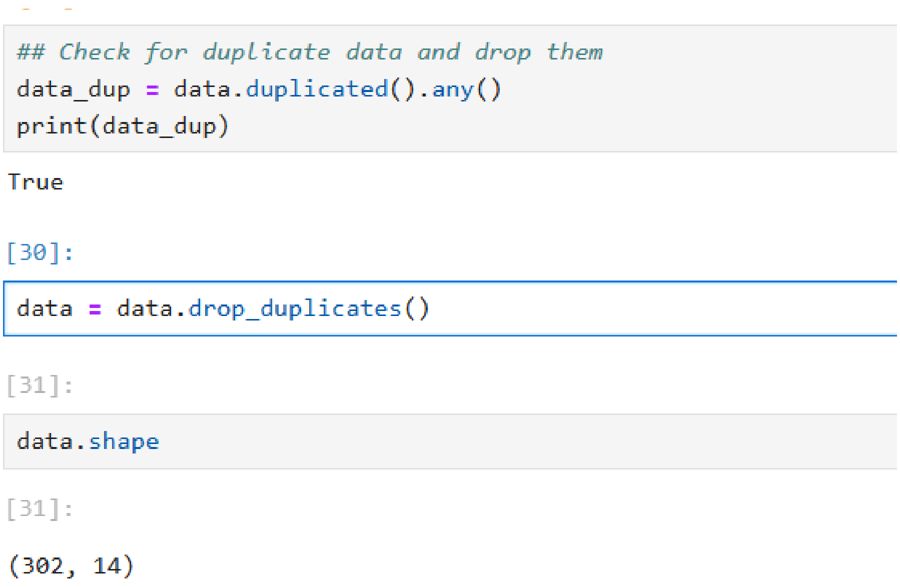
```
data.isnull().sum() ## Check Null Values in the dataset
```

```
[28]:

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
```

6. **Check for duplicate data and drop them.**

data_dup = data.duplicated().any()
print(data_dup)



```
## Check for duplicate data and drop them
data_dup = data.duplicated().any()
print(data_dup)
```

```
True
```

```
[30]:
```

```
data = data.drop_duplicates()
```

```
[31]:
```

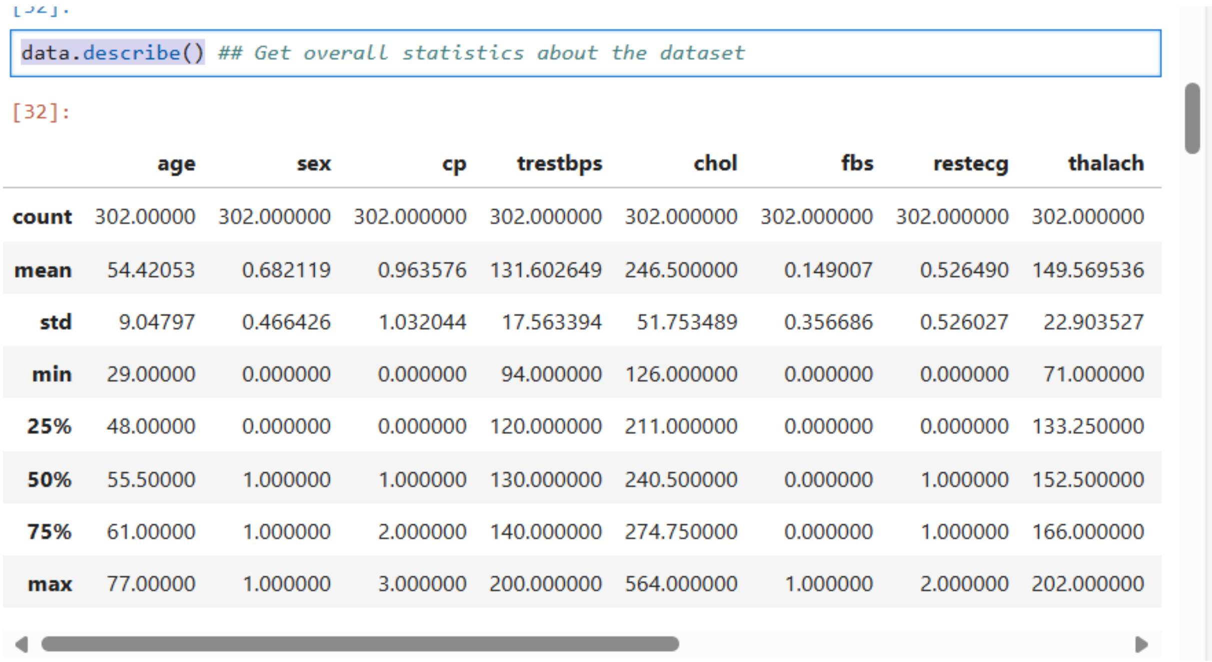
```
data.shape
```

```
[31]:
```

```
(302, 14)
```

7. **Get overall statistics about the dataset.**

data.describe()

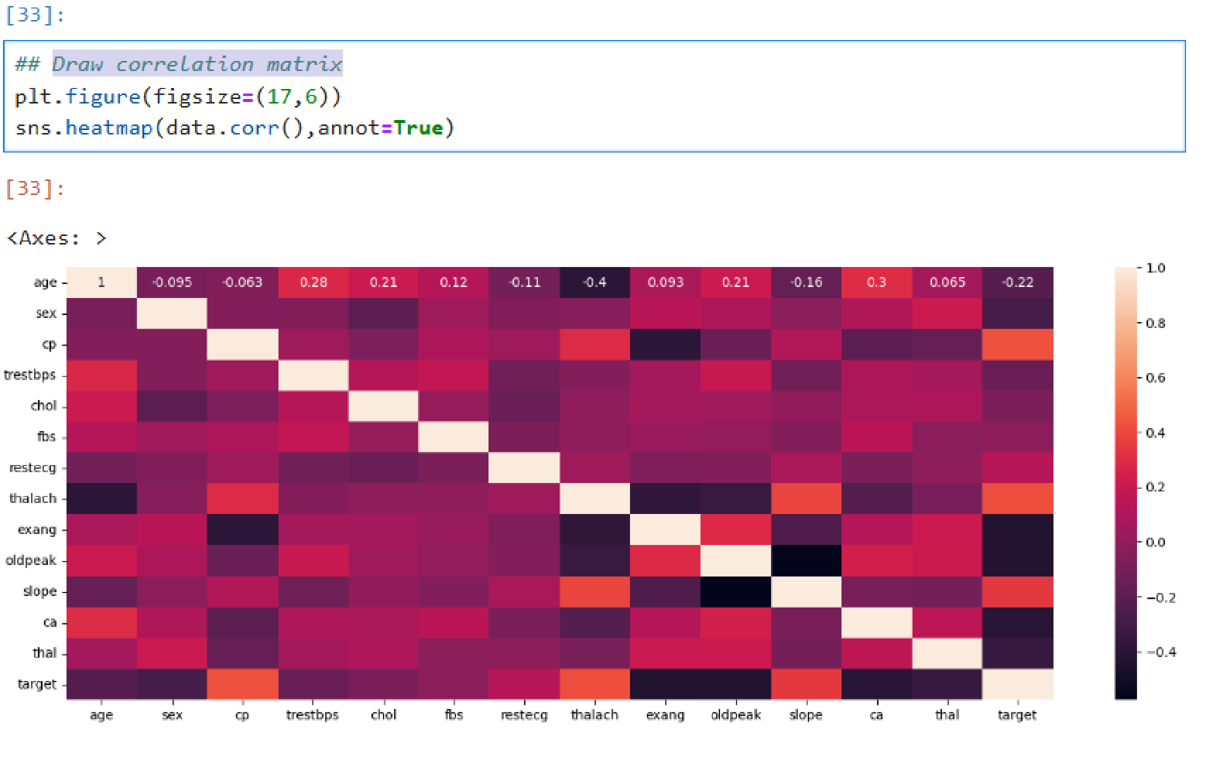


```
data.describe() ## Get overall statistics about the dataset
```

```
[32]:
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach
count	302.00000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000
mean	54.42053	0.682119	0.963576	131.602649	246.500000	0.149007	0.526490	149.569536
std	9.04797	0.466426	1.032044	17.563394	51.753489	0.356686	0.526027	22.903527
min	29.00000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.000000
25%	48.00000	0.000000	0.000000	120.000000	211.000000	0.000000	0.000000	133.250000
50%	55.50000	1.000000	1.000000	130.000000	240.500000	0.000000	1.000000	152.500000
75%	61.00000	1.000000	2.000000	140.000000	274.750000	0.000000	1.000000	166.000000
max	77.00000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000	202.000000

8. Draw correlation matrix.



```
[33]:
## Draw correlation matrix
plt.figure(figsize=(17,6))
sns.heatmap(data.corr(),annot=True)
```

```
[33]:

<Axes: >
```

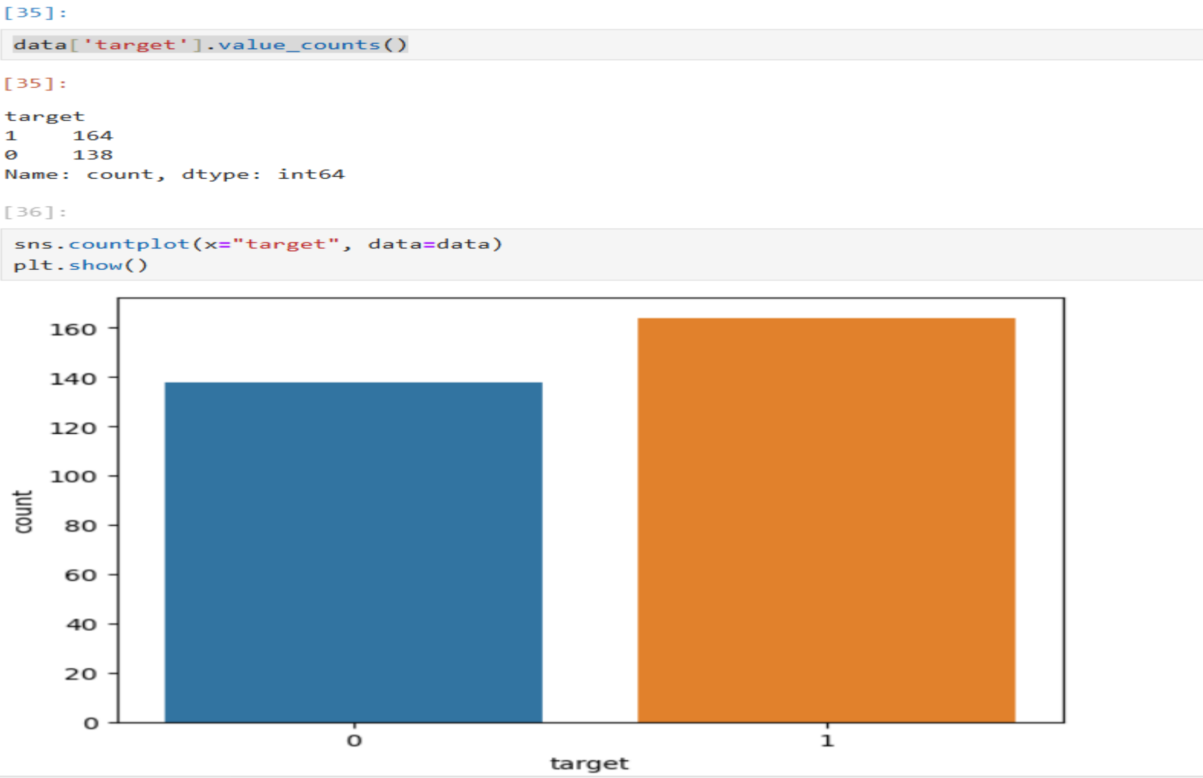
9. How many people have heart disease, and how many don't have heart disease in this dataset?

data.columns

data['target'].value_counts()

sns.countplot(x="target", data=data)

plt.show()



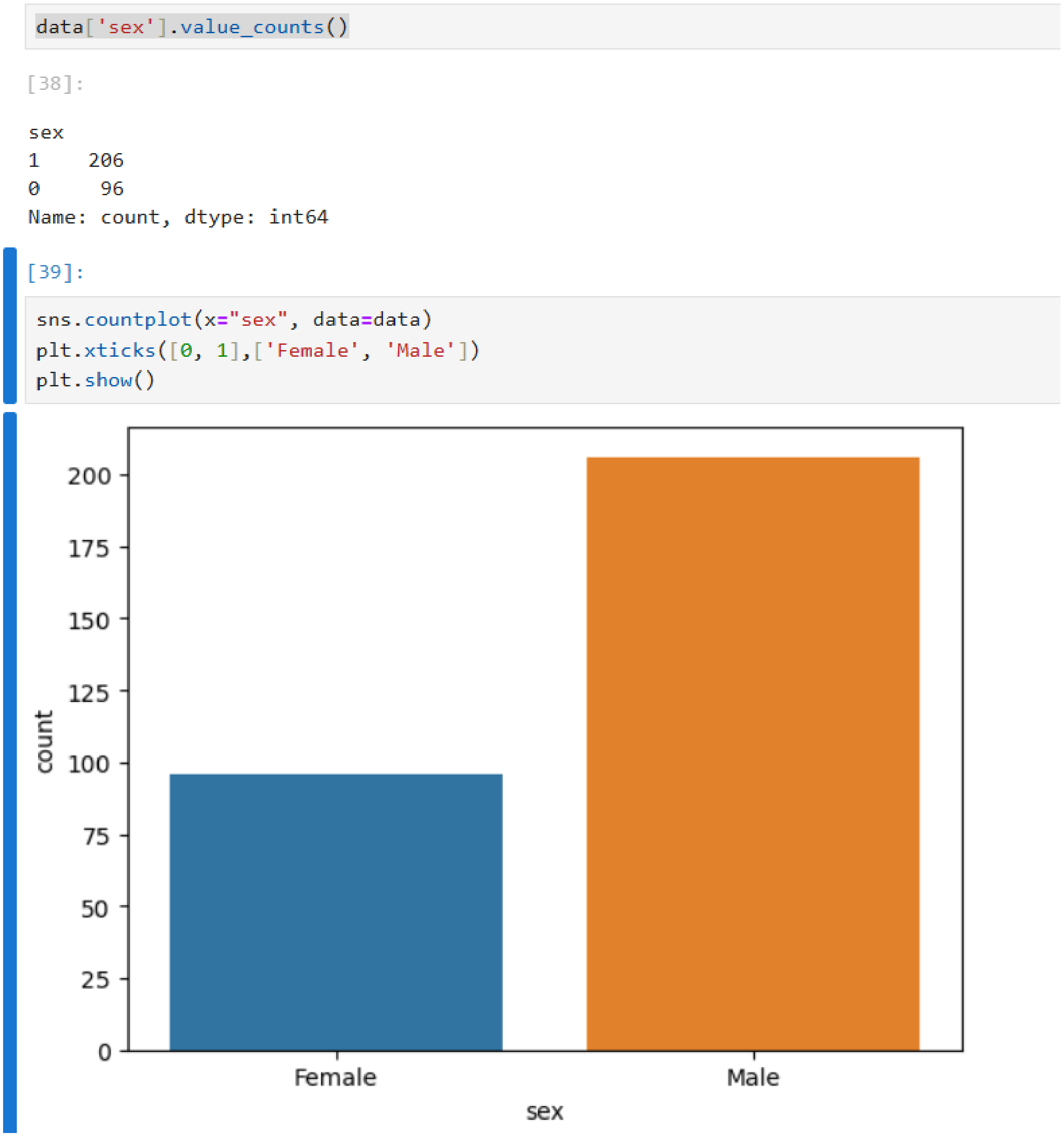
```
[35]:
data['target'].value_counts()
```

```
[35]:
target
1     164
0     138
Name: count, dtype: int64
```

```
[36]:
sns.countplot(x="target", data=data)
plt.show()
```

10. Find count of male and female in this dataset.

```
data.columns
data['sex'].value_counts()
sns.countplot(x="sex", data=data)
plt.xticks([0, 1],['Female', 'Male'])
plt.show()
```



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

[38]:

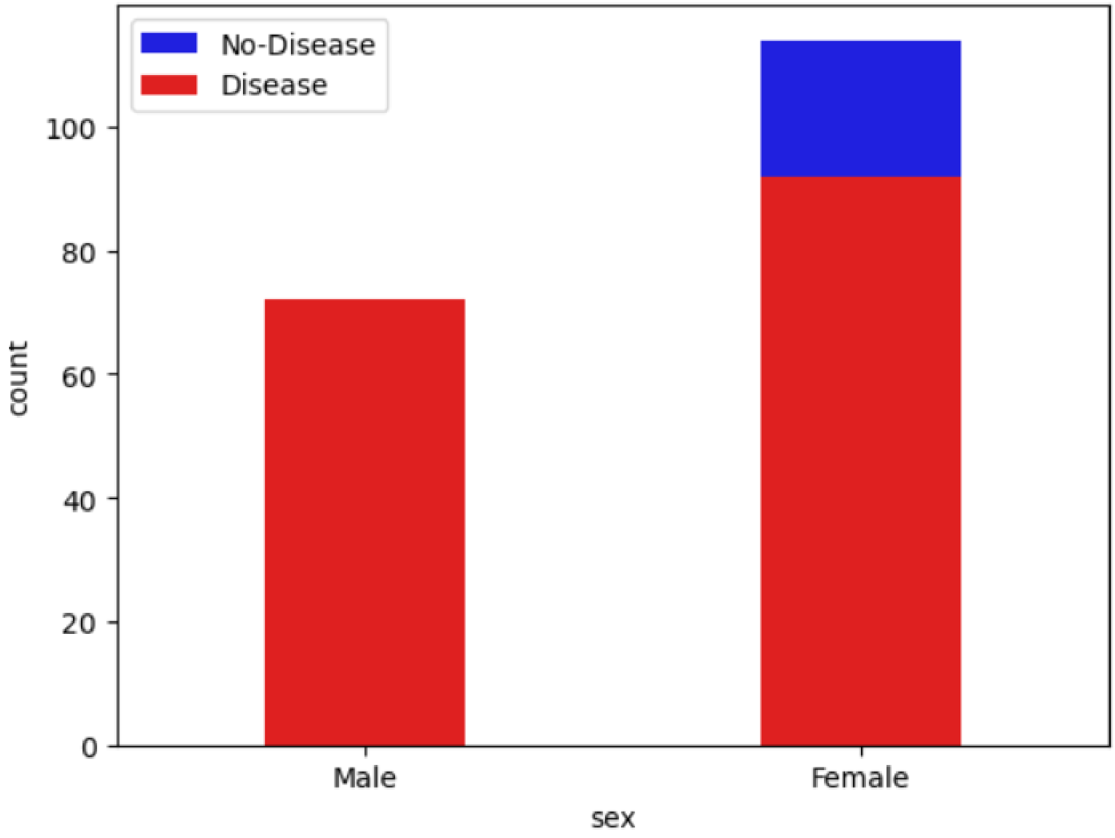
```
sex
1    206
0     96
Name: count, dtype: int64
```

[39]:

```
sns.countplot(x="sex", data=data)
plt.xticks([0, 1],['Female', 'Male'])
plt.show()
```

11. **Find Gender Distribution according to the target variable.**

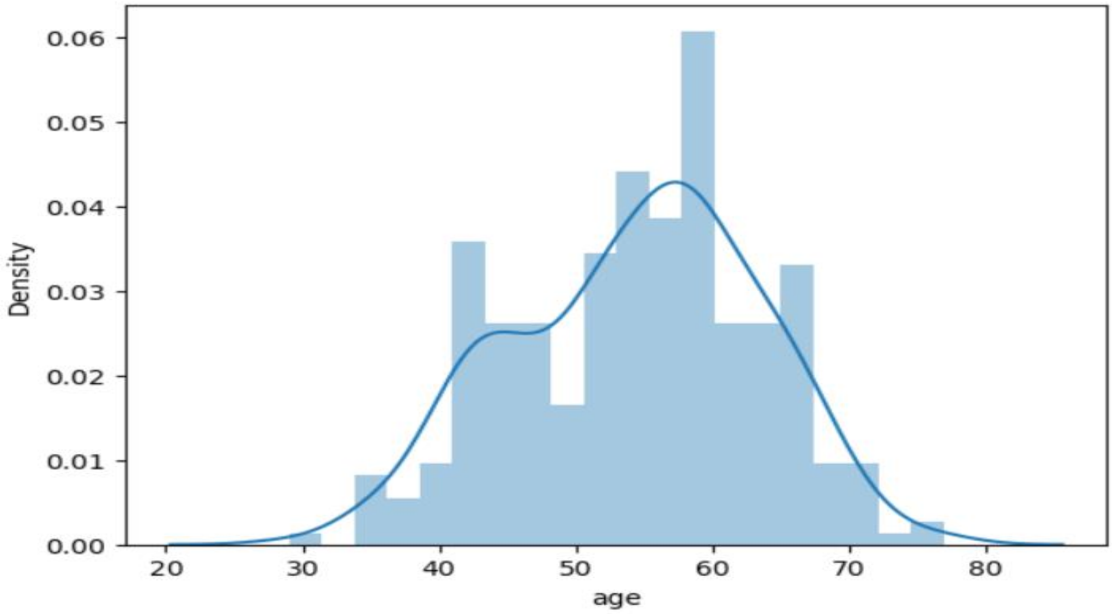
```
data_no_disease = data[data['target'] == 0]
data_disease = data[data['target'] == 1]
sns.countplot(x="sex", data=data_no_disease, color='blue', label='No-Disease', width=0.4)
sns.countplot(x="sex", data=data_disease, color='red', label='Disease', width=0.4)
plt.xticks([0, 1], ["Male", "Female"])
plt.legend(labels=['No-Disease', 'Disease'])
plt.show()
```



12. **Check Age distribution in the dataset.**

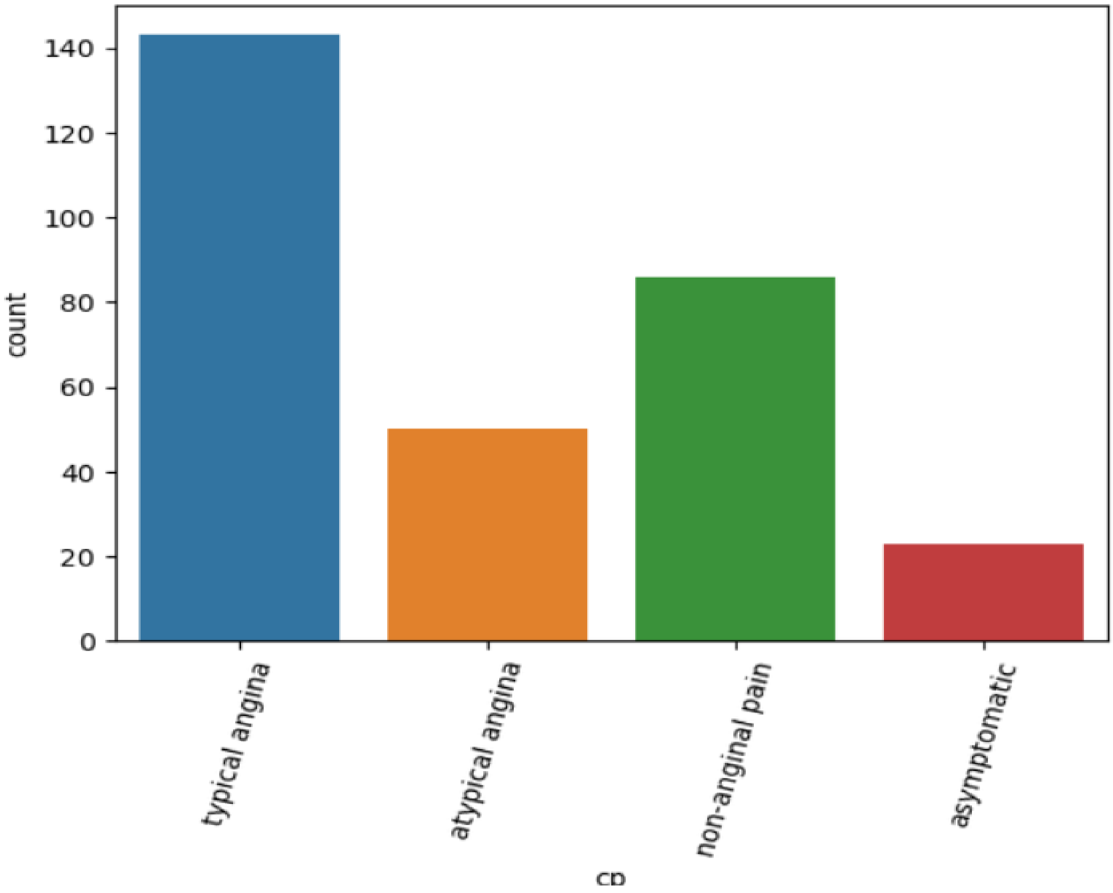
```
sns.distplot(data['age'],bins=20)

plt.show()
```



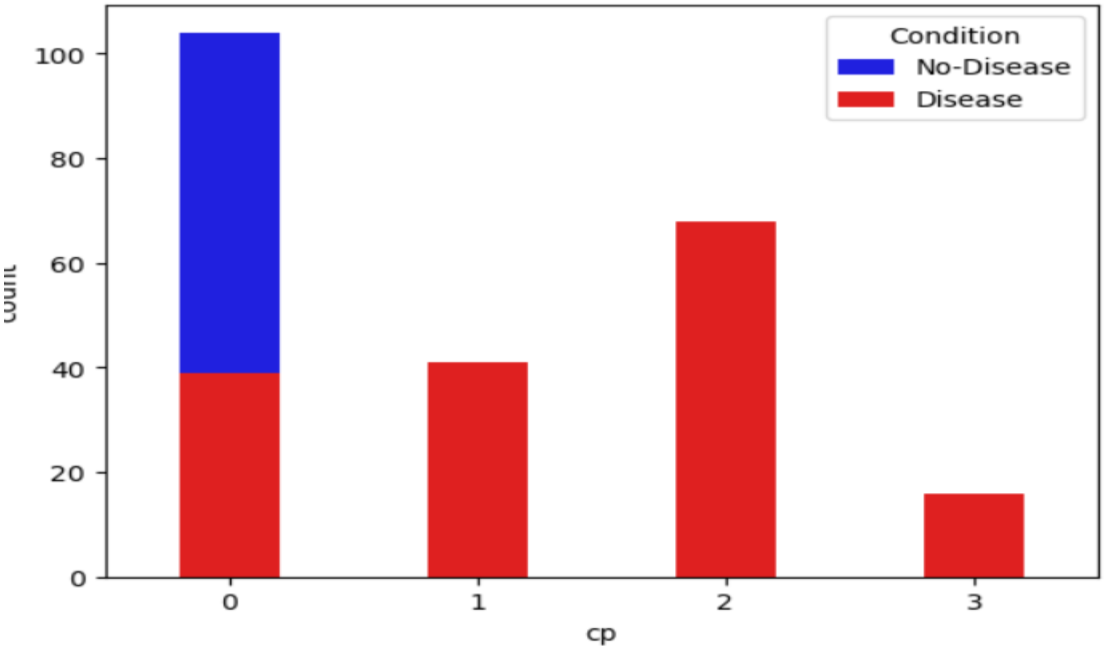
13. Check chest pain type.

```
sns.countplot(x="cp", data=data)
plt.xticks([0,1,2,3],["typical angina","atypical angina","non-anginal pain","asymptomatic"])
plt.xticks(rotation=75)
plt.show()
```



14. show the chest pain distribution as per target variable.

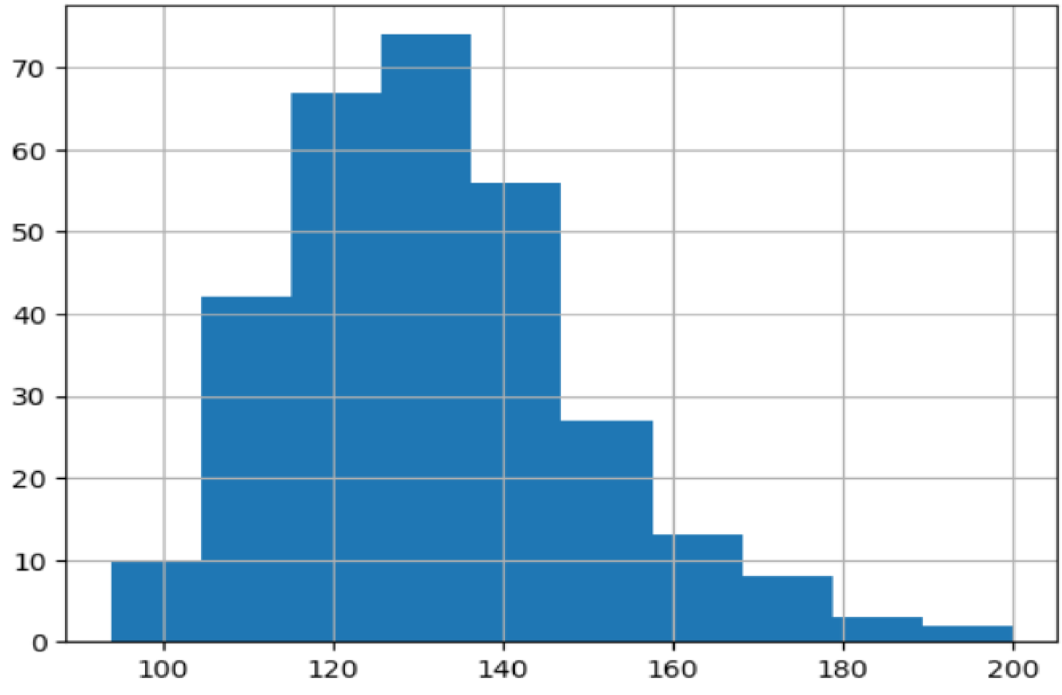
```
data_no_disease = data[data['target'] == 0]
data_disease = data[data['target'] == 1]
sns.countplot(x="cp", data=data_no_disease, color='blue', label='No-Disease', width=0.4)
sns.countplot(x="cp", data=data_disease, color='red', label='Disease', width=0.4)
plt.xticks([0, 1], ["Male", "Female"])
plt.legend(labels=['No-Disease', 'Disease'])
plt.show()
```



15. Check Resting blood pressure distribution.

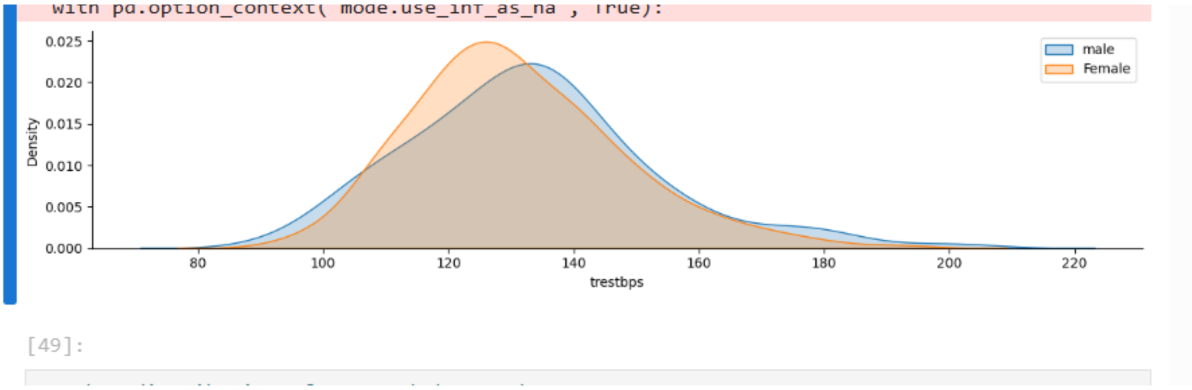
```
data.columns
```

```
data['trestbps'].hist()
```



16. compare Resting blood pressure as per sex column.

```
g = sns.FacetGrid(data, hue="sex", aspect=4)
g.map(sns.kdeplot, 'trestbps', fill=True)
plt.legend(labels=['male','Female'])
plt.show()
```

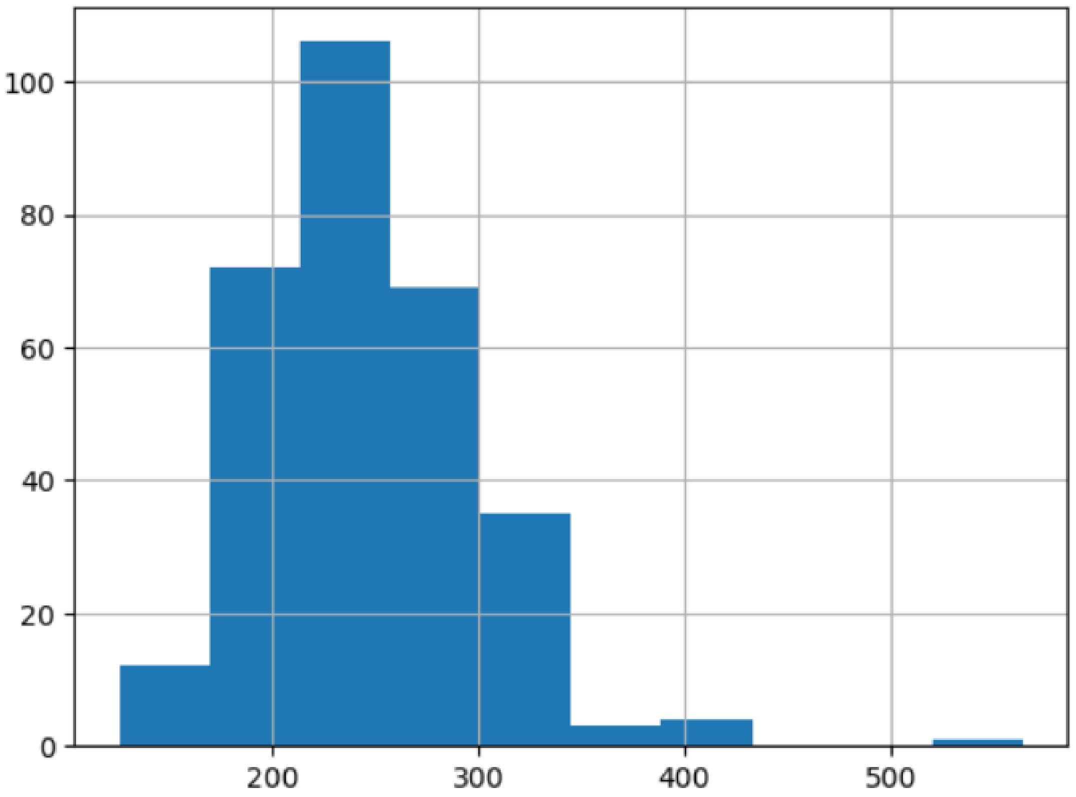


```
with pd.option_context('mode.use_inf_as_na', True):
```

17. show distribution of serum cholesterol.

```
data['chol'].hist()
```

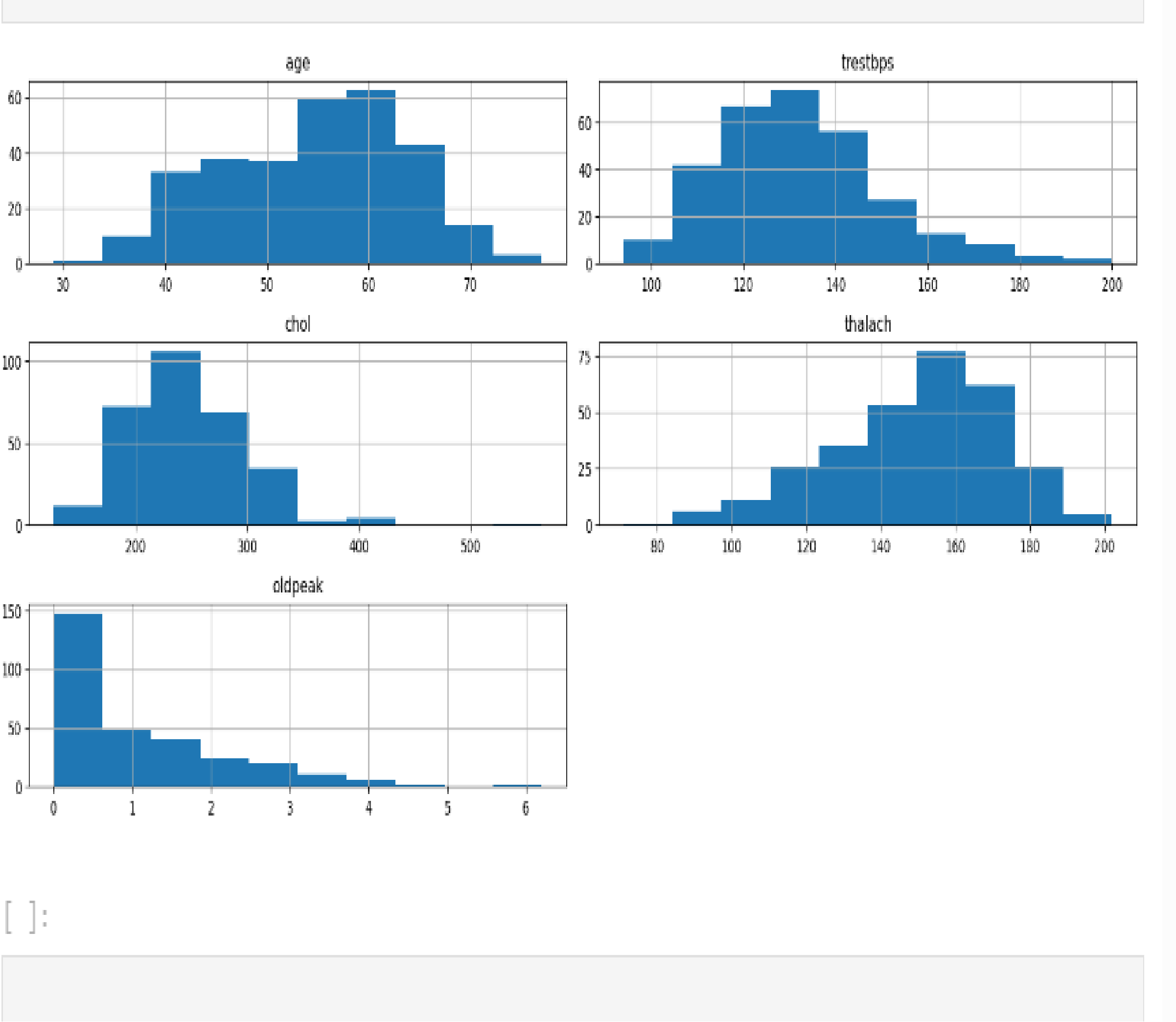
```
<Axes: >
```



18. **plot continuos variable.**
19.

```
cate_val=[]
cont_val=[]

for column in data.columns:
   if data[column].nunique() <=10:
      cate_val.append(column)
   else:
      cont_val.append(column)
cate_val
cont_val
data.hist(cont_val,figsize=(15,6))
plt.tight_layout()
plt.show()
```


age
0
20
40
60
30
40
50
60
70
trestbps
0
20
40
60
100
120
140
160
180
200
chol
0
50
100
200
300
400
500
thalach
0
25
50
75
80
100
120
140
160
180
200
oldpeak
0
50
100
150
0
1
2
3
4
5
6
[ ]: