heart-diesease-analysis

February 2, 2024

[14]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

[15]:
```
data = pd.read_csv(r"E:\PYTHON\Exploratory Data Analysis Projects\Heart␣
↪Diesease\Dataset\heart.csv")
```

Display top 5 rows of the dataset

[16]:
```
data.head()
```

[16]:

	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

Check the Last 5 rows

[17]:
```
data.tail()
```

[17]:

	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				
1022	47	1	0	110	275	0	0	118	1	1.0				
1023	50	0	0	110	254	0	0	159	0	0.0				
1024	54	1	0	120	188	0	1	113	0	1.4				

```
1021      1   1     3       0
1022      1   1     2       0
1023      2   0     2       1
1024      1   1     3       0
```

Find Shape of Dataset(No.of rows & columns)

[18]:
```
print("Number of Rows", data.shape[0])
print("Number of Columns", data.shape[1])
```

```
Number of Rows 1025
Number of Columns 14
```

[19]:
```
data.shape
```

[19]: (1025, 14)

Get Information About Our Dataset Like Total Number Rows, Total Number of Columns, Datatypes Of Each Column And Memory Requirement

[20]:
```
data.info()
```

```
<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
```

Check NullValues

[21]:
```
data.isnull().sum()
```

[21]:
```
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 for Duplicate Data and Drop Them

[22]:
```
data_dup=data.duplicated().any()
print(data_dup)
```

```
True
```

Dropping the Duplicates

[23]:
```
data= data.drop_duplicates()
```

[25]:
```
data.shape
```

[25]:
```
(302, 14)
```

Get Overall Statistics About the Dataset

[26]:
```
data.describe()
```

[26]:

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

	restecg	thalach	exang	oldpeak	slope	ca	\
count	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000	
mean	0.526490	149.569536	0.327815	1.043046	1.397351	0.718543	
std	0.526027	22.903527	0.470196	1.161452	0.616274	1.006748	

```
min      0.000000   71.000000    0.000000    0.000000    0.000000    0.000000
25%      0.000000  133.250000    0.000000    0.000000    1.000000    0.000000
50%      1.000000  152.500000    0.000000    0.800000    1.000000    0.000000
75%      1.000000  166.000000    1.000000    1.600000    2.000000    1.000000
max      2.000000  202.000000    1.000000    6.200000    2.000000    4.000000

             thal      target
count  302.000000  302.000000
mean     2.314570    0.543046
std      0.613026    0.498970
min      0.000000    0.000000
25%      2.000000    0.000000
50%      2.000000    1.000000
75%      3.000000    1.000000
max      3.000000    1.000000
```

Draw Correlation Matrix

[27]:
```
data.corr()
```

[27]:
```
                age       sex        cp  trestbps      chol       fbs  \
age        1.000000 -0.094962 -0.063107  0.283121  0.207216  0.119492
sex       -0.094962  1.000000 -0.051740 -0.057647 -0.195571  0.046022
cp        -0.063107 -0.051740  1.000000  0.046486 -0.072682  0.096018
trestbps   0.283121 -0.057647  0.046486  1.000000  0.125256  0.178125
chol       0.207216 -0.195571 -0.072682  0.125256  1.000000  0.011428
fbs        0.119492  0.046022  0.096018  0.178125  0.011428  1.000000
restecg   -0.111590 -0.060351  0.041561 -0.115367 -0.147602 -0.083081
thalach   -0.395235 -0.046439  0.293367 -0.048023 -0.005308 -0.007169
exang      0.093216  0.143460 -0.392937  0.068526  0.064099  0.024729
oldpeak    0.206040  0.098322 -0.146692  0.194600  0.050086  0.004514
slope     -0.164124 -0.032990  0.116854 -0.122873  0.000417 -0.058654
ca         0.302261  0.113060 -0.195356  0.099248  0.086878  0.144935
thal       0.065317  0.211452 -0.160370  0.062870  0.096810 -0.032752
target    -0.221476 -0.283609  0.432080 -0.146269 -0.081437 -0.026826

             restecg   thalach     exang   oldpeak     slope        ca  \
age        -0.111590 -0.395235  0.093216  0.206040 -0.164124  0.302261
sex        -0.060351 -0.046439  0.143460  0.098322 -0.032990  0.113060
cp          0.041561  0.293367 -0.392937 -0.146692  0.116854 -0.195356
trestbps   -0.115367 -0.048023  0.068526  0.194600 -0.122873  0.099248
chol       -0.147602 -0.005308  0.064099  0.050086  0.000417  0.086878
fbs        -0.083081 -0.007169  0.024729  0.004514 -0.058654  0.144935
restecg     1.000000  0.041210 -0.068807 -0.056251  0.090402 -0.083112
thalach     0.041210  1.000000 -0.377411 -0.342201  0.384754 -0.228311
exang      -0.068807 -0.377411  1.000000  0.286766 -0.256106  0.125377
oldpeak    -0.056251 -0.342201  0.286766  1.000000 -0.576314  0.236560
```

```
slope     0.090402  0.384754 -0.256106 -0.576314  1.000000 -0.092236
ca       -0.083112 -0.228311  0.125377  0.236560 -0.092236  1.000000
thal     -0.010473 -0.094910  0.205826  0.209090 -0.103314  0.160085
target    0.134874  0.419955 -0.435601 -0.429146  0.343940 -0.408992

              thal    target
age       0.065317 -0.221476
sex       0.211452 -0.283609
cp       -0.160370  0.432080
trestbps  0.062870 -0.146269
chol      0.096810 -0.081437
fbs      -0.032752 -0.026826
restecg  -0.010473  0.134874
thalach  -0.094910  0.419955
exang     0.205826 -0.435601
oldpeak   0.209090 -0.429146
slope    -0.103314  0.343940
ca        0.160085 -0.408992
thal      1.000000 -0.343101
target   -0.343101  1.000000
```

[28]:
```
plt.figure(figsize=(17,6))

sns.heatmap(data.corr(), annot=True)
```

[28]: <Axes: >

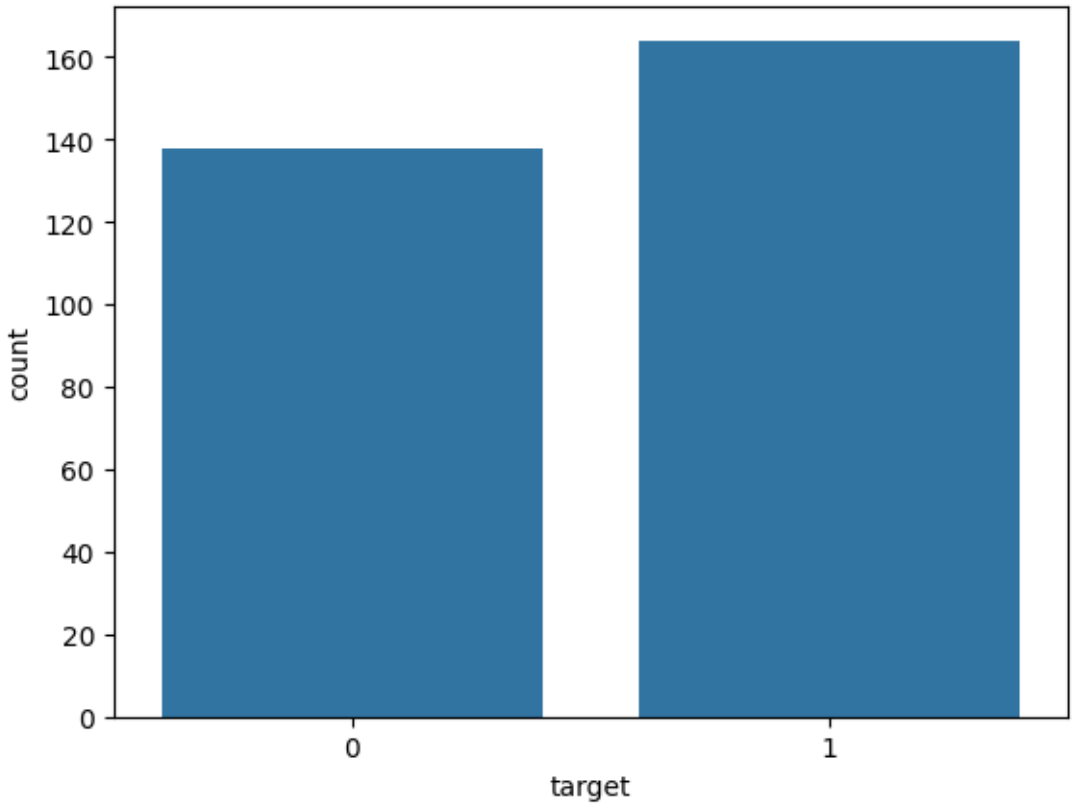
How many people have heart disease , and how many don't have heart disease in this dataset?

[31]:
```
data['target'].value_counts()
```

[31]:
```
target
1    164
0    138
Name: count, dtype: int64
```

[34]:
```
sns.countplot(x=data['target'])
```

[34]:
```
<Axes: xlabel='target', ylabel='count'>
```

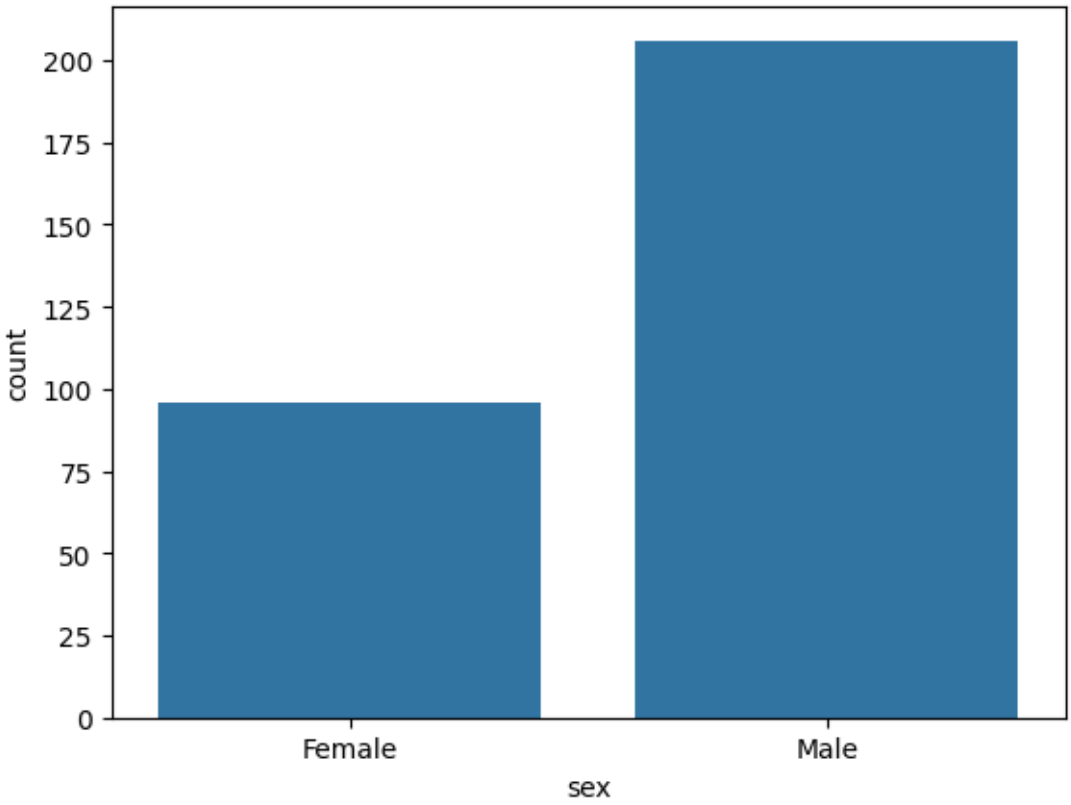


Find Count of Male& Female in this dataset

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

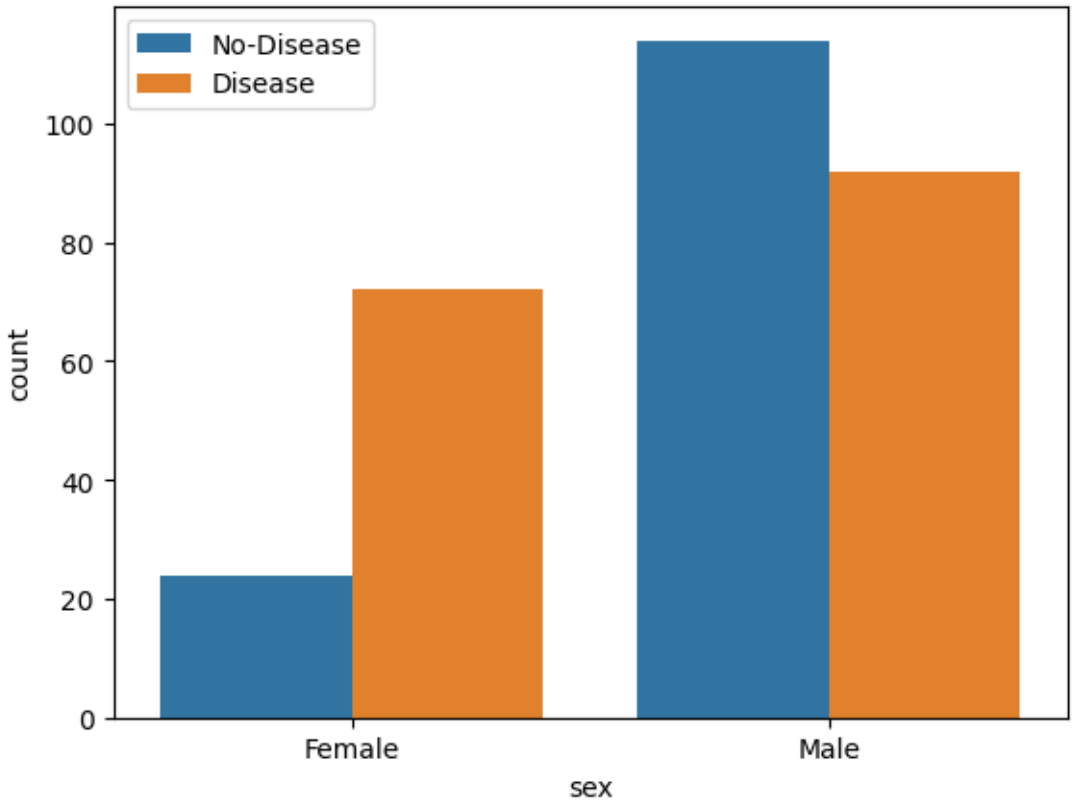
[35]:
```
sex
1    206
0     96
Name: count, dtype: int64
```

[37]:
```
sns.countplot(x=data['sex'])
plt.xticks([0,1],['Female','Male'])
plt.show()
```



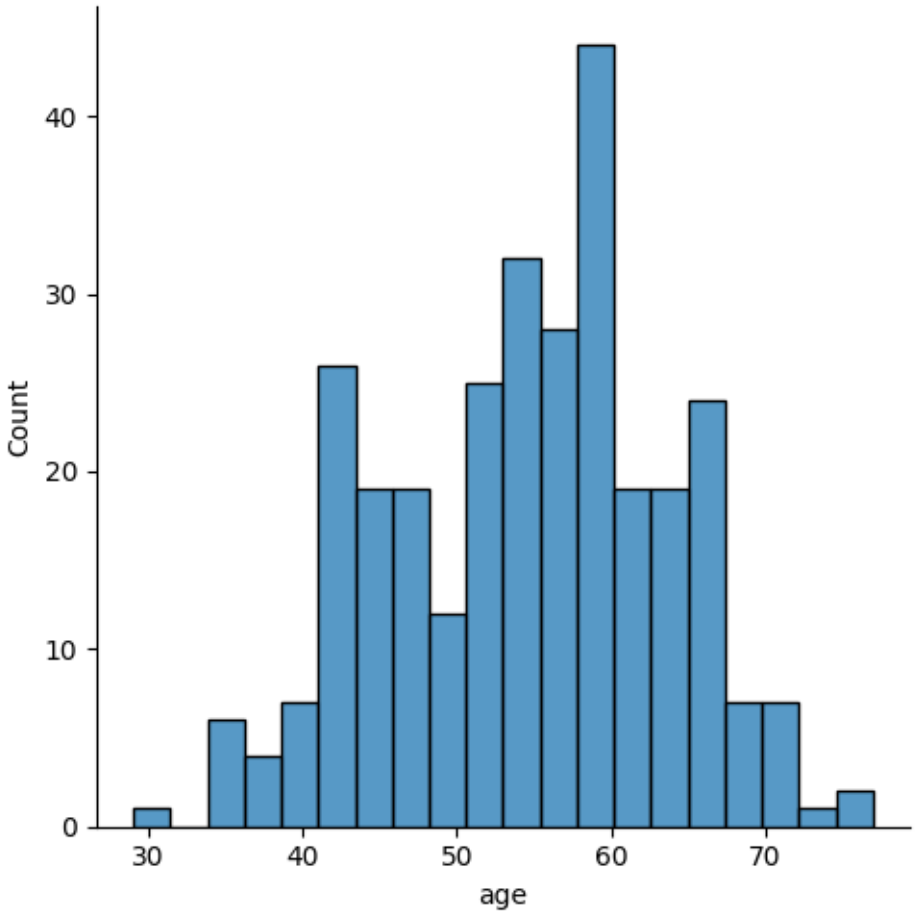
Find Gender Distribution According to the target variable

[39]:
```
sns.countplot(x='sex' , hue="target", data=data)
plt.xticks([1,0],['Male', 'Female'])
plt.legend(labels=['No-Disease', 'Disease'])
plt.show()
```



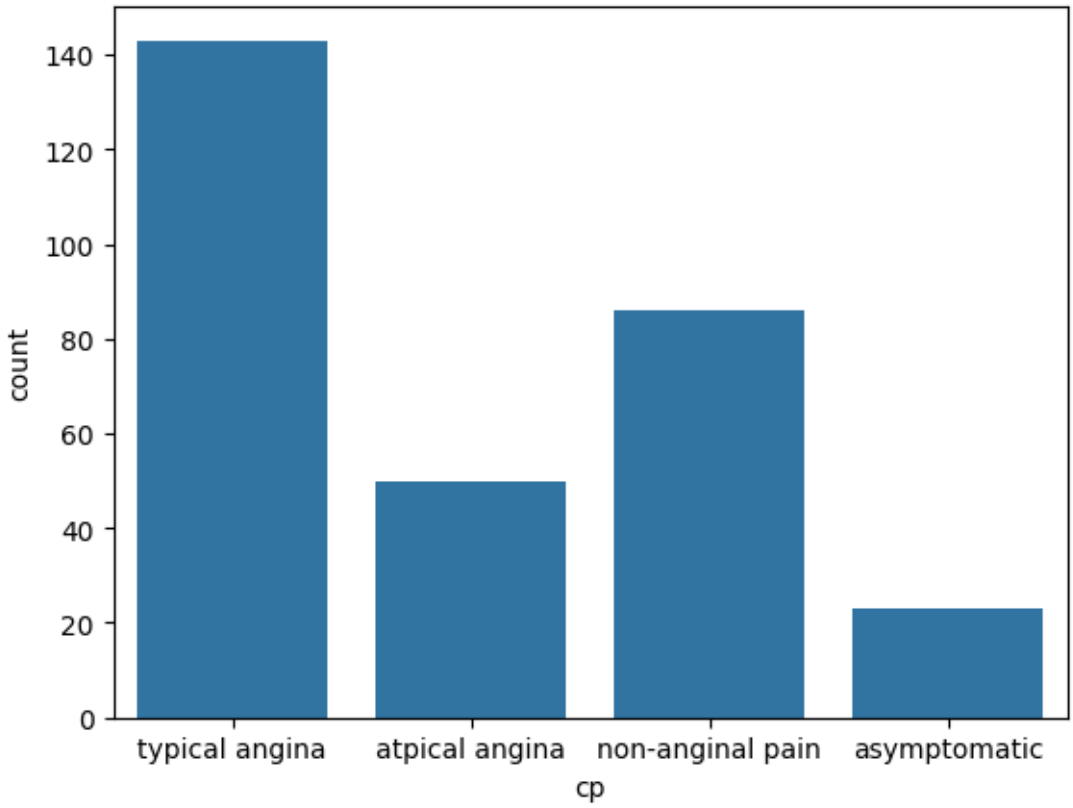
Check Age Distributionin The Data Set

[41]:
```
sns.displot(data['age'],bins=20)
plt.show()
```



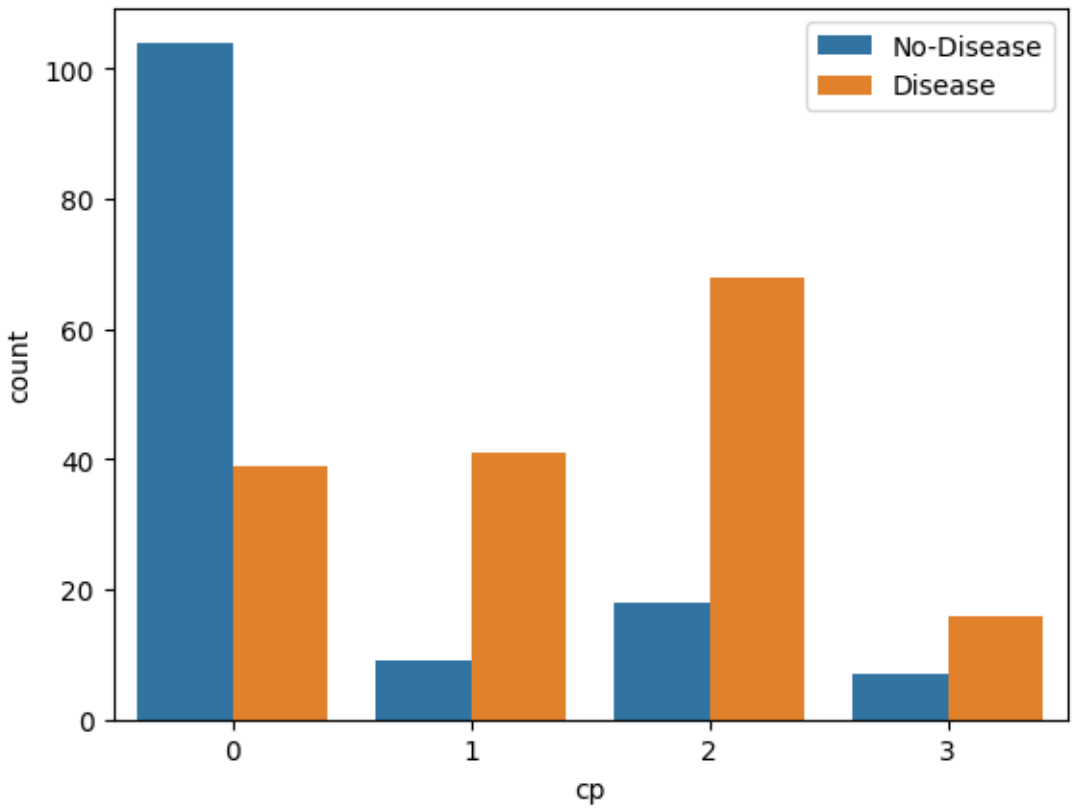
Check Chest Pain Type

[44]:
```
sns.countplot(x=data['cp'])
plt.xticks([0,1,2,3],["typical angina","atpical angina","non-anginal␣
↪pain","asymptomatic"])
plt.show()
```



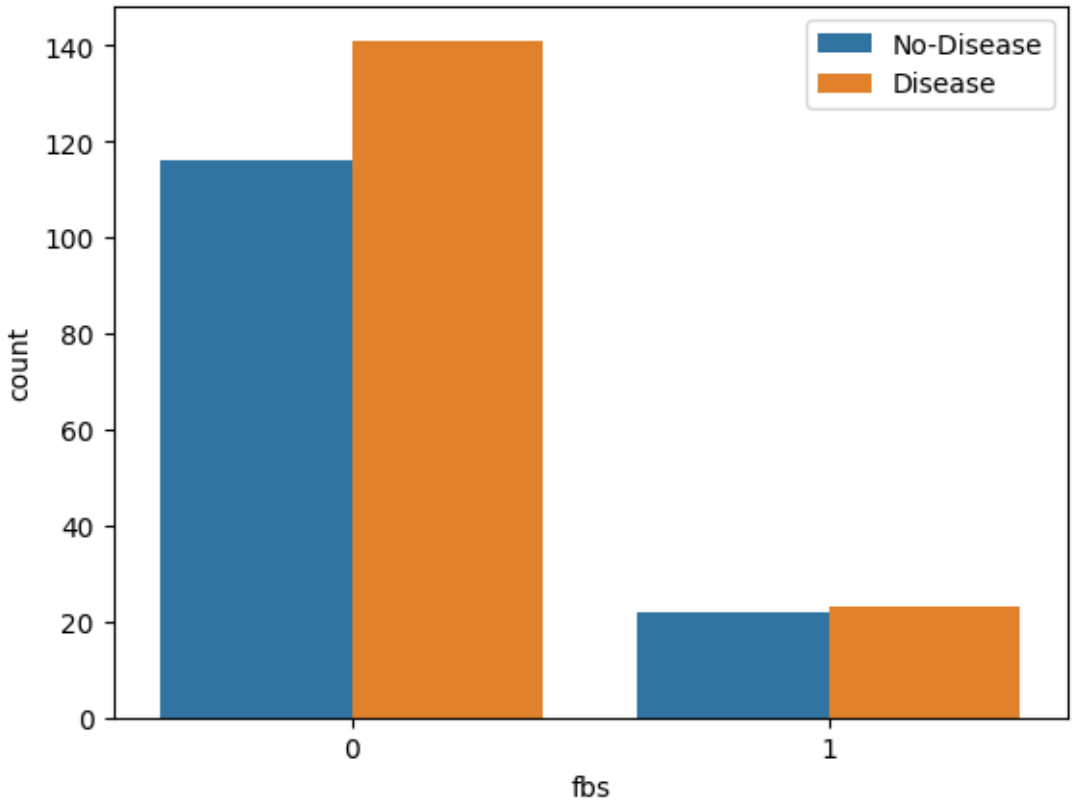
Show the Chest Pain Distribution As Per Target Variable

[47]:
```
sns.countplot(x="cp" , hue="target", data=data)
plt.legend(labels=["No-Disease", "Disease"])
plt.show()
```



Show Fasting Blood Sugar Distribution According Ti Target Variable

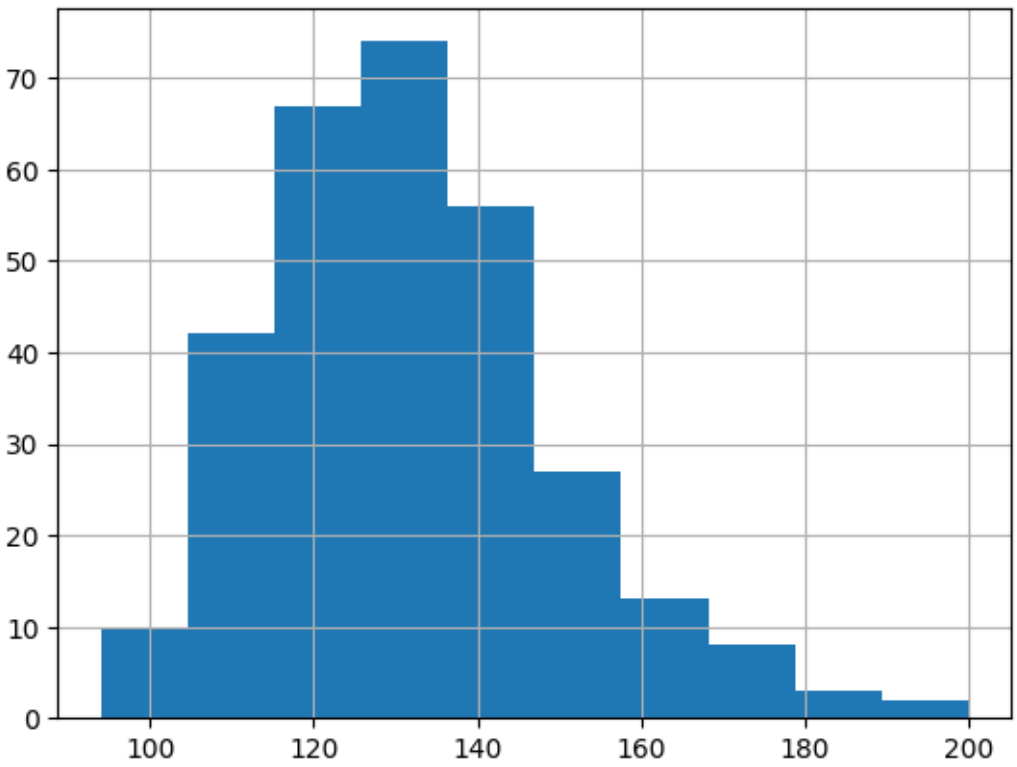
[48]:
```
sns.countplot(x="fbs" , hue="target", data=data)
plt.legend(labels=["No-Disease", "Disease"])
plt.show()
```



Check Resting Blood Pressure distribution

[49]:
```
data['trestbps'].hist()
```

[49]: <Axes: >



Compare Resting Blood Pressure As Per Sex Column

[51]:
```
g= sns.FacetGrid(data,hue="sex",aspect=4)

g.map(sns.kdeplot,'trestbps', shade=True)

plt.legend(labels=['Male', 'Female'])

plt.show()
```

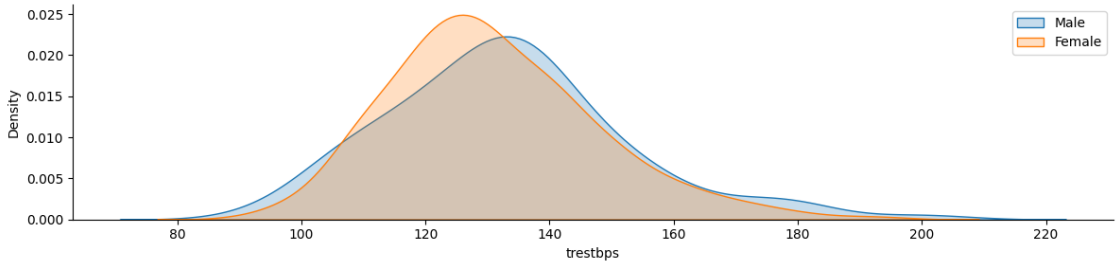
```
C:\Users\Admin\AppData\Roaming\Python\Python312\site-
packages\seaborn\axisgrid.py:854: FutureWarning:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  func(*plot_args, **plot_kwargs)
C:\Users\Admin\AppData\Roaming\Python\Python312\site-
packages\seaborn\axisgrid.py:854: FutureWarning:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.
```

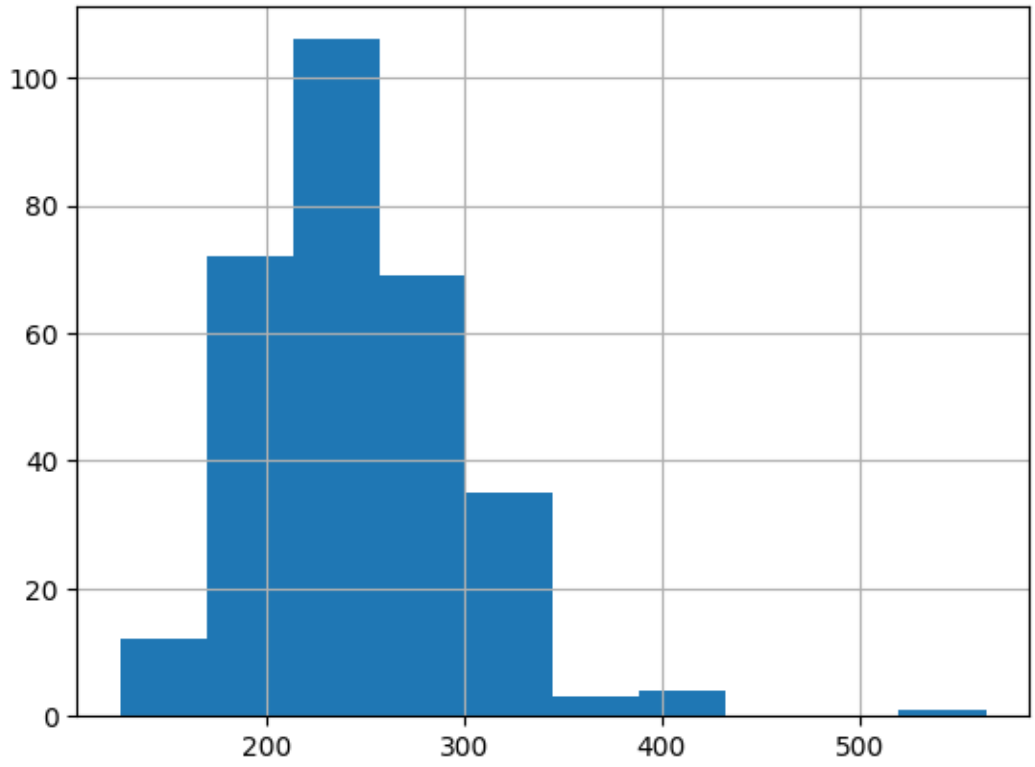
```
func(*plot_args, **plot_kwargs)
```



Show Distribution of Serum cholestrol

[52]:
```
data['chol'].hist()
```

[52]: <Axes: >



Plot Continious Variables

[54]:
```
data.columns
```

[54]:
```
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
      dtype='object')
```

[60]:
```
cate_val=[]
cont_val=[]

for column in data.columns:
     if data[column].nunique() <=10:
          cate_val.append(column)
else:
        cont_val.append(column)
```

[61]:
```
cate_val
```

[61]:
```
['sex', 'cp', 'fbs', 'restecg', 'exang', 'slope', 'ca', 'thal', 'target']
```

[62]:
```
cont_val
```

[62]:
```
['target']
```

[63]:
```
data.hist(cont_val,figsize=(15,6))
plt.tight_layout()
plt.show()
```