Heart-Disease-Analysis

September 16, 2023

[24]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

[25]:
```
data=pd.read_csv('heart/heart.csv')
```

[26]:
```
# print first 5 rowes
data.head(5)
```

[26]:

	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

[27]:
```
# print the last 5 rows
data.tail(5)
```

[27]:

	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

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

[28]:
```
data.shape
```

[28]:
```
(1025, 14)
```

[29]:
```
print("number of rows: ",data.shape[0])
print("number of coulums: ", data.shape[1])
```

```
number of rows:  1025
number of coulums:  14
```

[30]:
```
# checking information related to data
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
```

[31]:
```
# checking for the null values
data.isnull()
```

[31]:
```
        age    sex     cp  trestbps   chol    fbs  restecg  thalach  exang  \
0     False  False  False     False  False  False    False    False  False
1     False  False  False     False  False  False    False    False  False
2     False  False  False     False  False  False    False    False  False
```

```
3     False  False  False     False  False  False    False    False  False
4     False  False  False     False  False  False    False    False  False
…     …    …    …       …    …    …       …      …
1020  False  False  False     False  False  False    False    False  False
1021  False  False  False     False  False  False    False    False  False
1022  False  False  False     False  False  False    False    False  False
1023  False  False  False     False  False  False    False    False  False
1024  False  False  False     False  False  False    False    False  False

      oldpeak  slope     ca   thal  target
0       False  False  False  False   False
1       False  False  False  False   False
2       False  False  False  False   False
3       False  False  False  False   False
4       False  False  False  False   False
…       …    …    …    …     …
1020    False  False  False  False   False
1021    False  False  False  False   False
1022    False  False  False  False   False
1023    False  False  False  False   False
1024    False  False  False  False   False

[1025 rows x 14 columns]
```

[32]:
```
#converting null values to 0
data.isnull().sum()
```

[32]:
```
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
```

[33]:
```
# checking duplicate data and dropping it
data_dup=data.duplicated().any()
print(data_dup)
```

True

[34]:
```
# dropping
data=data.drop_duplicates()
```

[35]:
```
# now just checking  how much data reduced
data.shape
```

[35]: (302, 14)

[36]:
```
# getting overall statistics for datasets
data.describe()
```

[36]:

	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

[37]:
```
# Now to check the relationship between different column forming the␣
 ↪Correlation Matrix
# data.corr()
plt.figure(figsize=(17,6))
```

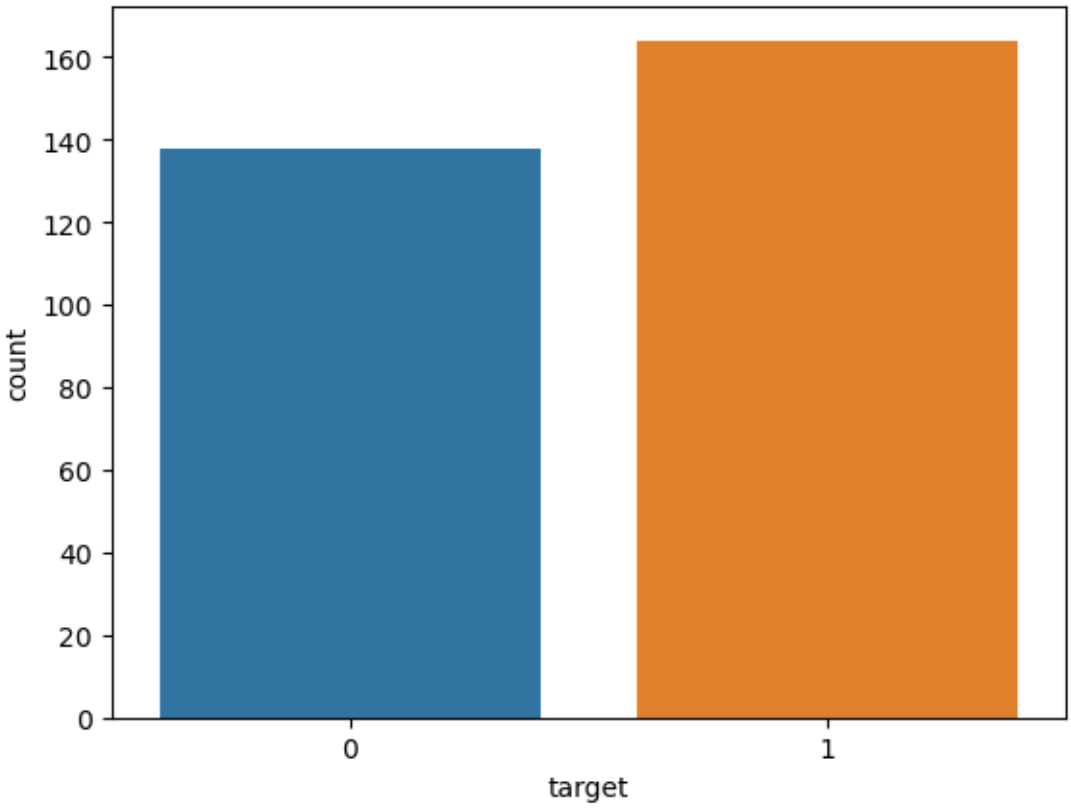
```
sns.heatmap(data.corr(),annot=True)
plt.show()
```

[38]:
```
# what correlation matrix do
```

[39]:
```
# calculating how many people have heart diseas and how many don't have this
data['target'].value_counts()
```

[39]:
```
target
1    164
0    138
Name: count, dtype: int64
```

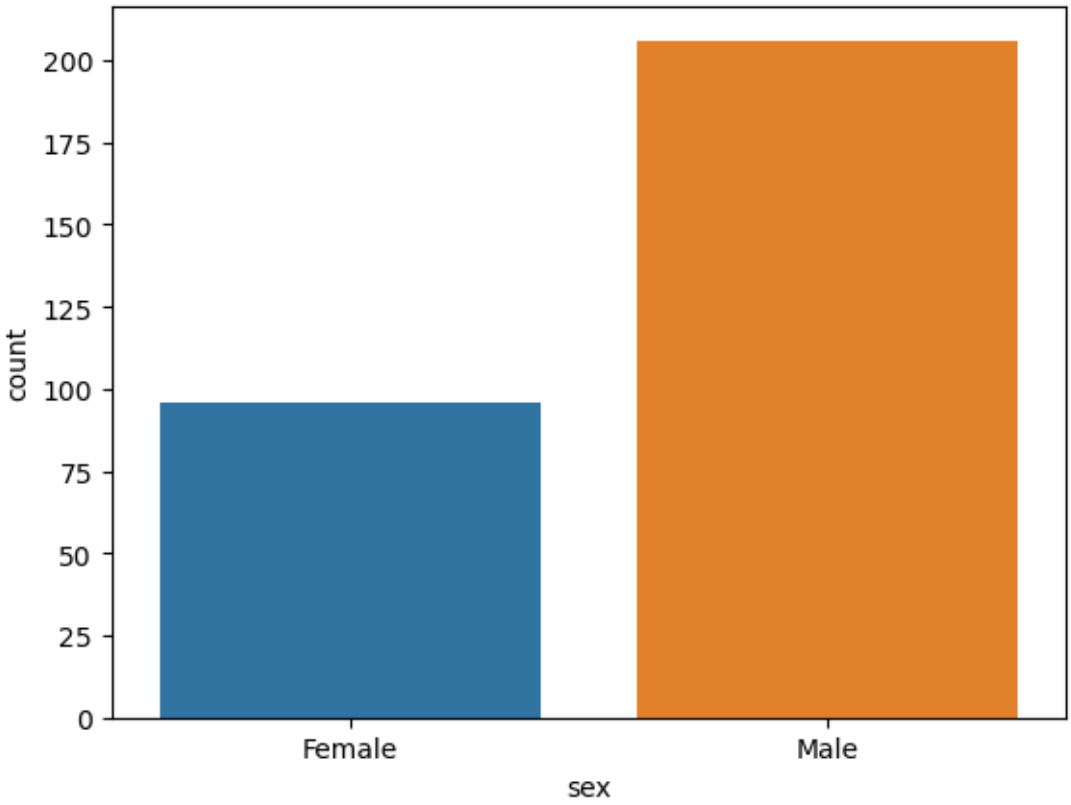
[40]:
```
# checking the
sns.countplot(x=data['target'])
plt.show()
# sns.countplot(x="target",data=data)
```



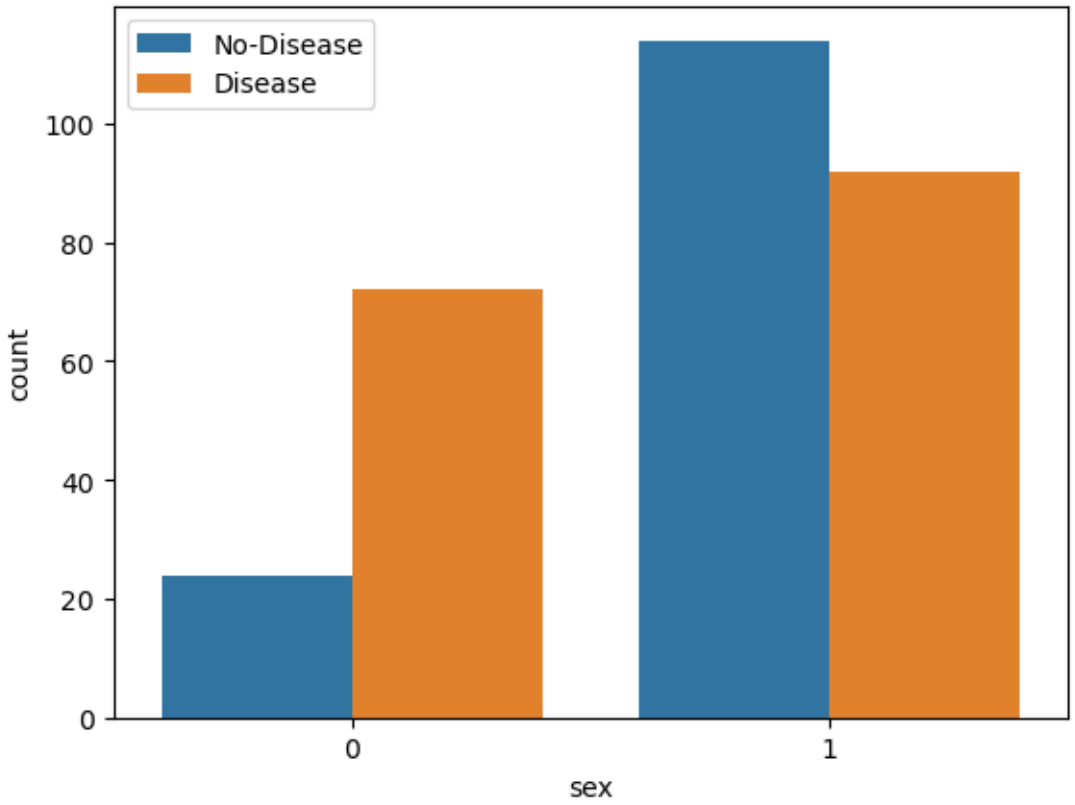
[41]:
```
# counting number of male and females
data['sex'].value_counts()
```

[41]:
```
sex
1    206
0     96
Name: count, dtype: int64
```

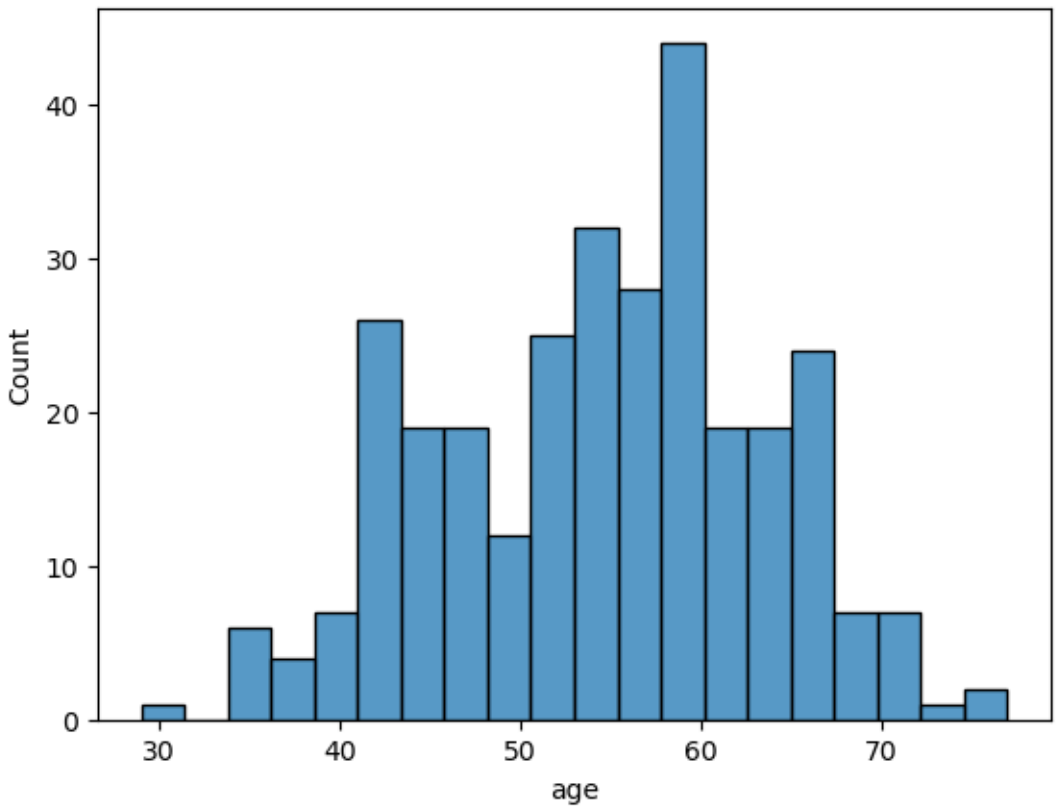
[42]:
```
# showing graphically the males and female
# sns.countplot(data['sex'])
sns.countplot(x='sex',data=data)
plt.xticks([0,1],['Female','Male'])
plt.show()
```



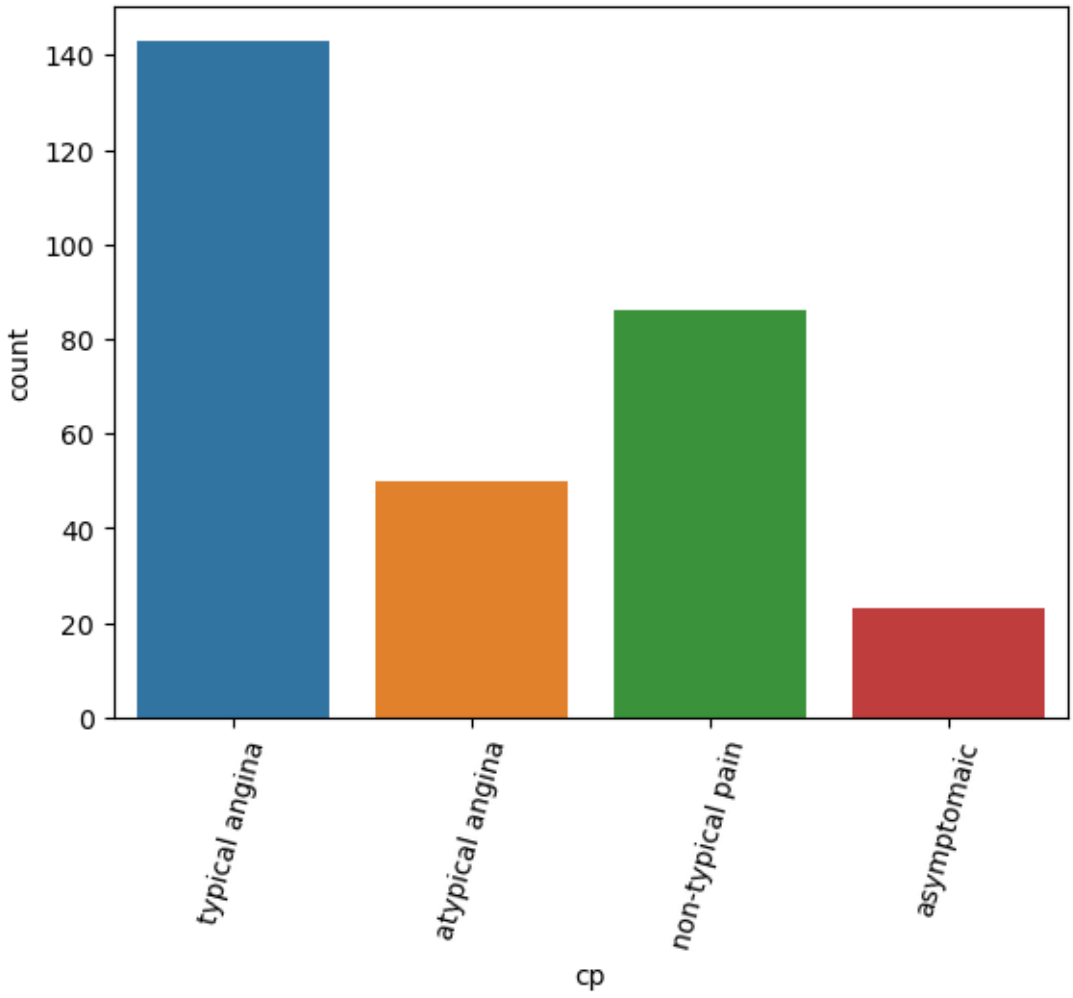
[43]:
```
sns.countplot(x='sex',hue="target",data=data)
# plt.xtricks([0,1],['Male','Female'])
plt.legend(labels=['No-Disease','Disease'])
plt.show()
```



[44]:
```
# checking for the people on age basis who were victim of heart disease
sns.histplot(data['age'],bins=20)
plt.show()
```



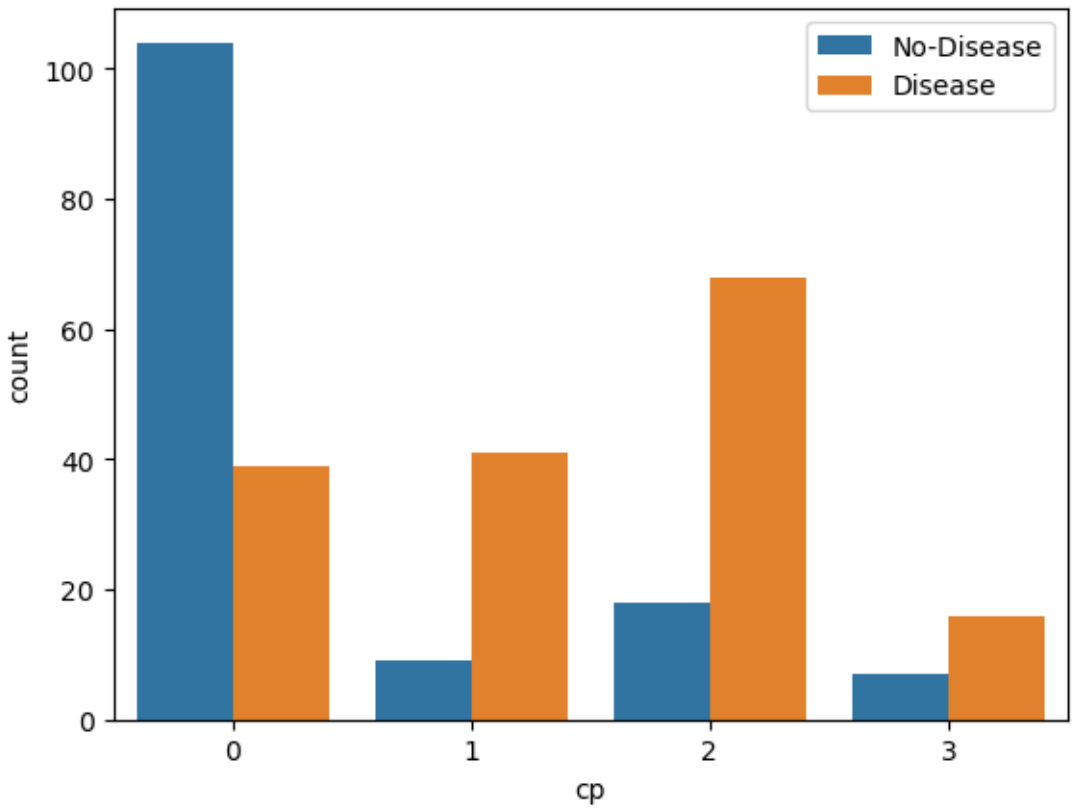
[45]:
```
#checking chest pain type
sns.countplot(x=data['cp'])
plt.xticks([0,1,2,3],["typical angina","atypical angina","non-typical␣
 ↪pain","asymptomaic"])
plt.xticks(rotation=75)
plt.show()
```



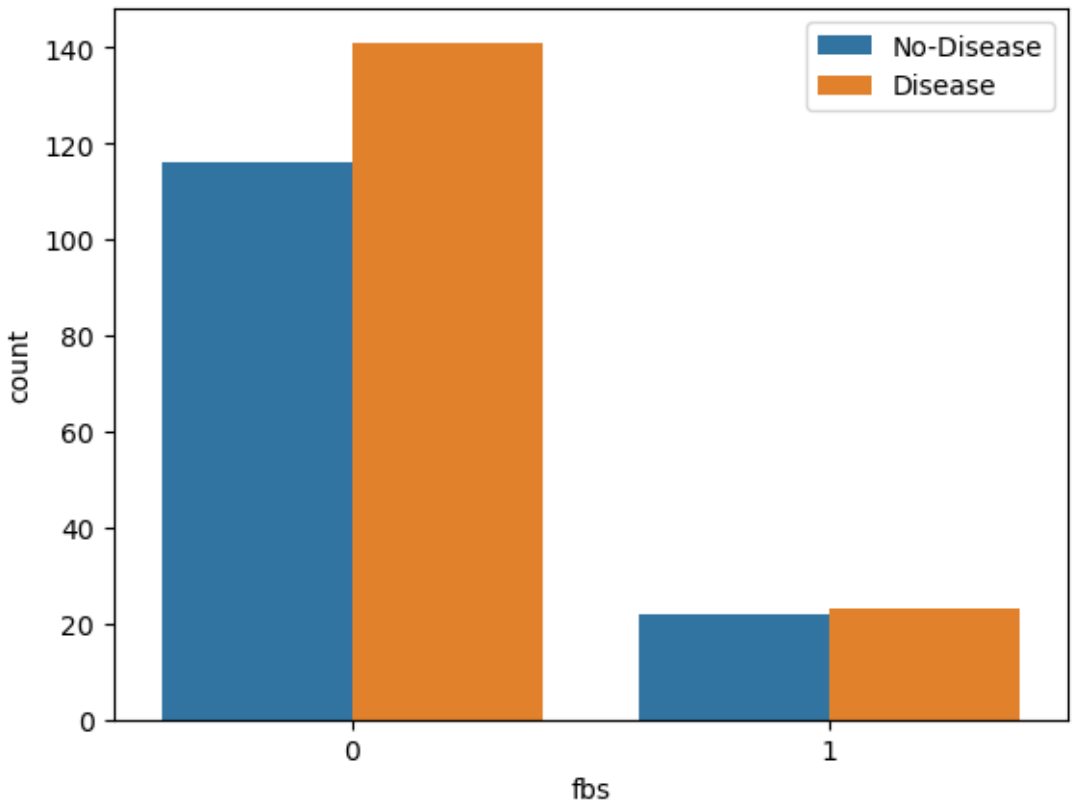
[46]:
```
data.columns
```

[46]:
```
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
      dtype='object')
```

[47]:
```
# identifying who suffer from the chest pain most
sns.countplot(x="cp",hue="target",data=data)
plt.legend(labels=["No-Disease","Disease"])
plt.show()
```



[48]:
```
# showing fasting blood sugar distibution according to target variable
sns.countplot(x="fbs",hue="target",data=data)
plt.legend(labels=["No-Disease","Disease"])
plt.show()
```

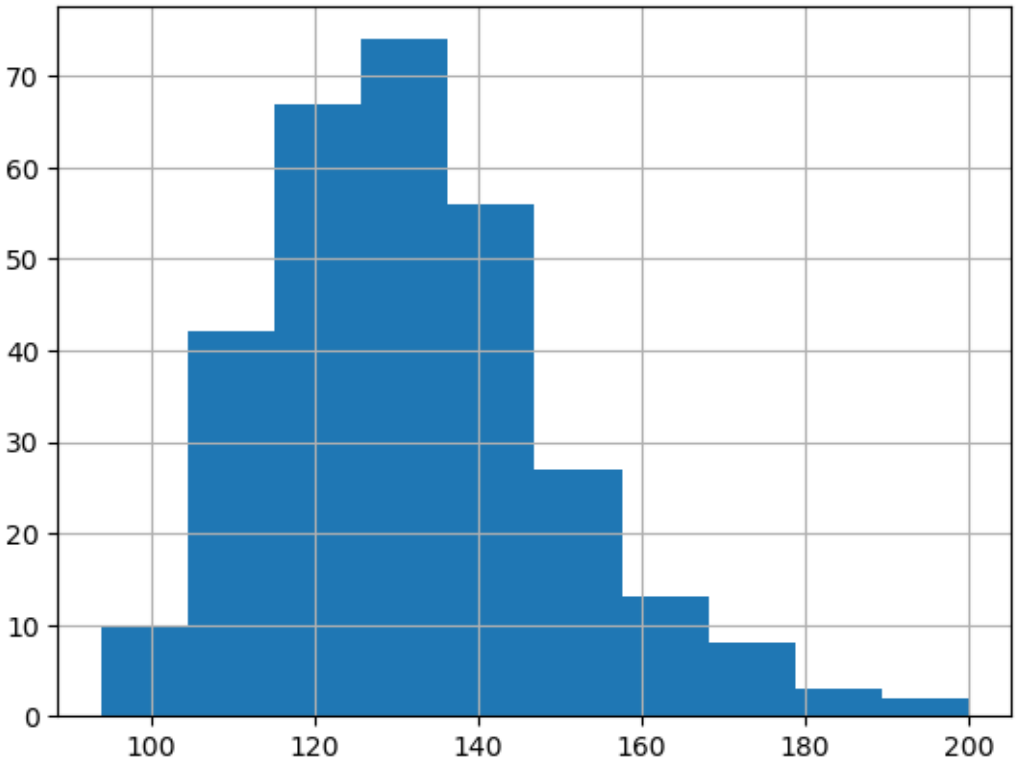


[49]:
```
# checking resting blood pressure distribution
data.columns
```

[49]:
```
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
      dtype='object')
```

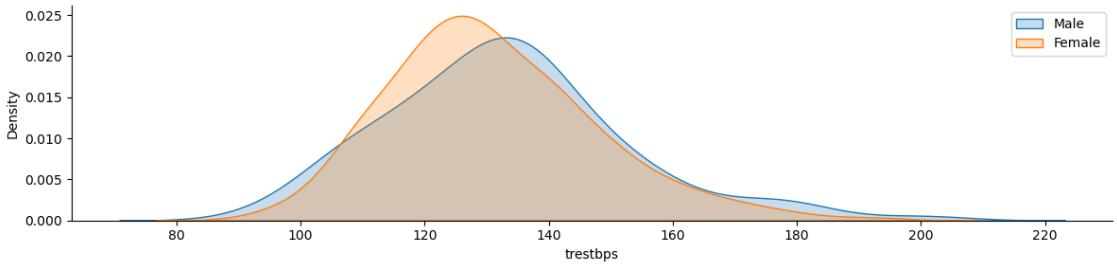
[50]:
```
data['trestbps'].hist()
```

[50]:
```
<AxesSubplot:>
```



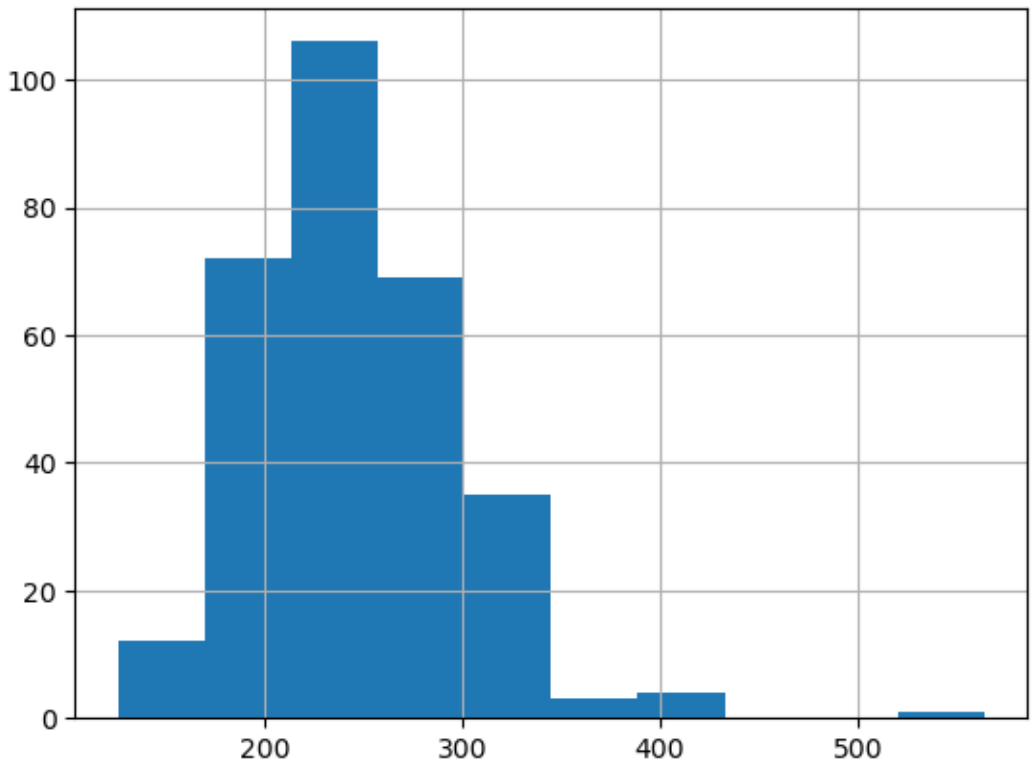
[51]:
```
# comparing resting blood pressure as per sex column
g=sns.FacetGrid(data,hue="sex",aspect=4)
g.map(sns.kdeplot,'trestbps',shade=True)
plt.legend(labels=['Male','Female'])
```

[51]: <matplotlib.legend.Legend at 0x7f241aff9c00>



[52]:
```
# showing distibution of serum cholestrol
data['chol'].hist()
```

[52]: <AxesSubplot:>



[53]:
```
# ploting continous variables
cate_val=[]
cont_val=[]
for column in data.columns:
    if data[column].nunique()<=10:
        cate_val.append(column)
    else:
        cont_val.append(column)
```

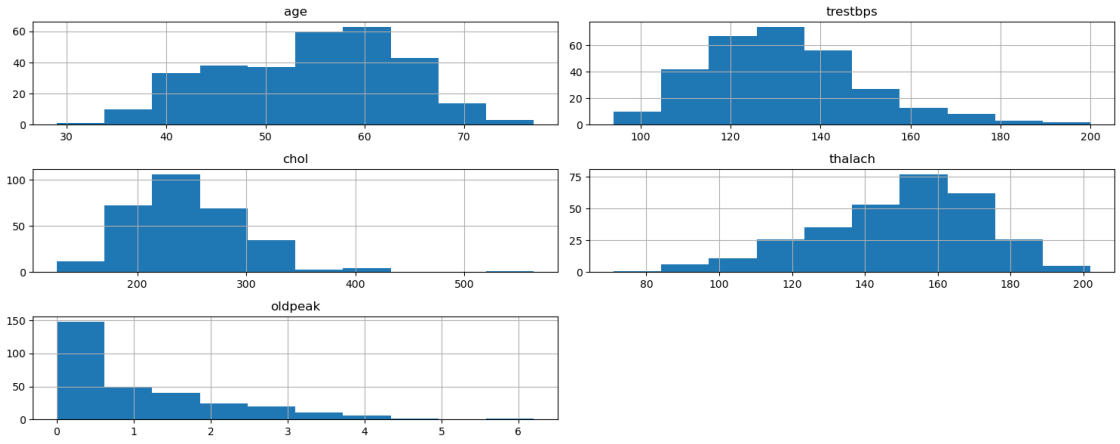
[54]:
```
cate_val
```

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

[55]:
```
cont_val
```

[55]: ['age', 'trestbps', 'chol', 'thalach', 'oldpeak']

[56]:
```
data.hist(cont_val,figsize=(15,6))
plt.tight_layout()
plt.show()
```


age
trestbps
chol
thalach
oldpeak