HEART DISEASE ANALYSIS

February 20, 2025

IMPORTING LIBRARIES

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Step 2: Loading and Exploring the Dataset

[2]:
```
df = pd.read_csv(r"C:\Users\Josphat\Desktop\DATA ANALYSIS VIDEOS AND␣
 ↪DATASETS-DATA THINKERS\HEART DISEASE.csv")
```

[3]:
```
df.head()
```

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

   ca  thal  target
0   2     3       0
1   0     3       0
2   0     3       0
3   1     3       0
4   3     2       0
```

[4]:
```
#finding the shape of the dataset
df.shape
print("Number of Rows:", df.shape[0])
print("Number of Columns:", df.shape[1])
```

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

[5]:
```
#checking for null values
df.isnull().sum()
```

```
[5]: 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]: #finding information about the dataset
     df.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
```

```
[7]: #finding columns of the dataset
     df.columns
```

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

[8]:

```
#finding the descriptive statistics of the dataset
df.describe()
```

[8]:

```
               age          sex           cp     trestbps        chol  \
count  1025.000000  1025.000000  1025.000000  1025.000000  1025.00000
mean     54.434146     0.695610     0.942439   131.611707   246.00000
std       9.072290     0.460373     1.029641    17.516718    51.59251
min      29.000000     0.000000     0.000000    94.000000   126.00000
25%      48.000000     0.000000     0.000000   120.000000   211.00000
50%      56.000000     1.000000     1.000000   130.000000   240.00000
75%      61.000000     1.000000     2.000000   140.000000   275.00000
max      77.000000     1.000000     3.000000   200.000000   564.00000

               fbs      restecg      thalach        exang      oldpeak  \
count  1025.000000  1025.000000  1025.000000  1025.000000  1025.000000
mean      0.149268     0.529756   149.114146     0.336585     1.071512
std       0.356527     0.527878    23.005724     0.472772     1.175053
min       0.000000     0.000000    71.000000     0.000000     0.000000
25%       0.000000     0.000000   132.000000     0.000000     0.000000
50%       0.000000     1.000000   152.000000     0.000000     0.800000
75%       0.000000     1.000000   166.000000     1.000000     1.800000
max       1.000000     2.000000   202.000000     1.000000     6.200000

             slope           ca         thal       target
count  1025.000000  1025.000000  1025.000000  1025.000000
mean      1.385366     0.754146     2.323902     0.513171
std       0.617755     1.030798     0.620660     0.500070
min       0.000000     0.000000     0.000000     0.000000
25%       1.000000     0.000000     2.000000     0.000000
50%       1.000000     0.000000     2.000000     1.000000
75%       2.000000     1.000000     3.000000     1.000000
max       2.000000     4.000000     3.000000     1.000000
```

[9]:

```
#checking for duplicated values and Dropping them
data_dup=df.duplicated().any()
print(data_dup)
```

```
True
```

[10]:

```
df.duplicated().sum()
```

[10]:

```
723
```

[11]:

```
df.shape
```

[11]:

```
(1025, 14)
```

[12]:

```
#dropping the duplicated values
df.drop_duplicates(inplace = True)
```

[13]:

```
df.shape
```

[13]:

```
(302, 14)
```

[14]:

```
df.describe()
```

[14]:

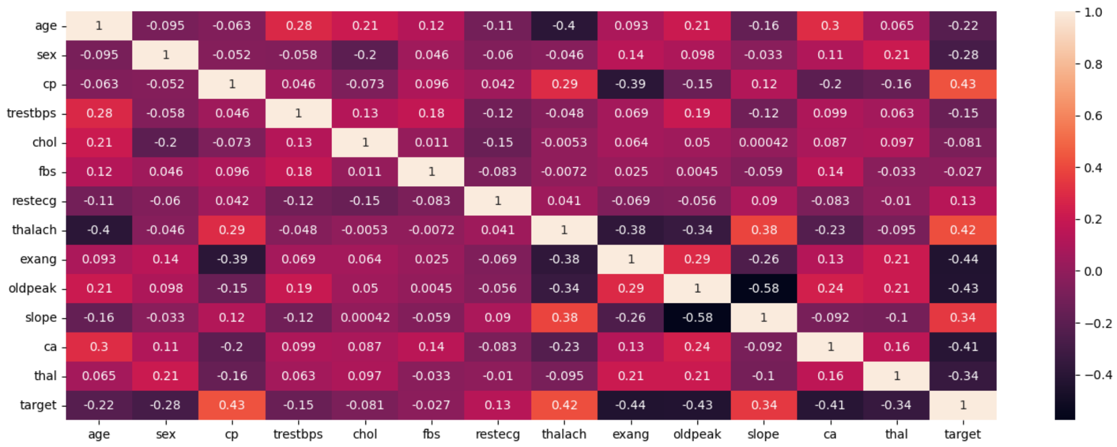
```
              age         sex          cp    trestbps        chol         fbs  \
count  302.00000  302.000000  302.000000  302.000000  302.000000  302.000000
mean    54.42053    0.682119    0.963576  131.602649  246.500000    0.149007
std      9.04797    0.466426    1.032044   17.563394   51.753489    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
```

[19]:

```
#Drawing correlation Matrix
plt.figure(figsize=(17,6))
sns.heatmap(df.corr(),annot = True)
```

[19]:

```
<Axes: >
```



	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
age	1	-0.095	-0.063	0.28	0.21	0.12	-0.11	-0.4	0.093	0.21	-0.16	0.3	0.065	-0.22
sex	-0.095	1	-0.052	-0.058	-0.2	0.046	-0.06	-0.046	0.14	0.098	-0.033	0.11	0.21	-0.28
cp	-0.063	-0.052	1	0.046	-0.073	0.096	0.042	0.29	-0.39	-0.15	0.12	-0.2	-0.16	0.43
trestbps	0.28	-0.058	0.046	1	0.13	0.18	-0.12	-0.048	0.069	0.19	-0.12	0.099	0.063	-0.15
chol	0.21	-0.2	-0.073	0.13	1	0.011	-0.15	-0.0053	0.064	0.05	0.00042	0.087	0.097	-0.081
fbs	0.12	0.046	0.096	0.18	0.011	1	-0.083	-0.0072	0.025	0.0045	-0.059	0.14	-0.033	-0.027
restecg	-0.11	-0.06	0.042	-0.12	-0.15	-0.083	1	0.041	-0.069	-0.056	0.09	-0.083	-0.01	0.13
thalach	-0.4	-0.046	0.29	-0.048	-0.0053	-0.0072	0.041	1	-0.38	-0.34	0.38	-0.23	-0.095	0.42
exang	0.093	0.14	-0.39	0.069	0.064	0.025	-0.069	-0.38	1	0.29	-0.26	0.13	0.21	-0.44
oldpeak	0.21	0.098	-0.15	0.19	0.05	0.0045	-0.056	-0.34	0.29	1	-0.58	0.24	0.21	-0.43
slope	-0.16	-0.033	0.12	-0.12	0.00042	-0.059	0.09	0.38	-0.26	-0.58	1	-0.092	-0.1	0.34
ca	0.3	0.11	-0.2	0.099	0.087	0.14	-0.083	-0.23	0.13	0.24	-0.092	1	0.16	-0.41
thal	0.065	0.21	-0.16	0.063	0.097	-0.033	-0.01	-0.095	0.21	0.21	-0.1	0.16	1	-0.34
target	-0.22	-0.28	0.43	-0.15	-0.081	-0.027	0.13	0.42	-0.44	-0.43	0.34	-0.41	-0.34	1

Question 1: How many people have Heart disease and How many don't have heart disease in this Dataset

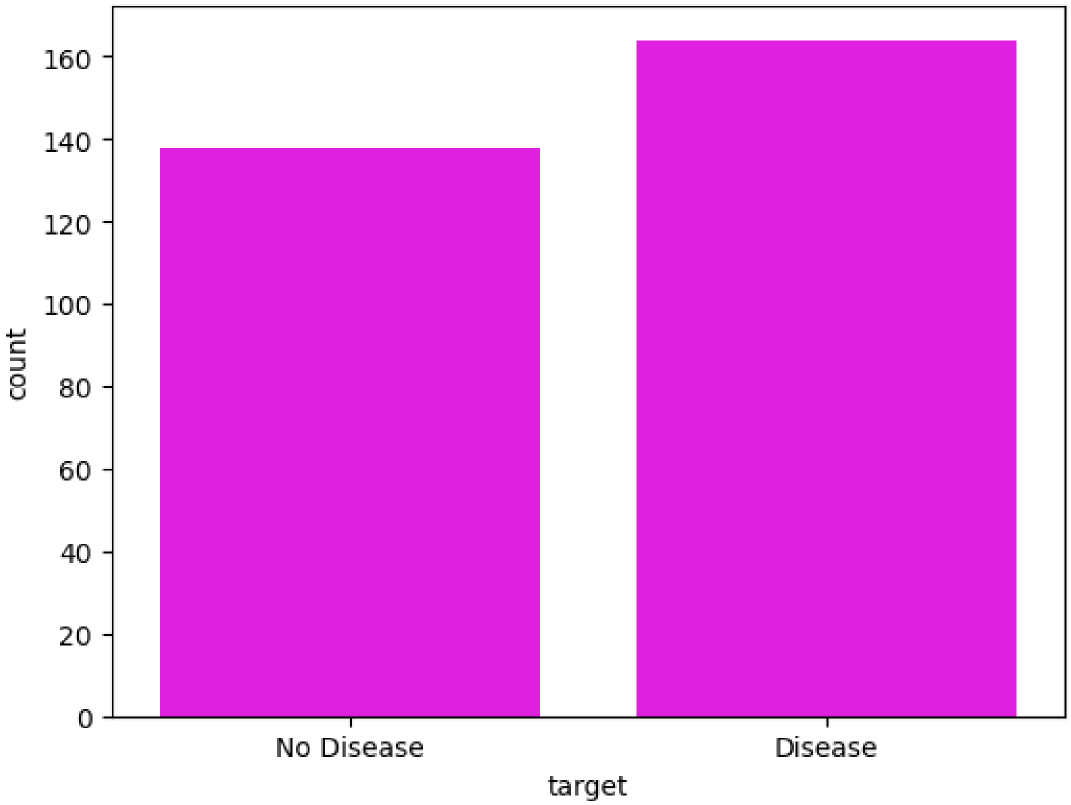
[21]:
```
df.columns
```

[21]:
```
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
      dtype='object')
```

[23]:
```
df["target"].value_counts()
```

[23]:
```
target
1    164
0    138
Name: count, dtype: int64
```

[108]:
```
sns.countplot(x='target', data=df, color='magenta')
plt.xticks([1,0],['Disease','No Disease'])
plt.show()
```



164 people have heart disease and 138 people have no heart disease

Question 2: Find count of Male and Female in the Dataset

```
[27]: df.columns
```

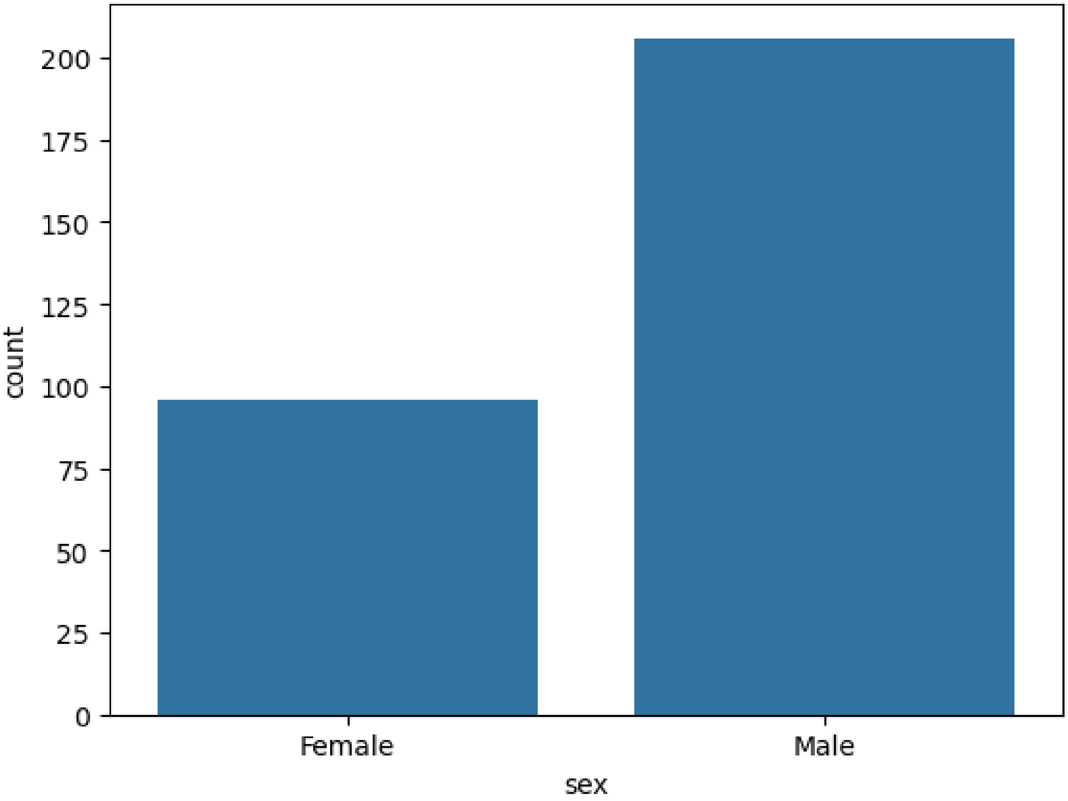
```
[27]: Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
              'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
             dtype='object')
```

```
[29]: df['sex'].value_counts()
```

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

206 people are male and 96 people are Female

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

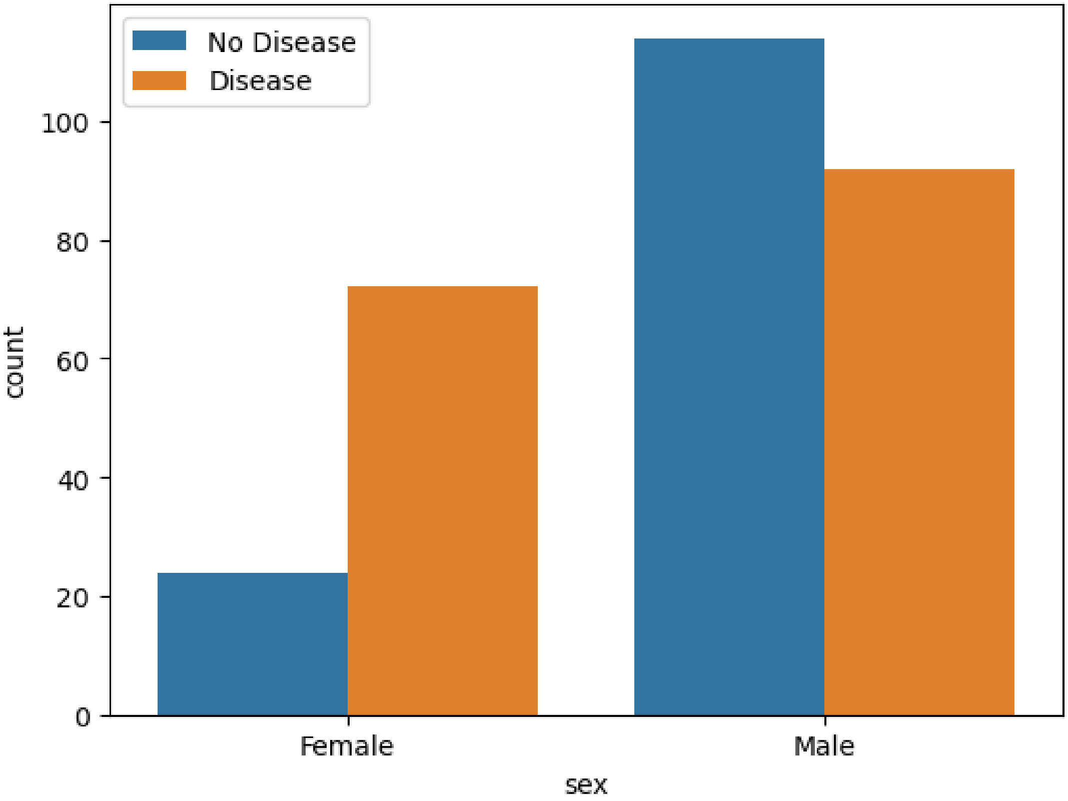


Question 3: Find gender distribution According to the target variable

```
[32]: df.columns
```

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

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



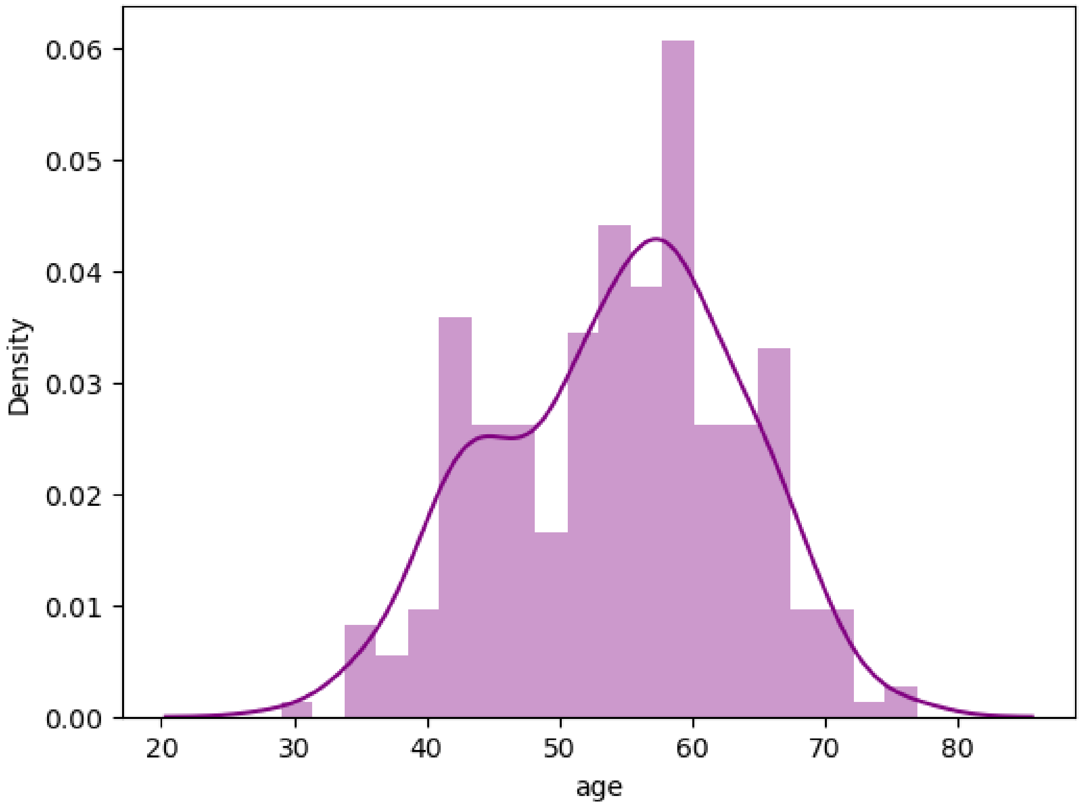
Question 4: Check Age Distribution in the Dataset

```
[44]: df.columns
```

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

```
[48]: sns.distplot(df.age, bins=20,color='purple')
      plt.show()
```

```
C:\Users\Josphat\AppData\Local\Temp\ipykernel_15108\415452091.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df.age, bins=20,color='purple')
```



Question 5: Check Chest pain type

```
* Value 0: typical angina
* Value 1: atypical angina
* Value 2: non-anginal pain
* Value 3: asymptomatic
```

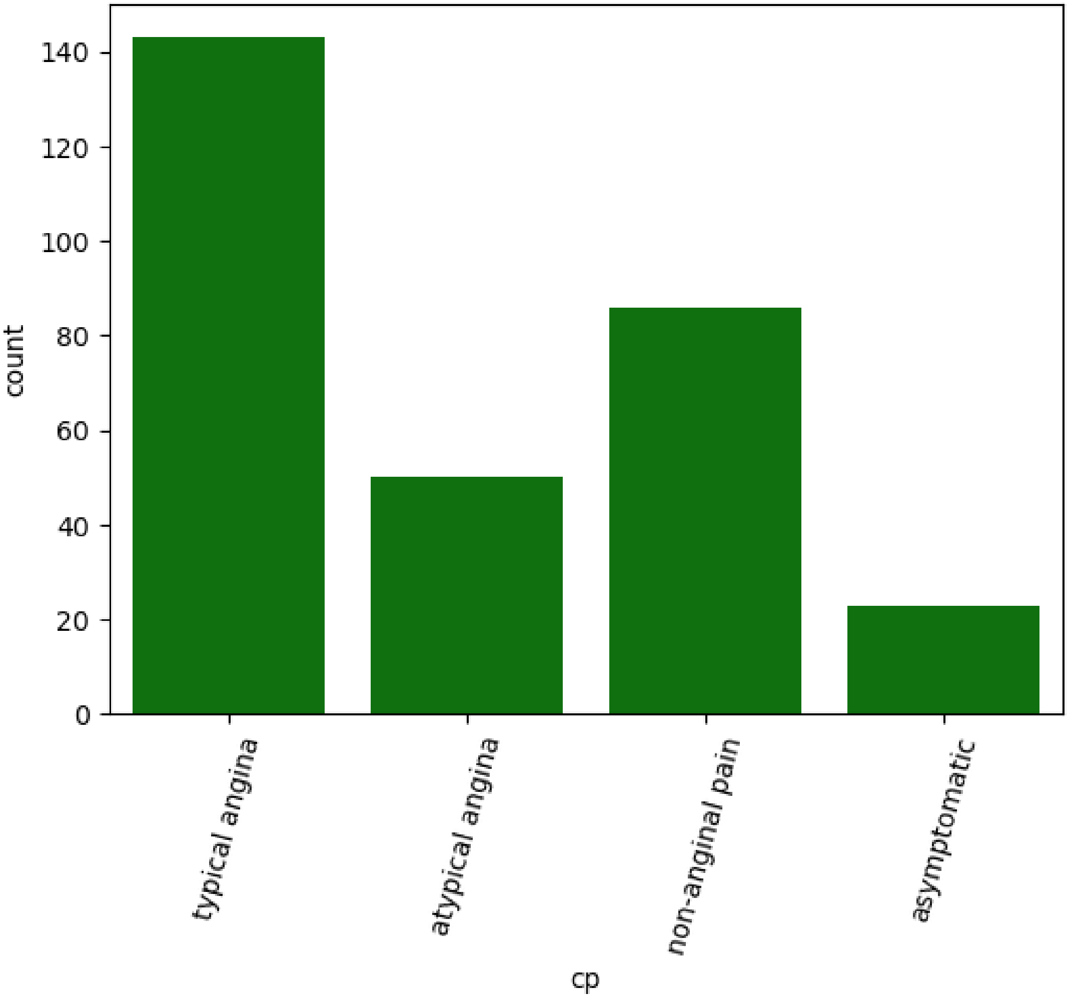
```
[49]: df.columns
```

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

```
[54]: df.cp.value_counts()
```

```
[54]: cp
      0    143
      2     86
      1     50
      3     23
      Name: count, dtype: int64
```

[56]:
```
sns.countplot(x='cp', data=df, color='green')
plt.xticks([0,1,2,3],['typical angina','atypical angina','non-anginal␣
 ↪pain','asymptomatic'])
plt.xticks(rotation=75)
plt.show()
```



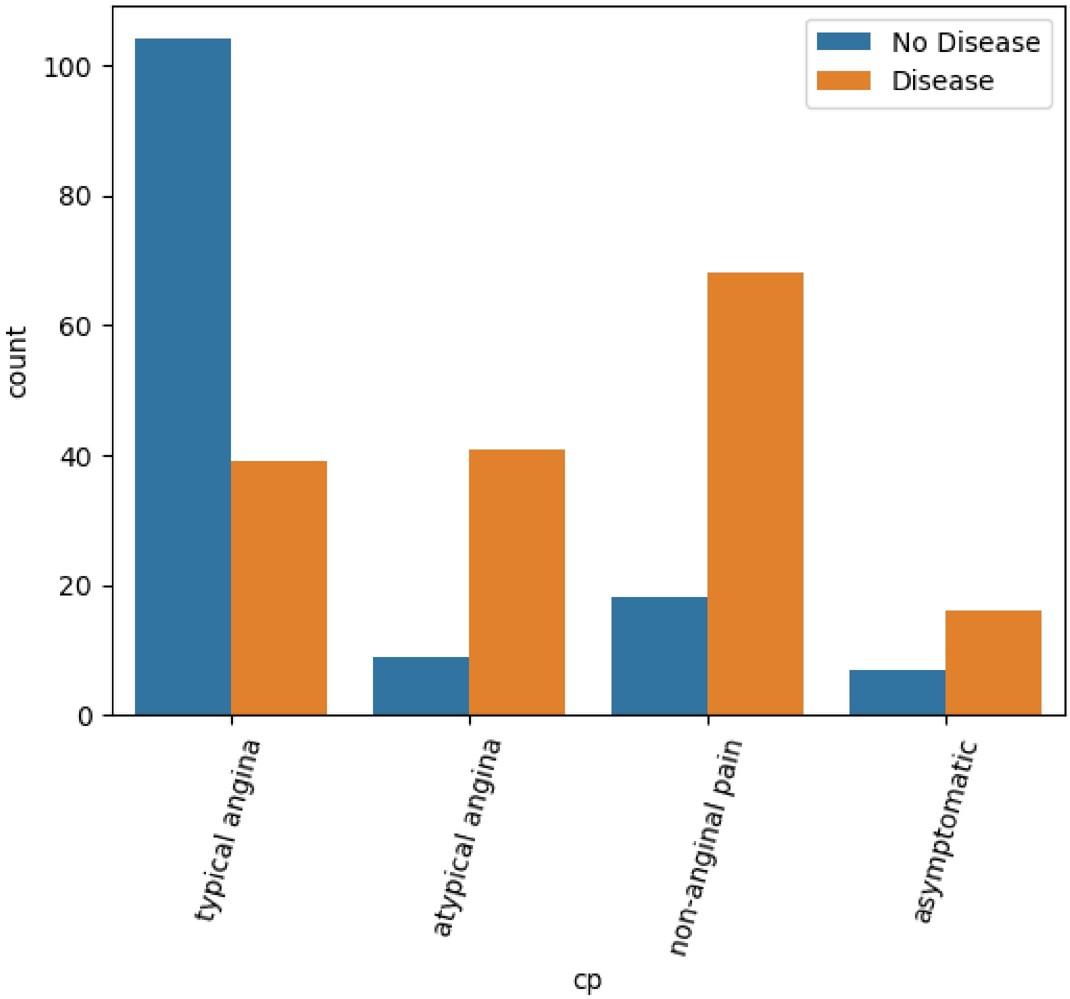
Question 6: Show chest pain Distribution as per Target variabe

[58]:
```
df.columns
```

[58]:
```
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
      dtype='object')
```

[61]:
```
sns.countplot(x='cp', hue='target',data=df)
```

```
plt.xticks([0,1,2,3],['typical angina','atypical angina','non-anginal␣
  ↪pain','asymptomatic'],rotation=75)
plt.legend(labels=['No Disease','Disease'])
plt.show()
```



[68]:
```
df[['cp','target']].value_counts().sort_values(ascending=False)
```

[68]:
```
cp  target
0   0         104
2   1          68
1   1          41
0   1          39
2   0          18
3   1          16
1   0           9
```

```
3    0             7
Name: count, dtype: int64
```

Question 7: Show Fasting Blood Sugar Distribution according to Target variable.

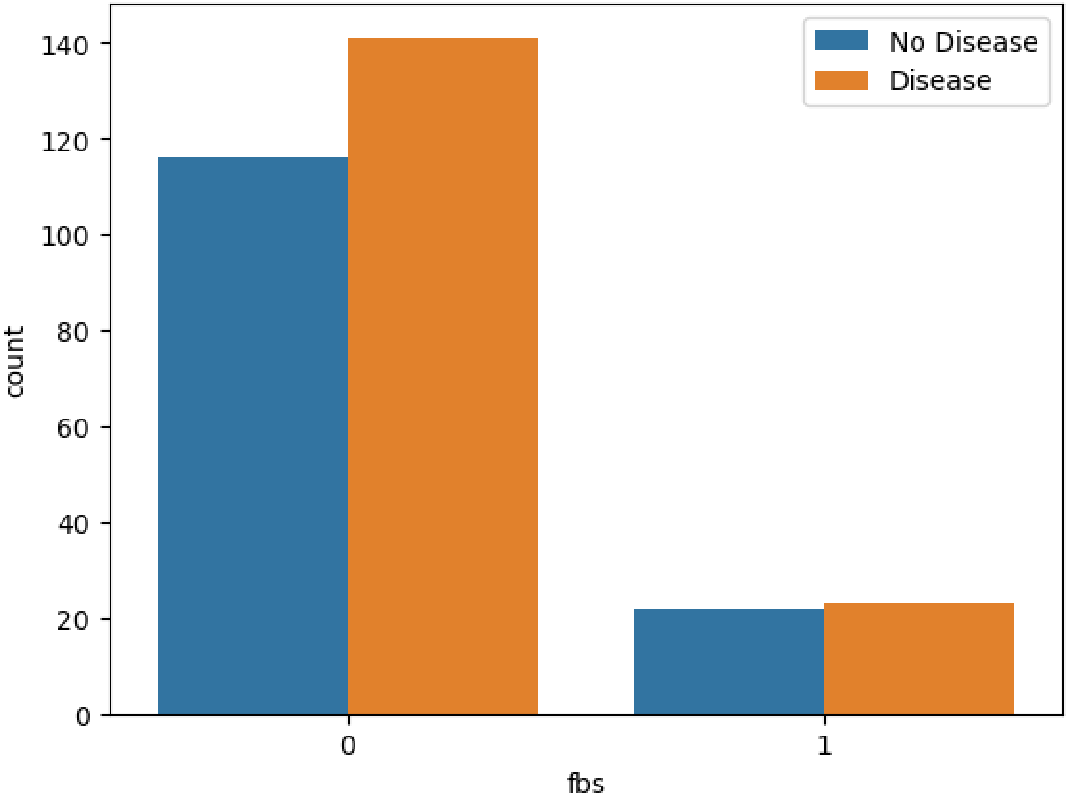
```
[69]: df.columns
```

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

```
[70]: df[['fbs','target']].value_counts().sort_values(ascending=False)
```

```
[70]: fbs  target
      0    1         141
           0         116
      1    1          23
           0          22
      Name: count, dtype: int64
```

```
[73]: sns.countplot(x='fbs', hue='target',data=df)
      plt.legend(labels=['No Disease','Disease'])
      plt.show()
```

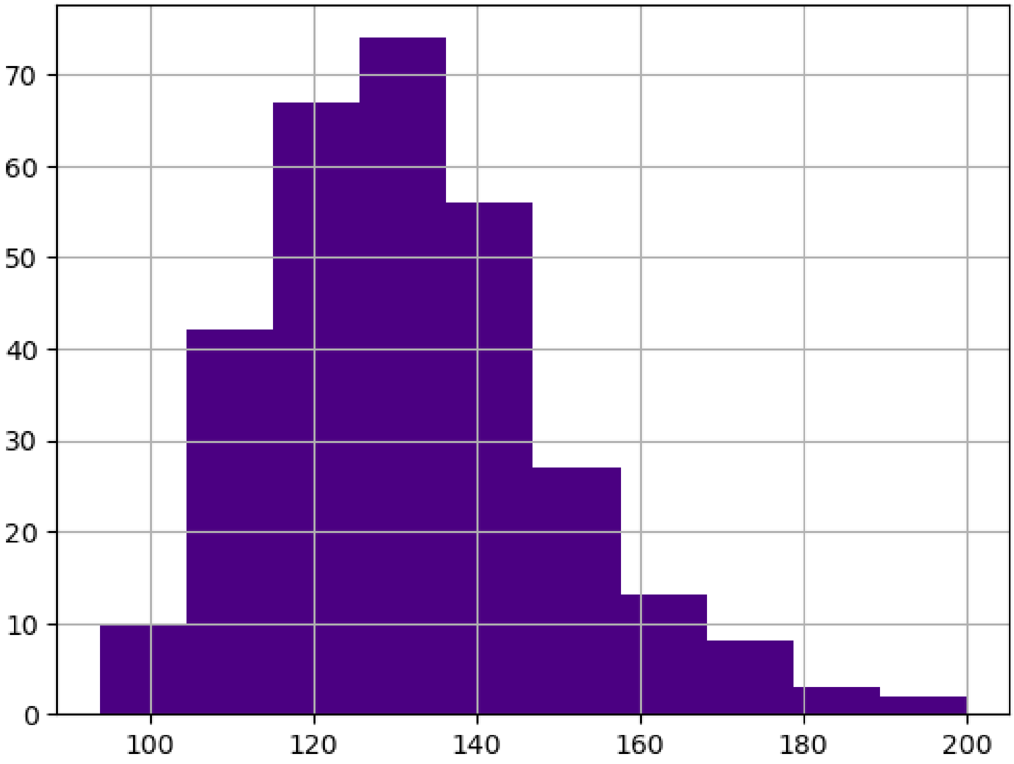


Question 8: Check Resting Blood Pressure Distribution

```
[74]: df.columns
```

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

```
[97]: df.trestbps.hist(color='indigo')
      plt.show()
```

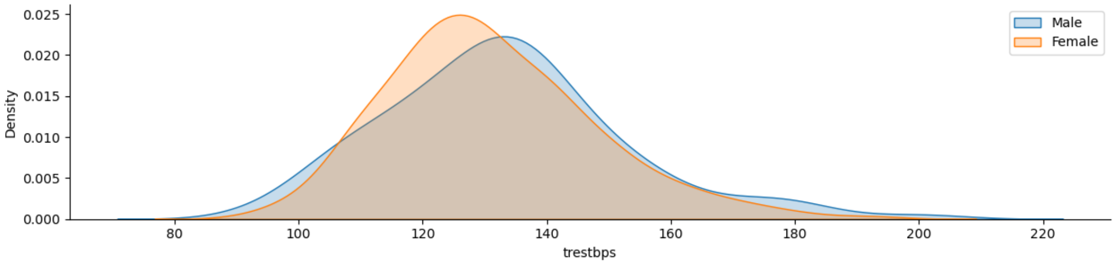


Question 9: compare Resting Blood Pressure as per sex column

```
[102]: g=sns.FacetGrid(df, hue='sex',aspect=4)
       g.map(sns.kdeplot,'trestbps',shade=True)
       plt.legend(labels=['Male','Female'])
       plt.show()
```

```
C:\Users\Josphat\anaconda3\new anaconda\Lib\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\Josphat\anaconda3\new anaconda\Lib\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)
```

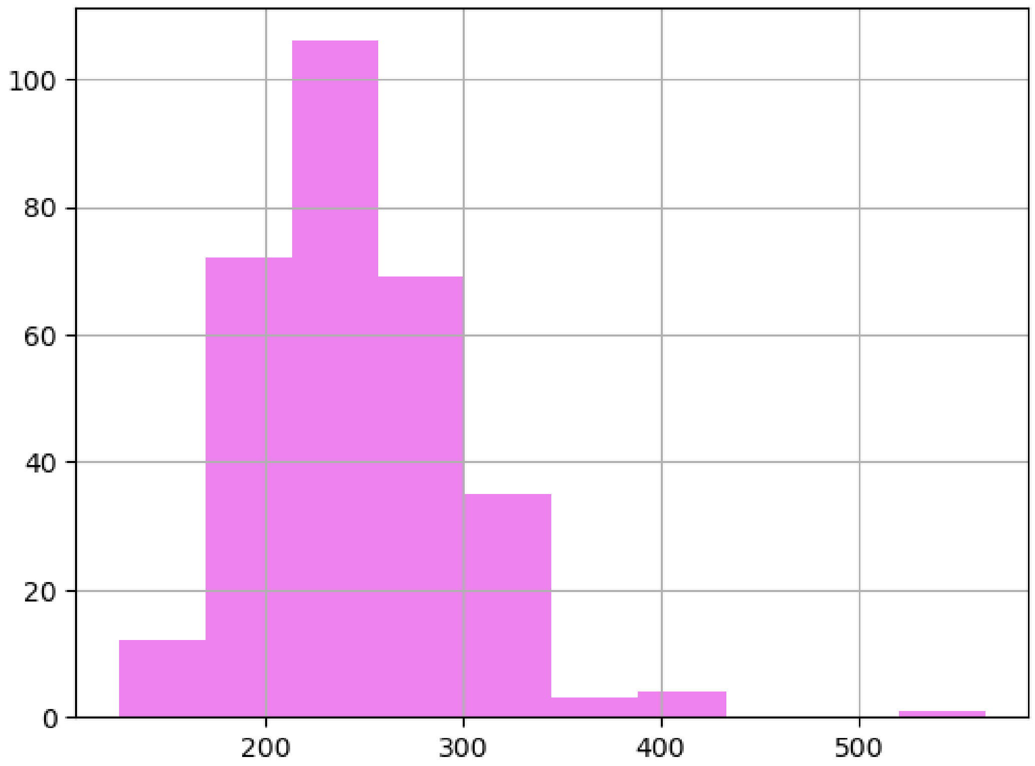


Question 10: Show distribution of Serum Cholestrol

[84]:
```
df.columns
```

[84]:
```
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
      dtype='object')
```

[104]:
```
df.chol.hist(color='violet')
plt.show()
```



Question 11: Plot Continuous Variables

[90]:

```
df.columns
```

[90]:

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

[91]:

```
cate_val= []
cont_val= []

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

[92]:

```
cate_val
```

[92]:

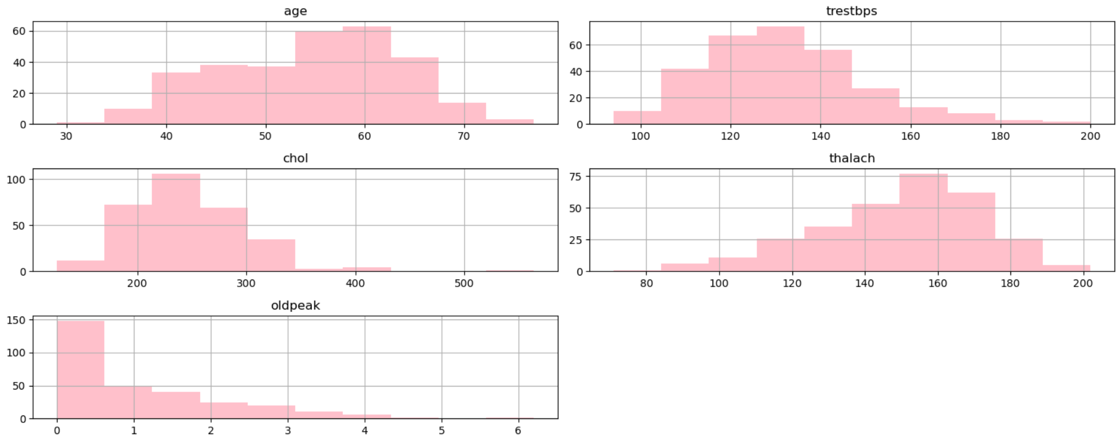
```
['sex', 'cp', 'fbs', 'restecg', 'exang', 'slope', 'ca', 'thal', 'target']
```

[93]:

```
cont_val
```

```
[93]: ['age', 'trestbps', 'chol', 'thalach', 'oldpeak']
```

```
[106]: df.hist(cont_val,figsize=(15,6),color='pink')
       plt.tight_layout()
       plt.show()
```



```
[ ]:
```