task-5

August 10, 2024

[2]:
```
#import all the libraries that we need.
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

[3]:
```
#importing our dataset.
from google.colab import drive
drive.mount('/content/drive')

data_path = '/content/drive/My Drive/heart.csv'  # Replace with your file path
df = pd.read_csv(data_path)
```

```
Mounted at /content/drive
```

[4]:
```
#Checking first five rows by calling df.head()
df.head()
```

[4]:

	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

[5]:
```
df.tail()
```

[5]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak
1020	59	1	1	140	221	0	1	164	1	0.0
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

      slope  ca  thal  target
1020      2   0     2       1
1021      1   1     3       0
1022      1   1     2       0
1023      2   0     2       1
1024      1   1     3       0
```

[6]:
```
#Take a look at the column names.
df.columns.values
```

[6]:
```
array(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg',
       'thalach', 'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
      dtype=object)
```

[7]:
```
#Checking for null values
df.isna().sum()
```

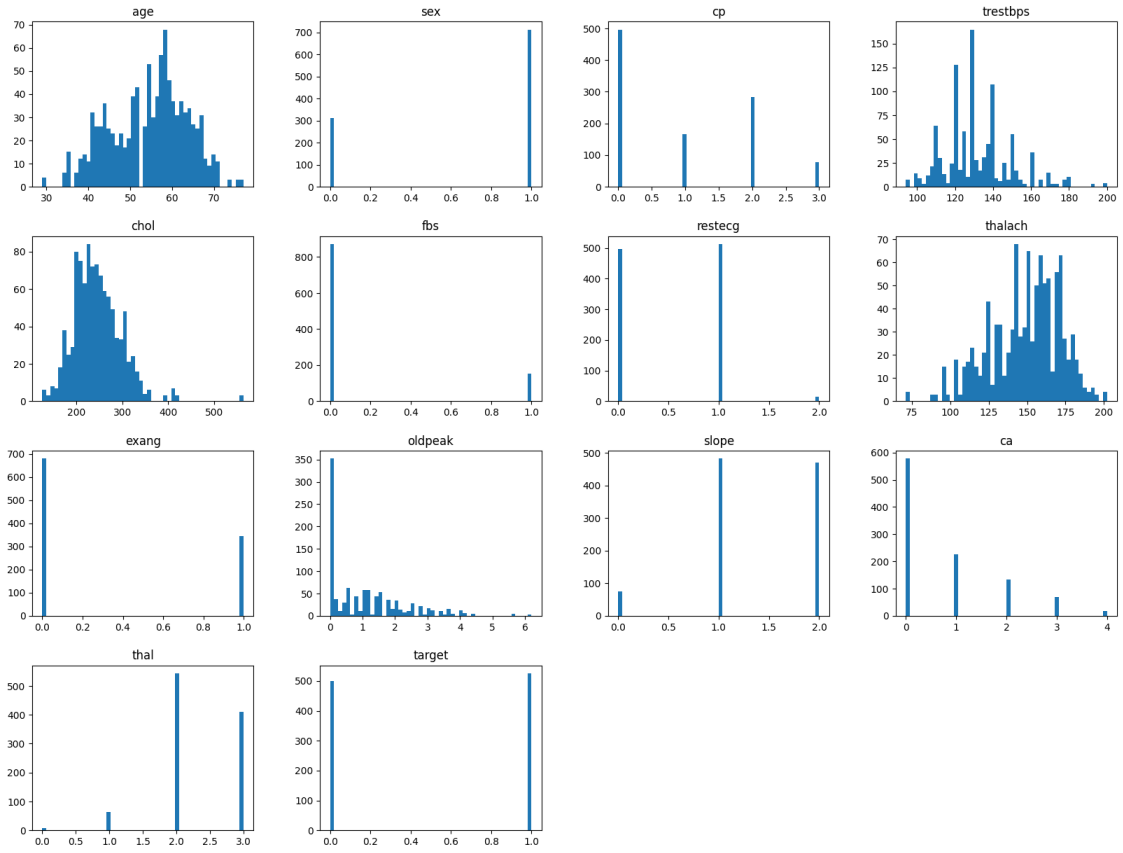
[7]:
```
age         0
sex         0
cp          0
trestbps    0
chol        0
fbs         0
restecg     0
thalach     0
exang       0
oldpeak     0
slope       0
ca          0
thal        0
target      0
dtype: int64
```

[8]:
```
#Concise summary of our 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
```

[9]:
```
#plotting histogram of all numeric values
df.hist(bins = 50, grid = False ,figsize=(20,15) );
```



[10]:
```
#Generating descriptive statistics.
df.describe()
```

[10]:

	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

[64]:
```
questions = ["1. How many people have heart disease and how many people doest␣
 ↪have heart disease?",
             "2. People of which sex has most heart disease?",
             "3. People of which sex has which type of chest pain most?",
             "4. People with which chest pain are most pron to have heart␣
 ↪disease?",
             "5.  Does fasting blood sugar level have an impact on the␣
 ↪likelihood of heart disease? ",
             "6. How does age correlate with heart disease?",
             "7. What is the average resting blood pressure of people with␣
 ↪heart disease?",
             "8. Is there a relationship between cholesterol levels and the␣
 ↪presence of heart disease?",
]

questions
```

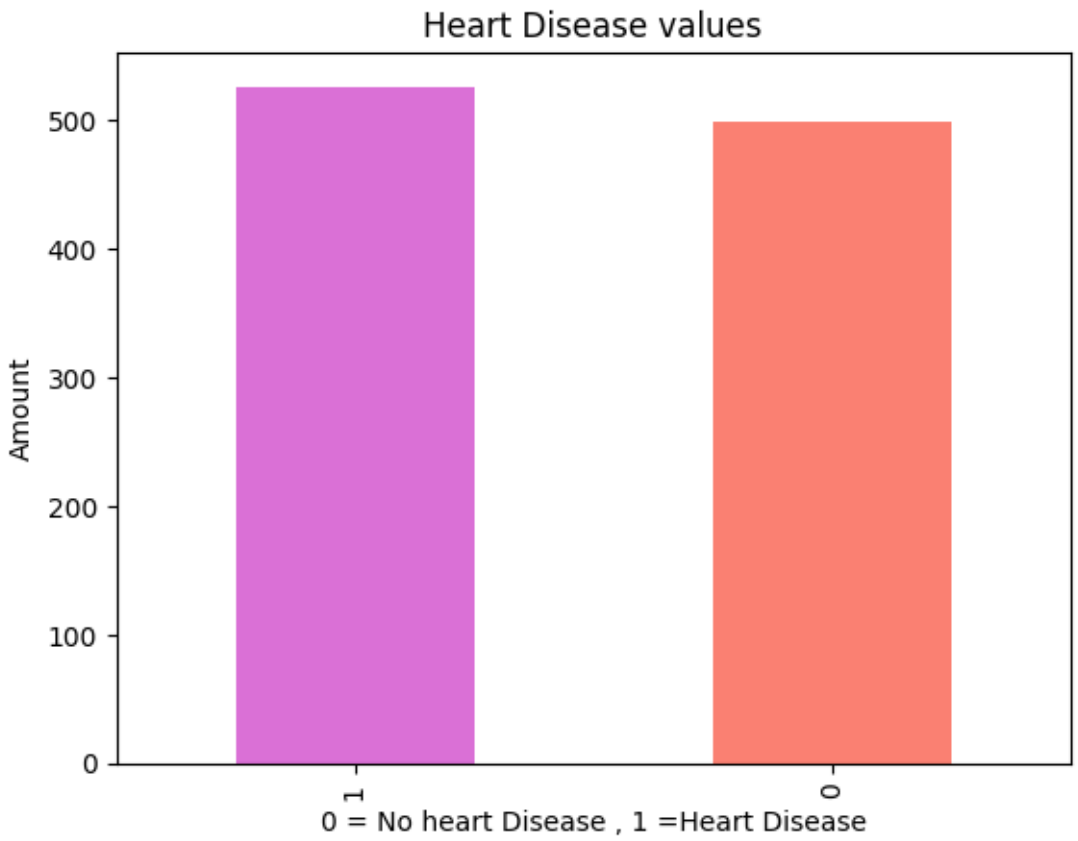
[64]:
```
['1. How many people have heart disease and how many people doest have heart
disease?',
 '2. People of which sex has most heart disease?',
 '3. People of which sex has which type of chest pain most?',
 '4. People with which chest pain are most pron to have heart disease?',
 '5.  Does fasting blood sugar level have an impact on the likelihood of heart
disease? ',
 '6. How does age correlate with heart disease?',
 '7. What is the average resting blood pressure of people with heart disease?',
 '8. Is there a relationship between cholesterol levels and the presence of
heart disease?']
```

[12]:
```
#Let's find the answer of first question.

#1.How many people have heart disease and how many people doesn't have heart␣
 ↪disease?

#getting the values
df.target.value_counts()
```

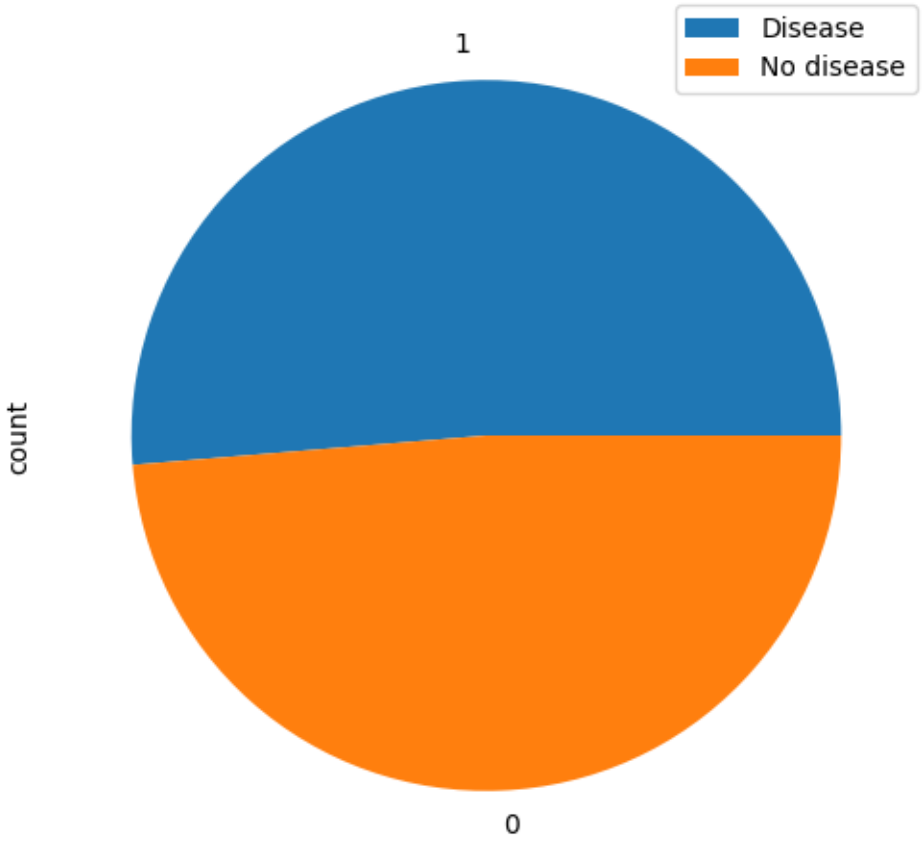
[12]:
```
target
1    526
0    499
Name: count, dtype: int64
```

[13]:
```
#Plotting bar chart
df.target.value_counts().plot(kind = 'bar' ,color=["orchid","salmon"])
plt.title("Heart Disease values")
plt.xlabel("0 = No heart Disease , 1 =Heart Disease")
plt.ylabel("Amount")
```

[13]:
```
Text(0, 0.5, 'Amount')
```



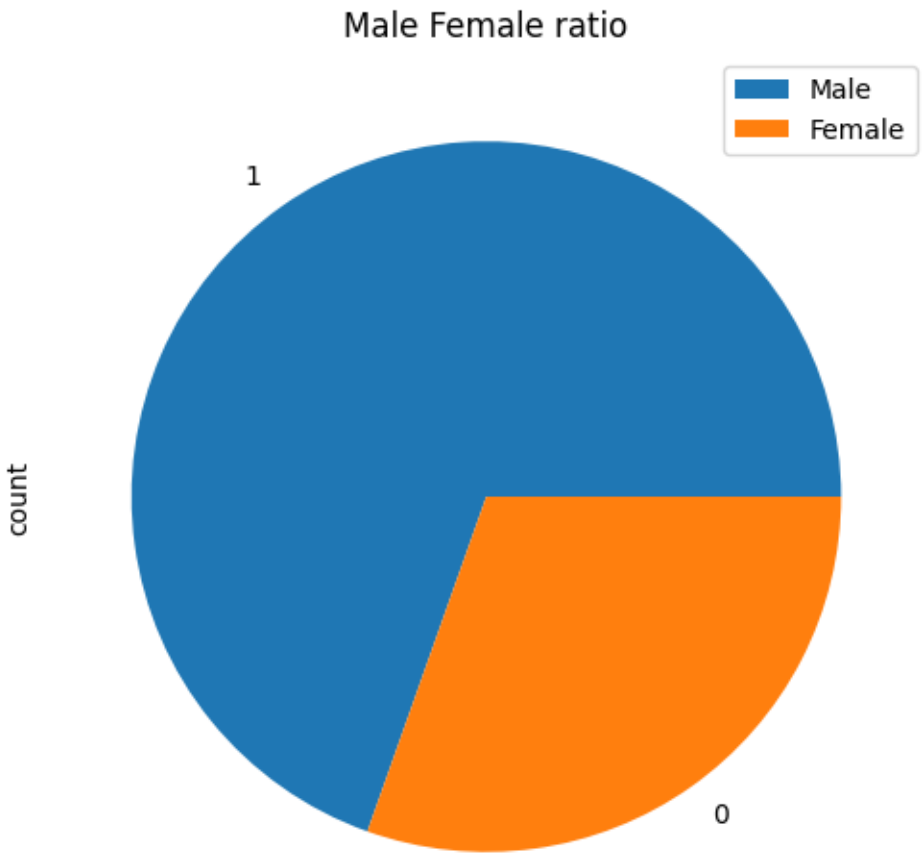
[14]:
```
#plotting a pie chart
df.target.value_counts().plot(kind ='pie' , figsize =(8,6))
plt.legend(["Disease","No disease"]);
```



[15]:
```
#'0' represent 'Female'

#'1' represent 'Male'

#SEX column part

#'0' represent 'No disease'

#Target column part

#Now let's check how many 'Male' and 'Female' are in the dataset

df.sex.value_counts()
```

[15]:
```
sex
1    713
0    312
Name: count, dtype: int64
```

[16]:
```
#plotting a pie chart
df.sex.value_counts().plot(kind ='pie' , figsize =(8,6))
plt.title("Male Female ratio")
plt.legend(["Male","Female"]);
```

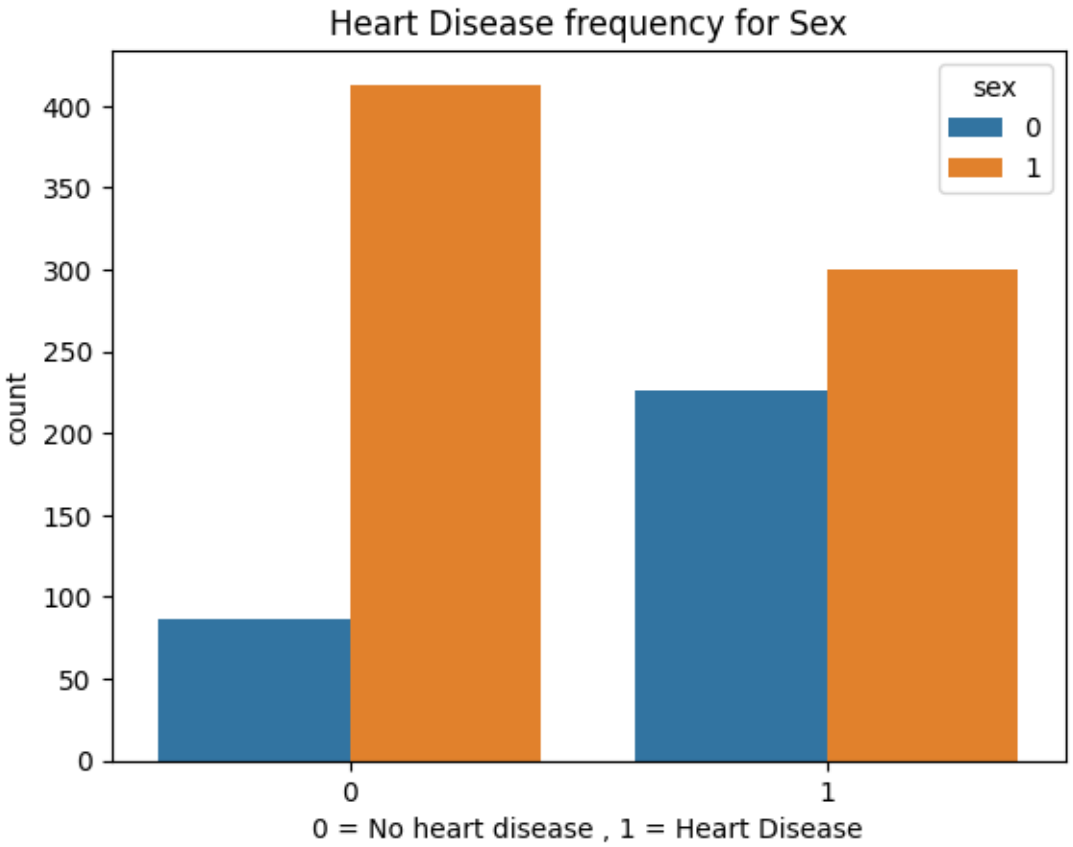


[17]:
```
#Let's find the answer of our 2nd question question.

#2.People of which sex has most heart disease?

pd.crosstab(df.target,df.sex)
```

[17]:

sex	0	1
target		
0	86	413
1	226	300

[18]:
```
sns.countplot(x='target',hue='sex',data=df);
plt.title("Heart Disease frequency for Sex")
plt.xlabel("0 = No heart disease , 1 = Heart Disease");
```



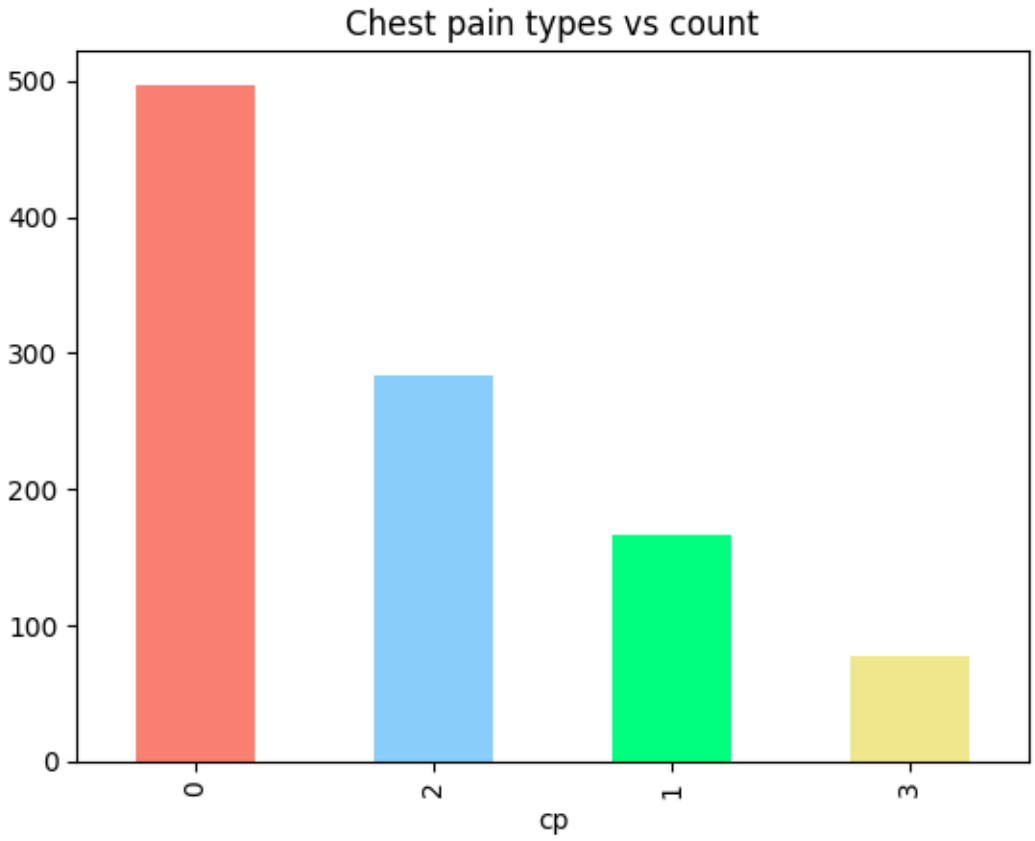
[19]:
```
#Number of male is more than double in our dataset than female.

#More than '455 male' has heart disease and '75% female' has heart disease.
```

[20]:
```
#Let's move to question 3

#3. 'People of which sex has which type of chest pain most?'

#counting values for different chest pain
df.cp.value_counts()
```

[20]:
```
cp
0    497
2    284
1    167
3     77
```

```
Name: count, dtype: int64
```

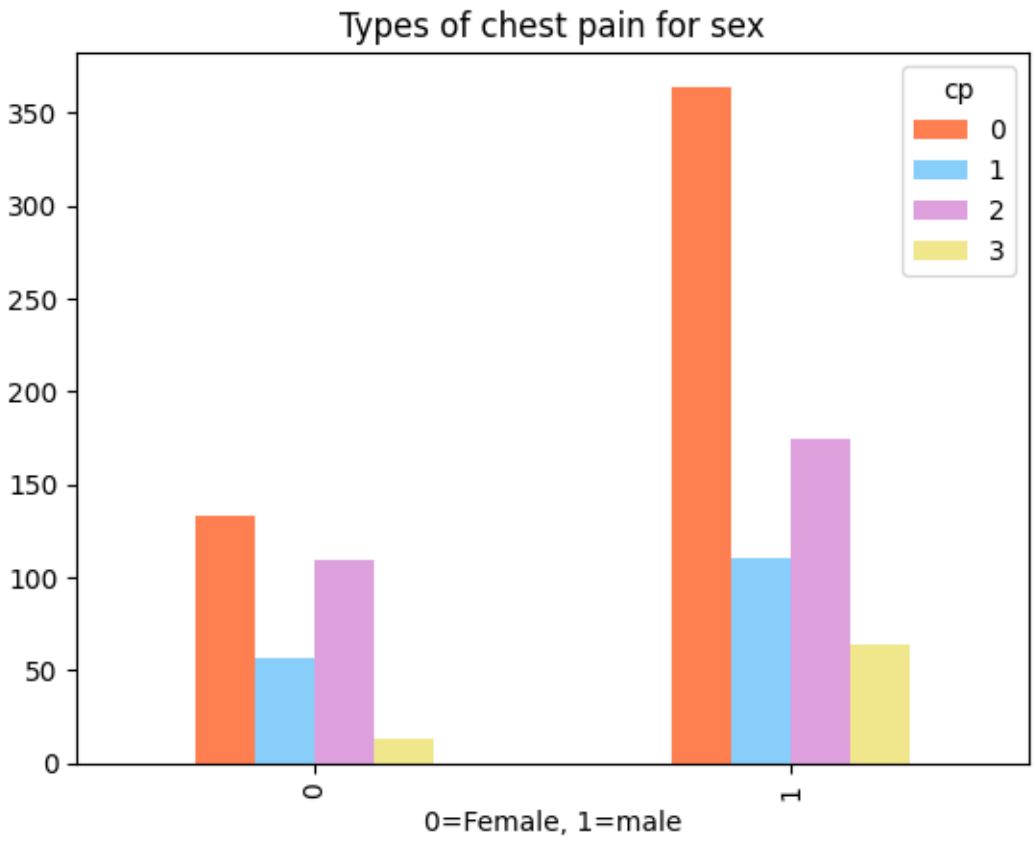
[21]:
```
#Plotting a bar chart
df.cp.value_counts().plot(kind␣
  ↪='bar',color=['salmon','lightskyblue','springgreen','khaki'])
plt.title("Chest pain types vs count");
```



[22]:
```
pd.crosstab(df.sex,df.cp)
```

[22]:
```
cp     0    1    2   3
sex
0    133   57  109  13
1    364  110  175  64
```

[23]:
```
pd.crosstab(df.sex,df.cp).
  ↪plot(kind='bar',color=['coral','lightskyblue','plum','khaki'])
plt.title("Types of chest pain for sex");
plt.xlabel('0=Female, 1=male');
```



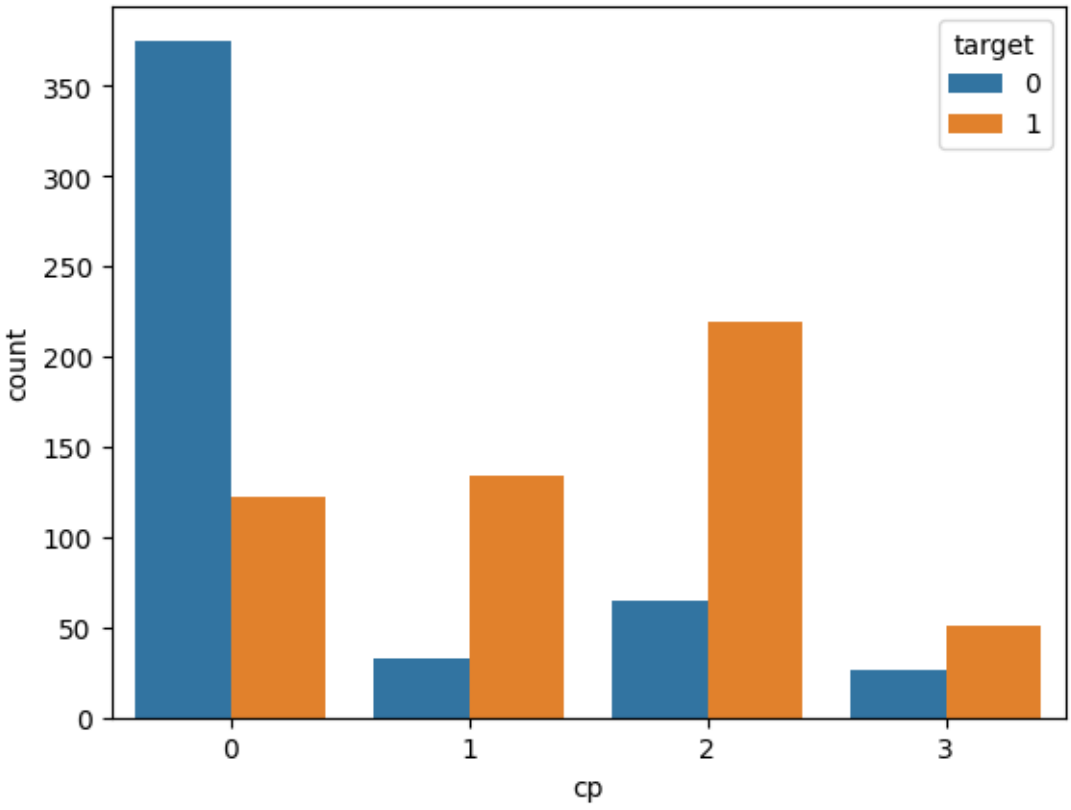
[24]:
```
#Most of the 'male' has 'type 0' chest painand Least of 'male' has 'type 4'␣
↪pain.

#in case of 'Female' 'type 0' and 'type 2' percentage is almost same .
```

[25]:
```
#Now question 4

#4.People with which chest pain are most pron to have heart disease?.

pd.crosstab(df.cp,df.target)
```

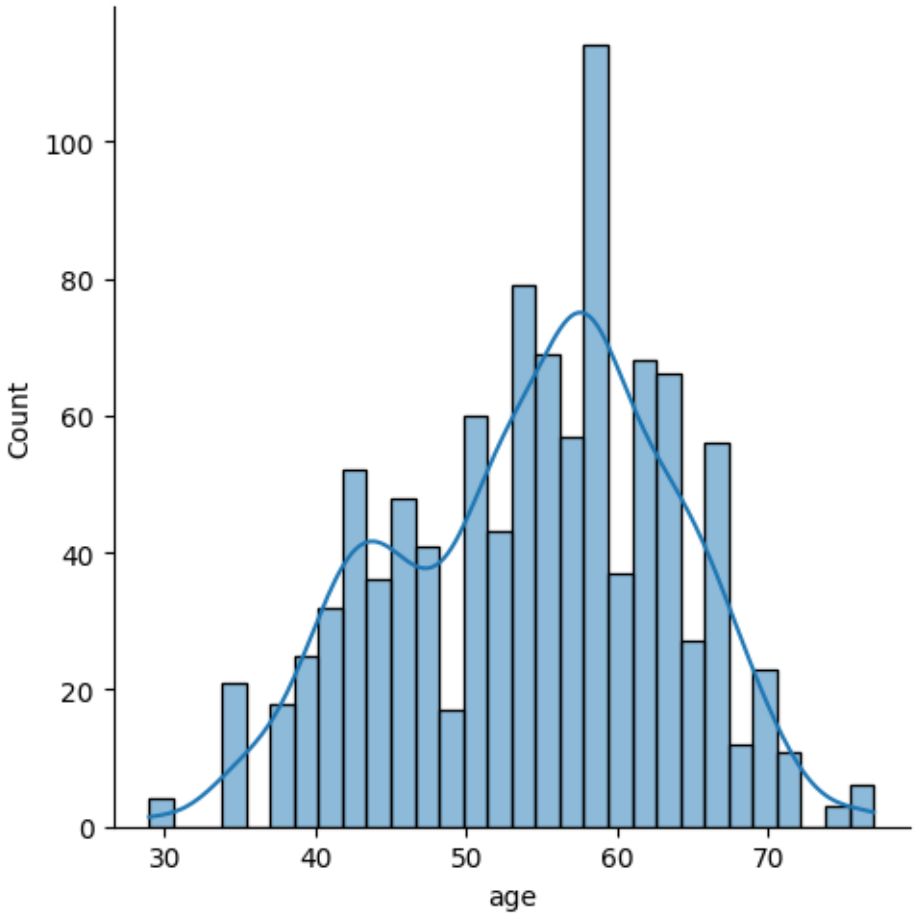
[25]:

target	0	1
cp		
0	375	122
1	33	134
2	65	219
3	26	51

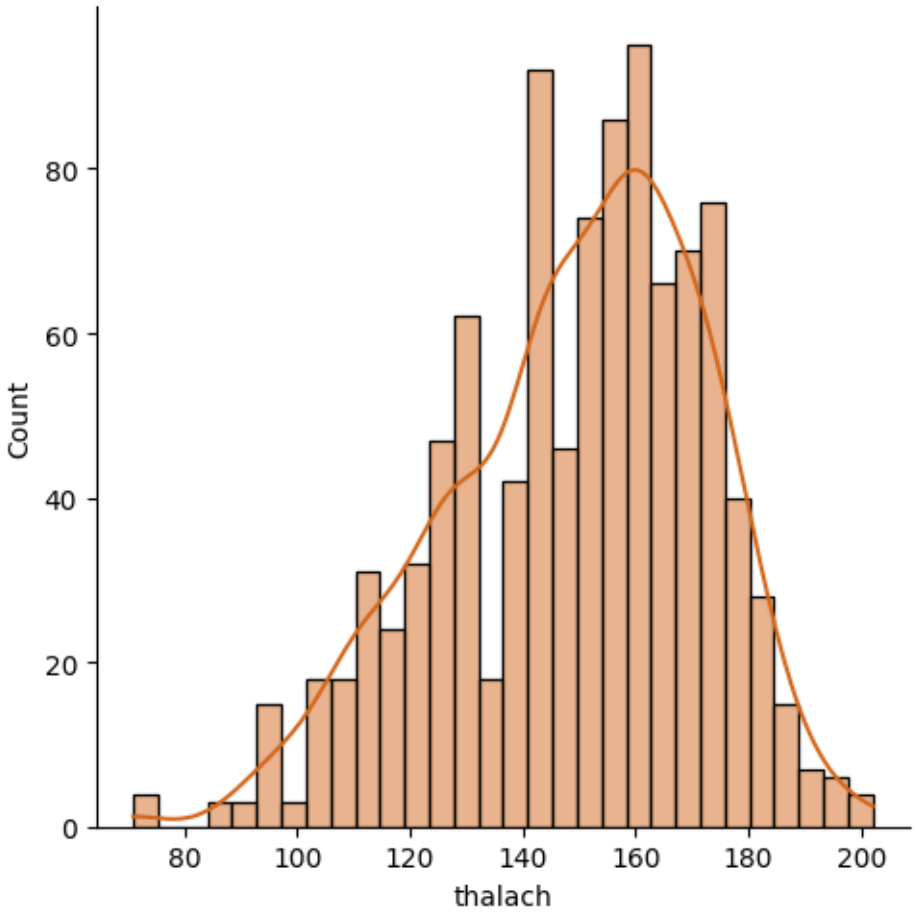
[26]:
```
sns.countplot(x = 'cp' , data = df,hue ='target');
```



[27]:
```
#Most of the people who has 'type 0' chest pain has the less chance for heart disease.

#And we see the opposite for other types.

#Now let's take a look at our age column.

# Create a distribution plot with normal ditribution curve
sns.displot(x='age',data=df,bins=30,kde=True);
```



[28]:
```
#'58-59' year old people are the most in the dataset.

#Let's plot another distribution plot for 'Maximum heart rate'.
sns.displot(x ='thalach', data = df, bins= 30, kde = True , color ='chocolate');
```

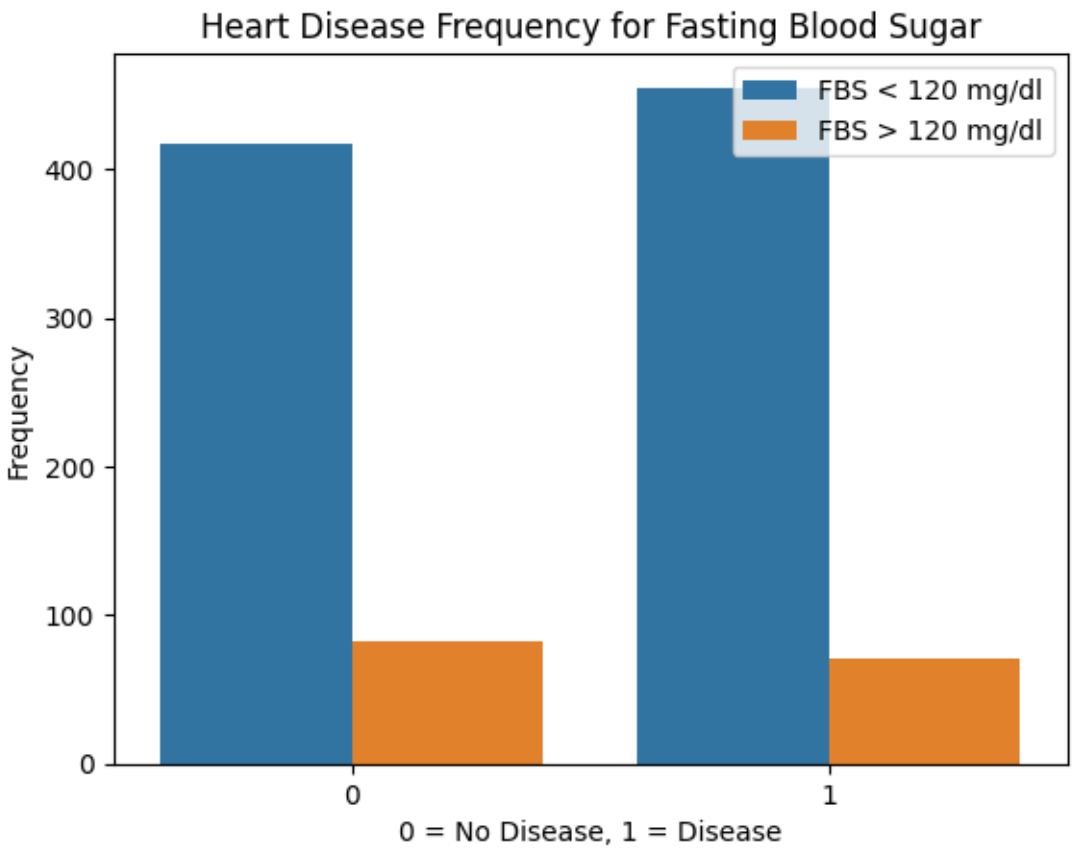


[29]:
```
#Now let's move on to question 5.

#5.Does fasting blood sugar level have an impact on the likelihood of heart␣
 ↪disease?

pd.crosstab(df.target,df.fbs)
```

[29]:
```
fbs       0   1
target
0       417  82
1       455  71
```

[30]:
```
sns.countplot(x='target',hue='fbs',data=df);
plt.title('Heart Disease Frequency for Fasting Blood Sugar')
plt.xlabel('0 = No Disease, 1 = Disease')
plt.ylabel('Frequency')
plt.legend(["FBS < 120 mg/dl", "FBS > 120 mg/dl"])
```

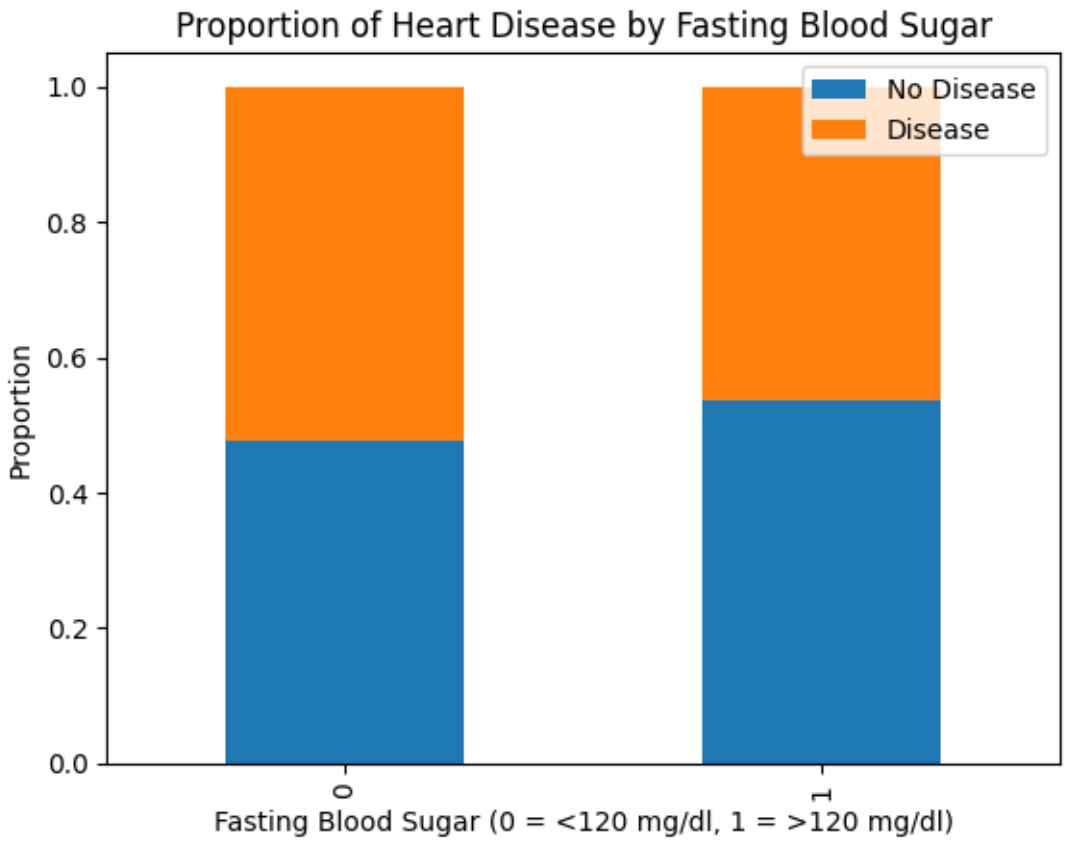
[30]: <matplotlib.legend.Legend at 0x7f194f58e920>



[31]:
```
#From the above it is clear that individuals with fasting blood sugar levels␣
↪above 120 mg/dl have higher frequency of heart disease.
```

[32]:
```
# Calculate proportions
fbs_proportions = pd.crosstab(df.fbs, df.target, normalize='index')

# Plot stacked bar chart
fbs_proportions.plot(kind='bar', stacked=True)
plt.title('Proportion of Heart Disease by Fasting Blood Sugar')
plt.xlabel('Fasting Blood Sugar (0 = <120 mg/dl, 1 = >120 mg/dl)')
plt.ylabel('Proportion')
plt.legend(["No Disease", "Disease"])
```

[32]: <matplotlib.legend.Legend at 0x7f194f61f9d0>



[33]:
```
#it suggests a higher proportion of heart disease among those with fasting␣
↪blood sugar levels above 120 mg/dl
```

[34]:
```
#Let's move on to question no 6.

#6.How does age correlate with heart disease?

pd.crosstab(df.age,df.target).head(10)
```

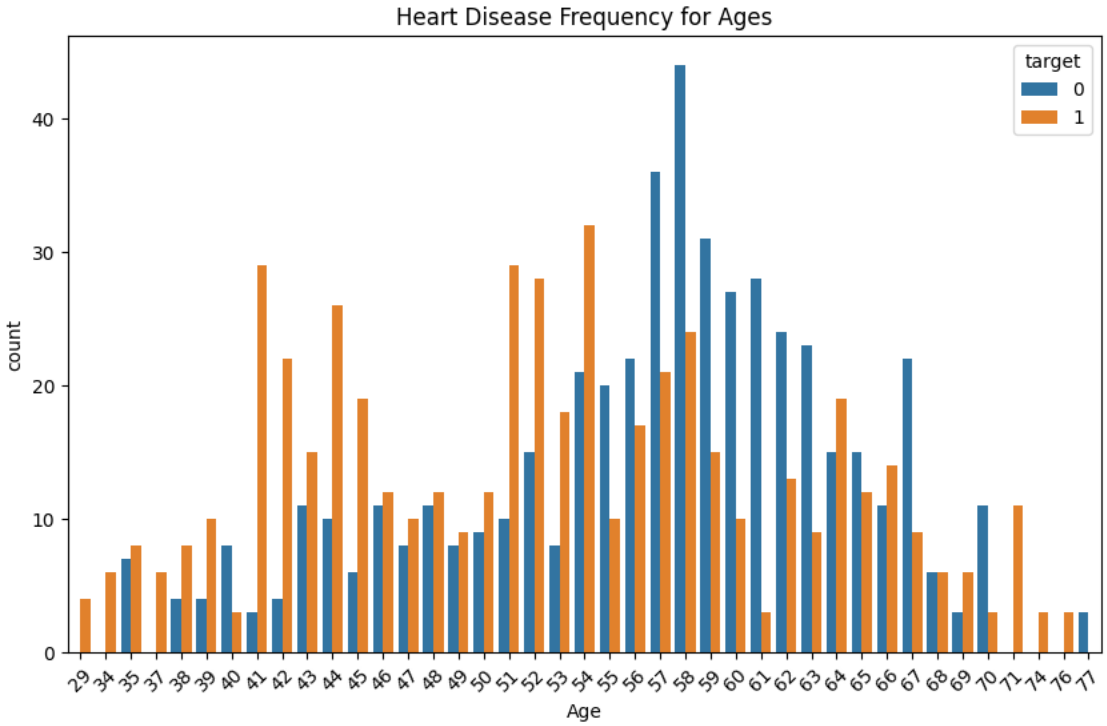
[34]:

target	0	1
age		
29	0	4
34	0	6
35	7	8
37	0	6
38	4	8
39	4	10
40	8	3
41	3	29
42	4	22

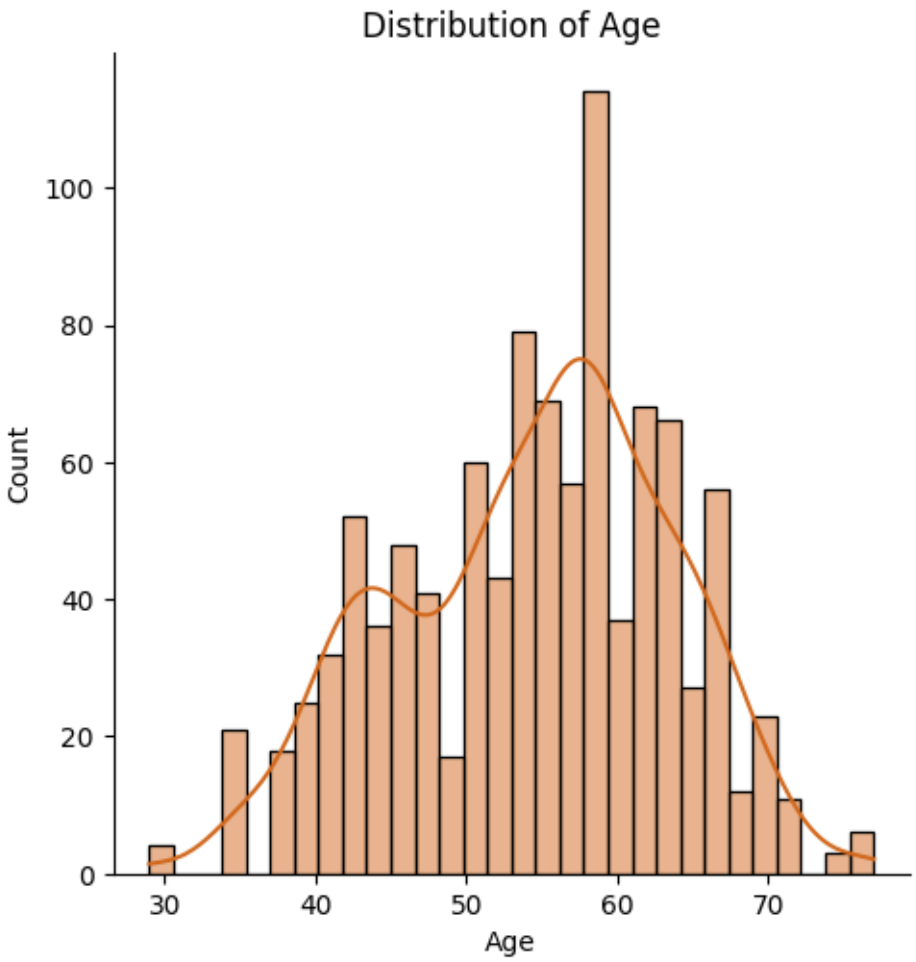
```
43      11  15
```

[35]:
```
import seaborn as sns
import matplotlib.pyplot as plt
```

[36]:
```
plt.figure(figsize=(10, 6))
sns.countplot(x = 'age' , data = df,hue ='target');
plt.title('Heart Disease Frequency for Ages')
plt.xlabel('Age')
plt.xticks(rotation=45)  # Rotate labels by 45 degrees
plt.show()
```



[37]:
```
sns.displot(x='age', data=df, bins=30, kde=True, color='chocolate')
plt.title('Distribution of Age')
plt.xlabel('Age')
plt.ylabel('Count')
plt.show()
```



[38]:
```
#Distribution approximatly normal, with a peak in around 50-60 years of age

#The highest incidence of heart rate is between 50 and 60 years of age.
```

[42]:
```
#Lets see the next question number 7.

#7.What is the average resting blood pressure of people with heart disease?

df.groupby('target').agg({'trestbps': 'mean'})
```

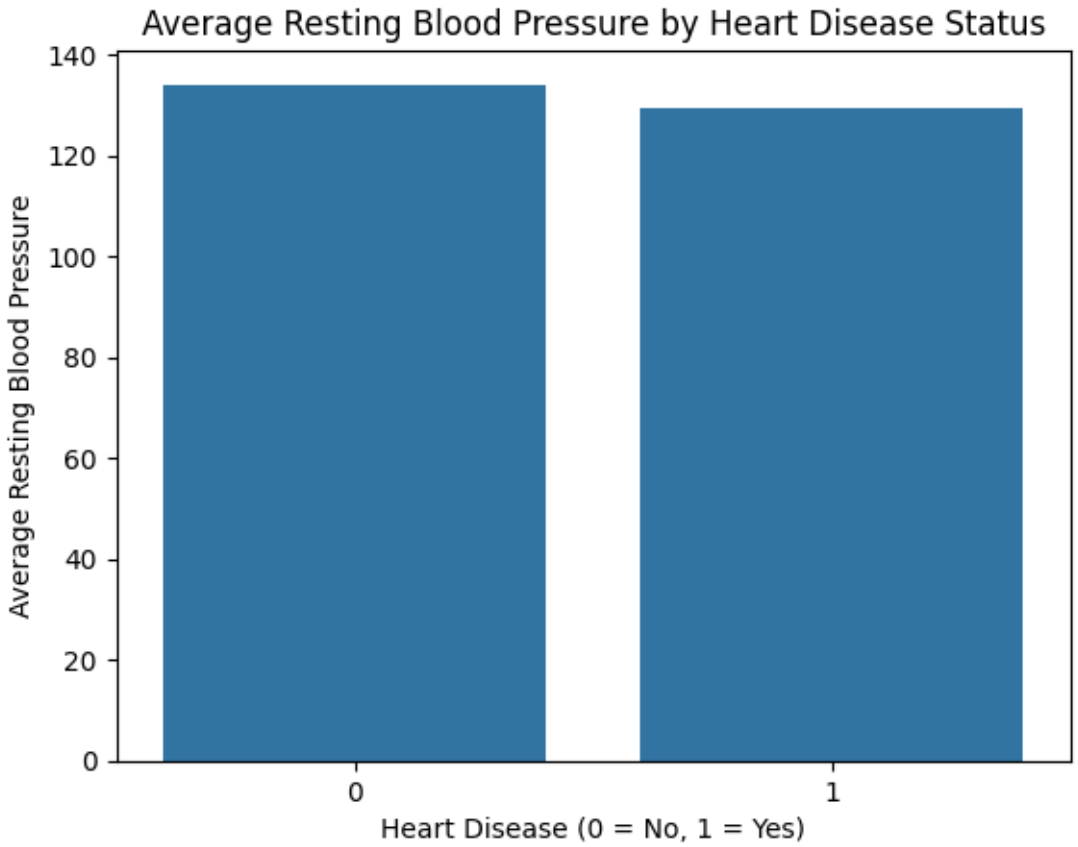
[42]:

target	trestbps
0	134.106212
1	129.245247

[59]:
```
import seaborn as sns
import matplotlib.pyplot as plt
import matplotlib.pyplot as plt
import pandas as pd

# Calculate value counts
value_counts = df.cp.value_counts()

# Calculate average blood pressure for each group
avg_bp = df.groupby('target')['trestbps'].mean()
```

[54]:
```
# Create bar plot
plt.figure()
sns.barplot(x=avg_bp.index, y=avg_bp.values)
plt.title('Average Resting Blood Pressure by Heart Disease Status')
plt.xlabel('Heart Disease (0 = No, 1 = Yes)')
plt.ylabel('Average Resting Blood Pressure')
```

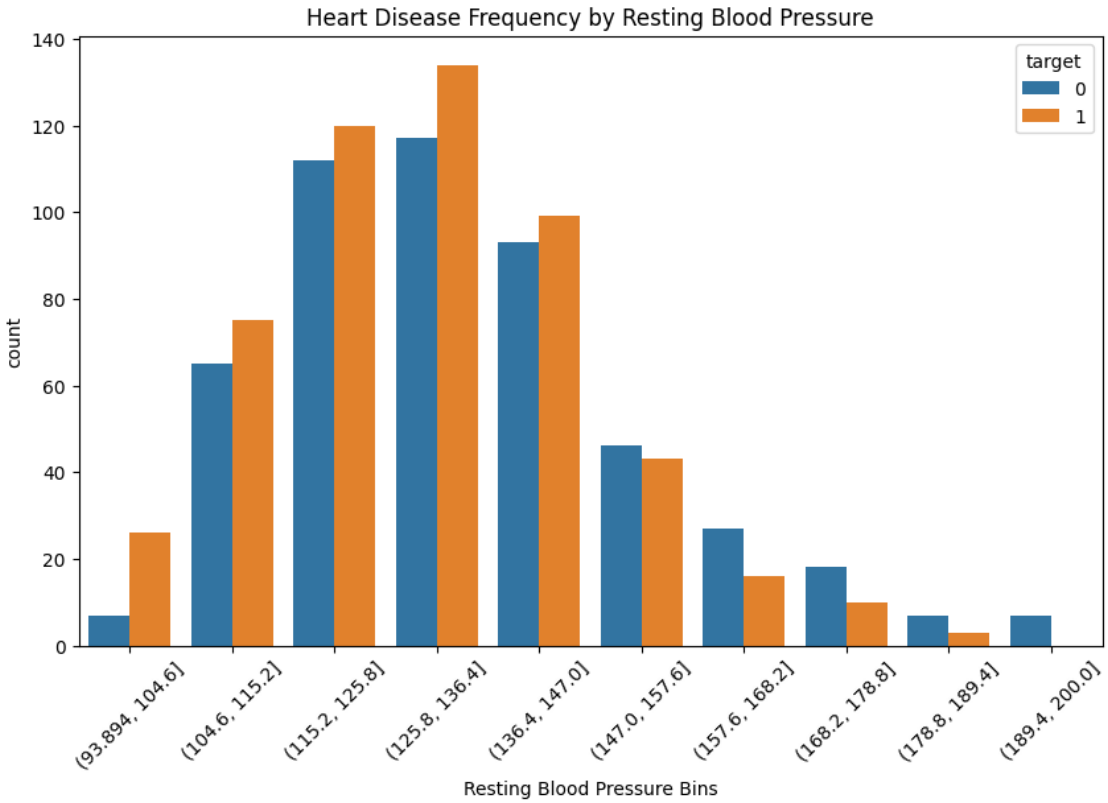
[54]:
```
Text(0, 0.5, 'Average Resting Blood Pressure')
```



[57]:
```
# Create bins for blood pressure
df['trestbps_bins'] = pd.cut(df['trestbps'], bins=10)

# Calculate average resting blood pressure for people with heart disease
avg_bp_with_disease = df[df['target'] == 1]['trestbps'].mean()

# Find the bin that contains the average
avg_bin = pd.cut([avg_bp_with_disease], bins=10)[0]
```

[58]:
```
# Create count plot
plt.figure(figsize=(10, 6))
ax = sns.countplot(x='trestbps_bins', data=df, hue='target')
plt.title('Heart Disease Frequency by Resting Blood Pressure')
plt.xlabel('Resting Blood Pressure Bins')
plt.xticks(rotation=45)
```

[58]:
```
([0, 1, 2, 3, 4, 5, 6, 7, 8, 9],
 [Text(0, 0, '(93.894, 104.6]'),
  Text(1, 0, '(104.6, 115.2]'),
  Text(2, 0, '(115.2, 125.8]'),
  Text(3, 0, '(125.8, 136.4]'),
  Text(4, 0, '(136.4, 147.0]'),
  Text(5, 0, '(147.0, 157.6]'),
  Text(6, 0, '(157.6, 168.2]'),
  Text(7, 0, '(168.2, 178.8]'),
  Text(8, 0, '(178.8, 189.4]'),
  Text(9, 0, '(189.4, 200.0]')])
```



[60]:
```
average_bp_with_disease = df[df['target'] == 1]['trestbps'].mean()
print(average_bp_with_disease)
```

```
129.24524714828897
```

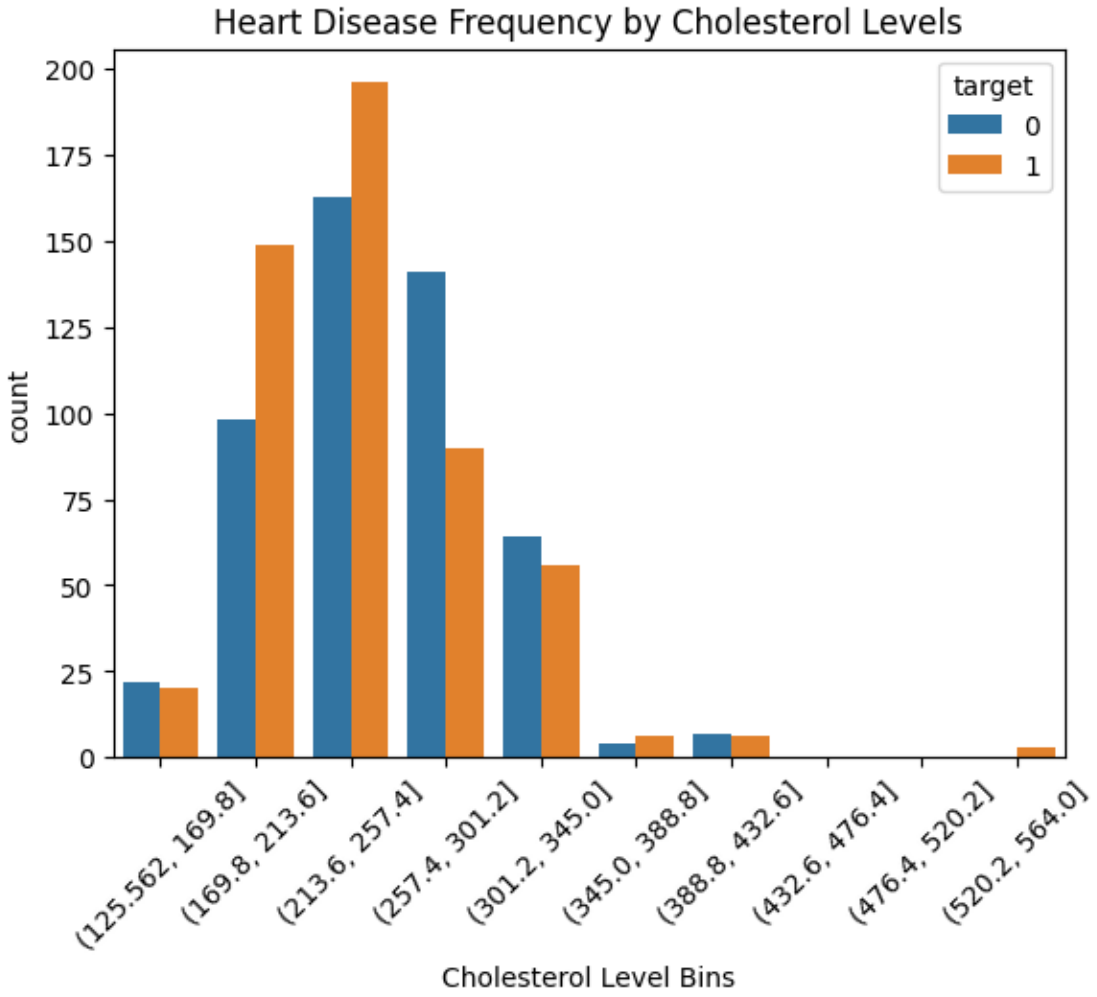
[ ]:
```
#The calculated average resting blood pressure for those with heart disease is␣
↪approximately 129
```

[61]:
```
#lets check the next question number 8.

#8.Is there a relationship between cholesterol levels and the presence of heart␣
↪disease?

df.groupby('target').agg({'chol': 'mean'})
```

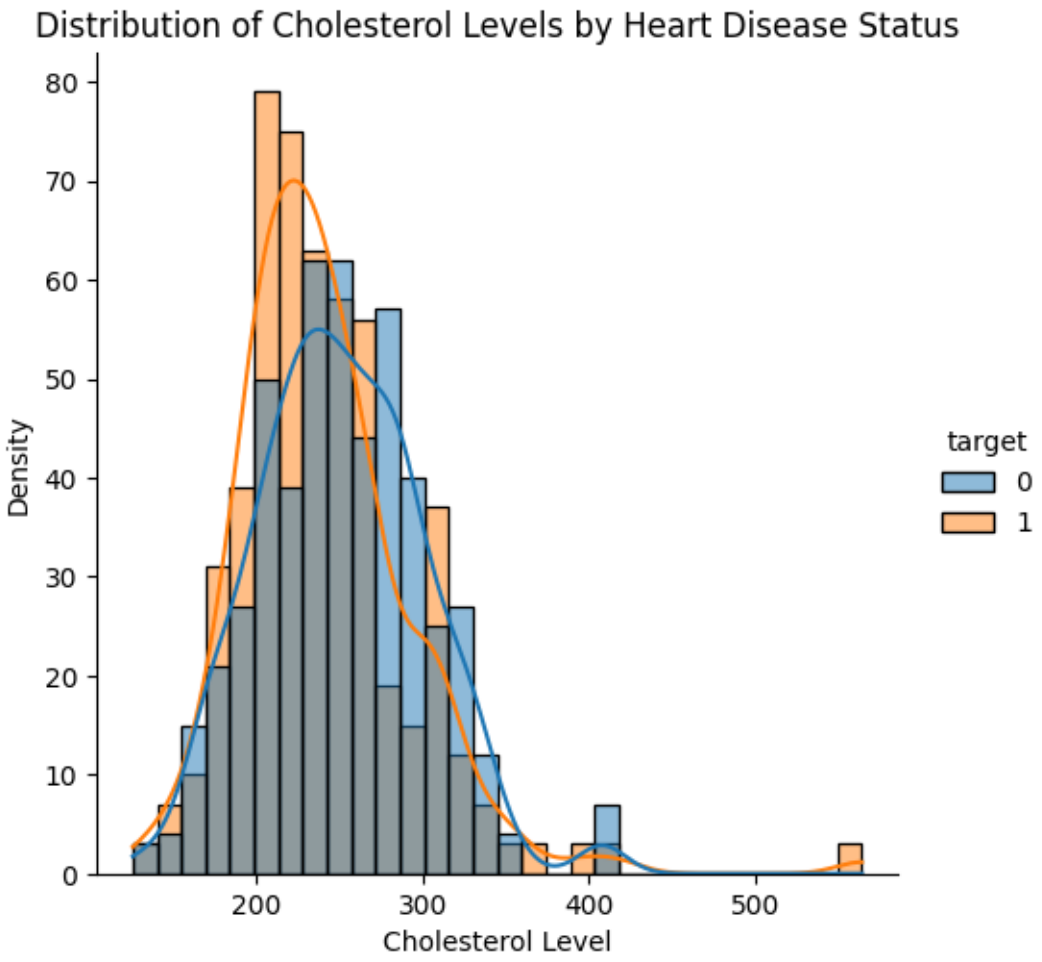
[61]:

target	chol
0	251.292585
1	240.979087

[62]:
```
# Create bins for cholesterol levels
df['chol_bins'] = pd.cut(df['chol'], bins=10)

plt.figure()
sns.countplot(x='chol_bins', data=df, hue='target')
plt.title('Heart Disease Frequency by Cholesterol Levels')
plt.xlabel('Cholesterol Level Bins')
plt.xticks(rotation=45)
plt.show()
```



[63]:
```
sns.displot(x='chol', data=df, bins=30, kde=True, hue='target')
plt.title('Distribution of Cholesterol Levels by Heart Disease Status')
plt.xlabel('Cholesterol Level')
plt.ylabel('Density')
plt.show()
```



[ ]:
```
# The visualizations suggest a potential association between cholesterol levels␣
↪and heart disease,

# as higher cholesterol levels appear to be more prevalent among individuals␣
↪with heart disease.
```