In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from IPython.display import Image
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
from sklearn.metrics import roc_curve
```

```
Matplotlib is building the font cache; this may take a moment.
```

In [2]:

```
data = pd.read_csv("heart_disease.csv.csv")
```

In [3]:

```
data
```

Out[3]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
298	57	0	0	140	241	0	1	123	1	0.2	1	0	3	0
299	45	1	3	110	264	0	1	132	0	1.2	1	0	3	0
300	68	1	0	144	193	1	1	141	0	3.4	1	2	3	0
301	57	1	0	130	131	0	1	115	1	1.2	1	1	3	0
302	57	0	1	130	236	0	0	174	0	0.0	1	1	2	0

303 rows × 14 columns

In [4]:

```
data.head()
```

Out[4]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1

In [5]:

```
data.tail()
```

Out[5]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
298	57	0	0	140	241	0	1	123	1	0.2	1	0	3	0
299	45	1	3	110	264	0	1	132	0	1.2	1	0	3	0
300	68	1	0	144	193	1	1	141	0	3.4	1	2	3	0
301	57	1	0	130	131	0	1	115	1	1.2	1	1	3	0
302	57	0	1	130	236	0	0	174	0	0.0	1	1	2	0

In [7]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       303 non-null    int64  
 1   sex       303 non-null    int64  
 2   cp        303 non-null    int64  
 3   trestbps  303 non-null    int64  
 4   chol      303 non-null    int64  
 5   fbs       303 non-null    int64  
 6   restecg   303 non-null    int64  
 7   thalach   303 non-null    int64  
 8   exang     303 non-null    int64  
 9   oldpeak   303 non-null    float64
 10  slope     303 non-null    int64  
 11  ca        303 non-null    int64  
 12  thal      303 non-null    int64  
 13  target    303 non-null    int64  
dtypes: float64(1), int64(13)
memory usage: 33.3 KB
```

In [8]:

```
data.describe()
```

Out[8]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.0
mean	54.366337	0.683168	0.966997	131.623762	246.264026	0.148515	0.528053	149.646865	0.326733	1.039604	1.3
std	9.082101	0.466011	1.032052	17.538143	51.830751	0.356198	0.525860	22.905161	0.469794	1.161075	0.6
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.000000	0.000000	0.000000	0.0
25%	47.500000	0.000000	0.000000	120.000000	211.000000	0.000000	0.000000	133.500000	0.000000	0.000000	1.0
50%	55.000000	1.000000	1.000000	130.000000	240.000000	0.000000	1.000000	153.000000	0.000000	0.800000	1.0
75%	61.000000	1.000000	2.000000	140.000000	274.500000	0.000000	1.000000	166.000000	1.000000	1.600000	2.0
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000	202.000000	1.000000	6.200000	2.0

In [9]:

```
####### null values ######

data.isnull().sum()
```

Out[9]:

```
age         0
sex         0
cp          0
trestbps    0
chol        0
fbs         0
restecg     0
```

```
restecg     0
thalach     0
exang       0
oldpeak     0
slope       0
ca          0
thal        0
target      0
dtype: int64
```

In [10]:

```
data.describe().T
```

Out[10]:

	count	mean	std	min	25%	50%	75%	max
age	303.0	54.366337	9.082101	29.0	47.5	55.0	61.0	77.0
sex	303.0	0.683168	0.466011	0.0	0.0	1.0	1.0	1.0
cp	303.0	0.966997	1.032052	0.0	0.0	1.0	2.0	3.0
trestbps	303.0	131.623762	17.538143	94.0	120.0	130.0	140.0	200.0
chol	303.0	246.264026	51.830751	126.0	211.0	240.0	274.5	564.0
fbs	303.0	0.148515	0.356198	0.0	0.0	0.0	0.0	1.0
restecg	303.0	0.528053	0.525860	0.0	0.0	1.0	1.0	2.0
thalach	303.0	149.646865	22.905161	71.0	133.5	153.0	166.0	202.0
exang	303.0	0.326733	0.469794	0.0	0.0	0.0	1.0	1.0
oldpeak	303.0	1.039604	1.161075	0.0	0.0	0.8	1.6	6.2
slope	303.0	1.399340	0.616226	0.0	1.0	1.0	2.0	2.0
ca	303.0	0.729373	1.022606	0.0	0.0	0.0	1.0	4.0
thal	303.0	2.313531	0.612277	0.0	2.0	2.0	3.0	3.0
target	303.0	0.544554	0.498835	0.0	0.0	1.0	1.0	1.0

In [11]:

```
########## data visualization ############

data.corr()
```

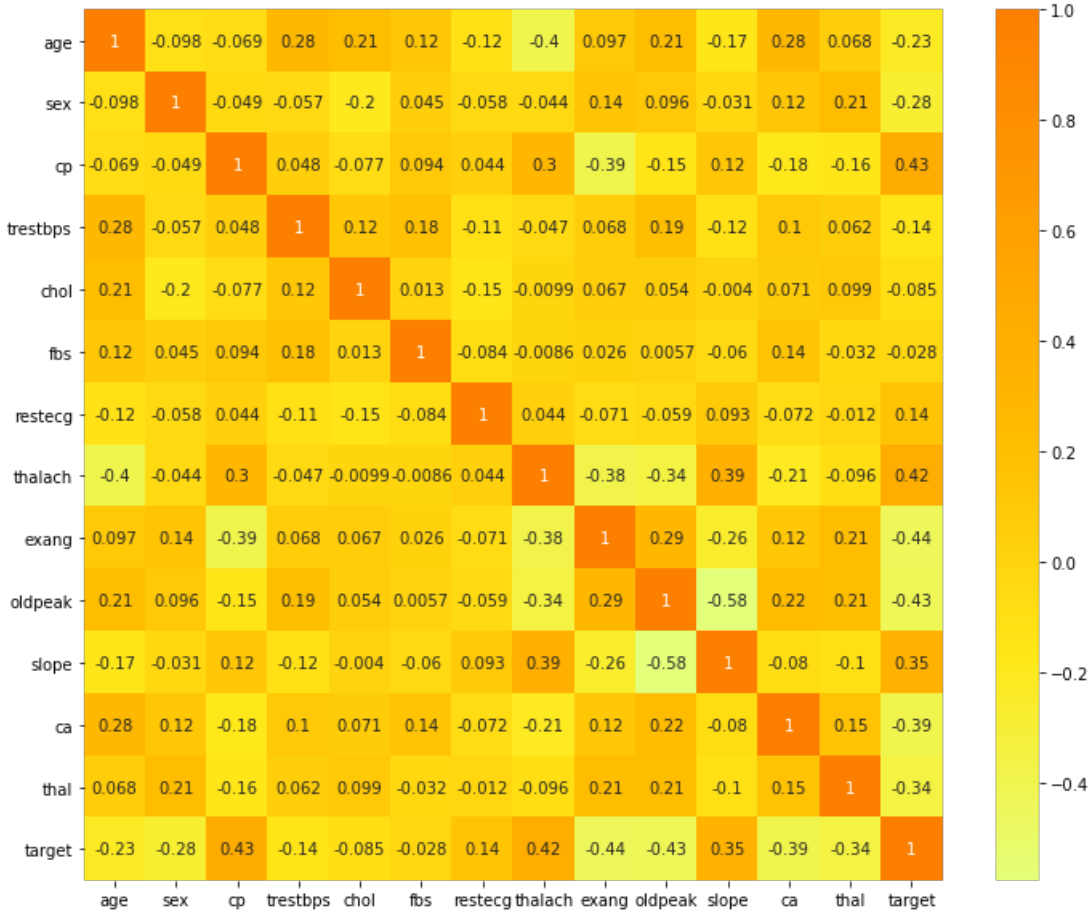
Out[11]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	
age	1.000000	-0.098447	-0.068653	0.279351	0.213678	0.121308	-0.116211	-0.398522	0.096801	0.210013	-0.168814	0.276326	0.06
sex	-0.098447	1.000000	-0.049353	-0.056769	-0.197912	0.045032	-0.058196	-0.044020	0.141664	0.096093	-0.030711	0.118261	0.21
cp	-0.068653	-0.049353	1.000000	0.047608	-0.076904	0.094444	0.044421	0.295762	-0.394280	-0.149230	0.119717	-0.181053	0.16
trestbps	0.279351	-0.056769	0.047608	1.000000	0.123174	0.177531	-0.114103	-0.046698	0.067616	0.193216	-0.121475	0.101389	0.06
chol	0.213678	-0.197912	-0.076904	0.123174	1.000000	0.013294	-0.151040	-0.009940	0.067023	0.053952	-0.004038	0.070511	0.09
fbs	0.121308	0.045032	0.094444	0.177531	0.013294	1.000000	-0.084189	-0.008567	0.025665	0.005747	-0.059894	0.137979	0.03
restecg	-0.116211	-0.058196	0.044421	-0.114103	-0.151040	-0.084189	1.000000	0.044123	-0.070733	-0.058770	0.093045	-0.072042	0.01
thalach	-0.398522	-0.044020	0.295762	-0.046698	-0.009940	-0.008567	0.044123	1.000000	-0.378812	-0.344187	0.386784	-0.213177	0.09
exang	0.096801	0.141664	-0.394280	0.067616	0.067023	0.025665	-0.070733	-0.378812	1.000000	0.288223	-0.257748	0.115739	0.20
oldpeak	0.210013	0.096093	-0.149230	0.193216	0.053952	0.005747	-0.058770	-0.344187	0.288223	1.000000	-0.577537	0.222682	0.21

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	
slope	-0.168814	-0.030711	0.119717	-0.121475	-0.004038	0.059894	0.093045	-0.386784	0.257748	0.577537	1.000000	0.080155	0.10
ca	0.276326	0.118261	-0.181053	0.101389	0.070511	0.137979	-0.072042	-0.213177	0.115739	0.222682	0.080155	1.000000	0.15
thal	0.068001	0.210041	-0.161736	0.062210	0.098803	-0.032019	-0.011981	-0.096439	0.206754	0.210244	-0.104764	0.151832	1.00
target	-0.225439	-0.280937	0.433798	-0.144931	-0.085239	-0.028046	0.137230	0.421741	-0.436757	-0.430696	0.345877	-0.391724	0.34

In [13]:

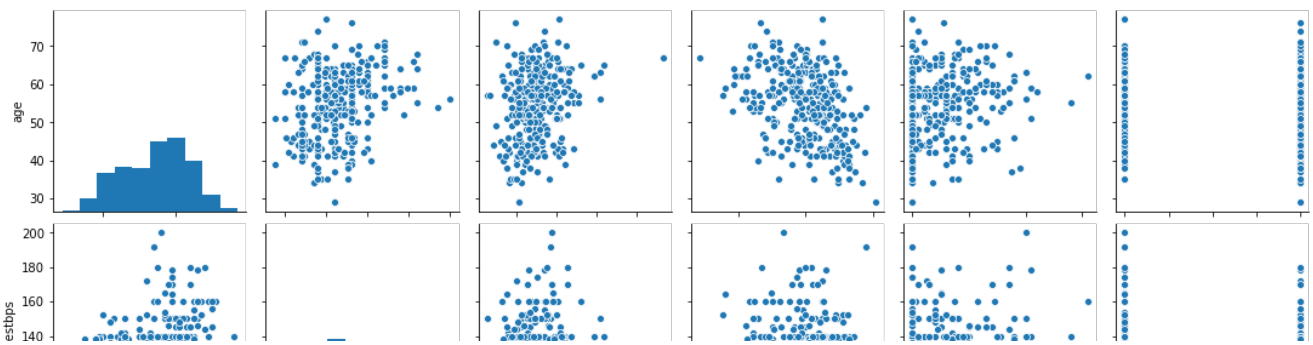
```
plt.figure(figsize=(12,10))
sns.heatmap(data.corr(), annot=True,cmap='Wistia')
plt.show()        ####### we can figure out that cp, thalach, slope is little bit correlated with t
arget #######
```

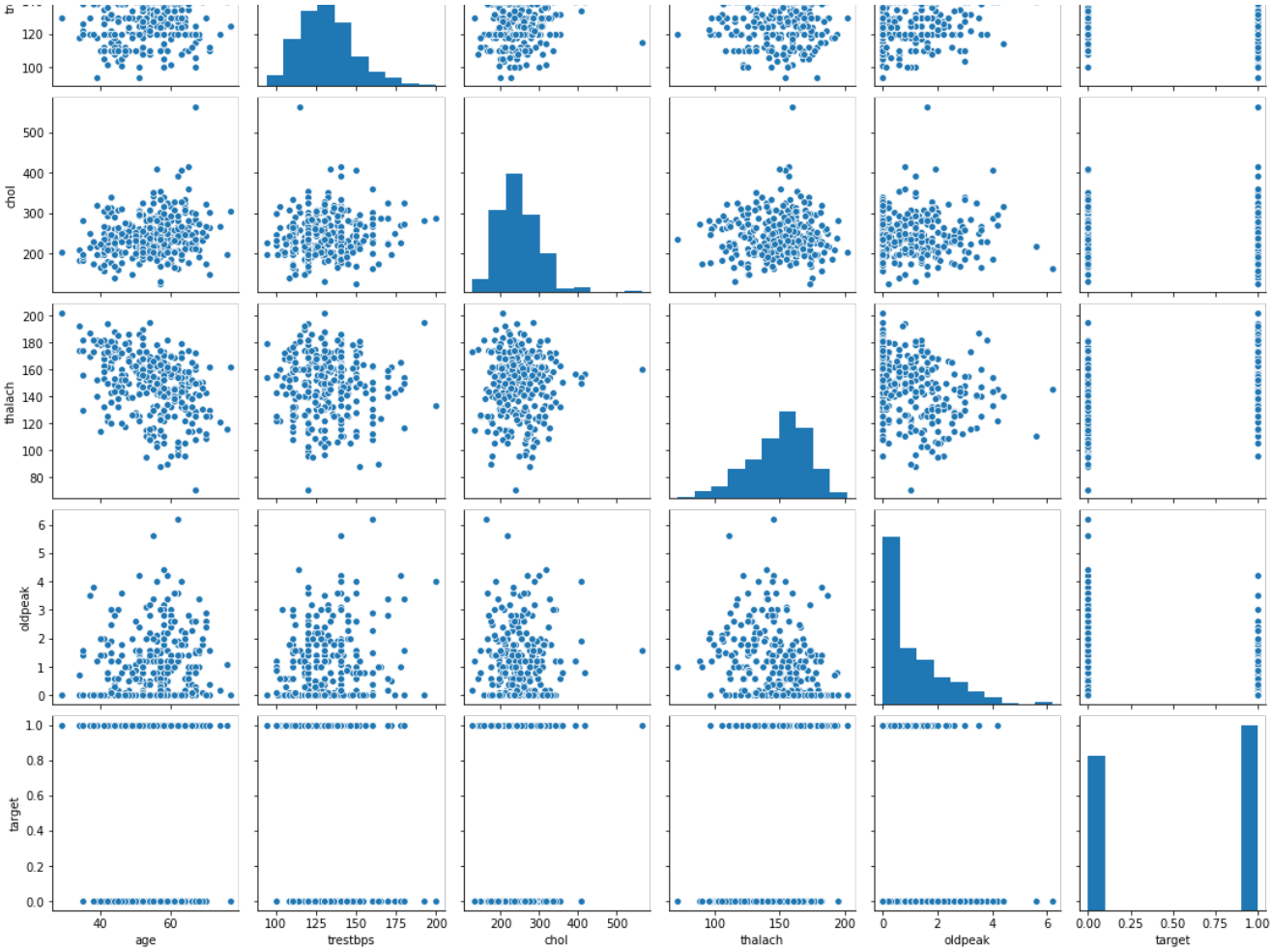


In [17]:

```
plt.figure(figsize=(15,10))
num = ['age', 'trestbps', 'chol', 'thalach', 'oldpeak', 'target' ]
sns.pairplot(data[num],kind = 'scatter',diag_kind='hist')
plt.show()
```

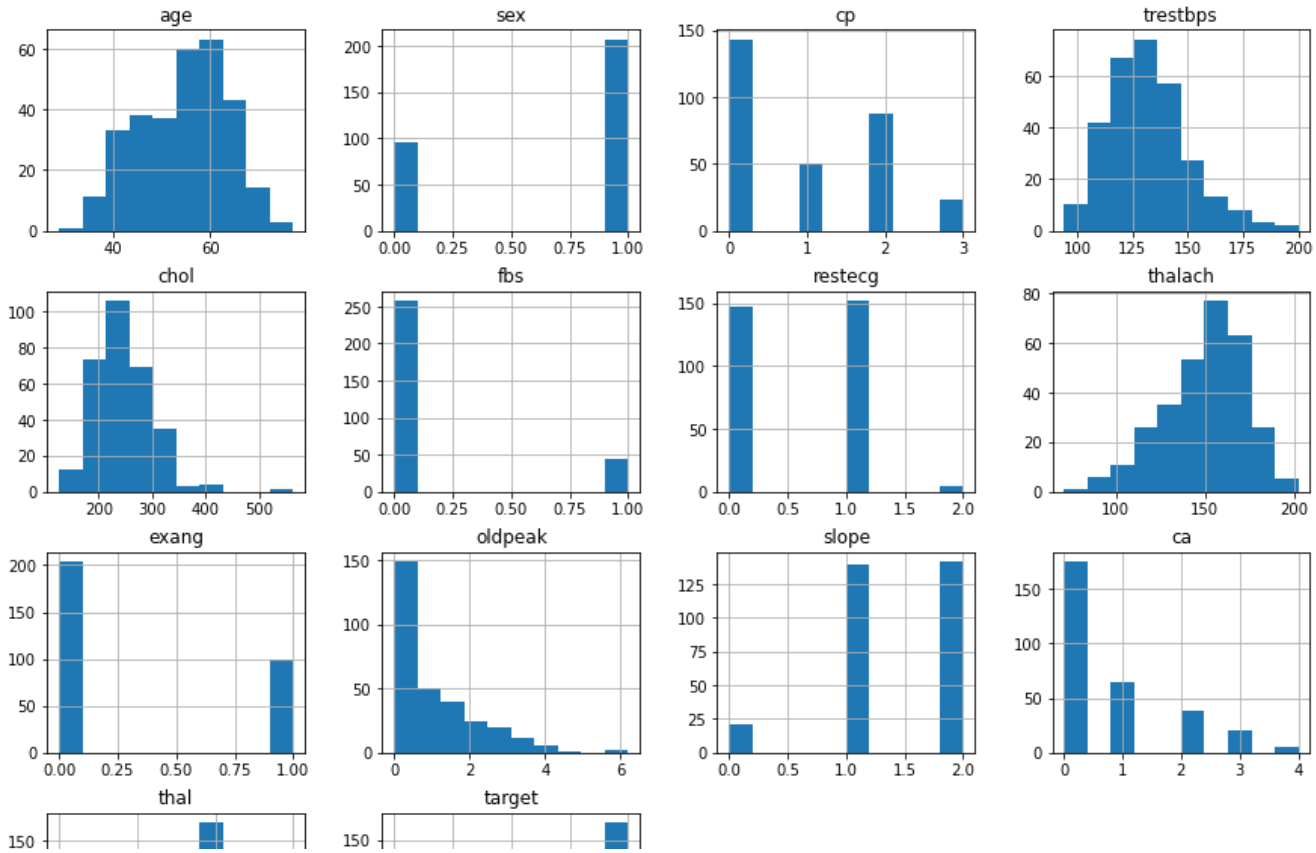
```
<Figure size 1080x720 with 0 Axes>
```

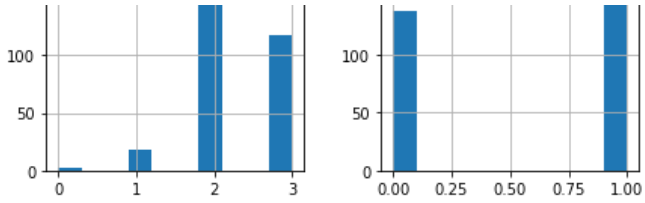




In [18]:

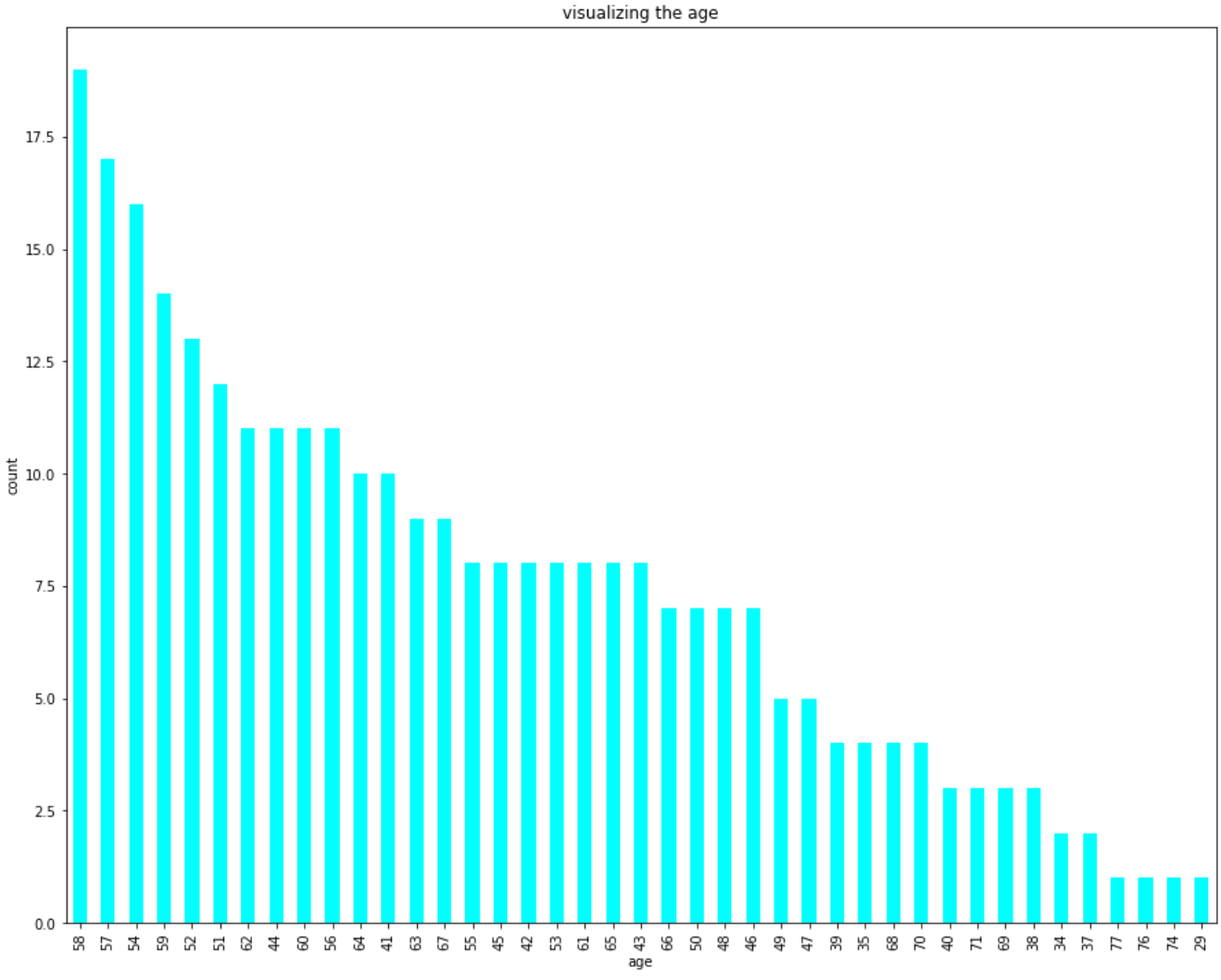
```
#### histogram plotting #######

data.hist(figsize = (15,12))
plt.show()
```





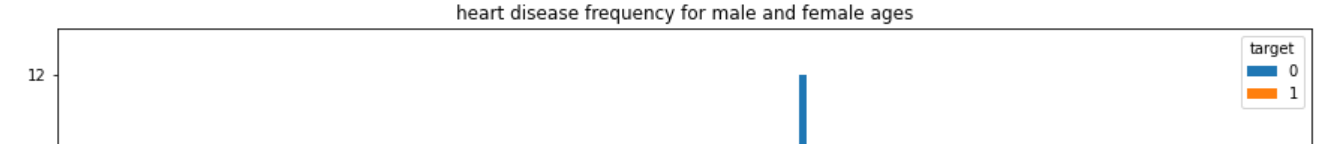
In [22]:

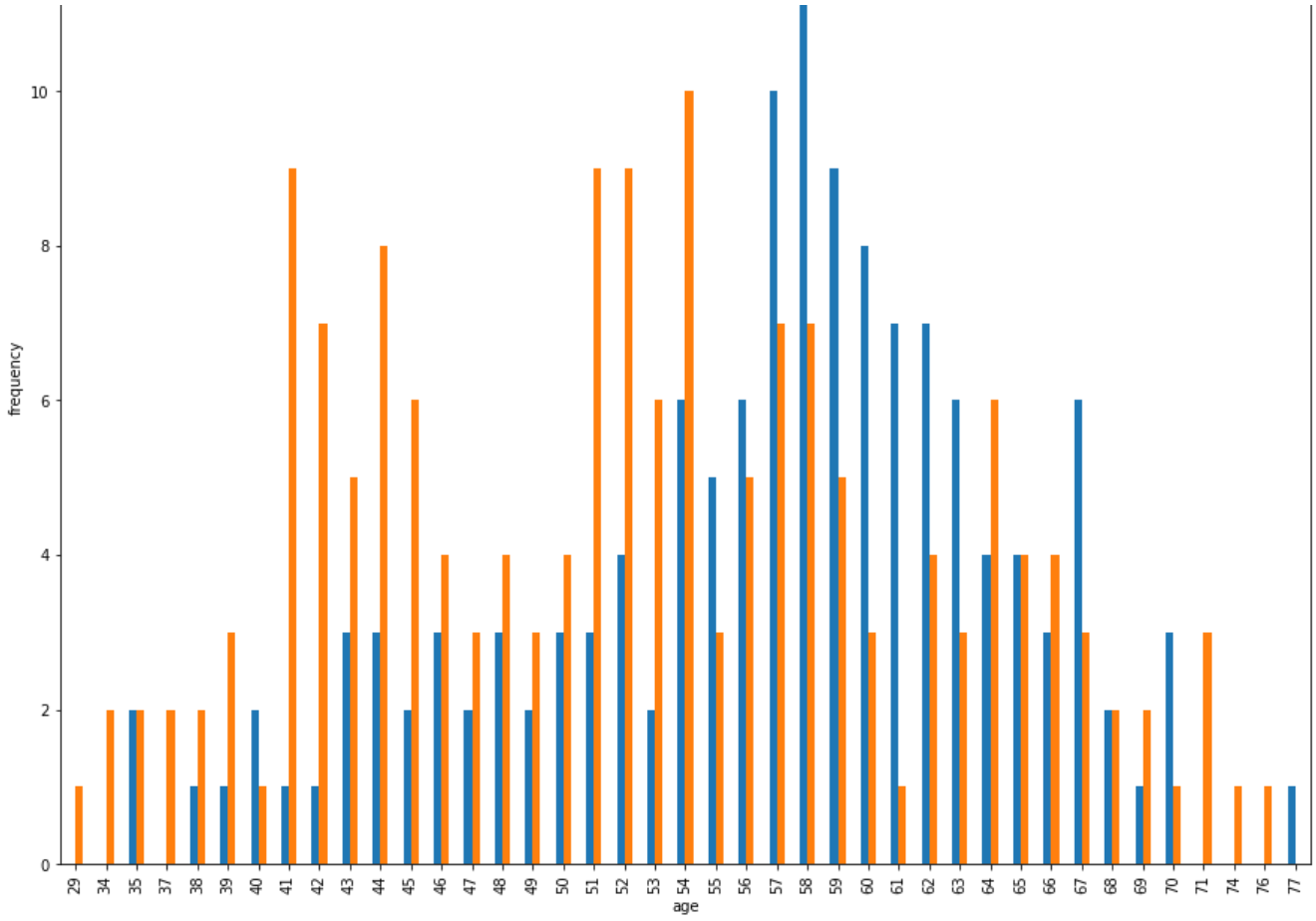
```
########## visualizing the age ##########

plt.subplots(figsize=(15,12))
data['age'].value_counts(normalize = True)
data['age'].value_counts(dropna=False).plot.bar(color = 'cyan')
plt.title('visualizing the age')
plt.xlabel('age')
plt.ylabel('count')
plt.show()
```



In [23]:

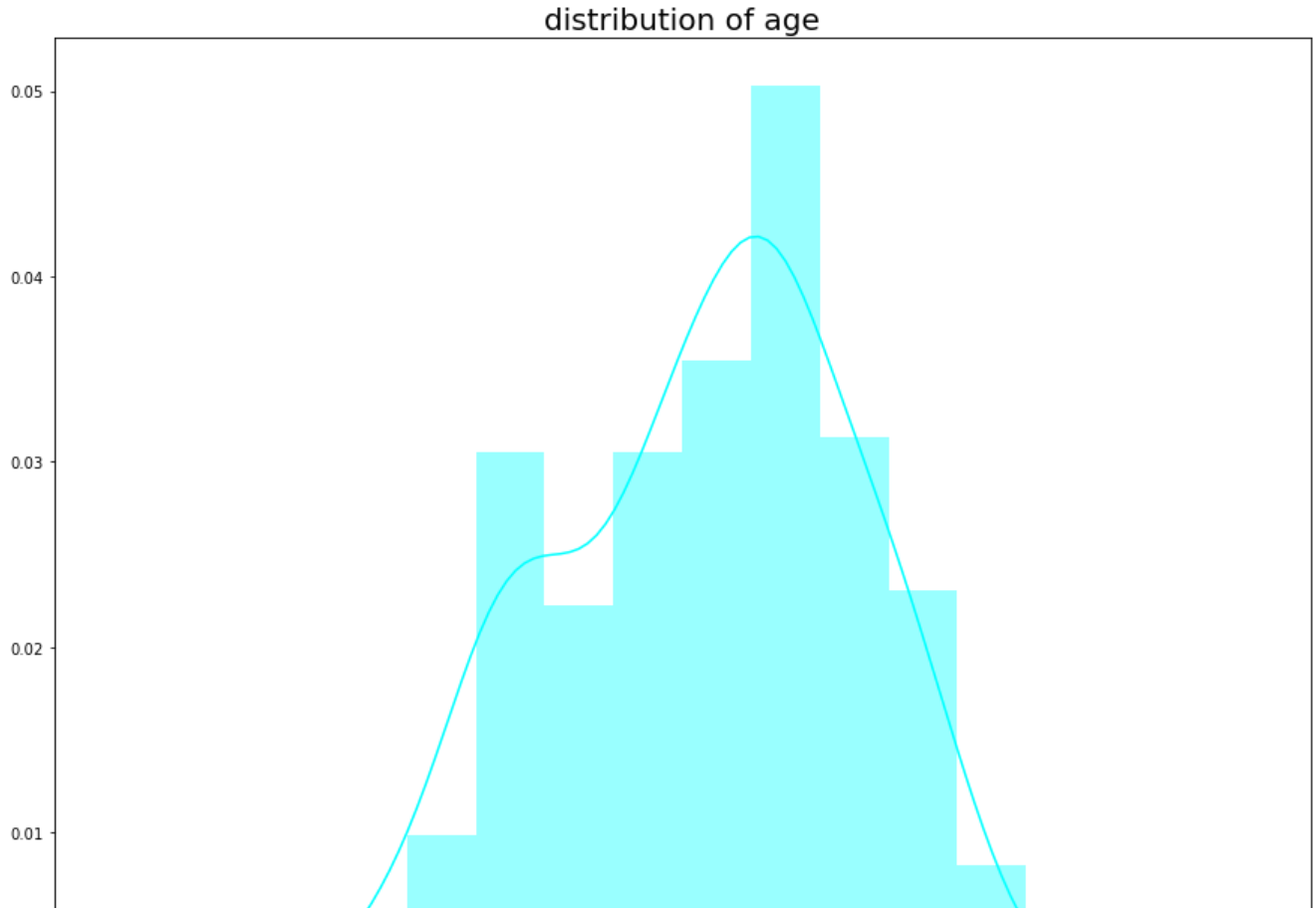
```
pd.crosstab(data.age,data.target).plot(kind = 'bar',figsize=(15,12))
plt.title('heart disease frequency for male and female ages')
plt.xlabel('age')
plt.ylabel('frequency')
plt.show()
```

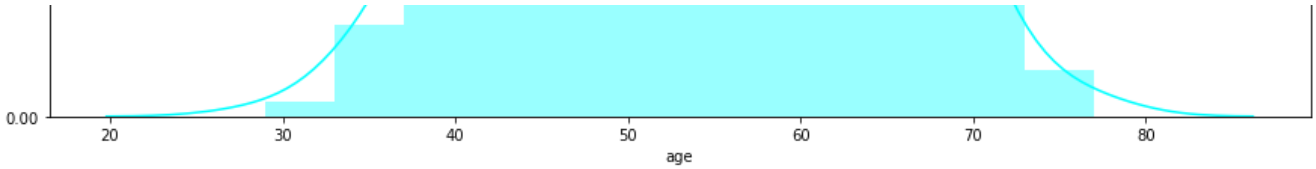




In [24]:

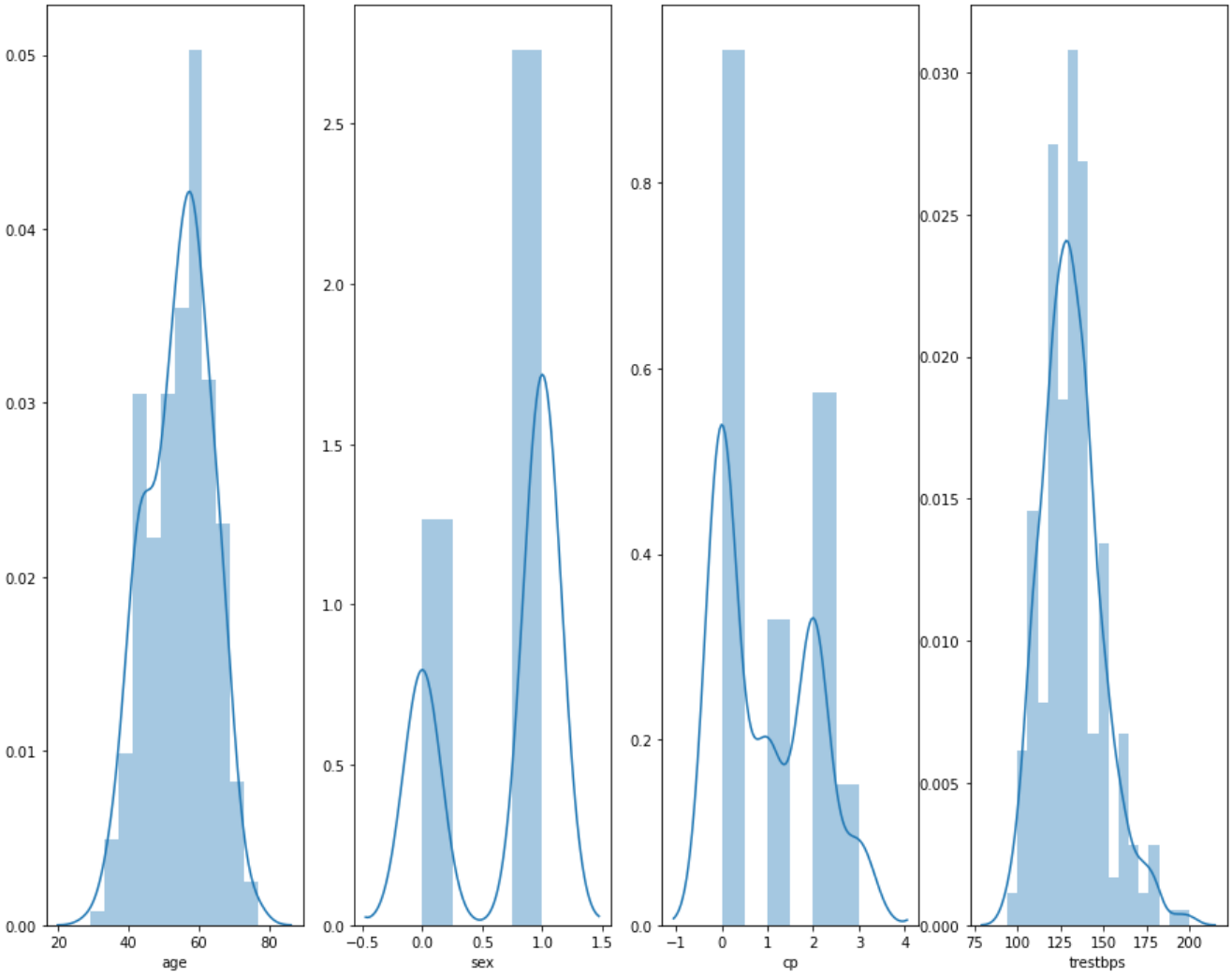
```
plt.subplots(figsize=(15,12))
sns.distplot(data['age'],color = 'cyan')
plt.title('distribution of age',fontsize = 20)
plt.show()
```





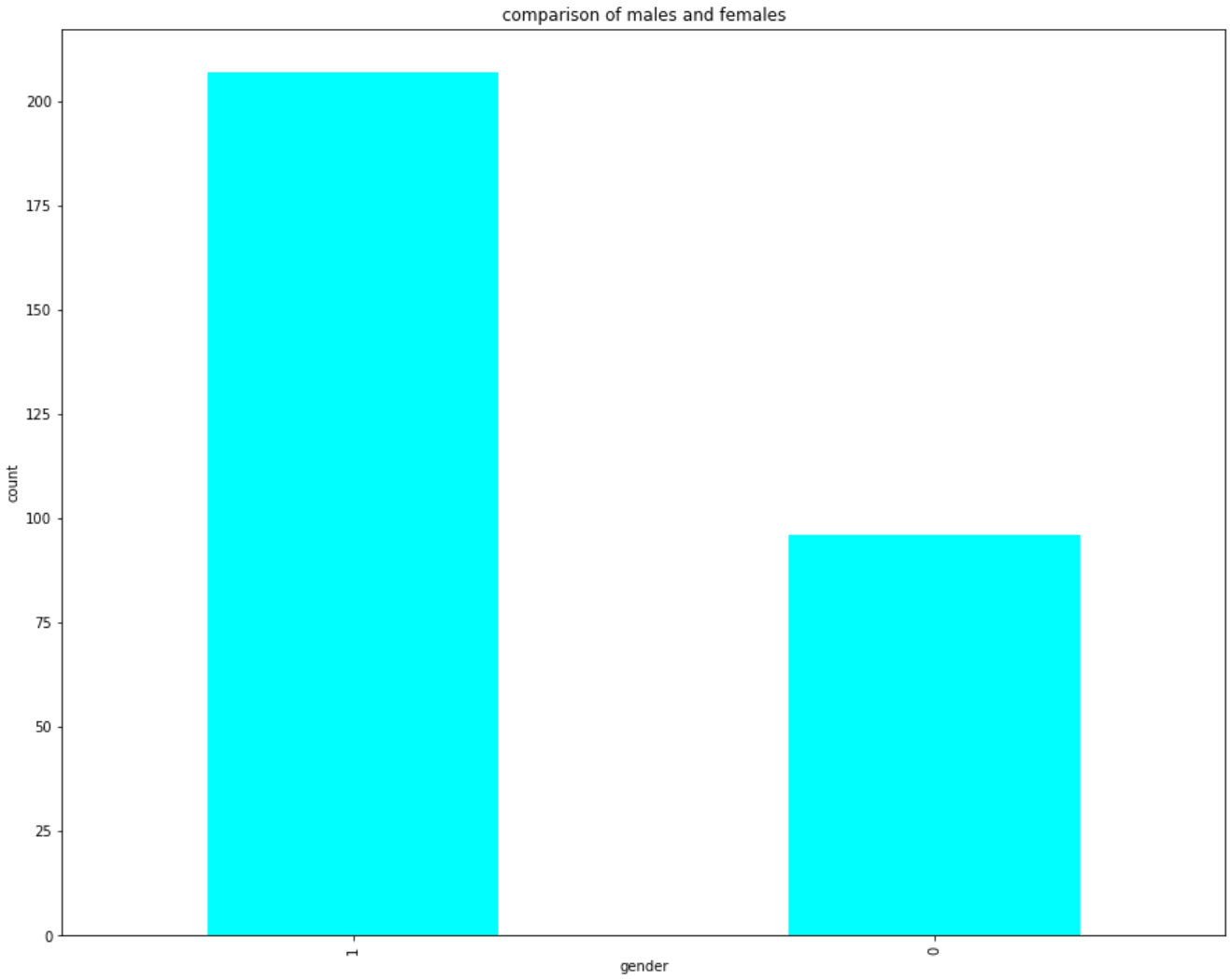
In [27]:

```
plt.subplots(figsize=(15,12))
plt.subplot(1,4,1)
sns.distplot(data['age'])

plt.subplot(1,4,2)
sns.distplot(data['sex'])

plt.subplot(1,4,3)
sns.distplot(data['cp'])

plt.subplot(1,4,4)
sns.distplot(data['trestbps'])

plt.show()
```



In [28]:

```
####### visualizing the male and female of the dataset ##################

plt.subplots(figsize=(15,12))
data['sex'].value_counts(normalize = True)
data['sex'].value_counts(dropna = False).plot.bar(color='cyan')
plt.title('comparison of males and females')
plt.xlabel('gender')
plt.ylabel('count')
plt.show()
```



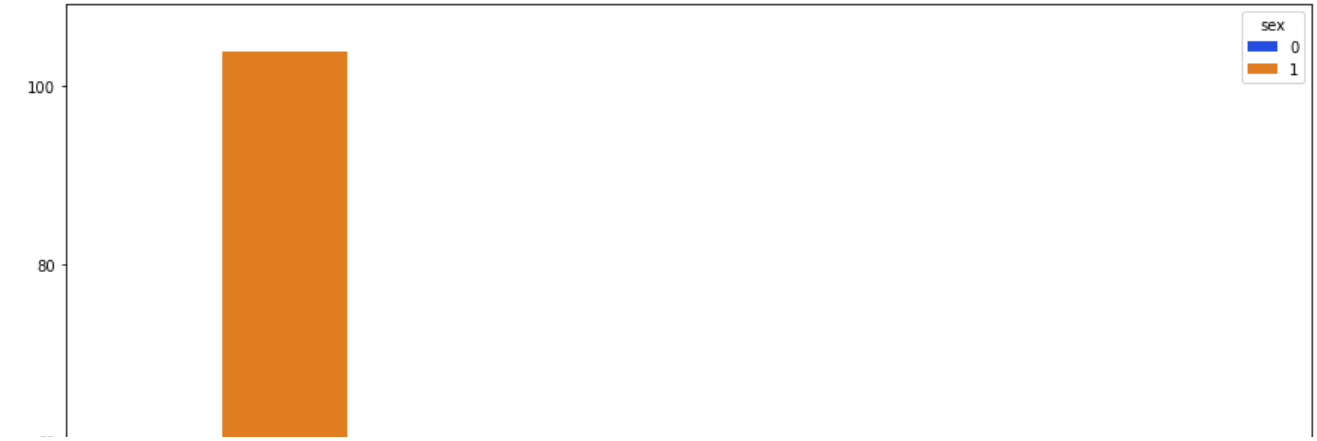
In [29]:

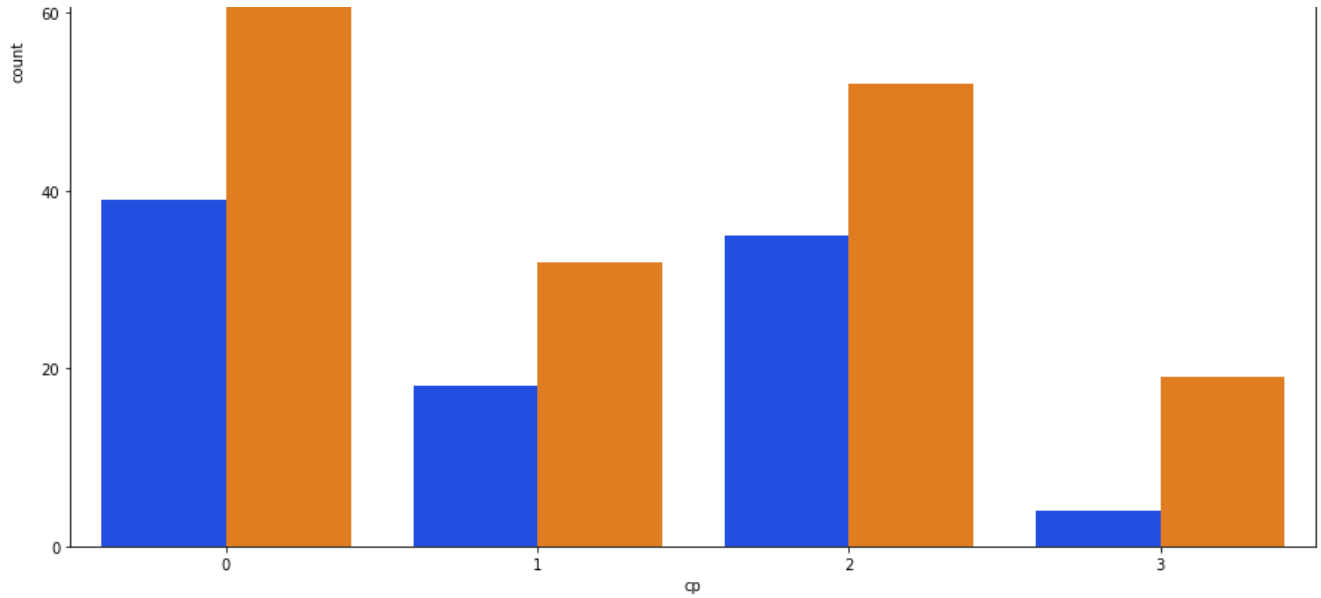
```
female_count=len(data[data.sex==0])
male_count=len(data[data.sex==1])
print("percentage of female patients:{:.2f}%".format((female_count/len(data.sex)*100)))
print("percentage of male patients:{:.2f}%".format((male_count/len(data.sex)*100)))
```

```
percentage of female patients:31.68%
percentage of male patients:68.32%
```

In [31]:

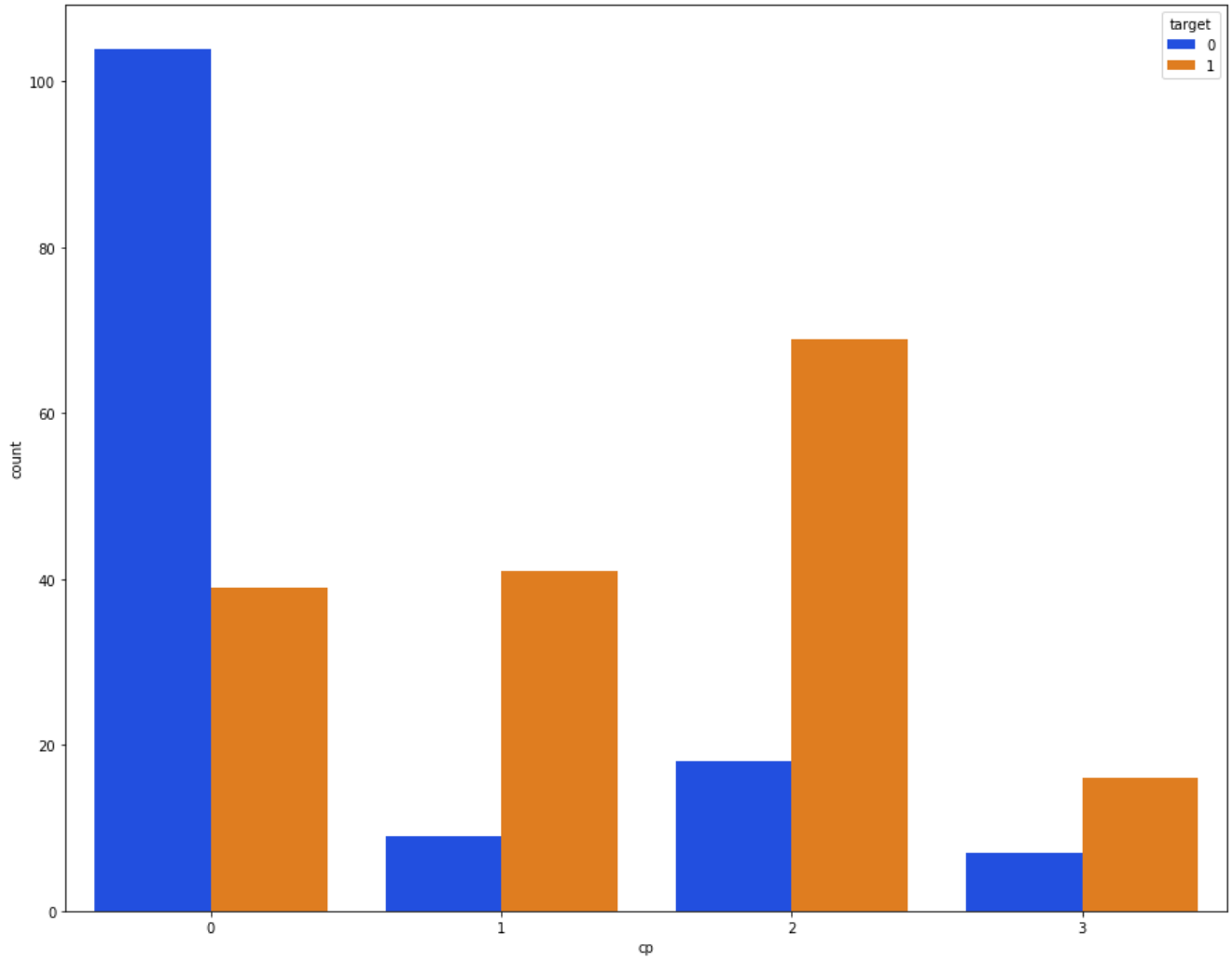
```
plt.subplots(figsize = (15,12))
sns.countplot(x = 'cp', data=data,hue = 'sex', palette = 'bright')
plt.show()
```





In [32]:

```
plt.subplots(figsize=(15,12))
sns.countplot(x='cp',data=data,hue='target',palette='bright')
plt.show()
```



In [33]:

```
!pip install missingno
```

```
Collecting missingno
  Using cached missingno-0.4.2-py3-none-any.whl (9.7 kB)
```

```
Requirement already satisfied: numpy in c:\users\dipsikha\anaconda3\lib\site-packages (from
missingno) (1.18.1)
Requirement already satisfied: seaborn in c:\users\dipsikha\anaconda3\lib\site-packages (from
missingno) (0.10.0)
Requirement already satisfied: scipy in c:\users\dipsikha\anaconda3\lib\site-packages (from
missingno) (1.4.1)
Requirement already satisfied: matplotlib in c:\users\dipsikha\anaconda3\lib\site-packages (from
missingno) (3.3.1)
Requirement already satisfied: pandas>=0.22.0 in c:\users\dipsikha\anaconda3\lib\site-packages
(from seaborn->missingno) (1.1.1)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\dipsikha\anaconda3\lib\site-packages
(from matplotlib->missingno) (1.1.0)
Collecting certifi>=2020.06.20
  Downloading certifi-2020.6.20-py2.py3-none-any.whl (156 kB)
Requirement already satisfied: cycler>=0.10 in c:\users\dipsikha\anaconda3\lib\site-packages (from
matplotlib->missingno) (0.10.0)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3 in
c:\users\dipsikha\anaconda3\lib\site-packages (from matplotlib->missingno) (2.4.6)
Requirement already satisfied: python-dateutil>=2.1 in c:\users\dipsikha\anaconda3\lib\site-
packages (from matplotlib->missingno) (2.8.1)
Requirement already satisfied: pillow>=6.2.0 in c:\users\dipsikha\anaconda3\lib\site-packages
(from matplotlib->missingno) (7.0.0)
Requirement already satisfied: pytz>=2017.2 in c:\users\dipsikha\anaconda3\lib\site-packages (from
pandas>=0.22.0->seaborn->missingno) (2019.3)
Requirement already satisfied: setuptools in c:\users\dipsikha\anaconda3\lib\site-packages (from
kiwisolver>=1.0.1->matplotlib->missingno) (45.2.0.post20200210)
Requirement already satisfied: six in c:\users\dipsikha\anaconda3\lib\site-packages (from
cycler>=0.10->matplotlib->missingno) (1.14.0)
Installing collected packages: missingno, certifi
  Attempting uninstall: certifi
    Found existing installation: certifi 2019.11.28
    Uninstalling certifi-2019.11.28:
      Successfully uninstalled certifi-2019.11.28
Successfully installed certifi-2020.6.20 missingno-0.4.2
```

In [34]:

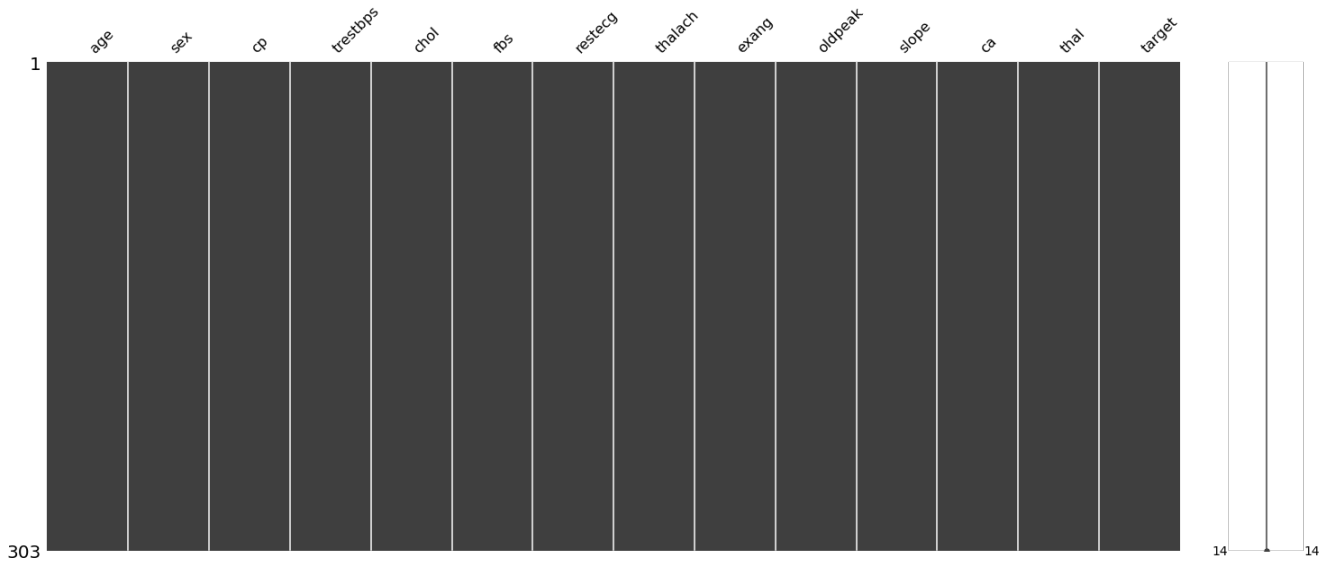
```
import missingno as msno
```

In [35]:

```
msno.matrix(data)
```

Out[35]:

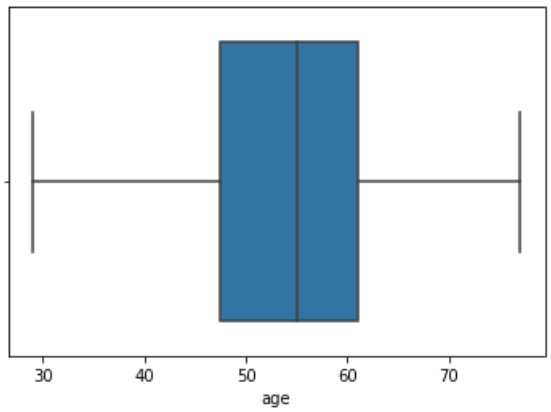
```
<AxesSubplot:>
```



In [36]:

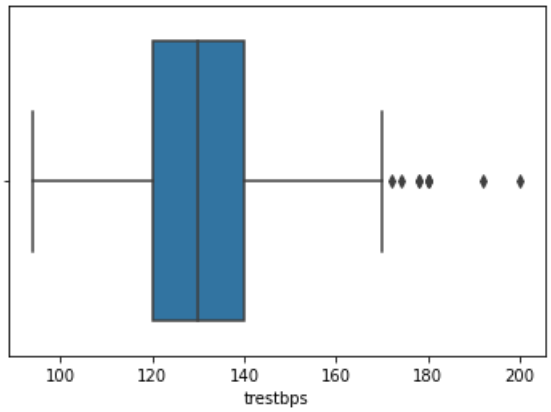
```
########## outlier detection ############
```

```
data['age'].describe()
sns.boxplot(data['age'])
plt.show()
```



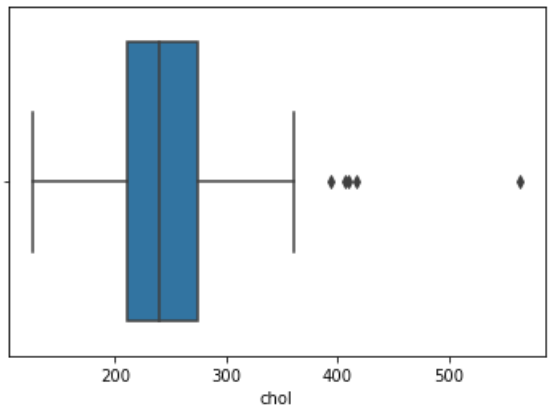
In [37]:

```
data['trestbps'].describe()
sns.boxplot(data['trestbps'])
plt.show()
```



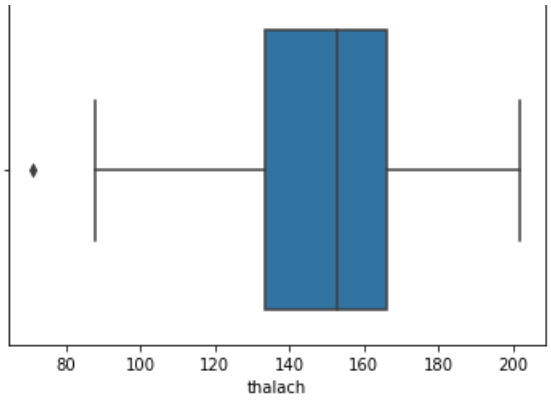
In [38]:

```
data['chol'].describe()
sns.boxplot(data['chol'])
plt.show()
```



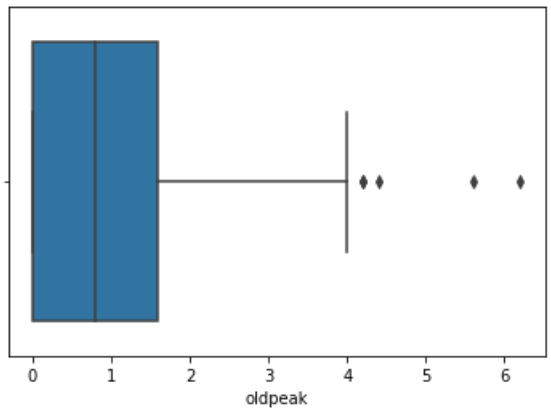
In [39]:

```
data['thalach'].describe()
sns.boxplot(data['thalach'])
plt.show()
```



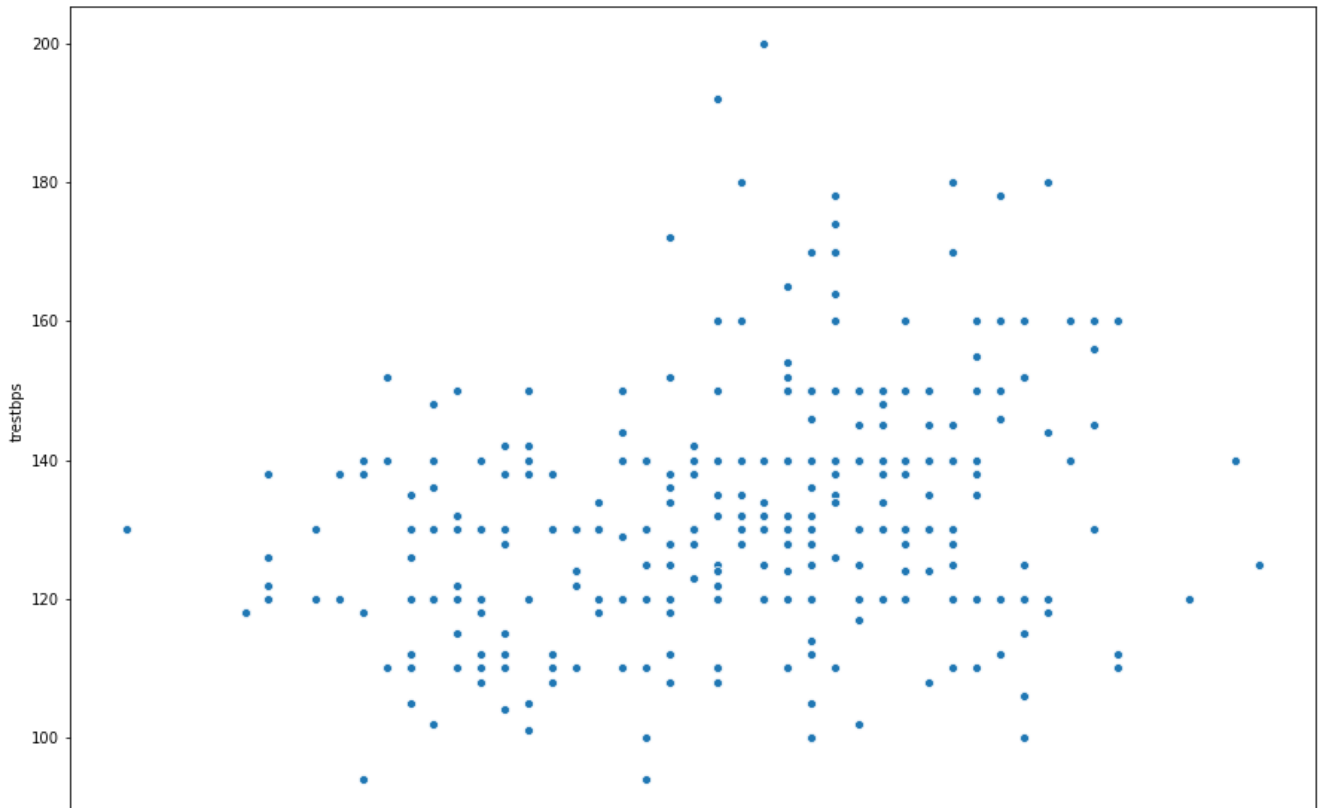
In [40]:

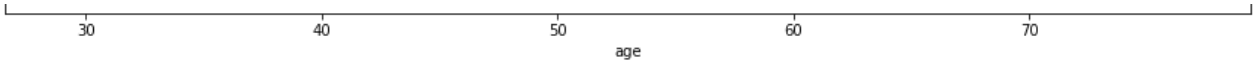
```
data['oldpeak'].describe()
sns.boxplot(data['oldpeak'])
plt.show()
```



In [41]:

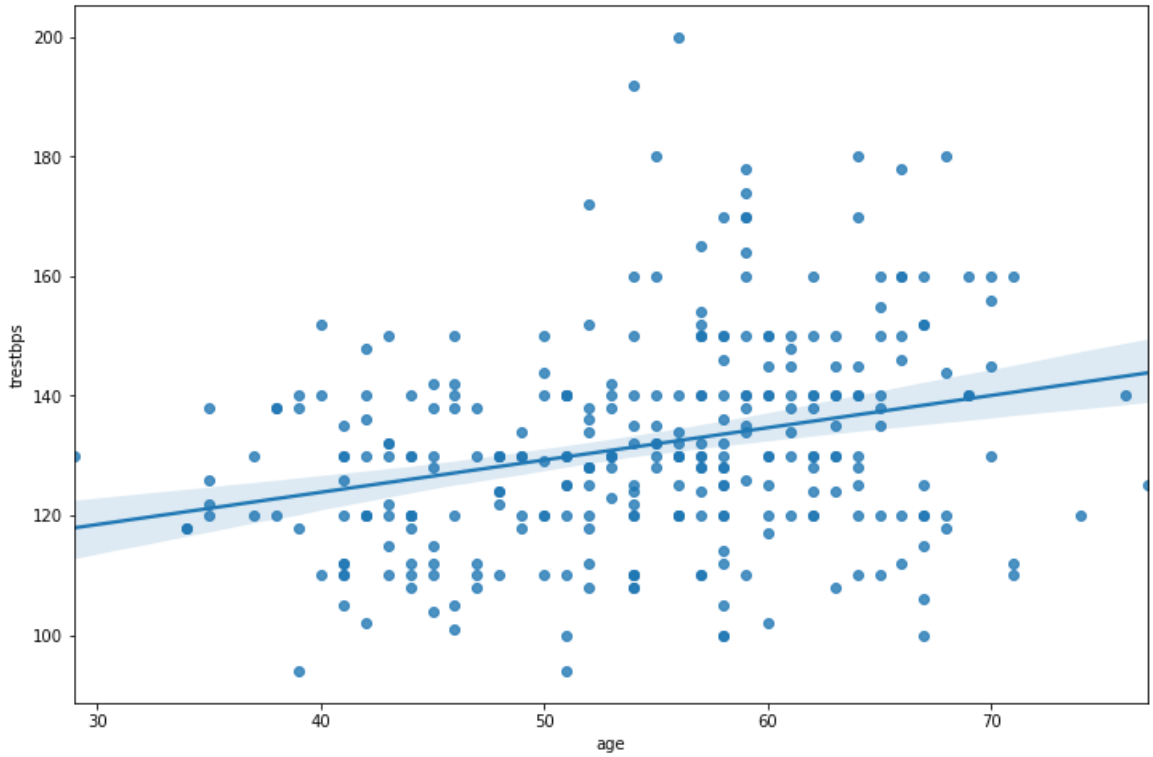
```
f,ax = plt.subplots(figsize=(15,10))
ax = sns.scatterplot(x='age',y='trestbps',data = data)
plt.show() #### there is no such correlation present between age and trestbps ###########
```





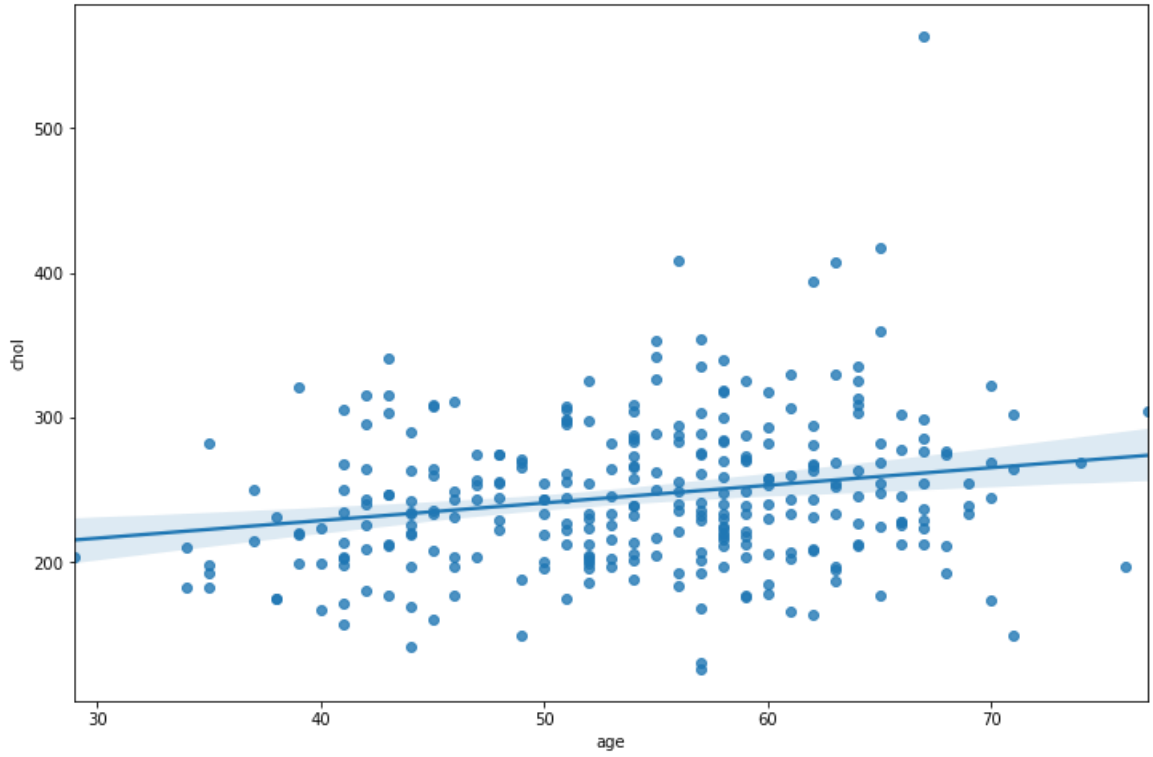
In [42]:

```
f,ax = plt.subplots(figsize=(12,8))
ax = sns.regplot(x='age',y='trestbps',data = data)
plt.show()    ######## no linearity present between age and trestbps ##########
```



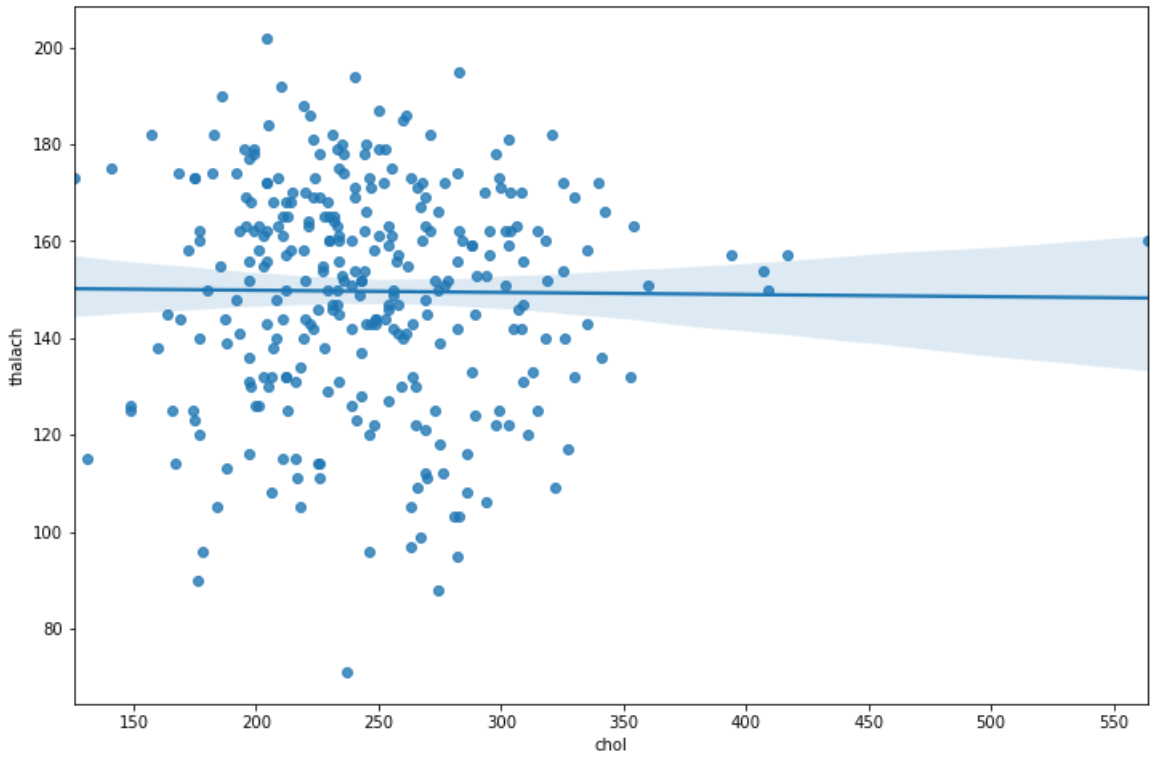
In [43]:

```
f,ax = plt.subplots(figsize=(12,8))
ax = sns.regplot(x='age',y='chol',data = data)
plt.show() ####### slightly better linear ############
```



In [44]:

```
f,ax = plt.subplots(figsize=(12,8))
ax = sns.regplot(x='chol',y='thalach',data = data)
plt.show() ###### completely no correlation ##########
```



In [ ]:

```
######## Results Outliers

The age variable does not contain any outlier.

trestbps variable contains outliers to the right side.

chol variable also contains outliers to the right side.

thalach variable contains a single outlier to the left side.

oldpeak variable contains outliers to the right side.

Those variables containing outliers needs further investigation. ################
```

In [45]:

```
###### data pre-processing ##########
###### changing the names of the columns for better understanding #############

data.columns = ['age', 'sex', 'chest_pain_type', 'resting_blood_pressure', 'cholesterol', 'fasting_
blood_sugar', 'rest_ecg', 'max_heart_rate_achieved',
       'exercise_induced_angina', 'st_depression', 'st_slope', 'num_major_vessels', 'thalassemia',
'target']
```

In [46]:

```
data['sex'][data['sex']==0]='female'
data['sex'][data['sex']==1]='male'

data['chest_pain_type'][data['chest_pain_type'] == 1] = 'typical angina'
data['chest_pain_type'][data['chest_pain_type'] == 2] = 'atypical angina'
data['chest_pain_type'][data['chest_pain_type'] == 3] = 'non-anginal pain'
data['chest_pain_type'][data['chest_pain_type'] == 4] = 'asymptomatic'

data['fasting_blood_sugar'][data['fasting_blood_sugar'] == 0] = 'lower than 120mg/ml'
```

```
data['fasting_blood_sugar'][data['fasting_blood_sugar'] == 1] = 'greater than 120mg/ml'

data['rest_ecg'][data['rest_ecg'] == 0] = 'normal'
data['rest_ecg'][data['rest_ecg'] == 1] = 'ST-T wave abnormality'
data['rest_ecg'][data['rest_ecg'] == 2] = 'left ventricular hypertrophy'

data['exercise_induced_angina'][data['exercise_induced_angina'] == 0] = 'no'
data['exercise_induced_angina'][data['exercise_induced_angina'] == 1] = 'yes'

data['st_slope'][data['st_slope'] == 1] = 'upsloping'
data['st_slope'][data['st_slope'] == 2] = 'flat'
data['st_slope'][data['st_slope'] == 3] = 'downsloping'

data['thalassemia'][data['thalassemia'] == 1] = 'normal'
data['thalassemia'][data['thalassemia'] == 2] = 'fixed defect'
data['thalassemia'][data['thalassemia'] == 3] = 'reversable defect'
```

```
C:\Users\Dipsikha\anaconda3\lib\site-packages\ipykernel_launcher.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  """Entry point for launching an IPython kernel.
C:\Users\Dipsikha\anaconda3\lib\site-packages\ipykernel_launcher.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  after removing the cwd from sys.path.
C:\Users\Dipsikha\anaconda3\lib\site-packages\ipykernel_launcher.py:9: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  if __name__ == '__main__':
C:\Users\Dipsikha\anaconda3\lib\site-packages\ipykernel_launcher.py:10: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  # Remove the CWD from sys.path while we load stuff.
C:\Users\Dipsikha\anaconda3\lib\site-packages\ipykernel_launcher.py:12: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  if sys.path[0] == '':
C:\Users\Dipsikha\anaconda3\lib\site-packages\ipykernel_launcher.py:16: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  app.launch_new_instance()
C:\Users\Dipsikha\anaconda3\lib\site-packages\ipykernel_launcher.py:17: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
C:\Users\Dipsikha\anaconda3\lib\site-packages\ipykernel_launcher.py:19: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
C:\Users\Dipsikha\anaconda3\lib\site-packages\ipykernel_launcher.py:23: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
C:\Users\Dipsikha\anaconda3\lib\site-packages\ipykernel_launcher.py:24: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
C:\Users\Dipsikha\anaconda3\lib\site-packages\ipykernel_launcher.py:25: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
```

In [48]:

```
data['sex'] = data['sex'].astype('object')
data['chest_pain_type'] = data['chest_pain_type'].astype('object')
data['fasting_blood_sugar'] = data['fasting_blood_sugar'].astype('object')
data['rest_ecg'] = data['rest_ecg'].astype('object')
data['exercise_induced_angina'] = data['exercise_induced_angina'].astype('object')
data['st_slope'] = data['st_slope'].astype('object')
data['thalassemia'] = data['thalassemia'].astype('object')
```

In [53]:

```
data.dtypes
```

Out[53]:

```
age                          int64
sex                         object
chest_pain_type             object
resting_blood_pressure       int64
cholesterol                  int64
fasting_blood_sugar         object
rest_ecg                    object
max_heart_rate_achieved      int64
exercise_induced_angina     object
st_depression              float64
st_slope                    object
num_major_vessels            int64
thalassemia                 object
target                       int64
dtype: object
```

In [54]:

```
data.head()
```

Out[54]:

	age	sex	chest_pain_type	resting_blood_pressure	cholesterol	fasting_blood_sugar	rest_ecg	max_heart_rate_achieved	exerci
0	63	male	non-anginal pain	145	233	greater than 120mg/ml	normal	150	
1	37	male	atypical angina	130	250	lower than 120mg/ml	ST-T wave abnormality	187	
2	41	female	typical angina	130	204	lower than 120mg/ml	normal	172	
3	56	male	typical angina	120	236	lower than 120mg/ml	ST-T wave abnormality	178	
4	57	female	0	120	354	lower than 120mg/ml	ST-T wave abnormality	163	

In [55]:

```
data = pd.get_dummies(data,drop_first=True)
```

In [56]:

```
data.head()
```

Out[56]:

	age	resting_blood_pressure	cholesterol	max_heart_rate_achieved	st_depression	num_major_vessels	target	sex_male	chest_pair
0	63	145	233	150	2.3	0	1	1	
1	37	130	250	187	3.5	0	1	1	

	age	resting_blood_pressure	cholesterol	max_heart_rate_achieved	st_depression	num_major_vessels	target	sex_male	chest_pair
2	41	130	204	172	1.4	0	1	0	
3	56	120	236	178	0.8	0	1	1	
4	57	120	354	163	0.6	0	1	0	

In [57]:

```
####### splitting the dataset into dependent and independent variable #############

X = data.drop('target',axis=1)
Y = data['target']
```

In [58]:

```
####### splitting the set into training and testing ###########
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(X,Y,test_size=0.2,random_state=0)
```

In [59]:

```
######## getting shapes ############

print("Shape of x_train :", x_train.shape)
print("Shape of x_test :", x_test.shape)
print("Shape of y_train :", y_train.shape)
print("Shape of y_test :", y_test.shape)
```

```
Shape of x_train : (242, 19)
Shape of x_test : (61, 19)
Shape of y_train : (242,)
Shape of y_test : (61,)
```

In [60]:

```
############## model building ###############

from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report

model = RandomForestClassifier(n_estimators = 50,max_depth=5)
model.fit(x_train,y_train)
y_predict = model.predict(x_test)
y_pred_quant=model.predict_proba(x_test)[:,1]
y_pred = model.predict(x_test)
```

In [61]:

```
####### evaluating the model ##########

print('training accuracy:',model.score(x_train,y_train))
print('testing accuracy:',model.score(x_test,y_test))
```

```
training accuracy: 0.9421487603305785
testing accuracy: 0.8688524590163934
```

In [63]:

```
###### classification report #########

cp = classification_report(y_test,y_pred)
print(cp)
```

```
              precision    recall  f1-score   support

           0       0.85      0.85      0.85        27
           1       0.88      0.88      0.88        34
```

```
    accuracy                           0.87        61
   macro avg       0.87      0.87      0.87        61
weighted avg       0.87      0.87      0.87        61
```

In [ ]:

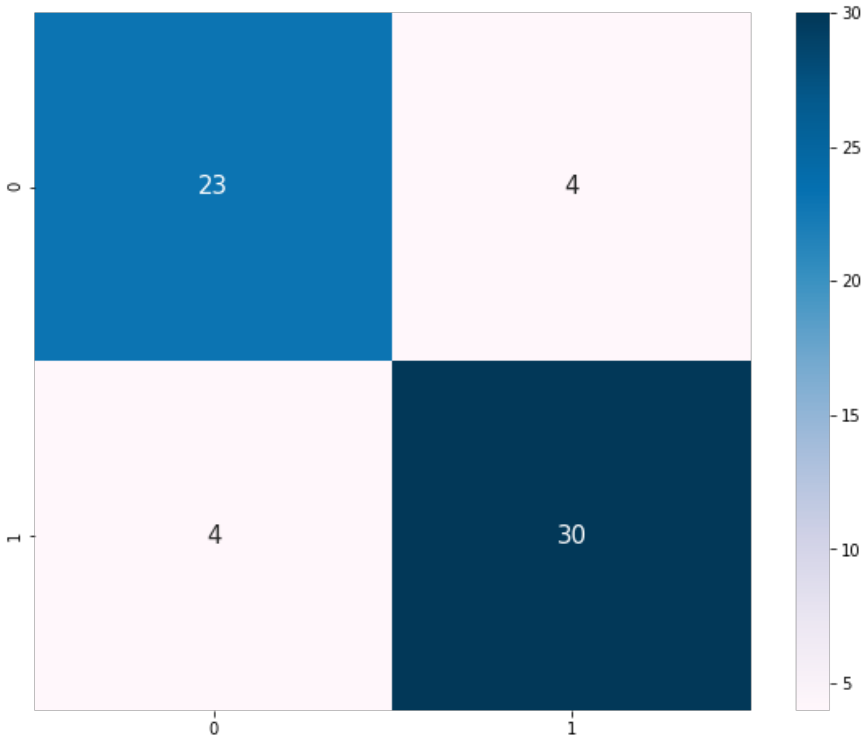
```
##### Precision quantifies the number of positive class predictions that actually belong to the po
sitive class(correctly predicted positive observation to the total predicted positive observation.
##### Recall quantifies the number of positive class predictions made out of all positive examples
in the dataset. true positives divided by the total number of true positives and false negatives.
#### F-Measure provides a single score that balances both the concerns of precision and recall in
one number.
######## as a measure of the quality of binary (two-class) classifications,
```

In [65]:

```
########## confusion matrix ###########

plt.subplots(figsize=(10,8))
cm=confusion_matrix(y_test,y_pred)
sns.heatmap(cm,annot=True,annot_kws={'size':15},cmap = 'PuBu')
```

Out[65]:

```
<AxesSubplot:>
```



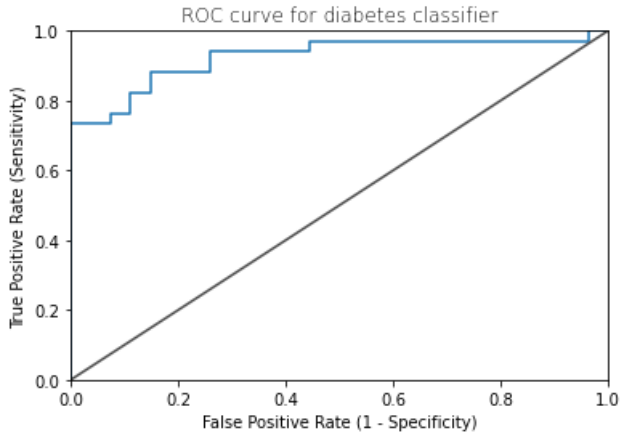
In [78]:

```
########## model explanation using graph ############

from sklearn.tree import export_graphviz
estimator = model.estimators_[1]
feature_names=[i for i in x_train.columns]

y_train_str = y_train.astype('str')
y_train_str[y_train_str == '0'] = 'no disease'
y_train_str[y_train_str == '1'] = 'disease'
y_train_str = y_train_str.values


export_graphviz(estimator, out_file='tree.dot',
                feature_names = feature_names,
                class_names = y_train_str,
                rounded = True, proportion = True,
```

```
                label='root',
                precision = 2, filled = True)
```

In [77]:

```
from sklearn.metrics import roc_curve
fpr,tpr,thresholds = roc_curve(y_test,y_pred_quant)
fig,ax = plt.subplots()
ax.plot(fpr,tpr)
ax.plot([0,1],[0,1], transform = ax.transAxes,ls="-",c = ".3")
plt.xlim([0.0,1.0])
plt.ylim([0.0,1.0])

plt.rcParams['figure.figsize'] = (15, 8)
plt.title('ROC curve for diabetes classifier', fontweight = 30)
plt.xlabel('False Positive Rate (1 - Specificity)')
plt.ylabel('True Positive Rate (Sensitivity)')
plt.show()
```



In [83]:

```
from sklearn import metrics
import xgboost as xgb
XGB = xgb.XGBClassifier(random_state = 1)
XGB.fit(x_train,y_train)
y_pred = XGB.predict(x_test)
acc_XGB = round(XGB.score(x_train,y_train)*100,2)
acc_XGB
print(classification_report(y_test,y_pred))
print("accuracy:",metrics.accuracy_score(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.75      0.89      0.81        27
           1       0.90      0.76      0.83        34

    accuracy                           0.82        61
   macro avg       0.82      0.83      0.82        61
weighted avg       0.83      0.82      0.82        61

accuracy: 0.819672131147541
```

In [85]:

```
###### decision tree #######

# Decision Tree
from sklearn.tree import DecisionTreeClassifier
decision_tree = DecisionTreeClassifier()
decision_tree.fit(x_train, y_train)
# predictions
yPred = decision_tree.predict(x_test)
acc_decision_tree = round(decision_tree.score(x_train,y_train)*100,2)
acc_decision_tree
```

Out[85]:

```
100.0
```

In [87]:

```
# K Nearest Neighbor:
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors = 3)
knn.fit(x_train, y_train)
y_pred = knn.predict(x_test)
acc_knn = round(knn.score(x_train, y_train) * 100, 2)
acc_knn
print(classification_report(y_test, y_pred))
print("accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.59      0.63      0.61        27
           1       0.69      0.65      0.67        34

    accuracy                           0.64        61
   macro avg       0.64      0.64      0.64        61
weighted avg       0.64      0.64      0.64        61

accuracy: 0.639344262295082
```

In [91]:

```
from sklearn.naive_bayes import GaussianNB
gaussian = GaussianNB()
gaussian.fit(x_train, y_train)
y_pred = gaussian.predict(x_test)
acc_gaussian = round(gaussian.score(x_train, y_train) * 100, 2)
acc_gaussian
print(classification_report(y_test, y_pred))
print("accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.83      0.89      0.86        27
           1       0.91      0.85      0.88        34

    accuracy                           0.87        61
   macro avg       0.87      0.87      0.87        61
weighted avg       0.87      0.87      0.87        61

accuracy: 0.8688524590163934
```

In [ ]:

In [ ]:

In [ ]:

In [ ]: