In [7]:
```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
from pandas import read_csv
```

In [8]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

In [9]:
```
filename = 'heart_disease_uci.csv'
df = read_csv(filename)

df.sample(5)
```

Out[9]:

	id	age	sex	dataset	cp	trestbps	chol	fbs	restecg	thalch	exang
49	50	53	Male	Cleveland	non-anginal	130.0	197.0	True	lv hypertrophy	152.0	False
522	523	57	Male	Hungary	atypical angina	140.0	265.0	False	st-t abnormality	145.0	True
64	65	54	Male	Cleveland	asymptomatic	120.0	188.0	False	normal	113.0	False
880	881	62	Male	VA Long Beach	asymptomatic	NaN	170.0	False	st-t abnormality	120.0	True
889	890	57	Male	VA Long Beach	atypical angina	180.0	285.0	True	st-t abnormality	120.0	False

In [10]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 920 entries, 0 to 919
Data columns (total 16 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   id        920 non-null    int64  
 1   age       920 non-null    int64  
 2   sex       920 non-null    object 
 3   dataset   920 non-null    object 
 4   cp        920 non-null    object 
 5   trestbps  861 non-null    float64
 6   chol      890 non-null    float64
 7   fbs       830 non-null    object 
 8   restecg   918 non-null    object 
 9   thalch    865 non-null    float64
 10  exang     865 non-null    object 
 11  oldpeak   858 non-null    float64
 12  slope     611 non-null    object 
 13  ca        309 non-null    float64
 14  thal      434 non-null    object 
 15  num       920 non-null    int64  
dtypes: float64(5), int64(3), object(8)
memory usage: 115.1+ KB
```

In [11]:
```
df.duplicated().sum()
```

Out[11]:
```
0
```

In [12]:
```
df.isna().sum()
```

Out[12]:
```
id            0
age           0
sex           0
dataset       0
cp            0
trestbps     59
chol         30
fbs          90
restecg       2
thalch       55
exang        55
oldpeak      62
slope       309
ca          611
thal        486
num           0
dtype: int64
```

In [13]:
```
#dropping null values
df.dropna(inplace=True)
df.isnull().sum()
```

Out[13]:
```
id          0
age         0
sex         0
dataset     0
cp          0
trestbps    0
chol        0
fbs         0
restecg     0
thalch      0
exang       0
oldpeak     0
slope       0
ca          0
thal        0
num         0
dtype: int64
```

In [14]:
```
# Converting DataTypes Of Features
```

In [15]:
```
df.sex.value_counts()
```

Out[15]:
```
Male      203
Female     96
Name: sex, dtype: int64
```

In [16]:
```
df['sex']=df['sex'].apply(lambda x:0 if x=='Male' else 1)
df['sex'].value_counts()
```

Out[16]:
```
0    203
1     96
Name: sex, dtype: int64
```

In [17]:
```
df['dataset'].value_counts()
```

Out[17]:
```
Cleveland        297
Hungary            1
VA Long Beach      1
Name: dataset, dtype: int64
```

In [18]:
```
df['dataset']=df['dataset'].apply(lambda x:0 if x== 'Cleveland' else 1 if x=='Hung
df['dataset'].value_counts()
```

Out[18]:
```
0    297
1      1
2      1
Name: dataset, dtype: int64
```

In [19]:
```
df['cp'].value_counts()
```

Out[19]:
```
asymptomatic       144
non-anginal         83
atypical angina     49
typical angina      23
Name: cp, dtype: int64
```

In [20]:
```
df['cp']=df['cp'].apply(lambda x:0 if x== 'typical angina' else 1 if x=='asymptoma
df['cp'].value_counts()
```

Out[20]:
```
1    144
2     83
3     49
0     23
Name: cp, dtype: int64
```

In [21]:
```
df['fbs'].value_counts()
```

Out[21]:
```
False    256
True      43
Name: fbs, dtype: int64
```

In [22]:
```
df['fbs']=pd.get_dummies(df['fbs'],drop_first=True)
```

In [23]:
```
df['exang'].value_counts()
```

Out[23]:
```
False    200
True      99
Name: exang, dtype: int64
```

In [24]:
```
df['exang']=pd.get_dummies(df['exang'],drop_first=True)
```

In [25]:
```
df['restecg'].value_counts()
```

Out[25]:
```
normal              149
lv hypertrophy      146
st-t abnormality      4
Name: restecg, dtype: int64
```

In [26]:
```
df['restecg']=df['restecg'].apply(lambda x:0 if x== 'lv hypertrophy' else 1 if x==
df['restecg'].value_counts()
```

Out[26]:
```
1    149
0    146
2      4
Name: restecg, dtype: int64
```

In [27]:
```
df['slope'].value_counts()
```

Out[27]:
```
flat           139
upsloping      139
downsloping     21
Name: slope, dtype: int64
```

In [28]:
```
df['slope']=df['slope'].apply(lambda x:0 if x== 'downsloping' else 1 if x=='flat'e
df['slope'].value_counts()
```

Out[28]:
```
1    139
2    139
0     21
Name: slope, dtype: int64
```

In [29]:
```
df['thal'].value_counts()
```

Out[29]:
```
normal               164
reversable defect    117
fixed defect          18
Name: thal, dtype: int64
```

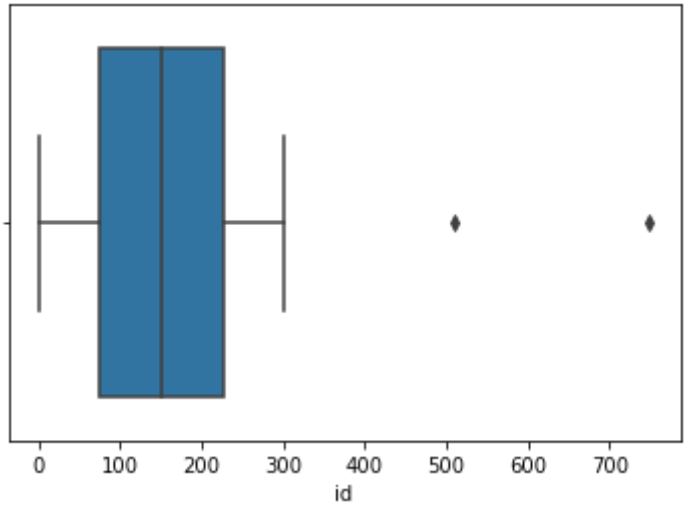
In [30]:
```
df['thal']=df['thal'].apply(lambda x:0 if x== 'fixed defect' else 1 if x=='normal'
df['thal'].value_counts()
```

Out[30]:
```
1    164
2    117
0     18
Name: thal, dtype: int64
```

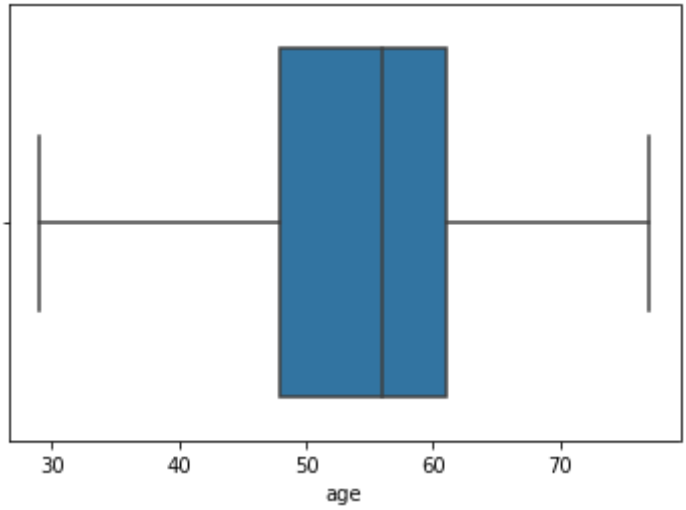
In [31]:
```
#**Treating Outliers**
```

In [32]:
```
for i in df.iloc[0:,:-1].columns:
    sns.boxplot(df[i],data=df)
    print(i)
    plt.show()
```

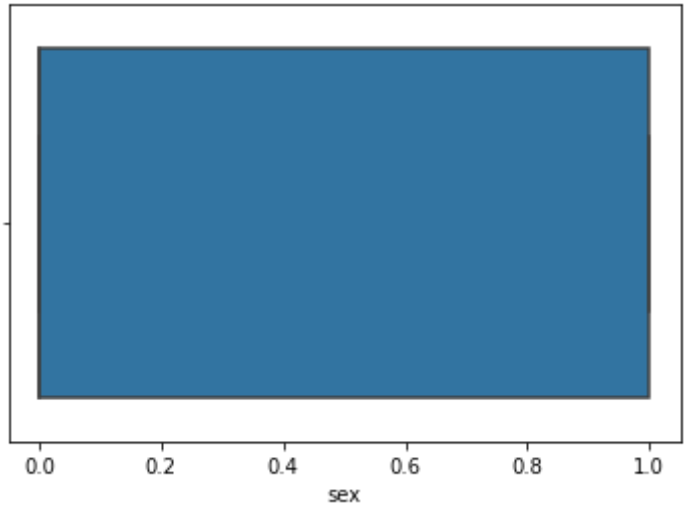
id



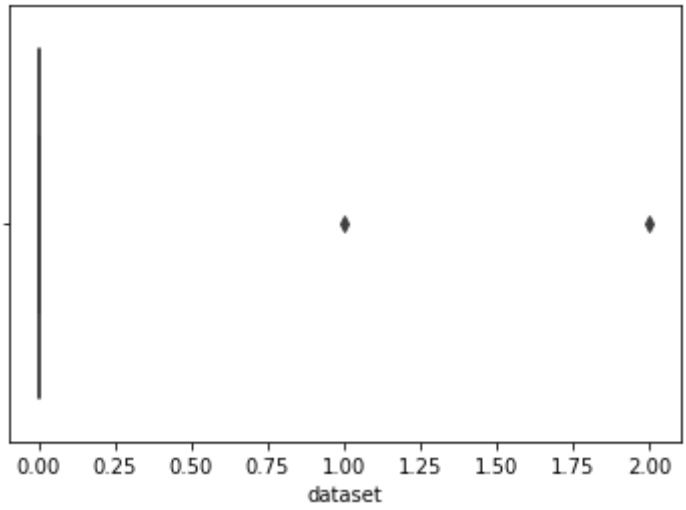
age



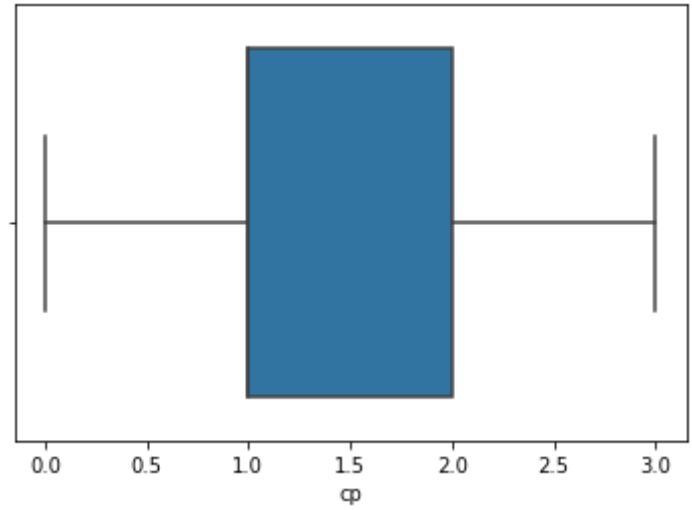
sex



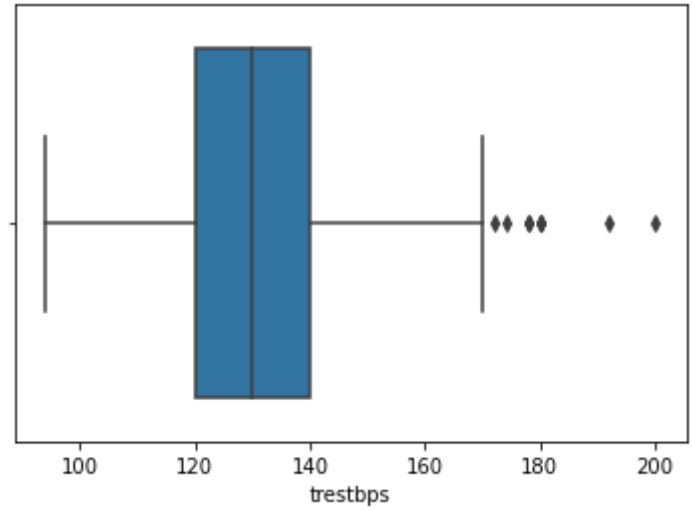
dataset



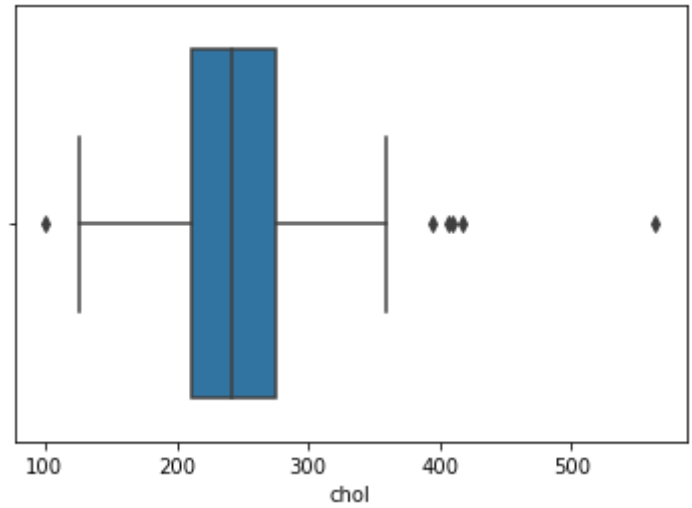
cp



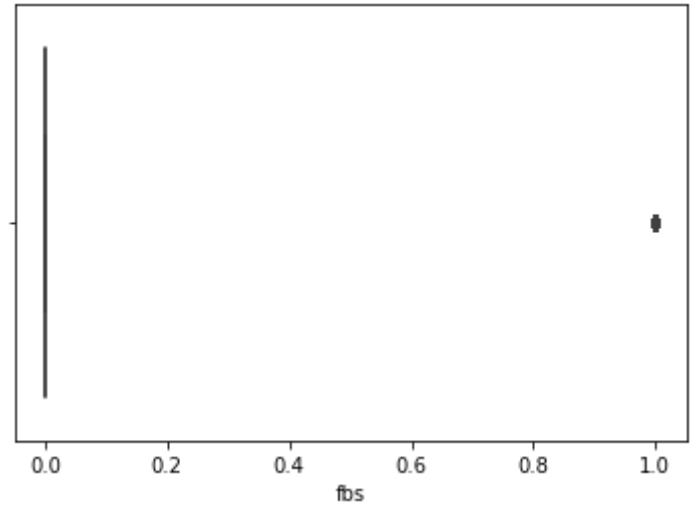
trestbps



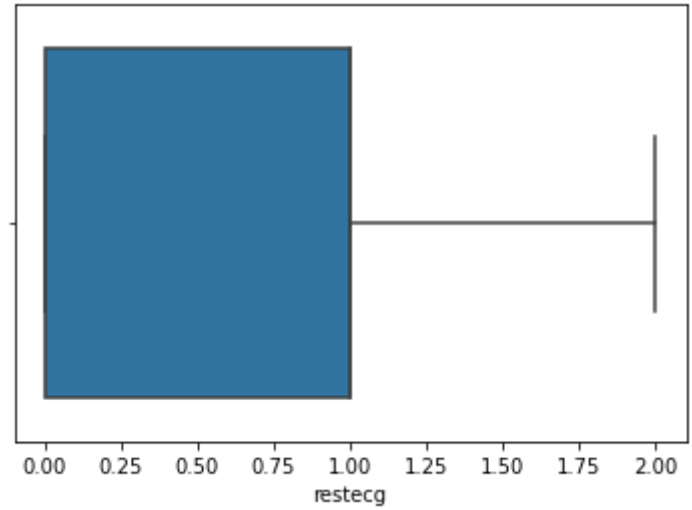
chol



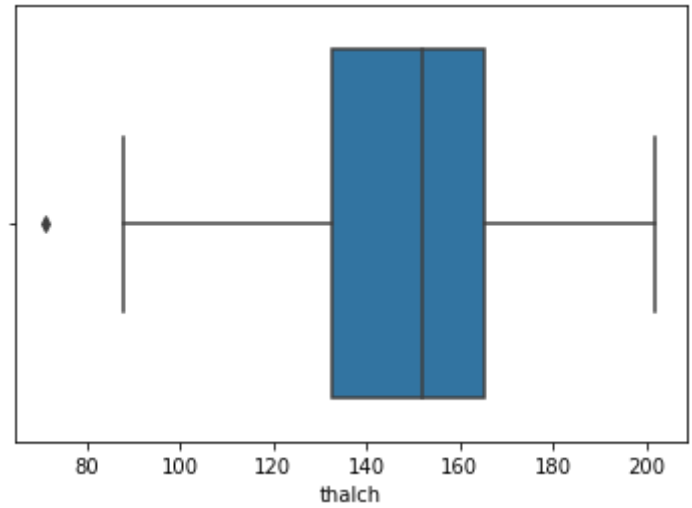
fbs



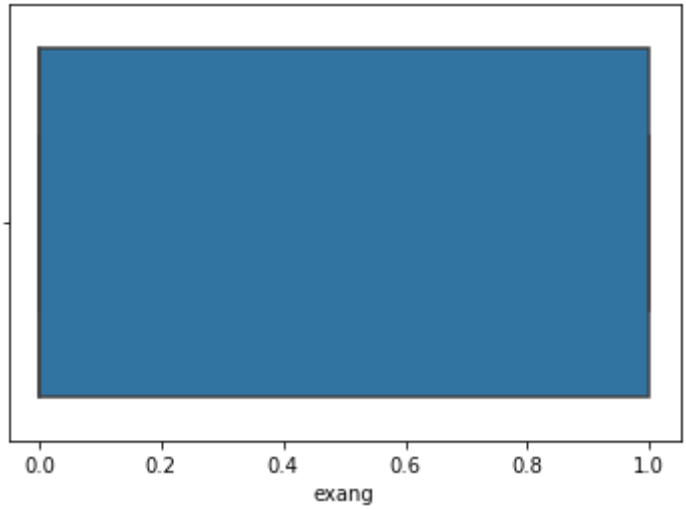
restecg



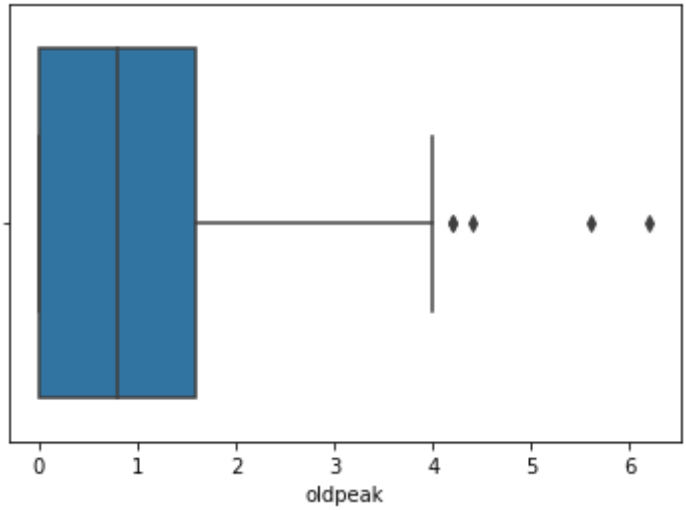
thalch



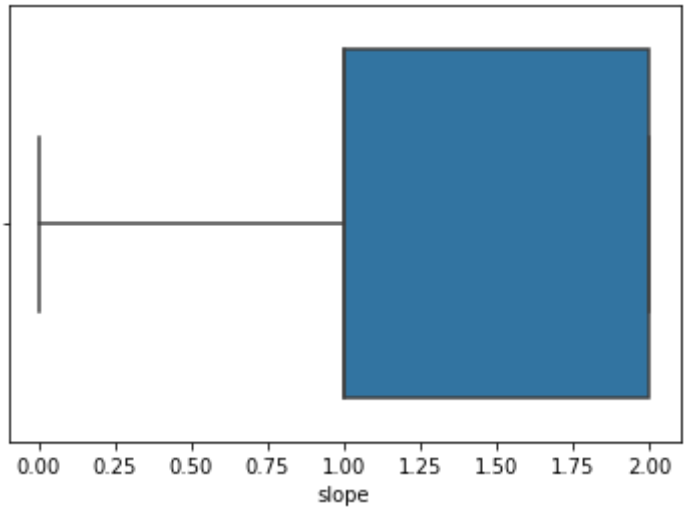
exang



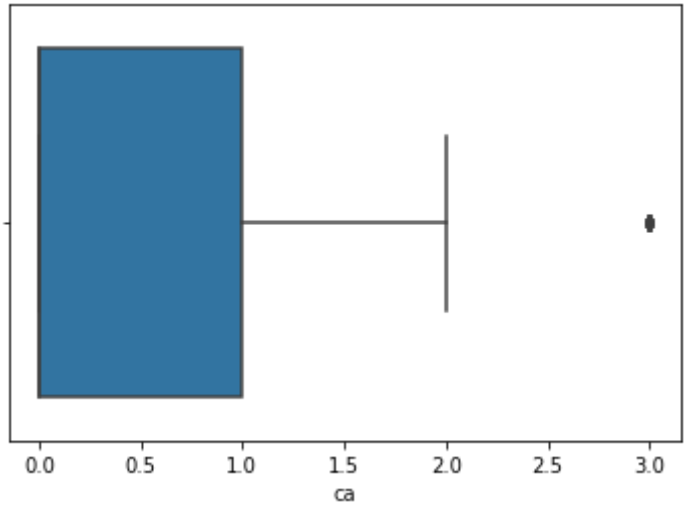
oldpeak



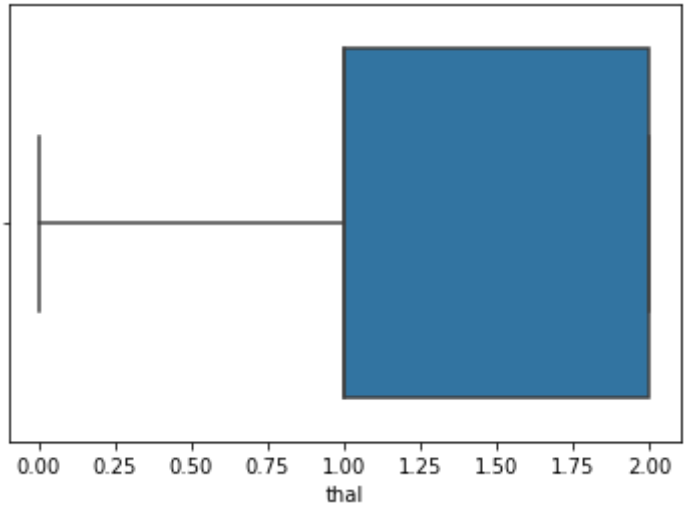
slope



ca



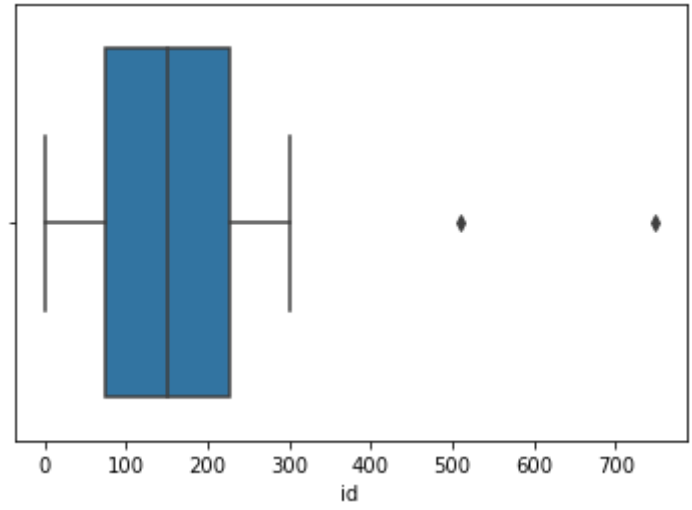
thal



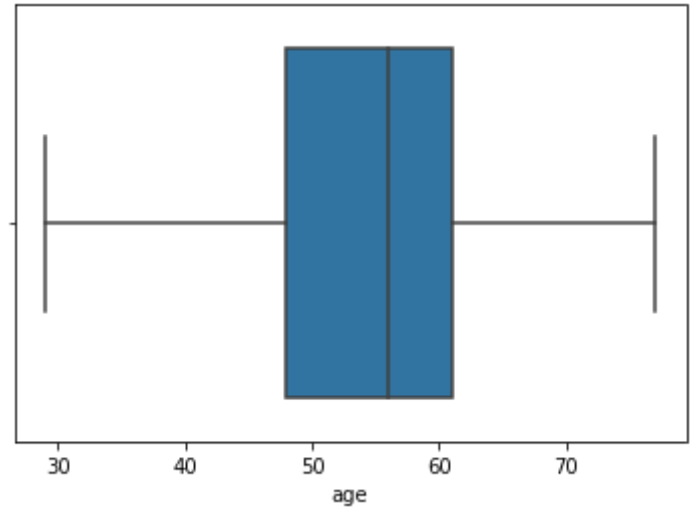
In [33]:
```
l=[5,6,9,11,13]
def outlier(df):
    for x in df.iloc[:,l].columns :
        Q1=df[x].quantile(0.25)
        Q3=df[x].quantile(0.75)
        IQR=Q3-Q1
        Lower = Q1-(1.5*IQR)
        Upper = Q3+(1.5*IQR)
        df.loc[:,x]= np.where(df[x].values > Upper, Upper-1, df[x].values)
        df.loc[:,x]= np.where(df[x].values < Lower, Lower+1, df[x].values)

outlier(df)
```

In [34]:
```
for i in df.iloc[0:,:-1].columns:
    sns.boxplot(df[i],data=df)
    print(i)
    plt.show()
```

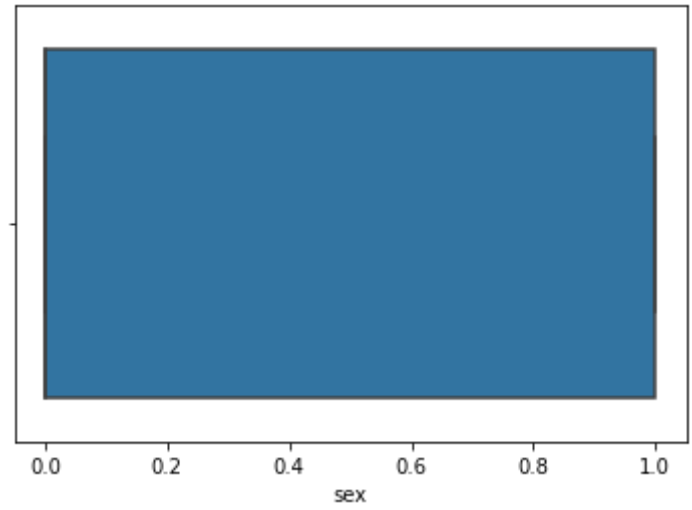
id



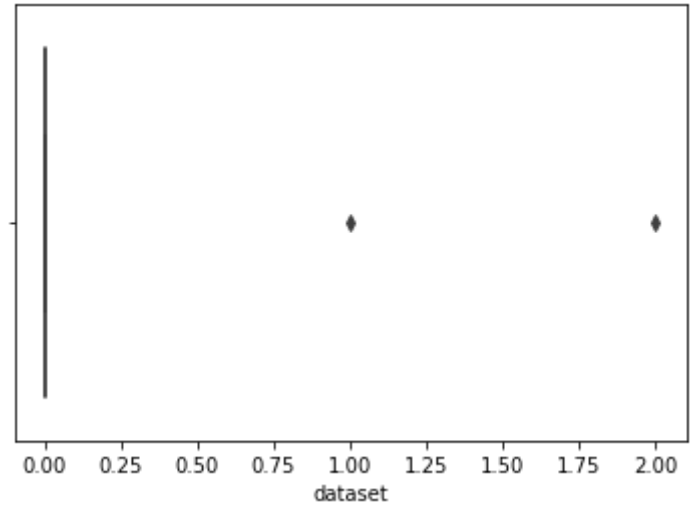
age



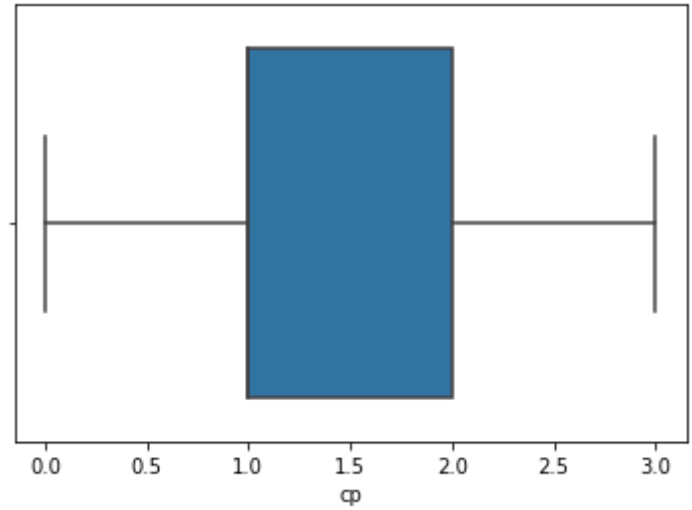
sex



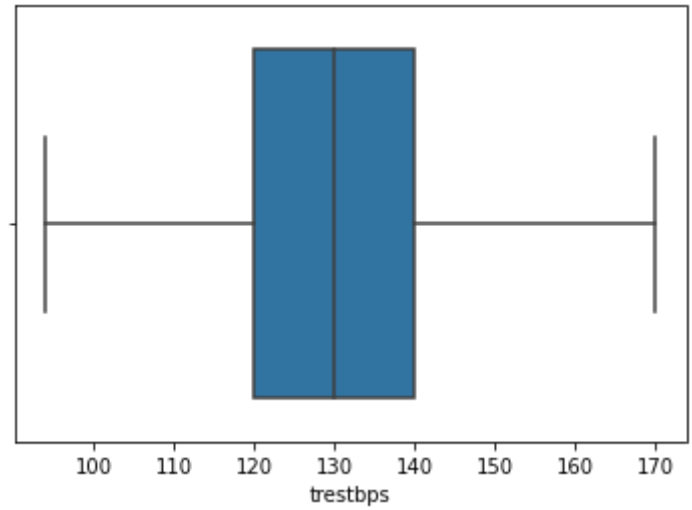
dataset



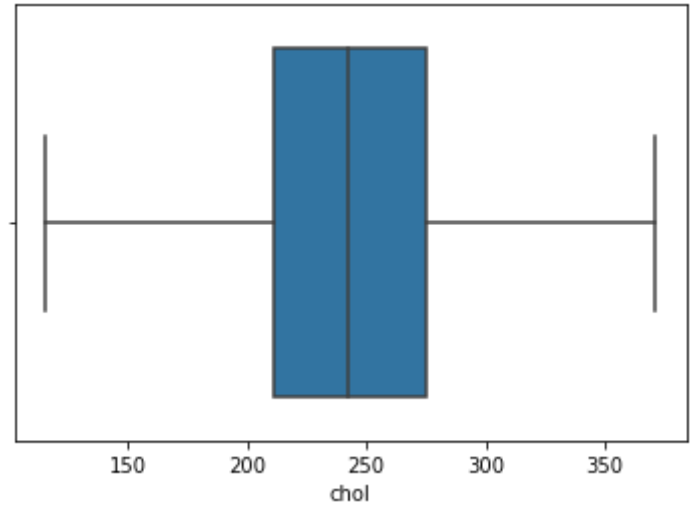
cp



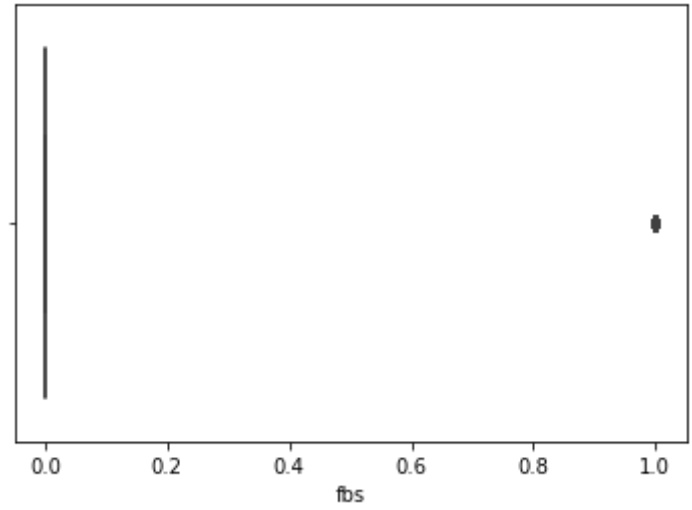
trestbps



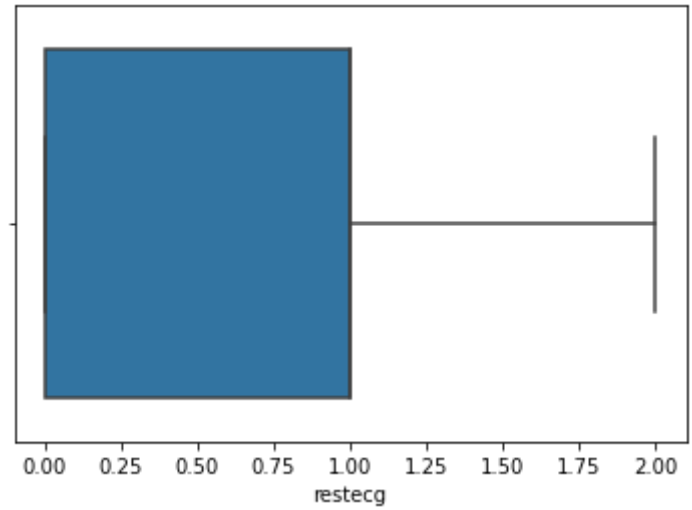
chol



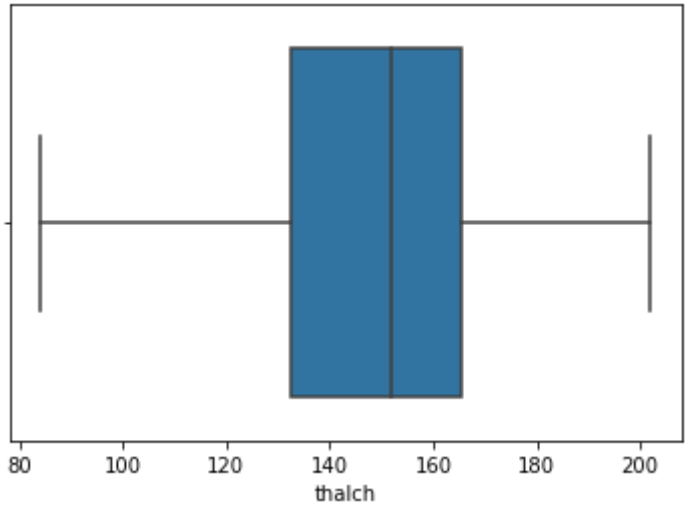
fbs



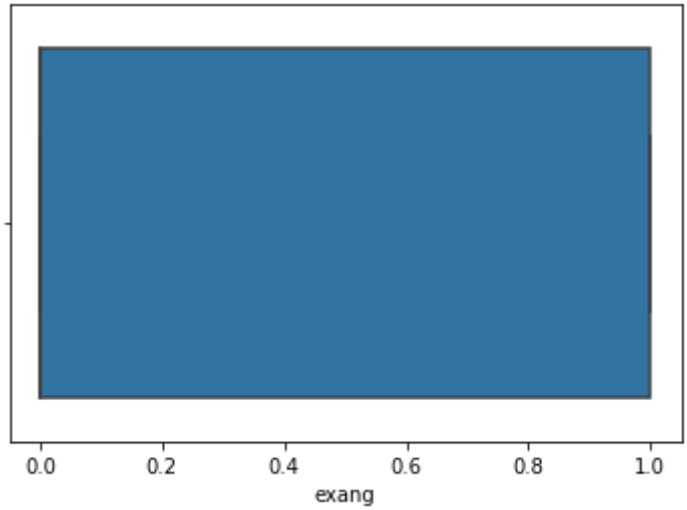
restecg



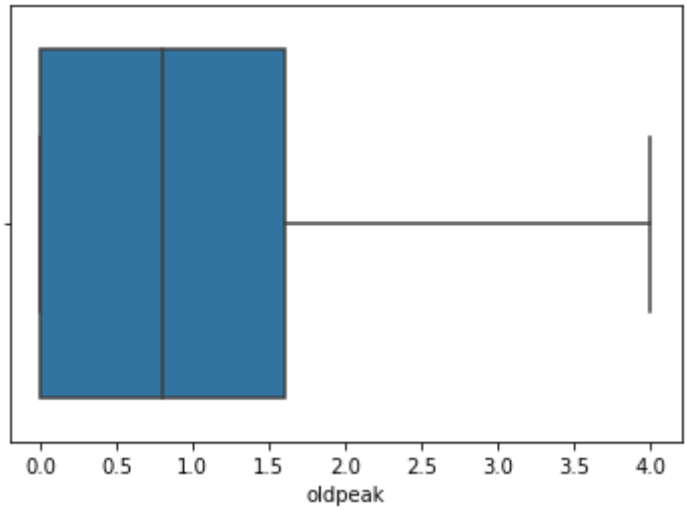
thalch



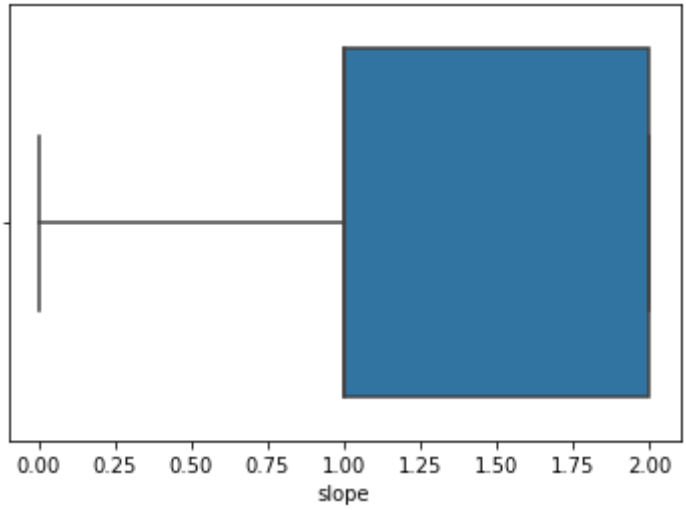
exang



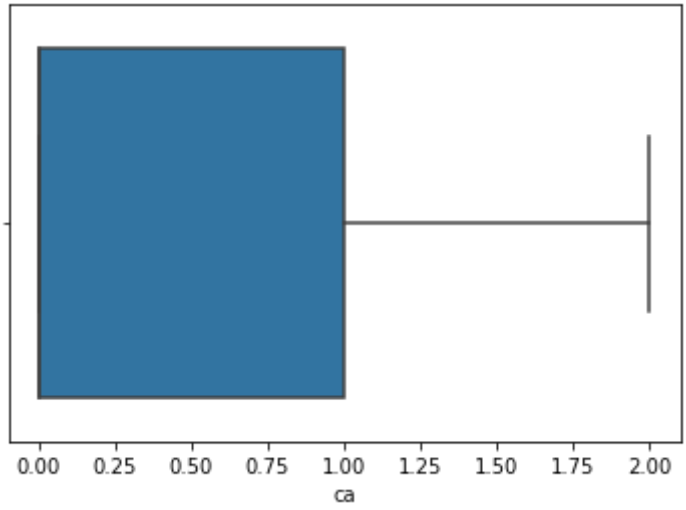
oldpeak



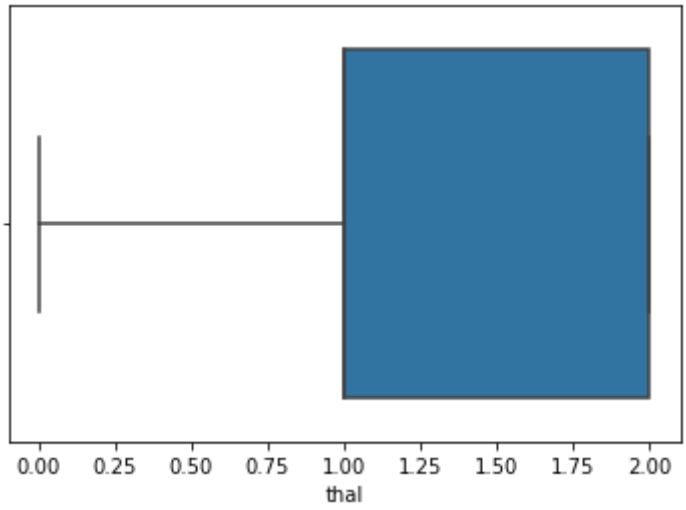
slope



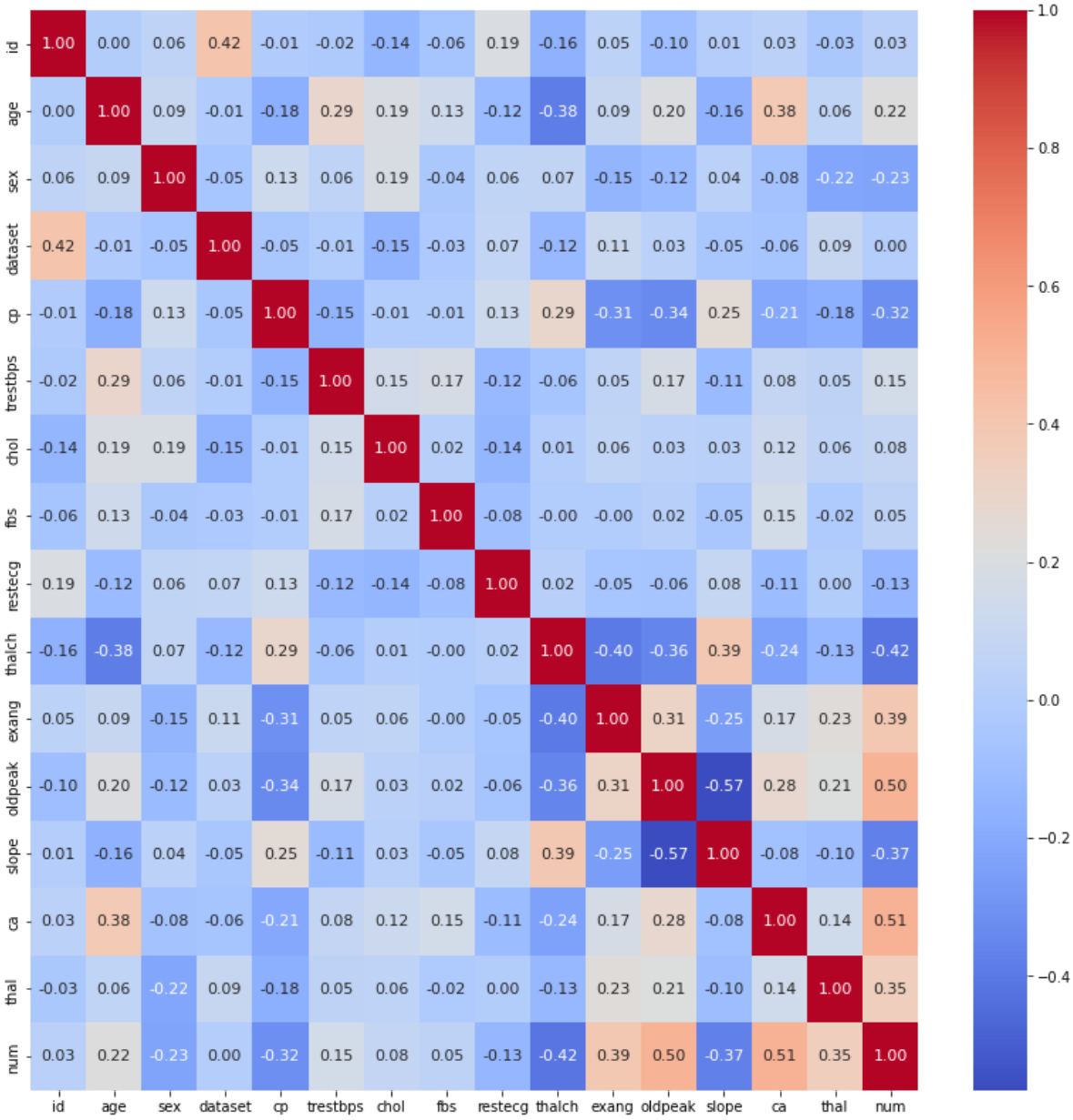
ca



thal



In [35]:
```
# heatmap

corr = df.corr()
plt.figure(figsize=(14,14))
sns.heatmap(corr, annot=True, fmt= '.2f',annot_kws={'size': 11}, cmap= 'coolwarm')
plt.show()
print(corr)
```



In [36]:
```
df.corr()['num']
```

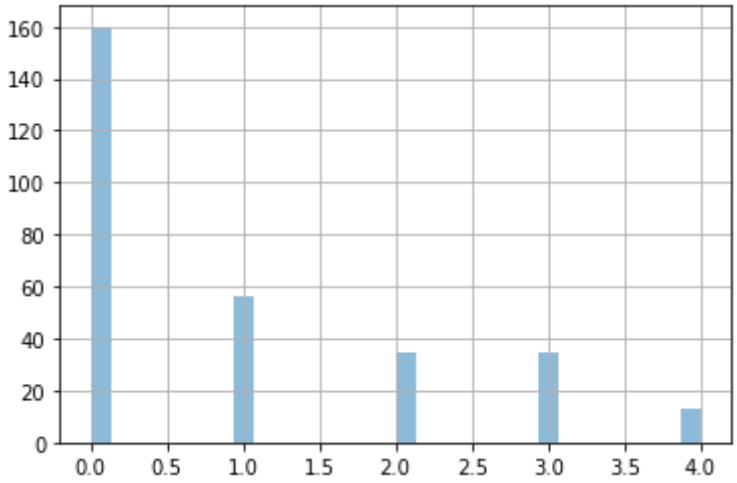
	id	age	sex	dataset	cp	trestbps
id	1.000000	0.001379	0.056881	0.418747	-0.013401	-0.024613
age	0.001379	1.000000	0.093693	-0.013136	-0.175512	0.293668
sex	0.056881	0.093693	1.000000	-0.053518	0.127962	0.056022
dataset	0.418747	-0.013136	-0.053518	1.000000	-0.048106	-0.006236
cp	-0.013401	-0.175512	0.127962	-0.048106	1.000000	-0.152634
trestbps	-0.024613	0.293668	0.056022	-0.006236	-0.152634	1.000000
chol	-0.141215	0.191070	0.186183	-0.151336	-0.009014	0.145120
fbs	-0.060584	0.132752	-0.036869	-0.031895	-0.008054	0.169248
restecg	0.192594	-0.115722	0.062598	0.070336	0.126842	-0.123906
thalch	-0.158837	-0.383826	0.066918	-0.124484	0.285467	-0.061313
exang	0.045102	0.092506	-0.148956	0.110614	-0.310202	0.054722
oldpeak	-0.098590	0.201406	-0.117872	0.034387	-0.338644	0.172363
slope	0.007997	-0.157545	0.036213	-0.049865	0.251822	-0.113459
ca	0.034542	0.377351	-0.076393	-0.060059	-0.205159	0.077103
thal	-0.034021	0.059228	-0.217901	0.089030	-0.176072	0.048840
num	0.031397	0.221787	-0.226636	0.003390	-0.317051	0.153360

	chol	fbs	restecg	thalch	exang	oldpeak
id	-0.141215	-0.060584	0.192594	-0.158837	0.045102	-0.098590
age	0.191070	0.132752	-0.115722	-0.383826	0.092506	0.201406
sex	0.186183	-0.036869	0.062598	0.066918	-0.148956	-0.117872
dataset	-0.151336	-0.031895	0.070336	-0.124484	0.110614	0.034387
cp	-0.009014	-0.008054	0.126842	0.285467	-0.310202	-0.338644
trestbps	0.145120	0.169248	-0.123906	-0.061313	0.054722	0.172363
chol	1.000000	0.018505	-0.142140	0.006227	0.059012	0.034726
fbs	0.018505	1.000000	-0.083047	-0.003723	-0.004809	0.020132
restecg	-0.142140	-0.083047	1.000000	0.019060	-0.053871	-0.063231
thalch	0.006227	-0.003723	0.019060	1.000000	-0.397895	-0.356900
exang	0.059012	-0.004809	-0.053871	-0.397895	1.000000	0.310870
oldpeak	0.034726	0.020132	-0.063231	-0.356900	0.310870	1.000000
slope	0.026479	-0.045958	0.083082	0.393096	-0.254650	-0.565984
ca	0.116040	0.153154	-0.109841	-0.240532	0.171478	0.283937
thal	0.055166	-0.020172	0.000182	-0.126320	0.233616	0.213387
num	0.076560	0.048892	-0.132639	-0.417319	0.389355	0.498471

	slope	ca	thal	num
id	0.007997	0.034542	-0.034021	0.031397
age	-0.157545	0.377351	0.059228	0.221787
sex	0.036213	-0.076393	-0.217901	-0.226636
dataset	-0.049865	-0.060059	0.089030	0.003390
cp	0.251822	-0.205159	-0.176072	-0.317051
trestbps	-0.113459	0.077103	0.048840	0.153360
chol	0.026479	0.116040	0.055166	0.076560
fbs	-0.045958	0.153154	-0.020172	0.048892
restecg	0.083082	-0.109841	0.000182	-0.132639
thalch	0.393096	-0.240532	-0.126320	-0.417319
exang	-0.254650	0.171478	0.233616	0.389355
oldpeak	-0.565984	0.283937	0.213387	0.498471
slope	1.000000	-0.080490	-0.102807	-0.374357
ca	-0.080490	1.000000	0.141878	0.507578
thal	-0.102807	0.141878	1.000000	0.350634
num	-0.374357	0.507578	0.350634	1.000000

Out[36]:
```
id          0.031397
age         0.221787
sex        -0.226636
dataset     0.003390
cp         -0.317051
trestbps    0.153360
chol        0.076560
fbs         0.048892
restecg    -0.132639
thalch     -0.417319
exang       0.389355
oldpeak     0.498471
slope      -0.374357
ca          0.507578
thal        0.350634
num         1.000000
Name: num, dtype: float64
```

In [37]:
```
# Plot a histogram

df.num.hist(bins=30, alpha=0.5)
plt.show()
```



In [38]:
```
df['num'].unique()
```

Out[38]:
```
array([0, 2, 1, 3, 4], dtype=int64)
```

In [39]:
```
#Splitting Data
```

In [40]:
```
from sklearn.model_selection import train_test_split
```

In [41]:
```
X=df.drop(['num','id'],axis=1)
y=df['num']

X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state=2)
```

In [42]:
```
#Building A Model
```

In [43]:
```
from sklearn.tree import DecisionTreeClassifier

clf=DecisionTreeClassifier(criterion= 'entropy', max_depth= 10, max_features= 10, r
```

In [44]:
```
#Scaling the data

from sklearn.preprocessing import MinMaxScaler
```

```
min_max=MinMaxScaler()

X_train_transformed=min_max.fit_transform(X_train)

X_test_transformed=min_max.transform(X_test)
```

In [45]:
```
clf.fit(X_train_transformed,y_train)
```

Out[45]:
```
DecisionTreeClassifier(criterion='entropy', max_depth=10, max_features=10,
                       min_samples_leaf=12, min_samples_split=25)
```

In [46]:
```
pred=clf.predict(X_test_transformed)
pred
```

Out[46]:
```
array([3, 0, 1, 0, 3, 0, 0, 3, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 2, 0, 0, 0,
       1, 0, 0, 0, 0, 2, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0,
       0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 3, 3, 2, 0, 3, 0, 0, 0, 0, 0, 1, 0,
       0, 1, 1, 0, 2, 0, 0, 0, 1, 0, 0, 0, 2, 1, 1, 1, 0, 0, 0, 0, 0, 0,
       0, 0], dtype=int64)
```

In [47]:
```
train_pred=clf.predict(X_train_transformed)
train_pred
```

Out[47]:
```
array([1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 3, 0, 0, 1, 3, 0, 0, 0, 1, 0, 0, 1,
       2, 0, 0, 2, 1, 0, 0, 0, 0, 0, 3, 0, 1, 0, 0, 0, 0, 0, 1, 2, 0, 0,
       1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 3, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0,
       1, 0, 2, 2, 0, 0, 0, 0, 3, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0,
       0, 1, 2, 2, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 2, 3, 0, 0, 0, 0,
       0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 3, 0, 0, 0, 0, 0,
       0, 1, 3, 1, 0, 0, 1, 3, 1, 0, 0, 0, 0, 0, 0, 1, 1, 3, 0, 2, 0, 2,
       0, 0, 0, 0, 0, 0, 3, 0, 0, 0, 1, 0, 1, 1, 1, 0, 3, 3, 0, 0, 0, 0,
       0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       1, 0, 2, 2, 0, 0, 0, 0, 2, 0, 3], dtype=int64)
```

In [48]:
```
#Evaluating The Model
```

In [49]:
```
from sklearn.metrics import accuracy_score
```

In [50]:
```
accuracy_score(y_test,pred)
```

Out[50]:
```
0.5888888888888889
```

In [51]:
```
accuracy_score(y_train,train_pred)
```

Out[51]:
```
0.6411483253588517
```

In [52]:
```
from sklearn import metrics
```

In [53]:
```
print(metrics.classification_report(y_test,pred))
```

	precision	recall	f1-score	support
0	0.75	0.90	0.82	51
1	0.28	0.31	0.29	16
2	0.40	0.25	0.31	8
3	0.00	0.00	0.00	10
4	0.00	0.00	0.00	5
accuracy			0.59	90
macro avg	0.29	0.29	0.28	90
weighted avg	0.51	0.59	0.55	90

In [54]:
```
print(metrics.classification_report(y_train,train_pred))
```

	precision	recall	f1-score	support
0	0.74	0.92	0.82	109
1	0.44	0.50	0.47	40
2	0.46	0.22	0.30	27
3	0.53	0.32	0.40	25
4	0.00	0.00	0.00	8
accuracy			0.64	209
macro avg	0.43	0.39	0.40	209
weighted avg	0.59	0.64	0.60	209

In [ ]:
```

```