```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
data=pd.read_csv("/content/sample_data/heart.csv")
```

```
data.head()
```

	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

```
data.shape
```

```
(1025, 14)
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1025 entries, 0 to 1024
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       1025 non-null   int64
 1   sex       1025 non-null   int64
 2   cp        1025 non-null   int64
 3   trestbps  1025 non-null   int64
 4   chol      1025 non-null   int64
 5   fbs       1025 non-null   int64
 6   restecg   1025 non-null   int64
 7   thalach   1025 non-null   int64
 8   exang     1025 non-null   int64
 9   oldpeak   1025 non-null   float64
 10  slope     1025 non-null   int64
 11  ca        1025 non-null   int64
 12  thal      1025 non-null   int64
 13  target    1025 non-null   int64
dtypes: float64(1), int64(13)
memory usage: 112.2 KB
```

```
data.isna().sum()
```

	0
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

```
data.duplicated().sum()
```

```
723
```

```
data=data.drop_duplicates(keep="first")
```

```
data.duplicated().sum()
```

```
0
```

```
data.shape
```

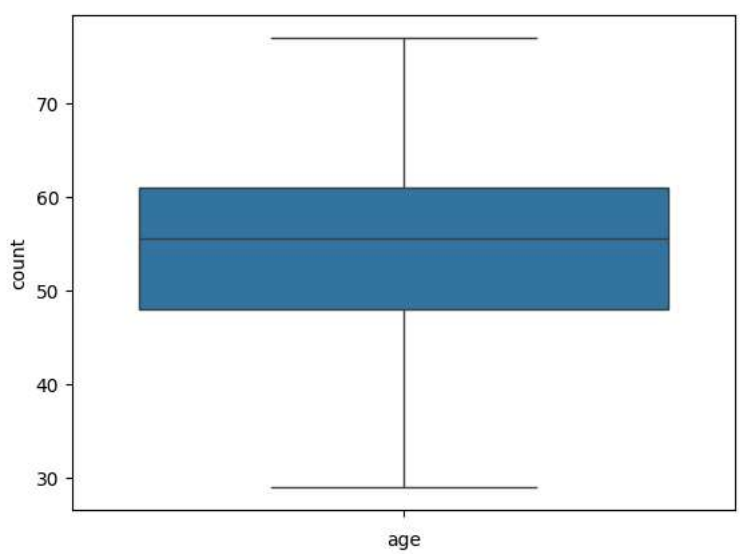
```
(302, 14)
```

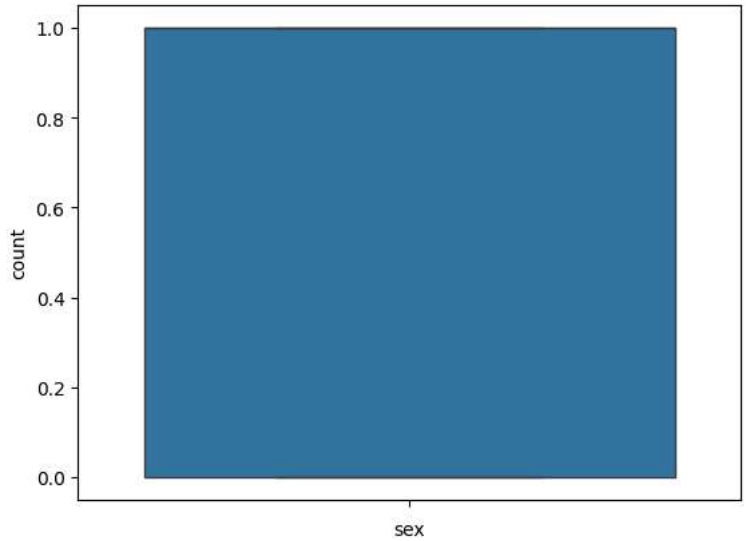
```
data.info()
```

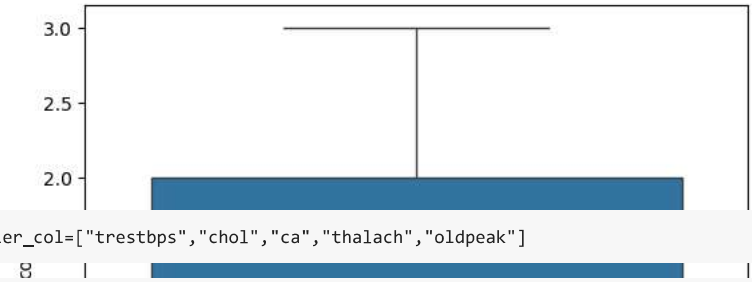
```
<class 'pandas.core.frame.DataFrame'>
Index: 302 entries, 0 to 878
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       302 non-null    int64
 1   sex       302 non-null    int64
 2   cp        302 non-null    int64
 3   trestbps  302 non-null    int64
 4   chol      302 non-null    int64
 5   fbs       302 non-null    int64
 6   restecg   302 non-null    int64
 7   thalach   302 non-null    int64
 8   exang     302 non-null    int64
 9   oldpeak   302 non-null    float64
 10  slope     302 non-null    int64
 11  ca        302 non-null    int64
 12  thal      302 non-null    int64
 13  target    302 non-null    int64
dtypes: float64(1), int64(13)
memory usage: 35.4 KB
```

```
#checking outlier
```

```
for i in data.columns:
  if ((data[i].dtypes !="object") & (i !="target")):
     sns.boxplot(data[i])
     plt.xlabel(i)
     plt.ylabel("count")
     plt.show()
```







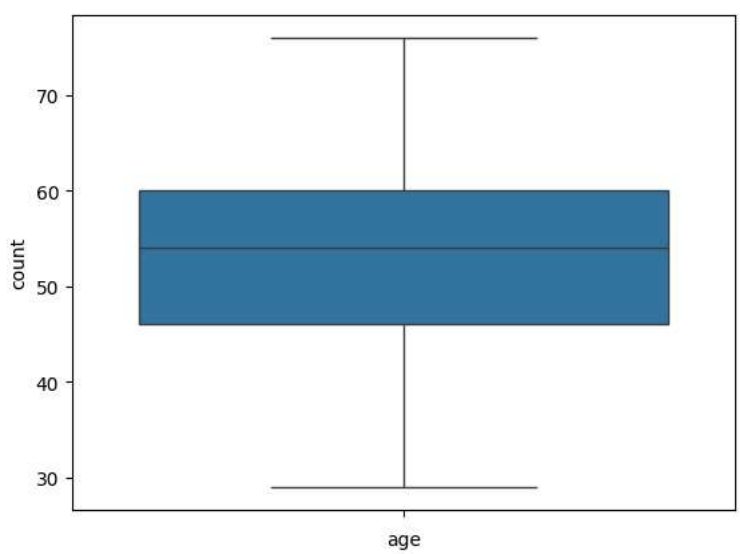

```
outlier_col=["trestbps","chol","ca","thalach","oldpeak"]
```

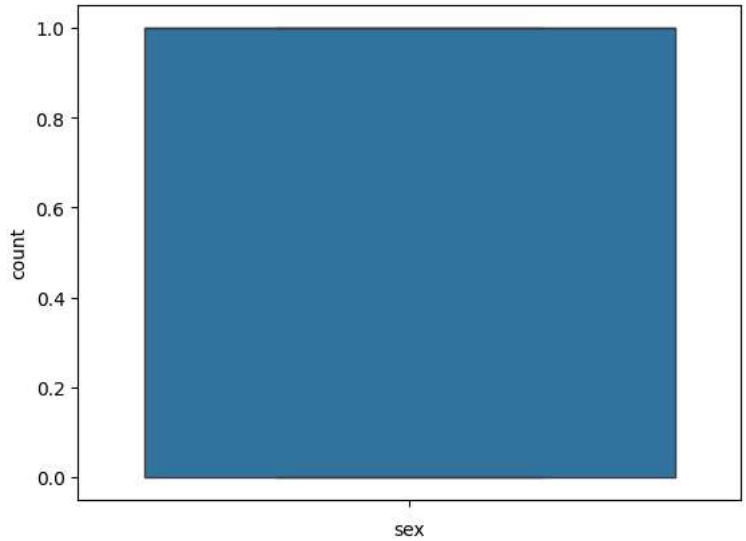
```
outlier_col
```

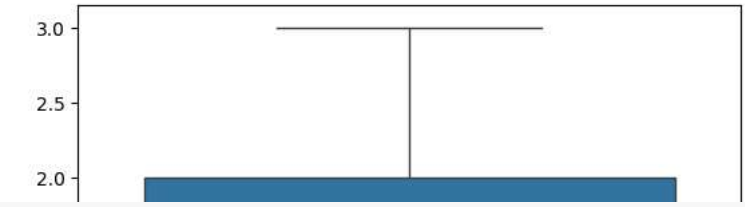
```
['trestbps', 'chol', 'ca', 'thalach', 'oldpeak']
```

```
for i in outlier_col:
  q1=data[i].quantile(0.25)
  q3=data[i].quantile(0.75)
  iqr=q3-q1
  lower=q1-1.5*iqr
  upper=q3+1.5*iqr
  data=data[(data[i]>=lower) & (data[i]<=upper)]
```

```
for i in data.columns:
  if ((data[i].dtypes !="object") & (i !="target")):
     sns.boxplot(data[i])
     plt.xlabel(i)
     plt.ylabel("count")
     plt.show()
```








```
data.shape
```

```
(263, 14)
```

```
for i in data.columns:
  if ((data[i].dtypes !="object") & (i !="target")):
    sns.distplot(data[i])
    plt.show()
```

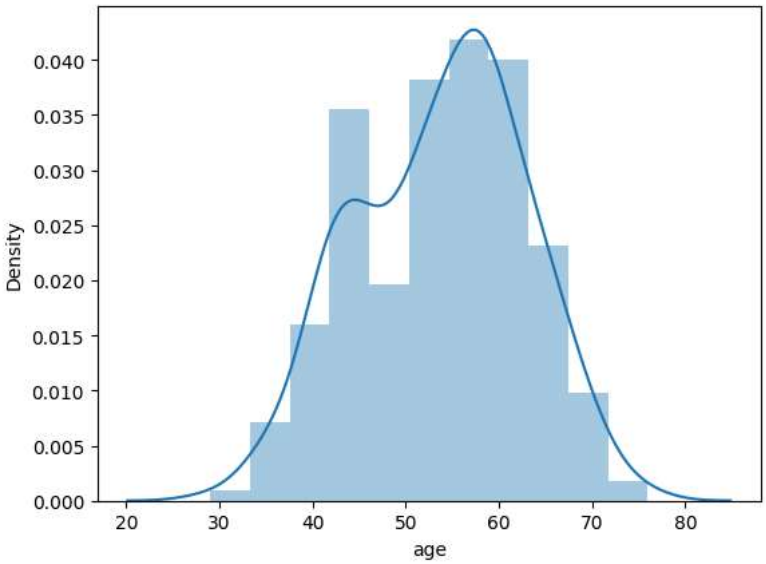
```
<ipython-input-31-b8b1b1ba79af>:3: UserWarning:

`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(data[i])
```




```
<ipython-input-31-b8b1b1ba79af>:3: 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(data[i])
```


```
# model building
x=data.drop(columns=['target'])
y=data['target']
```

```
x.head()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3
5	58	0	0	100	248	0	0	122	0	1.0	1	0	2

```
from sklearn.model_selection import train_test_split
```

```
xtrain,xtest,ytrain,ytest=train_test_split(x,y,test_size=0.2,random_state=42)
```

```
from sklearn.preprocessing import StandardScaler
```

```
sc=StandardScaler()
```

```
xtrain=sc.fit_transform(xtrain)
xtest=sc.fit_transform(xtest)
```

```
xtrain
```

```
array([[ 0.46149851,  0.6770032 , -0.95587907, ...,  0.90065484,
        -0.72249687,  1.18504289],
       [ 1.14080707, -1.47709789,  1.02180176, ...,  0.90065484,
        -0.72249687,  1.18504289],
       [-0.33102814, -1.47709789,  1.02180176, ...,  0.90065484,
        -0.72249687, -0.48515179],
       ...,
       [-1.0103367 , -1.47709789, -0.95587907, ..., -0.80328675,
        -0.72249687, -0.48515179],
       [-0.33102814, -1.47709789,  1.02180176, ...,  0.90065484,
        -0.72249687, -0.48515179],
       [ 0.80115279,  0.6770032 ,  2.01064218, ..., -0.80328675,
         2.03612754, -0.48515179]])
```

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

```
from sklearn.tree import DecisionTreeClassifier
```

```
similar flexibility) or  histplot  (an axes-level function for histograms).
```

```
dt=DecisionTreeClassifier()
```

https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

```
dt.fit(xtrain,ytrain)
```

```
DecisionTreeClassifier
DecisionTreeClassifier()
```

```
ypred=dt.predict(xtest)
```

```
from sklearn.metrics import *
```

```
confusion_matrix(ypred, ytest)
```

```
array([[16, 10],
       [ 5, 22]])
```

```
(16+22)/(16+10+5+22)
```

```
0.7169811320754716
```

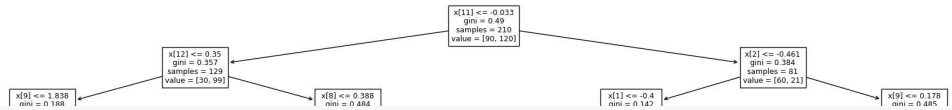
```
from sklearn.metrics import accuracy_score
```

```
accuracy_score(ytest,ypred)
```

```
0.7169811320754716
```

```
Please adapt your code to use either  displot  (a figure-level function with
```

```
plt.figure(figsize=(30,10))
from sklearn import tree
tree.plot_tree(dt)
plt.show()
```




```
# hyper parameter tuning
```

```
from sklearn.tree import DecisionTreeClassifier
```

```
from sklearn.model_selection import GridSearchCV
```

```
param_grid = {
    'criterion': ['gini', 'entropy'],
    'max_depth': [None, 5, 10, 15],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}
dt=DecisionTreeClassifier()
grid_search = GridSearchCV(dt, param_grid, cv=5)
grid_search.fit(x, y)
```

```
▸ GridSearchCV
▸ estimator: DecisionTreeClassifier
  ▸ DecisionTreeClassifier
```

```
best_params = grid_search.best_params_
best_dt = grid_search.best_estimator_
```