Practical Decision Tree

Decision Tree:

Decision tree is a one of the supervised machine learing algorithm. This algorithm can be used for regression and classification on problems. but mostly used classification problems.

The goal is to build up a classifire model that can predict the class or value of the target variable.

Graphical representation of all the possible solutions to a decision. Decisions are mainly based on some condition. Decision mode can easily be explained.

In [2]:
```
#import labrires
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn import tree
from sklearn.metrics import accuracy_score, confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt
```

In [3]:
```
#import dataset
data='heart_1.csv'
df=pd.read_csv(data)
df.head()
```

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

In [4]:
```
df.shape
```

Out[4]:
```
(303, 14)
```

In [5]:
```
df[df['chol']>300].shape
```

Out[5]:
```
(43, 14)
```

In [6]:
```
f=df[df['thal']==2]
```

In [7]:
```
f[f['target']==1].shape
```

Out[7]:
```
(130, 14)
```

In [8]:
```
df.shape
```

Out[8]:
```
(303, 14)
```

In [9]:
```
df.isnull().sum()
```

Out[9]:
```
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
```

In [10]:
```
df.dtypes
```

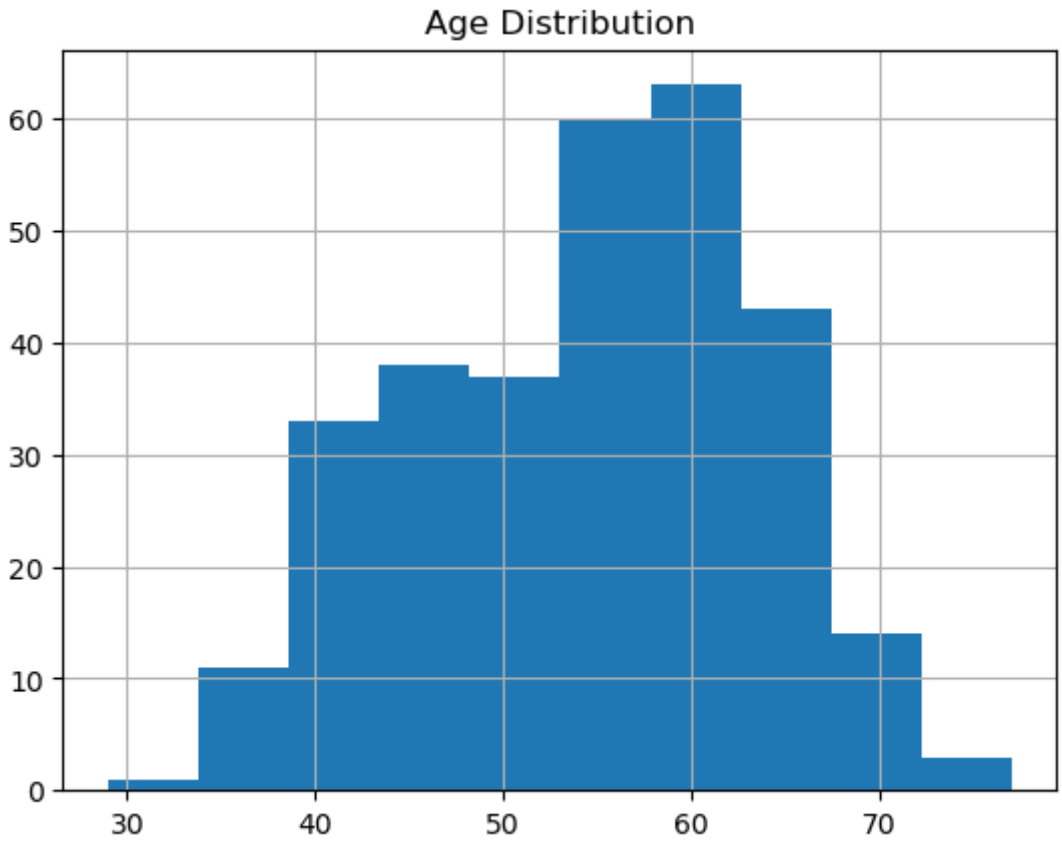
Out[10]:
```
age           int64
sex           int64
cp            int64
trestbps      int64
chol          int64
fbs           int64
restecg       int64
thalach       int64
exang         int64
oldpeak     float64
slope         int64
ca            int64
thal          int64
target        int64
dtype: object
```

In [11]:
```
df.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
```

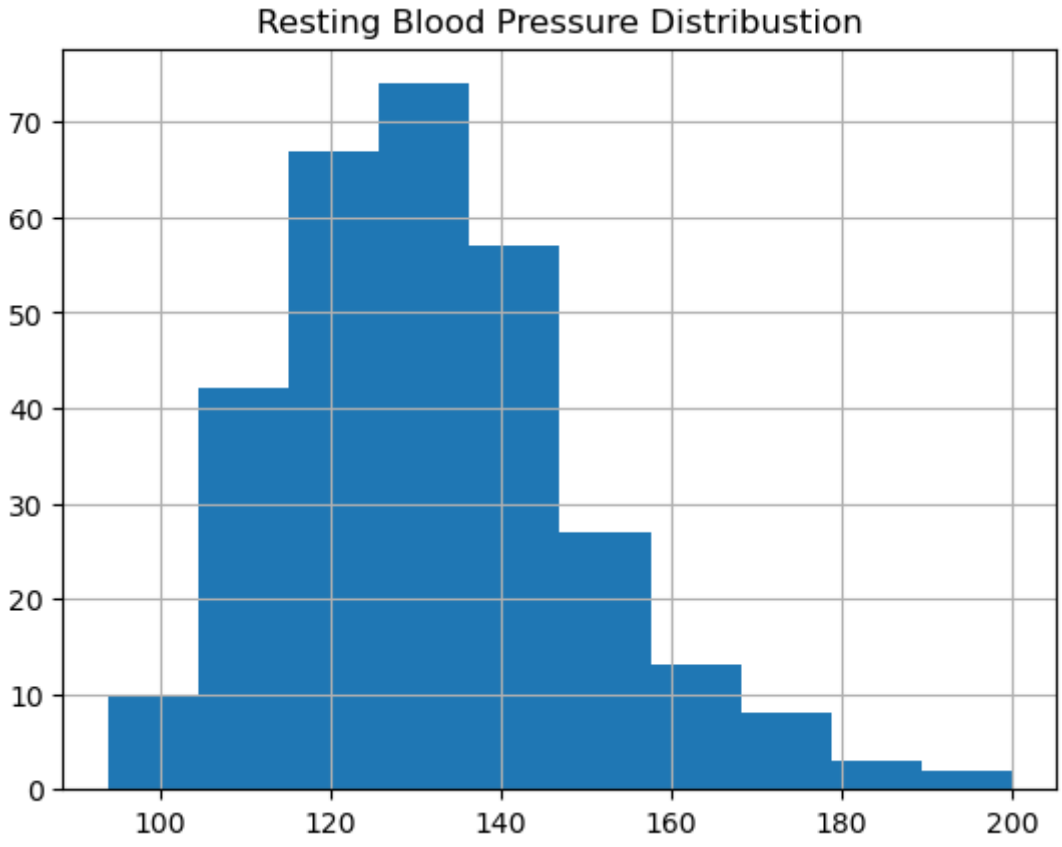
In [12]:
```
df['age'].hist(grid=True,bins=10);
plt.title('Age Distribution')
```

Out[12]:
```
Text(0.5, 1.0, 'Age Distribution')
```



In [13]:
```
df['trestbps'].hist()
plt.title('Resting Blood Pressure Distribustion')
```

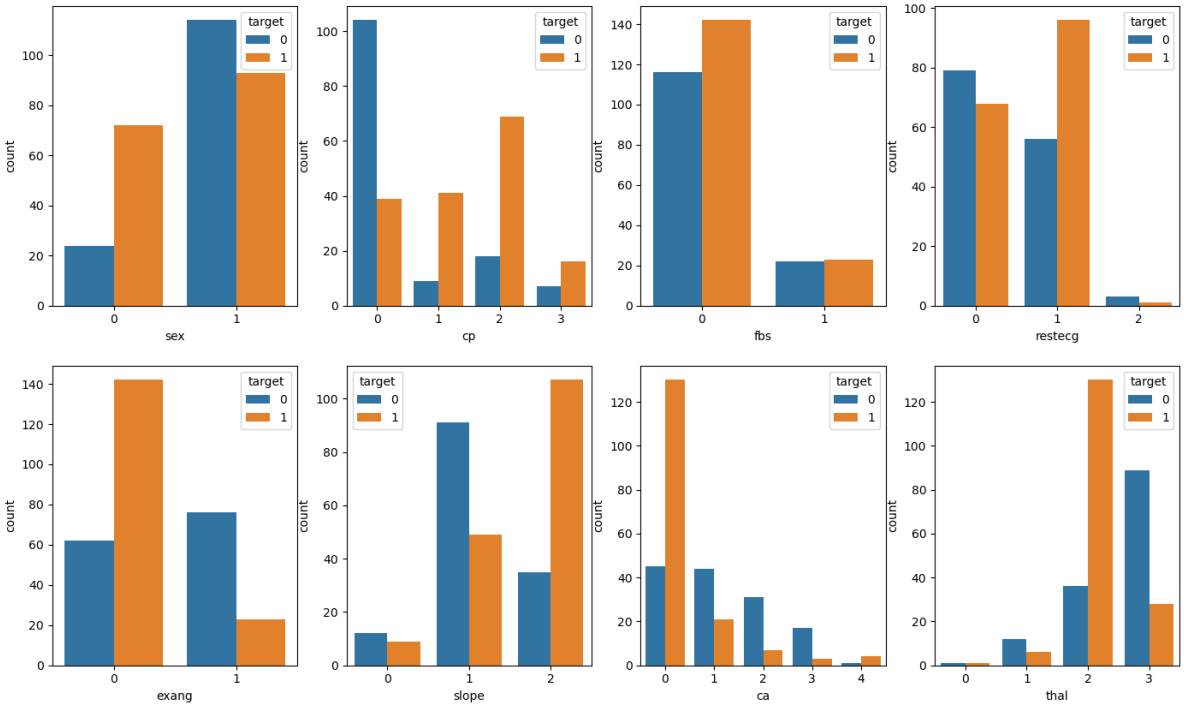
Out[13]:
```
Text(0.5, 1.0, 'Resting Blood Pressure Distribustion')
```



In the above graph, we are having a normal distribustion

In [14]:
```
fig, axes=plt.subplots(nrows=2,ncols=4,figsize=(17,10))
cat_feat=['sex','cp','fbs','restecg','exang','slope','ca','thal','target']

for idx, feature in enumerate(cat_feat):
    if feature !='target':
        ax=axes[int(idx/4),idx%4]
        sns.countplot(x=feature,hue='target',data=df,ax=ax)
```



Let's get some insights from this chart

Chest pain: the heart decreses diagrams is greater among the patientis that feel any chest pain.

Restqc-Electrocardisgraph results: The rate of heart deasease diagonoses higher for patients with a ST-T wabe abnormally.

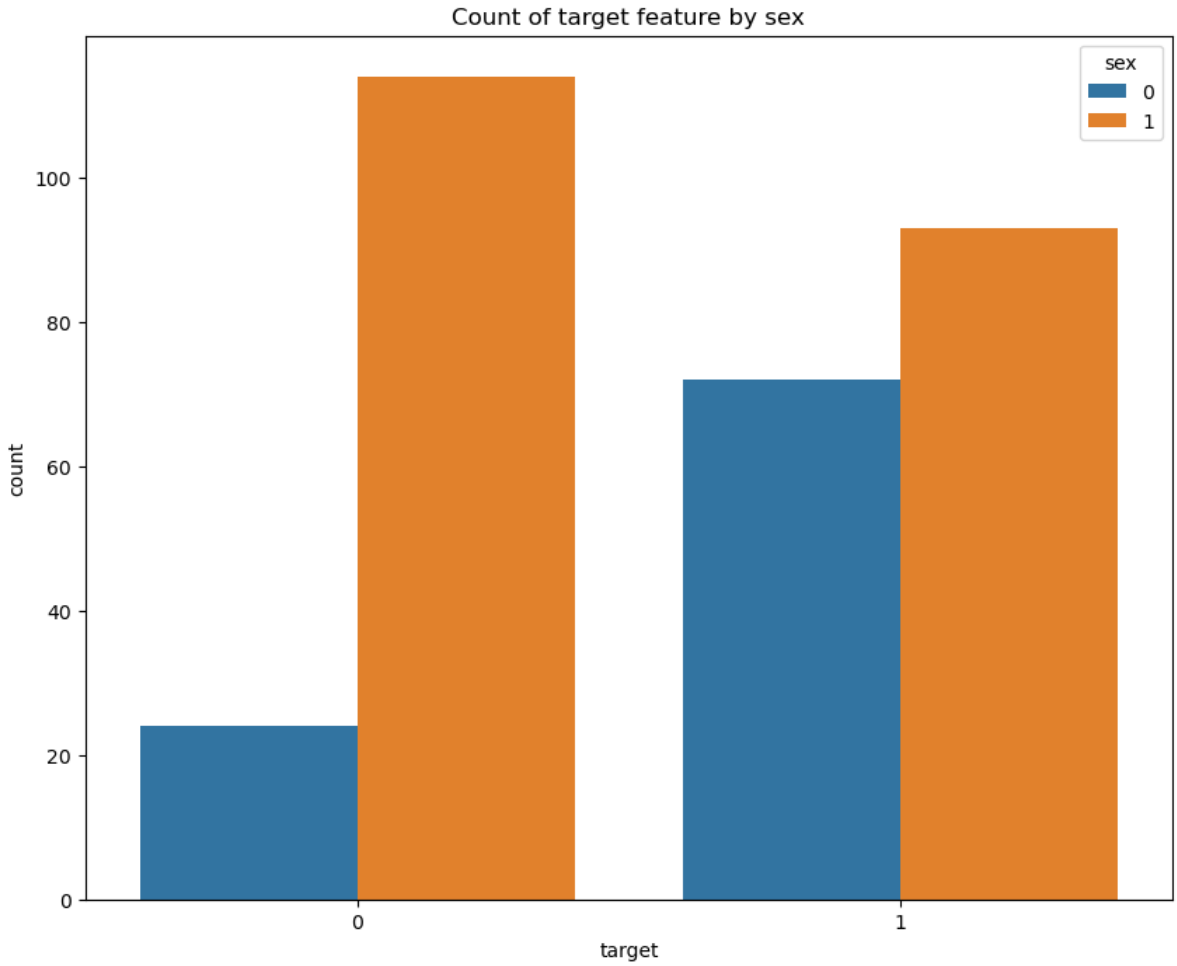
Slope:The ratio pf patients diagonsed with heart deasese is higher for slope=2.

ca: The diagonesed ration decresese of ca between 1 and 3.

Thal: The diagnosed ratio is higher for thal=2.

In [15]:
```
plt.rcParams['figure.figsize']=(10,8)
sns.countplot(x='target', hue='sex',data=df);
plt.title('Count of target feature by sex')
```

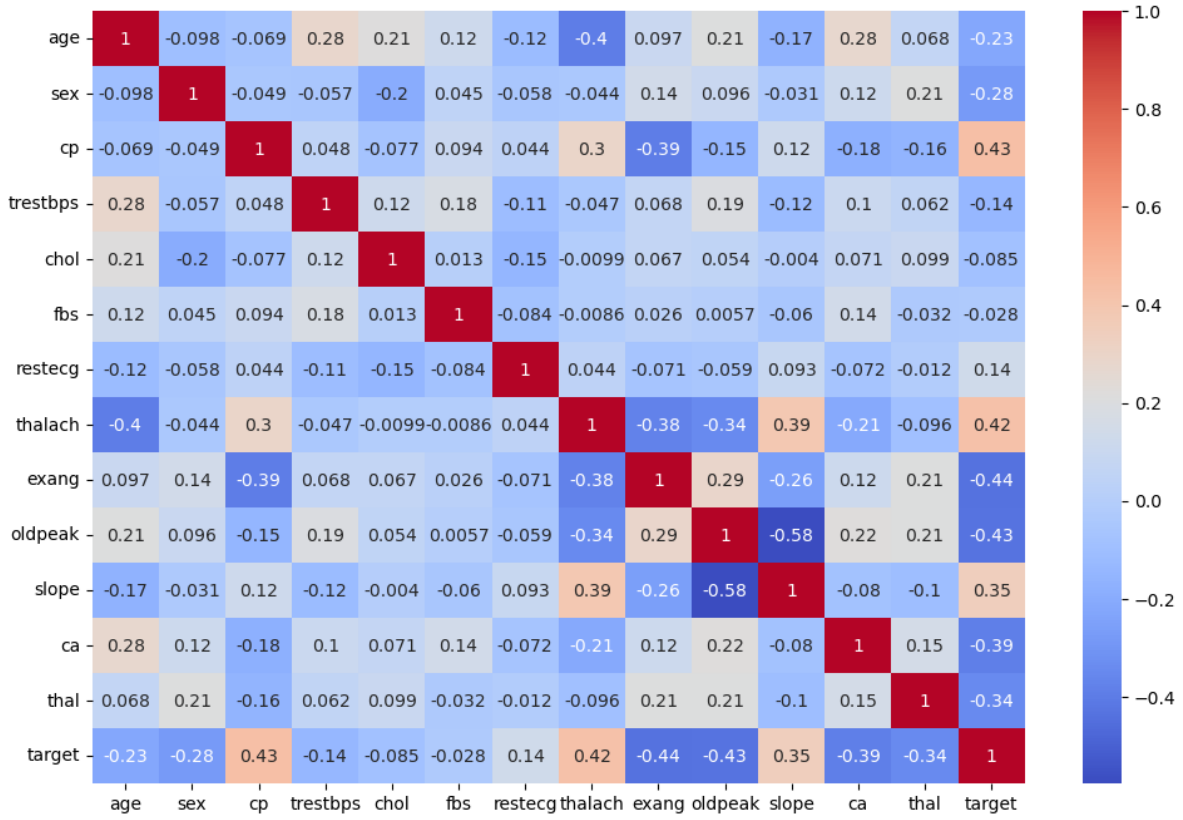
Out[15]:
```
Text(0.5, 1.0, 'Count of target feature by sex')
```



The amount of healthy male people is greater than the amount of unhealthy for women the number of unhealthy women is higher

In [16]:
```
plt.figure(figsize=(12,8))
sns.heatmap(df.corr(),annot=True,cmap='coolwarm')
```

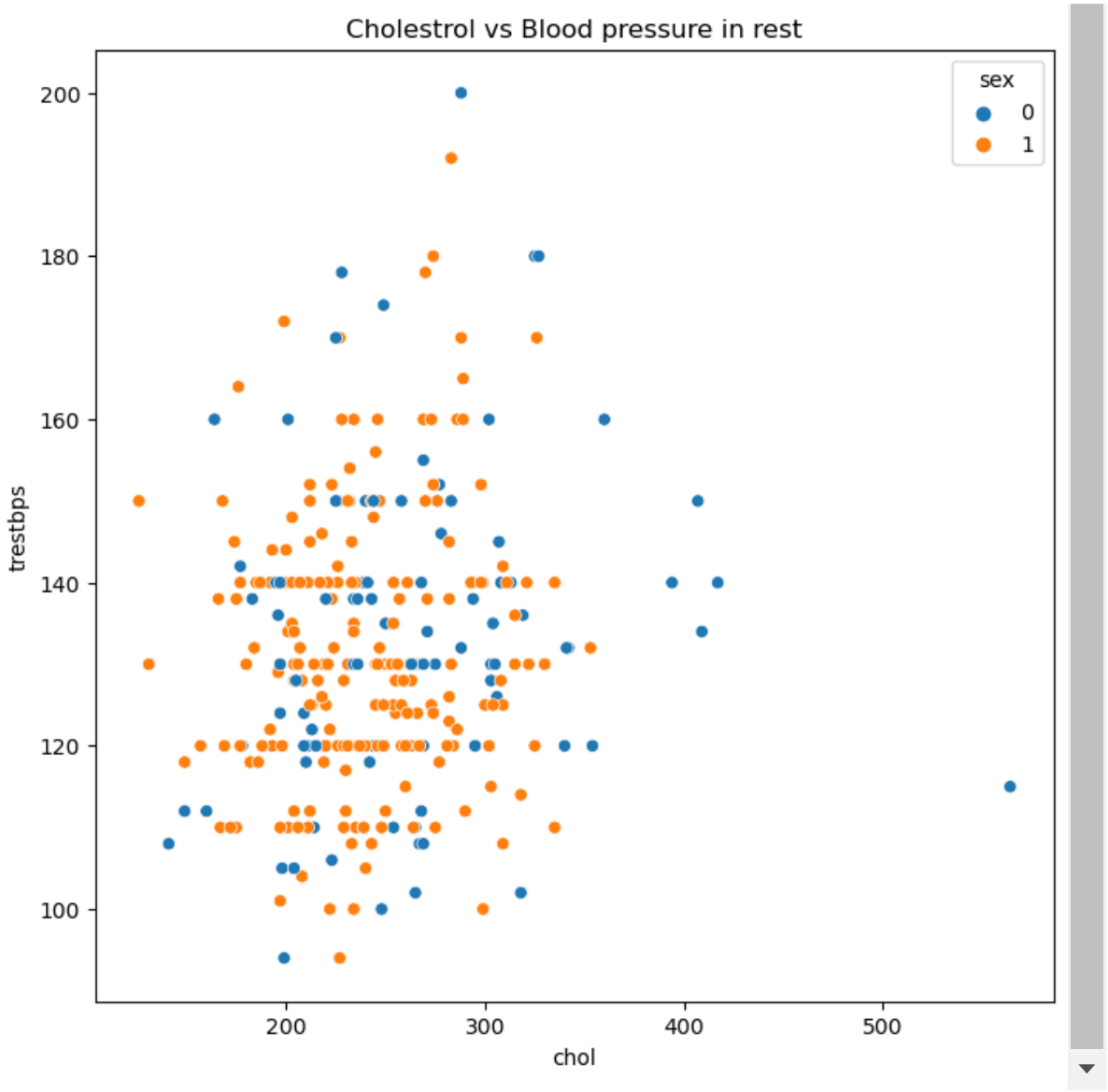
Out[16]:
```
<AxesSubplot:>
```



Apparently there are no feature with a pretty strong correlation (above [0.7])

In [21]:
```
plt.rcParams['figure.figsize']=(8,8)
sns.scatterplot(x='chol',y='trestbps',hue='sex',size=None,data=df)
plt.title('Cholestrol vs Blood pressure in rest')
```

Out[21]:
```
Text(0.5, 1.0, 'Cholestrol vs Blood pressure in rest')
```



As can be seen there is a paitient with high cholestrol. But there not a specific between those feel pain during exercise practice and those of not feel pain. we can use hue to fillter by sex. it's also possible to fillter using size='label_to_filter'

In [22]:
```
X=df.drop(columns=['target'])#independant variable
y=df['target']#dependant variable
print(X.shape)
print(y.shape)
```

```
(303, 13)
(303,)
```

In [23]:
```
X_train, X_test, y_train, y_test = train_test_split(X,y, random_state=0,test_size=(
```

In [25]:
```
print(X_train.shape)
print(X_test.shape)
```

```
(212, 13)
(91, 13)
```

In [26]:
```
clf=tree.DecisionTreeClassifier()
clf.fit(X_train,y_train)
#preduct
y_train_pred=clf.predict(X_train)
y_test_pred=clf.predict(X_test)
```

In [27]:
```
y_train_pred
```

Out[27]:
```
array([1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0,
       0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1,
       0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0,
       0, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0,
       0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1,
       0, 1, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0,
       0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0,
       0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0,
       1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0,
       1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0], dtype=int64)
```

In [28]:
```
y_test_pred
```

Out[28]:
```
array([0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 0,
       0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0,
       0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1,
       1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1,
       0, 1, 0], dtype=int64)
```

In [30]:
```
confusion_matrix(y_train_pred,y_train)
```

Out[30]:
```
array([[ 94,   0],
       [  0, 118]], dtype=int64)
```

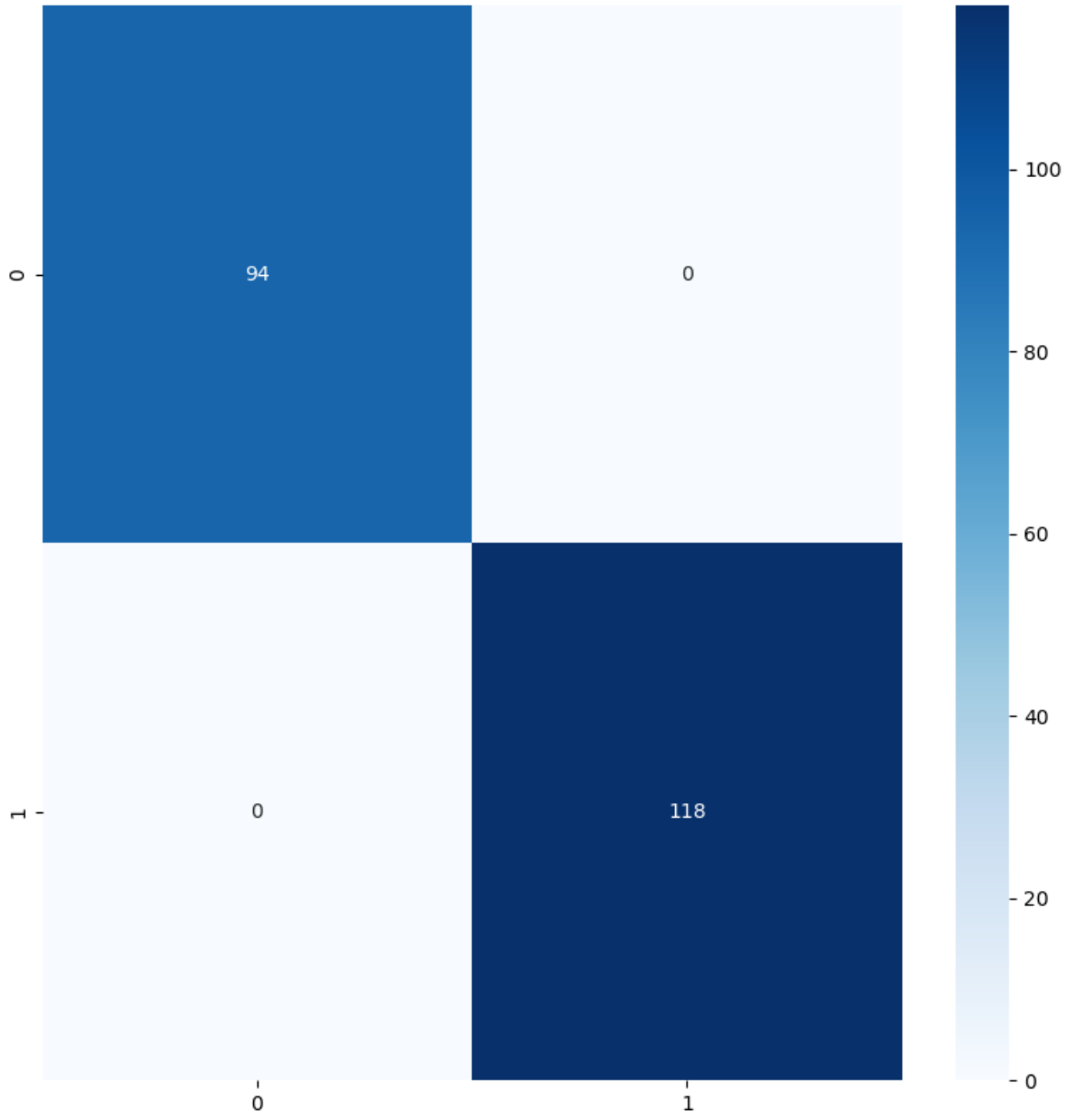
In [29]:
```
#helper function
def plot_confusionmatrix(y_train_pred, y_train,dom):
    print(f'{dom} Confusion matrix')
    cf=confusion_matrix(y_train_pred,y_train)
    sns.heatmap(cf,annot=True,cmap='Blues',fmt='g')
    plt.tight_layout()
    plt.show()
```

In [31]:
```
print(f'Train Score {accuracy_score(y_train_pred,y_train)}')
print(f'Test Score {accuracy_score(y_test_pred,y_test)}')
```

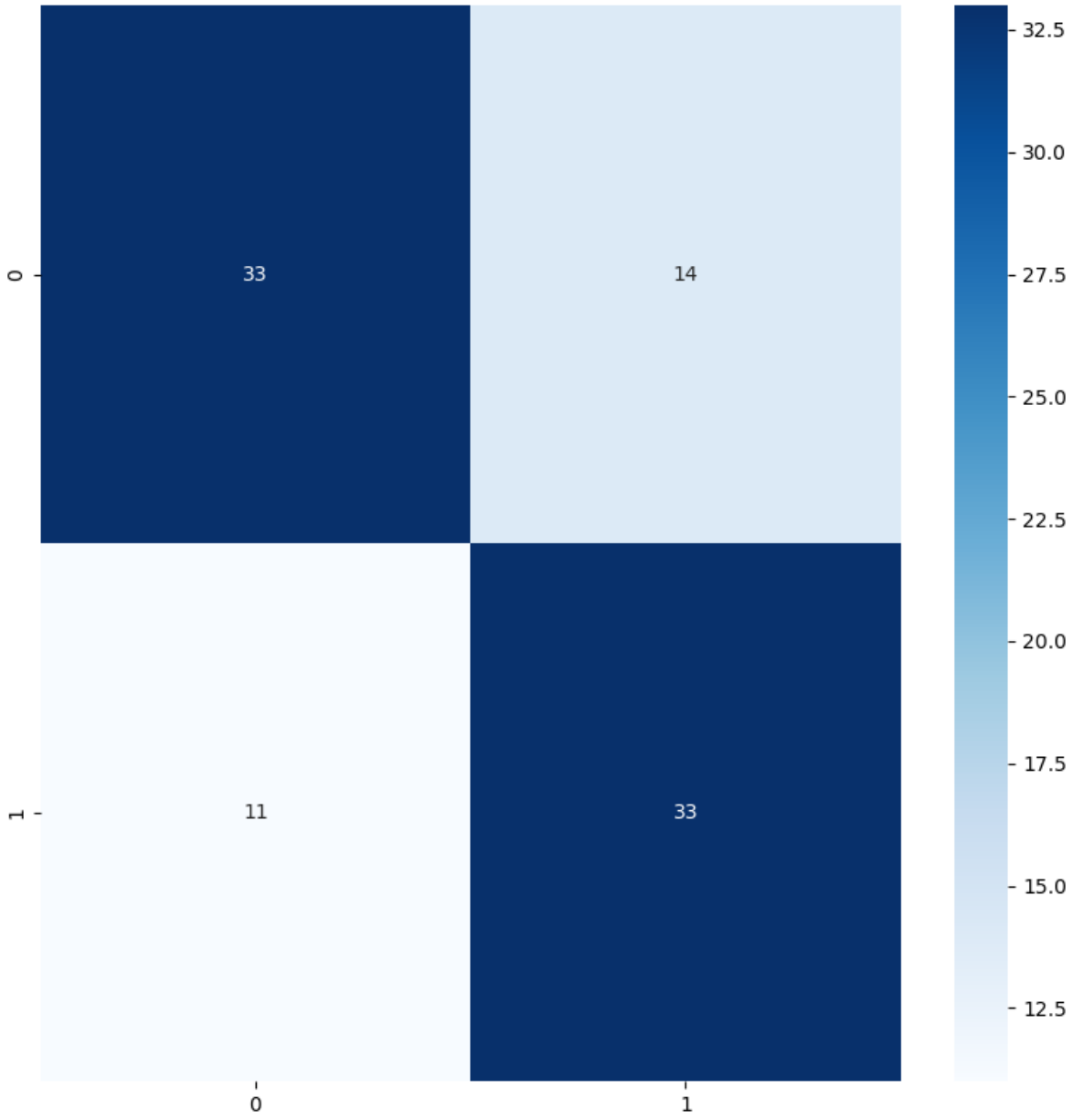
```
Train Score 1.0
Test Score 0.7252747252747253
```

In [32]:
```
plot_confusionmatrix(y_train_pred,y_train,dom='Train')
plot_confusionmatrix(y_test_pred,y_test,dom='Test')
```

```
Train Confusion matrix
```



Test Confusion matrix



In [48]:
```
c_parameter_name='max_depth'
c_parameter_value=[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15]
df=pd.DataFrame(columns=[c_parameter_name,'accuracy'])

for input_parameter in c_parameter_value:
    model=tree.DecisionTreeClassifier(max_depth=input_parameter,splitter='best')
    model.fit(X_train,y_train)
    y_pred=model.predict(X_test)
    acc_score=accuracy_score(y_test,y_pred)*100
    df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score}, ignore_
print(df)
```

	max_depth	accuracy
0	1.0	76.923077
1	2.0	73.626374
2	3.0	81.318681
3	4.0	76.923077
4	5.0	73.626374
5	6.0	72.527473
6	7.0	73.626374
7	8.0	71.428571
8	9.0	72.527473
9	10.0	72.527473
10	11.0	71.428571
11	12.0	71.428571
12	13.0	71.428571
13	14.0	75.824176
14	15.0	75.824176

```
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\3743572166.py:10: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score}, ignore_in
dex=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\3743572166.py:10: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score}, ignore_in
dex=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\3743572166.py:10: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score}, ignore_in
dex=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\3743572166.py:10: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score}, ignore_in
dex=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\3743572166.py:10: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score}, ignore_in
dex=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\3743572166.py:10: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score}, ignore_in
dex=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\3743572166.py:10: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score}, ignore_in
dex=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\3743572166.py:10: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score}, ignore_in
dex=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\3743572166.py:10: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score}, ignore_in
dex=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\3743572166.py:10: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score}, ignore_in
dex=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\3743572166.py:10: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score}, ignore_in
dex=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\3743572166.py:10: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score}, ignore_in
dex=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\3743572166.py:10: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score}, ignore_in
```

```
dex=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\3743572166.py:10: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score}, ignore_in
dex=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\3743572166.py:10: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score}, ignore_in
dex=True)
```

In [54]:

```
from sklearn.tree import DecisionTreeClassifier
for input_parameter in c_parameter_value:
    model= DecisionTreeClassifier(max_depth=input_parameter, splitter='best')
    model.fit(X_train,y_train)
    y_pred1=model.predict(X_test)
    acc_score=accuracy_score(y_test,y_pred)*100
    df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score},ignore_i
```

```
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\2617876694.py:7: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score},ignore_ind
ex=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\2617876694.py:7: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score},ignore_ind
ex=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\2617876694.py:7: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score},ignore_ind
ex=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\2617876694.py:7: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score},ignore_ind
ex=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\2617876694.py:7: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score},ignore_ind
ex=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\2617876694.py:7: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score},ignore_ind
ex=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\2617876694.py:7: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score},ignore_ind
ex=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\2617876694.py:7: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score},ignore_ind
ex=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\2617876694.py:7: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score},ignore_ind
ex=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\2617876694.py:7: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score},ignore_ind
ex=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\2617876694.py:7: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score},ignore_ind
ex=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\2617876694.py:7: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score},ignore_ind
ex=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\2617876694.py:7: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score},ignore_ind
```

```
ex=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\2617876694.py:7: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score},ignore_ind
ex=True)
C:\Users\Admin\AppData\Local\Temp\ipykernel_11352\2617876694.py:7: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  df=df.append({c_parameter_name:input_parameter, 'accuracy':acc_score},ignore_ind
ex=True)
```

In [47]:
```
from sklearn.metrics import classification_report
print(classification_report(y_test_pred,y_test))
```

```
              precision    recall  f1-score   support

           0       0.75      0.70      0.73        47
           1       0.70      0.75      0.73        44

    accuracy                           0.73        91
   macro avg       0.73      0.73      0.73        91
weighted avg       0.73      0.73      0.73        91
```

In [ ]:
```

```