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

```
from google.colab import files

uploaded=files.upload()
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving heart.csv to heart.csv
```

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import matplotlib.pyplot as plt # this is used for the plot the graph
import seaborn as sns # used for plot interactive graph.
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import average_precision_score
from sklearn.model_selection import cross_val_score
from sklearn.metrics import precision_recall_curve
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import roc_curve
from sklearn.metrics import f1_score
from sklearn.metrics import auc
from sklearn.svm import SVC
%matplotlib inline
```

```
df=pd.read_csv('heart.csv')
df.head(5)
```

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

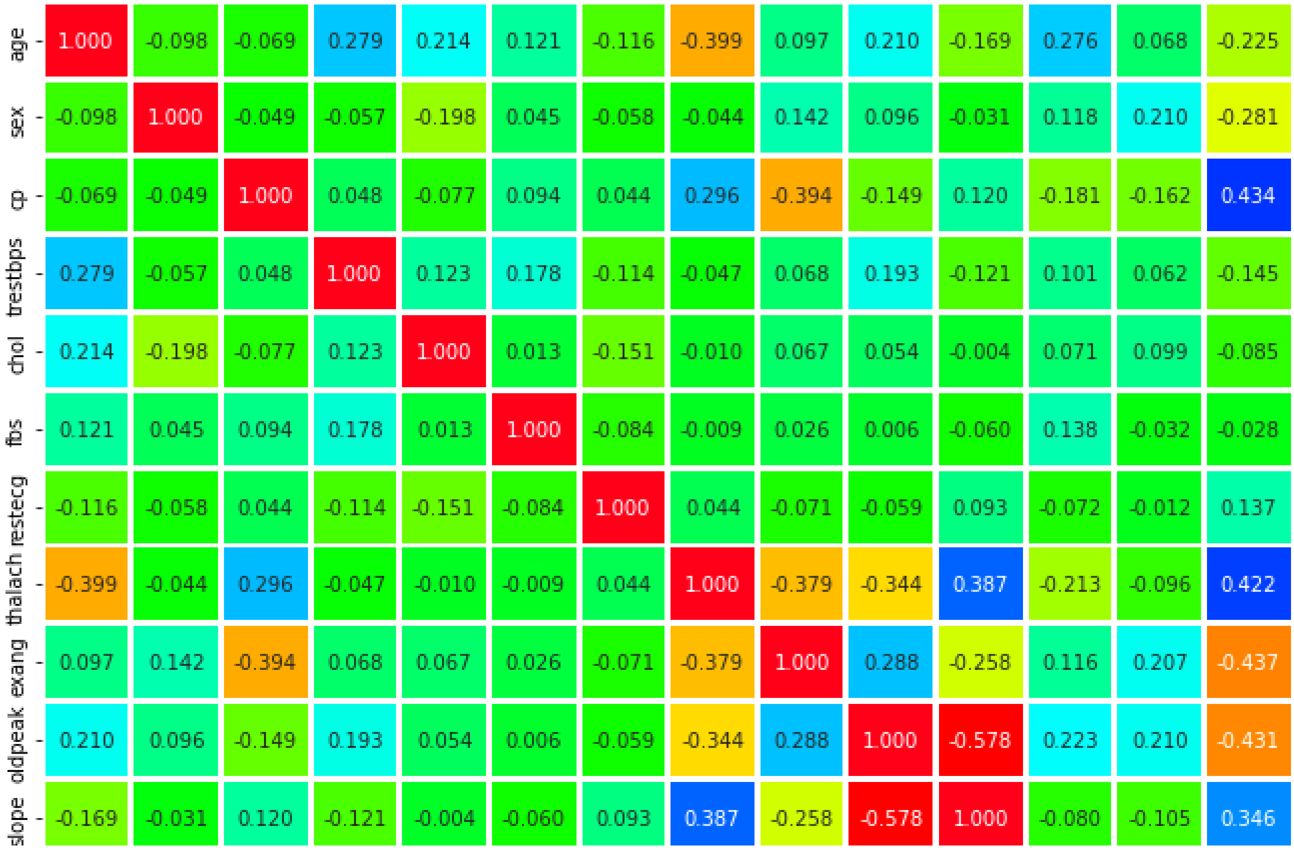
```
df.describe()
```

	age	sex	cp	trestbps	chol	fbs	restec
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.00000
mean	54.366337	0.683168	0.966997	131.623762	246.264026	0.148515	0.52805
std	9.082101	0.466011	1.032052	17.538143	51.830751	0.356198	0.52586
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.00000
25%	47.500000	0.000000	0.000000	120.000000	211.000000	0.000000	0.00000
50%	55.000000	1.000000	1.000000	130.000000	240.000000	0.000000	1.00000
75%	61.000000	1.000000	2.000000	140.000000	274.500000	0.000000	1.00000
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	2.00000

```
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
```

```
plt.figure(figsize=(14,10))
sns.heatmap(df.corr(),annot=True,cmap='hsv',fmt='.3f',linewidths=2)
plt.show()
```




```
df.groupby('cp',as_index=False)['target'].mean()
```

	cp	target
0	0	0.272727
1	1	0.820000
2	2	0.793103
3	3	0.695652

```
df.groupby('slope',as_index=False)['target'].mean()
```

	slope	target
0	0	0.428571
1	1	0.350000
2	2	0.753521

```
df.groupby('thal',as_index=False)['target'].mean()
```

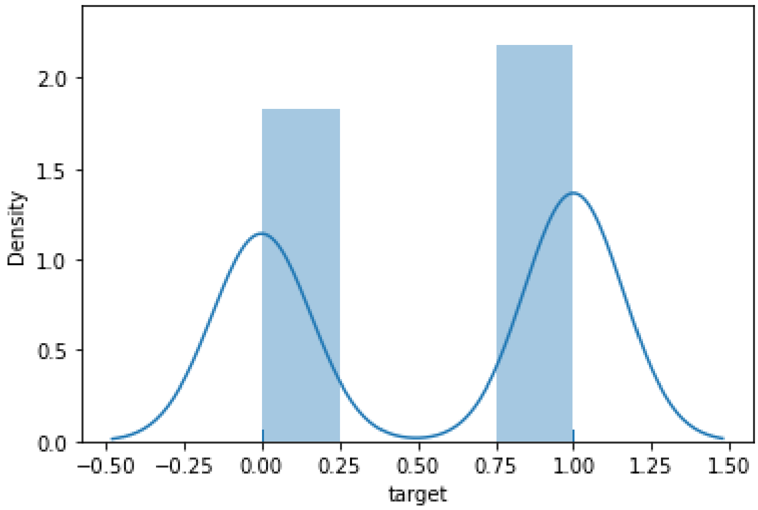
thal	target

```
df.groupby('target').mean()
```

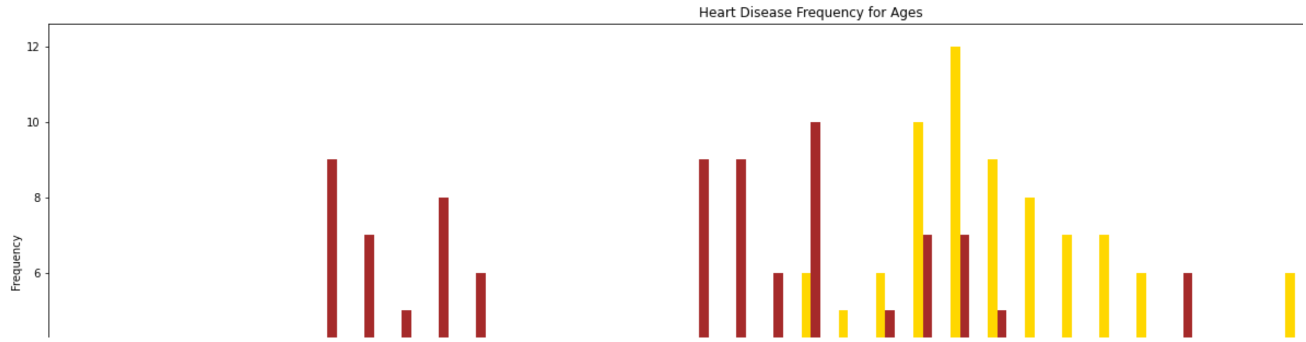
	age	sex	cp	trestbps	chol	fbs	restecg	th
target								
0	56.601449	0.826087	0.478261	134.398551	251.086957	0.159420	0.449275	139.1
1	52.496970	0.563636	1.375758	129.303030	242.230303	0.139394	0.593939	158.4

```
sns.distplot(df['target'],rug=True)
plt.show()
```

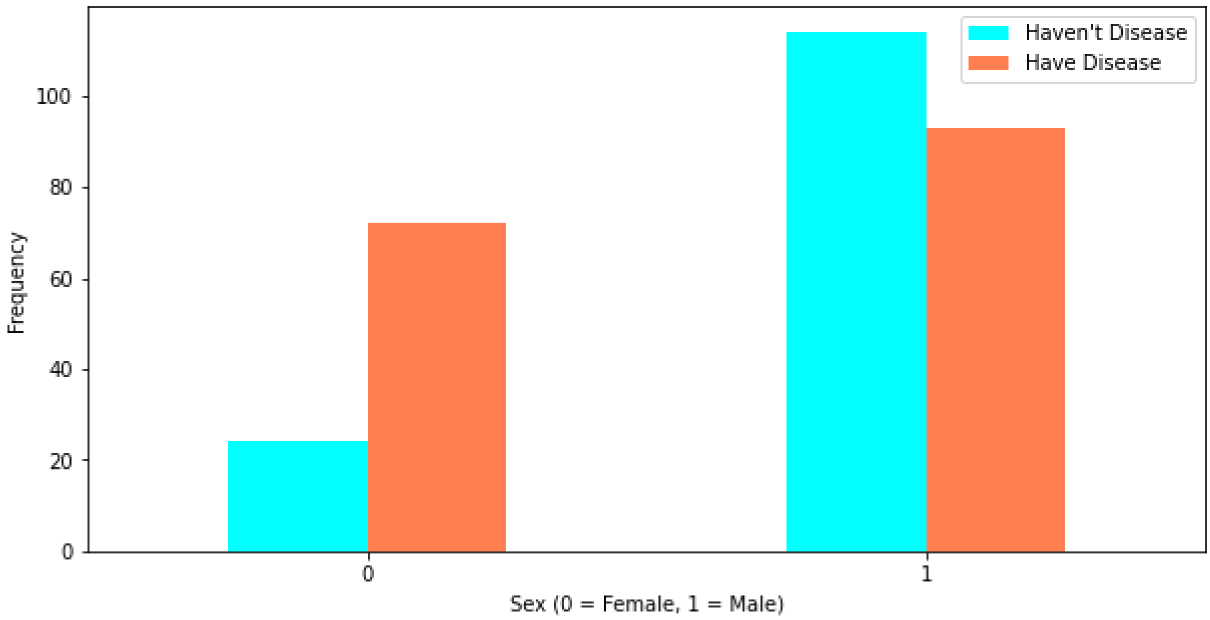
```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning:
  warnings.warn(msg, FutureWarning)
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2103: FutureWarning:
  warnings.warn(msg, FutureWarning)
```




```
pd.crosstab(df.age,df.target).plot(kind="bar",figsize=(25,8),color=['gold','brown' ])
plt.title('Heart Disease Frequency for Ages')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.show()
```

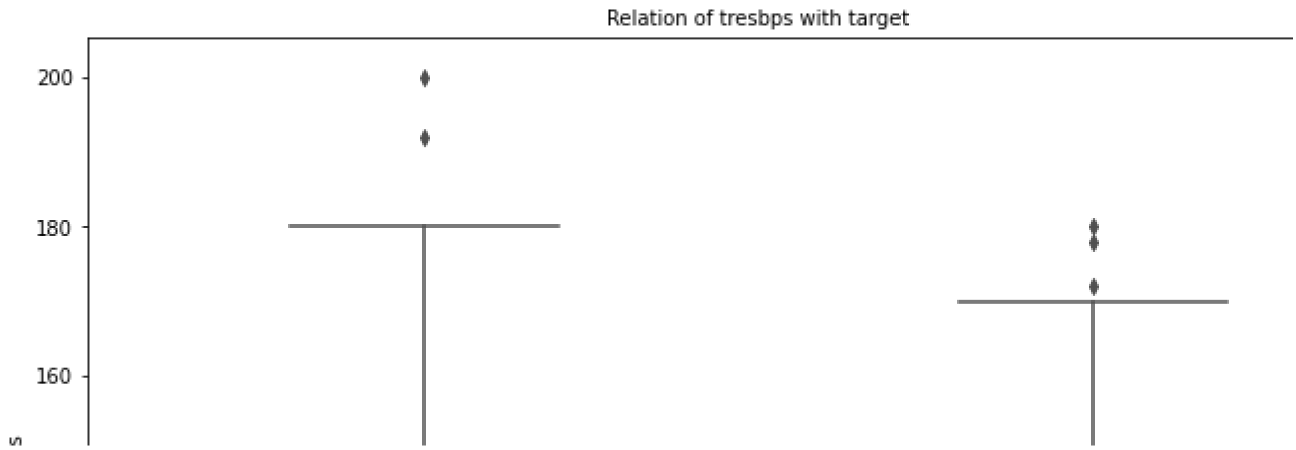



```
pd.crosstab(df.sex,df.target).plot(kind="bar",figsize=(10,5),color=['cyan','coral' ])
plt.xlabel('Sex (0 = Female, 1 = Male)')
plt.xticks(rotation=0)
plt.legend(["Haven't Disease", "Have Disease"])
plt.ylabel('Frequency')
plt.show()
```



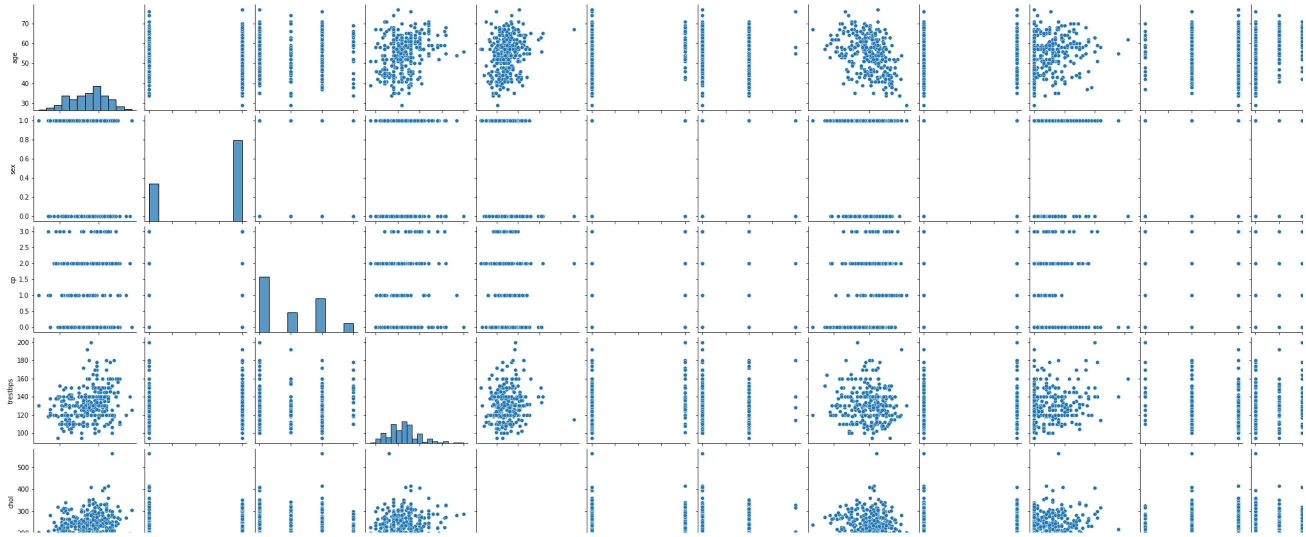

```
plt.figure(figsize=(12,8))
sns.boxplot(df['target'], df['trestbps'], palette = 'rainbow')
plt.title('Relation of tresbps with target', fontsize = 10)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass
  FutureWarning
Text(0.5, 1.0, 'Relation of tresbps with target')
```

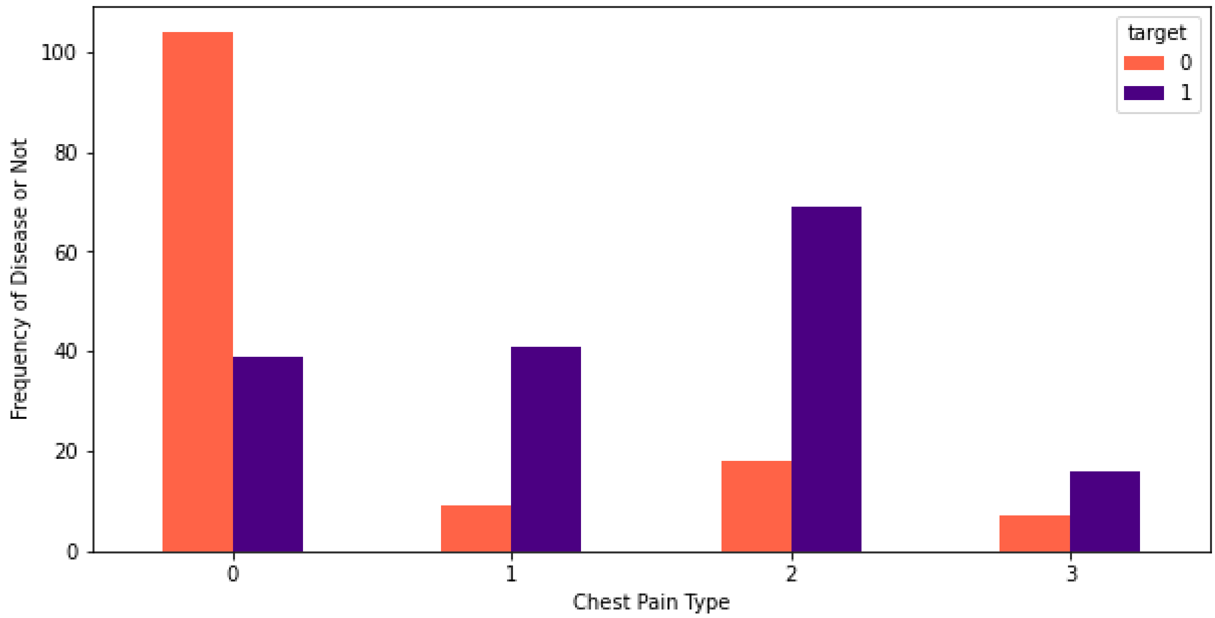


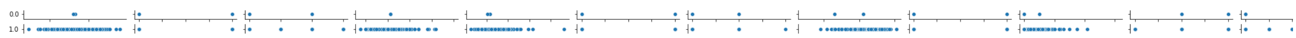

```
sns.pairplot(data=df)
```

```
<seaborn.axisgrid.PairGrid at 0x7f2933d3dad0>
```



```
pd.crosstab(df.cp,df.target).plot(kind="bar",figsize=(10,5),color=['tomato','indigo' ])
plt.xlabel('Chest Pain Type')
plt.xticks(rotation = 0)
plt.ylabel('Frequency of Disease or Not')
plt.show()
```



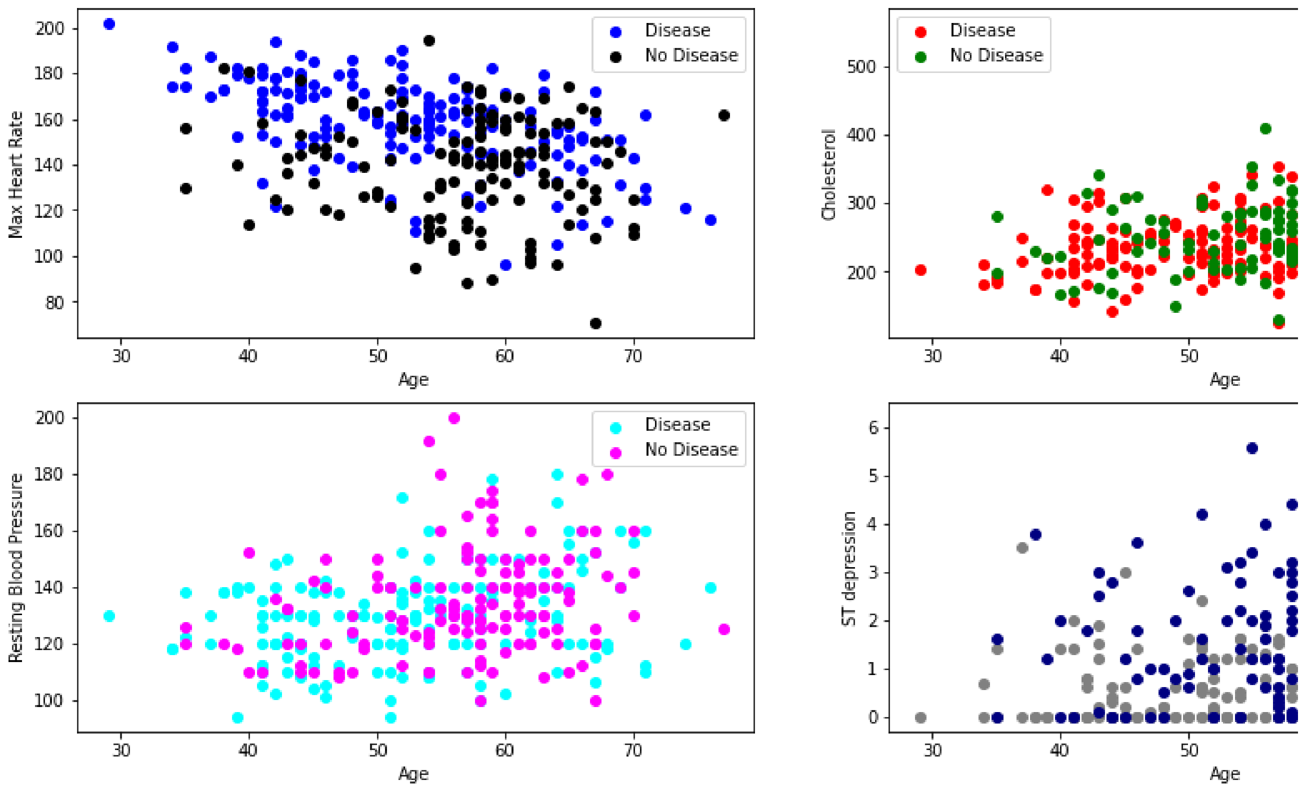


```
plt.figure(figsize=(16,8))
plt.subplot(2,2,1)
plt.scatter(x=df.age[df.target==1],y=df.thalach[df.target==1],c='blue')
plt.scatter(x=df.age[df.target==0],y=df.thalach[df.target==0],c='black')
plt.xlabel('Age')
plt.ylabel('Max Heart Rate')
plt.legend(['Disease','No Disease'])

plt.subplot(2,2,2)
plt.scatter(x=df.age[df.target==1],y=df.chol[df.target==1],c='red')
plt.scatter(x=df.age[df.target==0],y=df.chol[df.target==0],c='green')
plt.xlabel('Age')
plt.ylabel('Cholesterol')
plt.legend(['Disease','No Disease'])
```

```
plt.subplot(2,2,3)
plt.scatter(x=df.age[df.target==1],y=df.trestbps[df.target==1],c='cyan')
plt.scatter(x=df.age[df.target==0],y=df.trestbps[df.target==0],c='fuchsia')
plt.xlabel('Age')
plt.ylabel('Resting Blood Pressure')
plt.legend(['Disease','No Disease'])

plt.subplot(2,2,4)
plt.scatter(x=df.age[df.target==1],y=df.oldpeak[df.target==1],c='grey')
plt.scatter(x=df.age[df.target==0],y=df.oldpeak[df.target==0],c='navy')
plt.xlabel('Age')
plt.ylabel('ST depression')
plt.legend(['Disease','No Disease'])
plt.show()
```




```
chest_pain=pd.get_dummies(df['cp'],prefix='cp',drop_first=True)
df=pd.concat([df,chest_pain],axis=1)
df.drop(['cp'],axis=1,inplace=True)
sp=pd.get_dummies(df['slope'],prefix='slope')
th=pd.get_dummies(df['thal'],prefix='thal')
rest_ecg=pd.get_dummies(df['restecg'],prefix='restecg')
frames=[df,sp,th,rest_ecg]
df=pd.concat(frames,axis=1)
df.drop(['slope','thal','restecg'],axis=1,inplace=True)
```

```
df.head(5)
```

	age	sex	trestbps	chol	fbs	thalach	exang	oldpeak	ca	target	cp_1	cp_2	cp
0	63	1	145	233	1	150	0	2.3	0	1	0	0	
1	37	1	130	250	0	187	0	3.5	0	1	0	1	
2	41	0	130	204	0	172	0	1.4	0	1	1	0	
3	56	1	120	236	0	178	0	0.8	0	1	1	0	
4	57	0	120	354	0	163	1	0.6	0	1	0	0	

```
X = df.drop(['target'], axis = 1)
y = df.target.values
```

```
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 =
```

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

```
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D
from keras.layers import Activation, Dropout, Flatten, Dense
import keras
from keras.models import Sequential
from keras.layers import Dense
import warnings
```

```
classifier = Sequential()

# Adding the input layer and the first hidden layer
classifier.add(Dense(11,kernel_initializer = 'uniform', activation = 'relu', input_shape =

# Adding the second hidden layer
classifier.add(Dense(11,kernel_initializer = 'uniform', activation = 'relu'))
# Adding the output layer
classifier.add(Dense(1,kernel_initializer= 'uniform', activation = 'sigmoid'))

# Compiling the ANN
classifier.compile(optimizer = 'adam', loss = 'binary_crossentropy', metrics = ['accuracy'
```

```
classifier.fit(X_train, y_train, batch_size = 10, epochs = 100)
```

```
Epoch 1/100
25/25 [==============================] - 1s 3ms/step - loss: 0.6929 - accuracy: 0.
Epoch 2/100
25/25 [==============================] - 0s 3ms/step - loss: 0.6901 - accuracy: 0.
Epoch 3/100
25/25 [==============================] - 0s 2ms/step - loss: 0.6757 - accuracy: 0.
Epoch 4/100
25/25 [==============================] - 0s 2ms/step - loss: 0.6333 - accuracy: 0.
Epoch 5/100
25/25 [==============================] - 0s 3ms/step - loss: 0.5643 - accuracy: 0.
Epoch 6/100
25/25 [==============================] - 0s 2ms/step - loss: 0.4864 - accuracy: 0.
Epoch 7/100
25/25 [==============================] - 0s 3ms/step - loss: 0.4286 - accuracy: 0.
Epoch 8/100
25/25 [==============================] - 0s 2ms/step - loss: 0.3912 - accuracy: 0.
Epoch 9/100
25/25 [==============================] - 0s 3ms/step - loss: 0.3713 - accuracy: 0.
Epoch 10/100
25/25 [==============================] - 0s 3ms/step - loss: 0.3599 - accuracy: 0.
Epoch 11/100
25/25 [==============================] - 0s 3ms/step - loss: 0.3509 - accuracy: 0.
Epoch 12/100
25/25 [==============================] - 0s 2ms/step - loss: 0.3445 - accuracy: 0.
Epoch 13/100
25/25 [==============================] - 0s 2ms/step - loss: 0.3389 - accuracy: 0.
Epoch 14/100
25/25 [==============================] - 0s 3ms/step - loss: 0.3332 - accuracy: 0.
Epoch 15/100
25/25 [==============================] - 0s 2ms/step - loss: 0.3298 - accuracy: 0.
Epoch 16/100
25/25 [==============================] - 0s 2ms/step - loss: 0.3275 - accuracy: 0.
Epoch 17/100
25/25 [==============================] - 0s 3ms/step - loss: 0.3243 - accuracy: 0.
Epoch 18/100
25/25 [==============================] - 0s 2ms/step - loss: 0.3228 - accuracy: 0.
Epoch 19/100
25/25 [==============================] - 0s 2ms/step - loss: 0.3192 - accuracy: 0.
Epoch 20/100
25/25 [==============================] - 0s 3ms/step - loss: 0.3163 - accuracy: 0.
Epoch 21/100
25/25 [==============================] - 0s 2ms/step - loss: 0.3155 - accuracy: 0.
Epoch 22/100
25/25 [==============================] - 0s 3ms/step - loss: 0.3136 - accuracy: 0.
Epoch 23/100
25/25 [==============================] - 0s 2ms/step - loss: 0.3113 - accuracy: 0.
Epoch 24/100
25/25 [==============================] - 0s 2ms/step - loss: 0.3094 - accuracy: 0.
Epoch 25/100
25/25 [==============================] - 0s 2ms/step - loss: 0.3084 - accuracy: 0.
Epoch 26/100
25/25 [==============================] - 0s 3ms/step - loss: 0.3066 - accuracy: 0.
Epoch 27/100
25/25 [==============================] - 0s 3ms/step - loss: 0.3058 - accuracy: 0.
Epoch 28/100
25/25 [==============================] - 0s 2ms/step - loss: 0.3043 - accuracy: 0.
Epoch 29/100
```