In [ ]:

```
from google.colab import files
uploaded = files.upload()
```

Choose File **No file selected**

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving heart-statlog_csv (1).csv to heart-statlog_csv (1).csv
```

In [ ]:

```
import pandas as pd
import numpy as np
import tensorflow as tf
from tensorflow import keras
from sklearn.model_selection import train_test_split
import io
df = pd.read_csv(io.BytesIO(uploaded['heart-statlog_csv (1).csv']))
df.head()
```

Out[ ]:

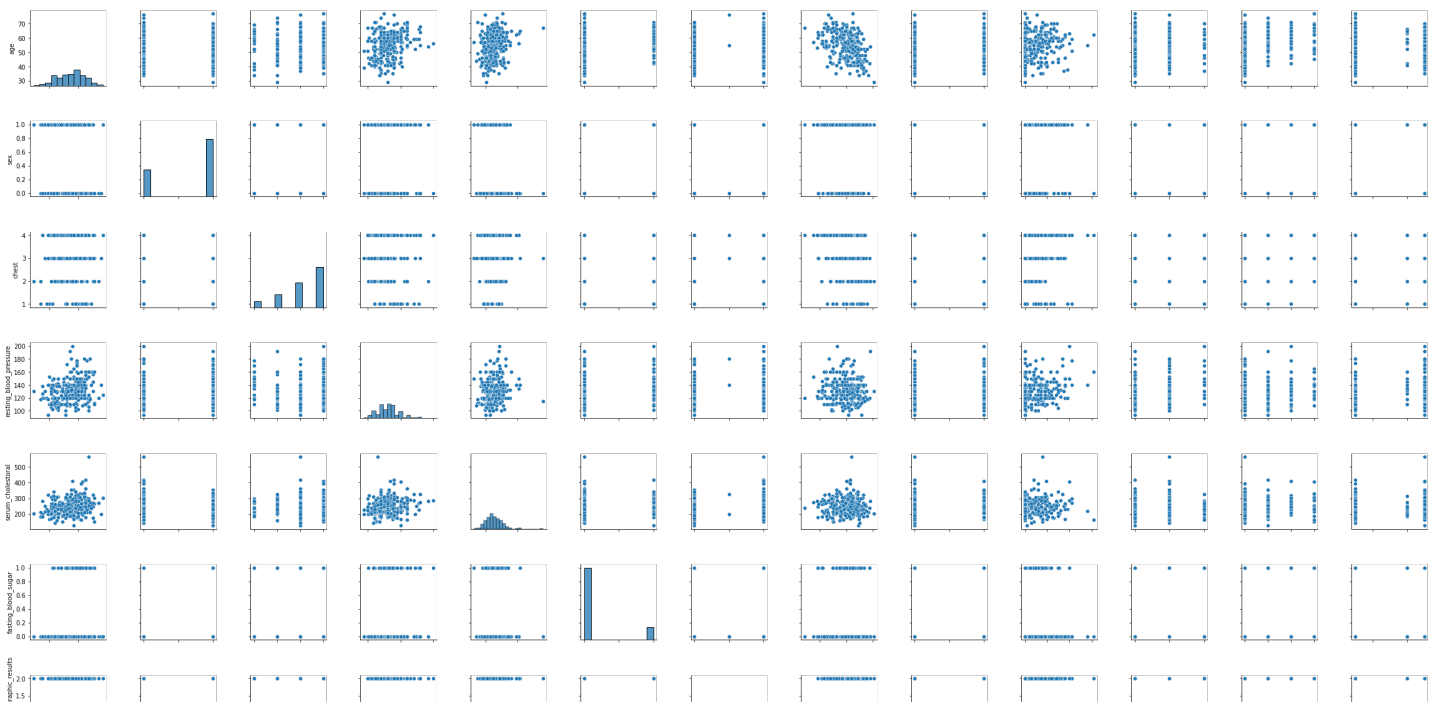
	age	sex	chest	resting_blood_pressure	serum_cholestoral	fasting_blood_sugar	resting_electrocardiographic_results	maxir
0	70	1	4	130	322	0	2	
1	67	0	3	115	564	0	2	
2	57	1	2	124	261	0	0	
3	64	1	4	128	263	0	0	
4	74	0	2	120	269	0	2	

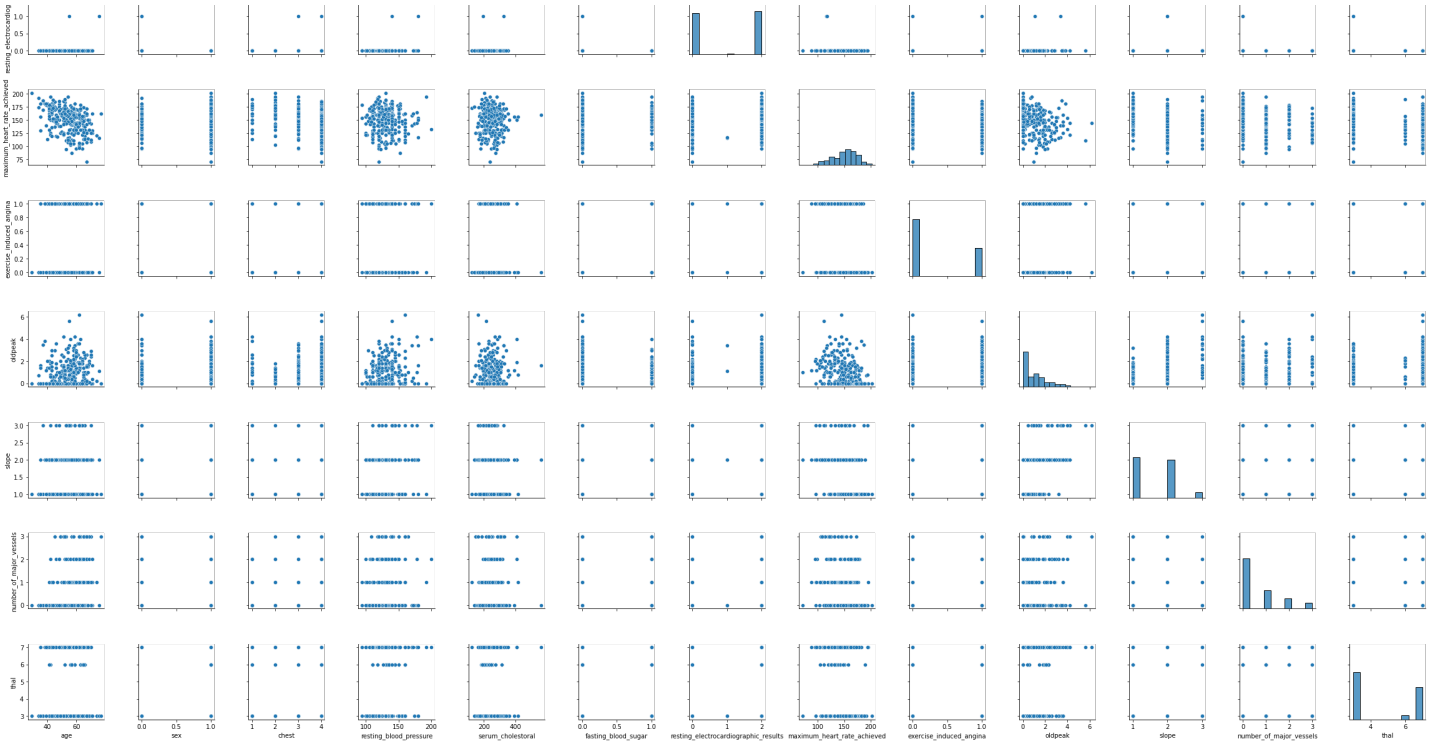
In [ ]:

```
import matplotlib.pyplot as plt
import seaborn as sns
sns.pairplot(df)
```

Out[ ]:

```
<seaborn.axisgrid.PairGrid at 0x7fa80f76c090>
```





confusion matrix target attribute is class

In [ ]:

```
df.shape
```

Out[ ]:

```
(270, 14)
```

In [ ]:

```
df.dtypes
```

Out[ ]:

```
age                                     int64
sex                                     int64
chest                                   int64
resting_blood_pressure                  int64
serum_cholestoral                       int64
fasting_blood_sugar                     int64
resting_electrocardiographic_results    int64
maximum_heart_rate_achieved             int64
exercise_induced_angina                 int64
oldpeak                               float64
slope                                   int64
number_of_major_vessels                 int64
thal                                    int64
class                                  object
dtype: object
```

In [ ]:

```
df1 = pd.read_csv(io.BytesIO(uploaded['heart-statlog_csv (1).csv']))
df1.head()
```

Out[ ]:

	age	sex	chest	resting_blood_pressure	serum_cholestoral	fasting_blood_sugar	resting_electrocardiographic_results	maxir
0	70	1	4	130	322	0	2	
1	67	0	3	115	564	0	2	
2	57	1	2	124	261	0	0	
3	64	1	4	128	263	0	0	

	age	sex	chest	resting_blood_pressure	serum_cholestoral	fasting_blood_sugar	resting_electrocardiographic_results	maxir
4	74	0	2	120	269	0	2	

In [ ]:

```
dummy= pd.get_dummies(df['class'])
df = pd.concat((df1,dummy) , axis=1)
df = df.drop(['present','class' ], axis=1)
df.rename(columns={"absent":"class"}, inplace = True)
df.head()
```

Out[ ]:

	age	sex	chest	resting_blood_pressure	serum_cholestoral	fasting_blood_sugar	resting_electrocardiographic_results	maxir
0	70	1	4	130	322	0	2	
1	67	0	3	115	564	0	2	
2	57	1	2	124	261	0	0	
3	64	1	4	128	263	0	0	
4	74	0	2	120	269	0	2	

In [ ]:

```
from sklearn.neural_network import MLPClassifier
from sklearn.datasets import make_classification
from sklearn.model_selection import train_test_split
df2=df.copy()
X = df2[['age', 'sex', 'chest', 'resting_blood_pressure', 'serum_cholestoral', 'fasting_
blood_sugar' ,'resting_electrocardiographic_results', 'maximum_heart_rate_achieved', 'exe
rcise_induced_angina', 'oldpeak', 'slope', 'number_of_major_vessels', 'thal']]
Y = df2['class']
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, train_size = 0.80)
clf = MLPClassifier(random_state=1, max_iter=300).fit(X_train, Y_train)
clf.predict_proba(X_test)
```

Out[ ]:

```
array([[0.95339675, 0.04660325],
       [0.30897871, 0.69102129],
       [0.77373008, 0.22626992],
       [0.34981534, 0.65018466],
       [0.16556116, 0.83443884],
       [0.26075007, 0.73924993],
       [0.16920934, 0.83079066],
       [0.93345061, 0.06654939],
       [0.11451301, 0.88548699],
       [0.07917015, 0.92082985],
       [0.91353214, 0.08646786],
       [0.1749768 , 0.8250232 ],
       [0.87877317, 0.12122683],
       [0.29169291, 0.70830709],
       [0.95405374, 0.04594626],
       [0.63601254, 0.36398746],
       [0.2715714 , 0.7284286 ],
       [0.12703685, 0.87296315],
       [0.95947824, 0.04052176],
       [0.01629224, 0.98370776],
       [0.22458907, 0.77541093],
       [0.79471413, 0.20528587],
       [0.82170425, 0.17829575],
       [0.28594142, 0.71405858],
       [0.85745635, 0.14254365],
       [0.20823367, 0.79176633],
       [0.1157022 , 0.8842978 ],
       [0.13421946, 0.86578054],
       [0.02763052, 0.97236948],
       [0.30814669, 0.69185331],
       [0.06220627, 0.93779373]
```

```
       [0.87462571, 0.12537429],
       [0.26234081, 0.73765919],
       [0.07471395, 0.92528605],
       [0.3029383 , 0.6970617 ],
       [0.05137776, 0.94862224],
       [0.06244856, 0.93755144],
       [0.6113274 , 0.3886726 ],
       [0.01929344, 0.98070656],
       [0.87553684, 0.12446316],
       [0.88478793, 0.11521207],
       [0.01530858, 0.98469142],
       [0.21152499, 0.78847501],
       [0.29815254, 0.70184746],
       [0.82235076, 0.17764924],
       [0.53460689, 0.46539311],
       [0.0340291 , 0.9659709 ],
       [0.13826474, 0.86173526],
       [0.01563712, 0.98436288],
       [0.45305915, 0.54694085],
       [0.15268013, 0.84731987],
       [0.07060396, 0.92939604],
       [0.89222312, 0.10777688],
       [0.10354575, 0.89645425]])
```

In [ ]:

```
clf.predict(X_test)
```

Out[ ]:

```
array([0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0,
       0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1,
       0, 0, 1, 1, 1, 1, 1, 1, 0, 1], dtype=uint8)
```

In [ ]:

```
Y_predic = clf.predict(X_test)
```

In [ ]:

```
clf.score(X_test, Y_test)
```

Out[ ]:

```
0.8333333333333334
```

In [ ]:

```
X_test.head()
```

Out[ ]:

	age	sex	chest	resting_blood_pressure	serum_cholestoral	fasting_blood_sugar	resting_electrocardiographic_results	ma
268	57	1	4	140	192	0	0	
53	63	0	2	140	195	0	0	
192	54	1	2	108	309	0	0	
32	37	0	3	120	215	0	0	
179	50	1	3	129	196	0	0	

In [ ]:

```
Y_test.head()
```

Out[ ]:

```
268    1
53     1
```

```
192     1
32      1
179     1
Name: class, dtype: uint8
```

Confusion Matrix

In [ ]:

```
from sklearn import metrics
from sklearn.metrics import classification_report, confusion_matrix,accuracy_score,log_lo
ss , roc_auc_score,  precision_score, recall_score, f1_score, matthews_corrcoef
CM=confusion_matrix(Y_test,Y_predic)
sns.heatmap(CM, annot=True)
TN = CM[0][0]
FN = CM[1][0]
TP = CM[1][1]
FP = CM[0][1]
specificity = TN/(TN+FP)
loss_log = log_loss(Y_test, Y_predic)
acc= accuracy_score(Y_test,Y_predic)
roc=roc_auc_score(Y_test, Y_predic)
prec = precision_score(Y_test, Y_predic)
rec = recall_score(Y_test, Y_predic)
f1 = f1_score(Y_test, Y_predic)
mathew = matthews_corrcoef(Y_test,Y_predic)
model_results =pd.DataFrame([['ANN',acc, prec,rec,specificity, f1,roc,mathew,loss_log]])
model_results.columns = ['Model', 'Accuracy','Precision', 'Sensitivity','Specificity' ,'
F1-Score','Recall Score','Mathew coefficient','log_loss']
model_results.head()
```

Out[ ]:

	Model	Accuracy	Precision	Sensitivity	Specificity	F1-Score	Recall Score	Mathew coefficient	log_loss
0	ANN	0.777778	0.777778	0.875	0.636364	0.823529	0.755682	0.533002	7.675402