```
import pandas as pd

from matplotlib import pyplot as plt
```

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

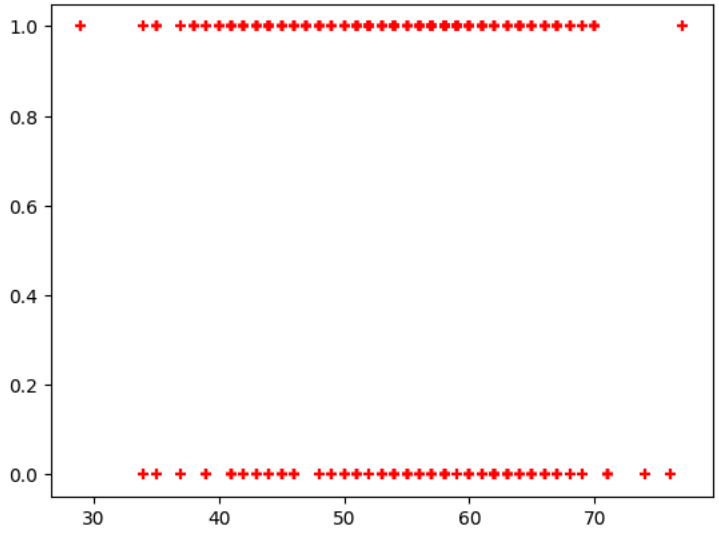
Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving heart.csv to heart.csv

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

```
plt.scatter(df.age,df.sex,marker='+',color='red')
```

```
<matplotlib.collections.PathCollection at 0x7aab49293e90>
```




```
df.shape
```

```
(1025, 14)
```

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

```
x_train,x_test,y_train,y_test = train_test_split(df[['age','sex']],df.target,test_size=0.2)
```

```
x_test
```

	age	sex
782	64	0
734	52	1
203	64	1
75	47	1
774	48	0
...	...	...
531	65	0
934	42	1
333	54	0
454	65	0
968	53	1

205 rows × 2 columns

```
from sklearn.linear_model import LogisticRegression
```

```
model = LogisticRegression()
```

```
model.fit(x_train,y_train)
```

```
▾ LogisticRegression
LogisticRegression()
```

```
model.predict(x_test)
```

```
array([1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0,
       1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0,
       0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1,
       1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 1, 0, 0,
       0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0,
       1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1,
       1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0,
       0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1,
       1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1,
       0, 0, 1, 1, 1, 1, 0])
```

```
model.predict_proba(x_test)
```

```
array([[0.38264809, 0.61735191],
       [0.55642019, 0.44357981],
       [0.73072384, 0.26927616],
       [0.47629885, 0.52370115],
       [0.18135517, 0.81864483],
       [0.75526761, 0.24473239],
       [0.36702381, 0.63297619],
       [0.70468108, 0.29531892],
       [0.66301885, 0.33698115],
       [0.49235776, 0.50764224],
       [0.41345158, 0.58654842],
       [0.71788484, 0.28211516],
       [0.73072384, 0.26927616],
       [0.55642019, 0.44357981],
       [0.2578712 , 0.7421288 ],
       [0.28323849, 0.71676151],
       [0.52448972, 0.47551028],
       [0.35221651, 0.64778349],
       [0.70468108, 0.29531892],
       [0.19109842, 0.80890158],
       [0.69112523, 0.30887477],
       [0.61865974, 0.38134026],
       [0.46028881, 0.53971119],
       [0.67723215, 0.32276785],
       [0.36757649, 0.63242351],
       [0.70468108, 0.29531892],
       [0.69112523, 0.30887477],
       [0.3376879 , 0.6623121 ],
       [0.33822002, 0.66177998],
```

```
        [0.35275933, 0.64724067],
        [0.32398   , 0.67602   ],
        [0.49295219, 0.50704781],
        [0.74318757, 0.25681243],
        [0.50843245, 0.49156755],
        [0.73072384, 0.26927616],
        [0.66301885, 0.33698115],
        [0.66301885, 0.33698115],
        [0.24575782, 0.75424218],
        [0.3979488 , 0.6020512 ],
        [0.42854531, 0.57145469],
        [0.66301885, 0.33698115],
        [0.79965255, 0.20034745],
        [0.66301885, 0.33698115],
        [0.63371039, 0.36628961],
        [0.66301885, 0.33698115],
        [0.35275933, 0.64724067],
        [0.64850451, 0.35149549],
        [0.35275933, 0.64724067],
        [0.67723215, 0.32276785],
        [0.61865974, 0.38134026],
        [0.13839146, 0.86160854],
        [0.63371039, 0.36628961],
        [0.73072384, 0.26927616],
        [0.11047169, 0.88952831],
        [0.12375701, 0.87624299],
        [0.20123646, 0.79876354],
        [0.2578712 , 0.7421288 ],
        [0.66301885, 0.33698115],
```

```
y_pred = model.predict(x_test)
var = model.score(x_test,y_test)
print('accuracy of the logistic regression classifier on the test set:(:,2f)',format(var))
```

```
accuracy of the logistic regression classifier on the test set:(:,2f) 0.6731707317073171
```

```
from sklearn.metrics import confusion_matrix
confusion_matrix=confusion_matrix(y_test,y_pred)
print(confusion_matrix)
```

```
[[64 32]
 [35 74]]
```

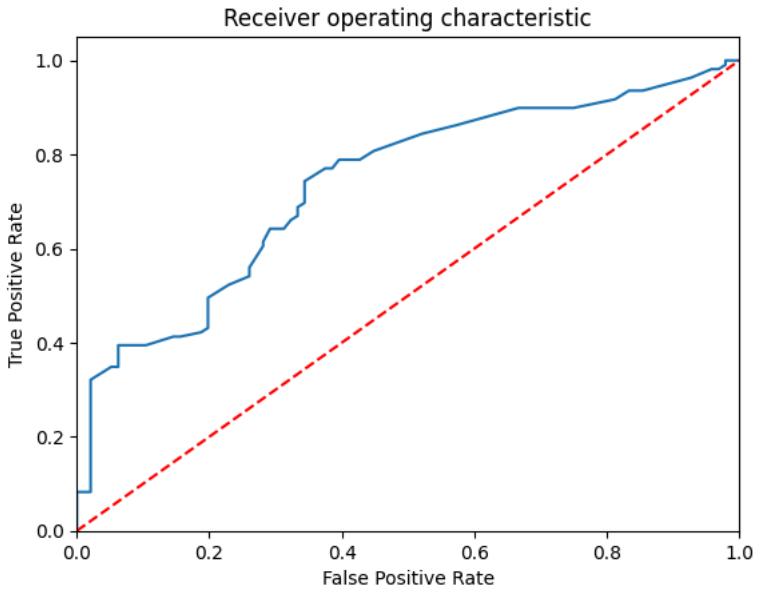
```
from sklearn.metrics import classification_report
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.65      0.67      0.66        96
           1       0.70      0.68      0.69       109

    accuracy                           0.67       205
   macro avg       0.67      0.67      0.67       205
weighted avg       0.67      0.67      0.67       205
```

```
from sklearn.metrics import roc_auc_score
from sklearn.metrics import roc_curve
logit_roc_auc = roc_auc_score(y_test, model.predict(x_test))
fpr, tpr, thresholds = roc_curve(y_test, model.predict_proba(x_test)[:,1])
plt.figure()
plt.plot(fpr, tpr, label='Logistic Regression (area = %0.2f)' % logit_roc_auc)
plt.plot([0, 1], [0,1],'r--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver operating characteristic')
plt.savefig('Log_ROC')
plt.show()
```




```
print(df.columns)
```

```
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
      dtype='object')
```