```
import numpy as np
import pandas as pd

hrt=pd.read_csv('/content/heart_miss.csv')
hrt.head()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope
0	63	1	3	145	233	1.0	0	150	0	2.3	0
1	37	1	2	130	250	0.0	1	187	0	3.5	0
2	41	0	1	130	204	0.0	0	172	0	1.4	2
3	56	1	1	120	236	0.0	1	178	0	0.8	2
4	57	0	0	120	354	0.0	1	163	1	0.6	2

```
hrt.isna().sum()
```

```
age         0
sex         0
cp          0
trestbps    0
chol        0
fbs         5
restecg     0
thalach     0
exang       0
oldpeak     0
slope       0
ca          0
thal        6
target      0
dtype: int64
```

```
hrt.describe()
```

	age	sex	cp	trestbps	chol	f
count	303.000000	303.000000	303.000000	303.000000	303.000000	298.0000
mean	54.366337	0.683168	0.966997	131.623762	246.264026	0.1510
std	9.082101	0.466011	1.032052	17.538143	51.830751	0.3586
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.0000
25%	47.500000	0.000000	0.000000	120.000000	211.000000	0.0000
50%	55.000000	1.000000	1.000000	130.000000	240.000000	0.0000
75%	61.000000	1.000000	2.000000	140.000000	274.500000	0.0000
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.0000

✓ 0s completed at 9:56 PM

```
hrt.dtypes
```

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

```
hrt['target'].value_counts()
```

```
1    165
0    138
Name: target, dtype: int64
```

```
hrt['fbs'].fillna(hrt['fbs'].mean(),inplace=True)
```

```
hrt['thal'].fillna(hrt['thal'].mean(),inplace=True)
```

```
x=hrt.iloc[:,:-1].values
y=hrt.iloc[:,-1].values
```

```
from sklearn.model_selection import train_test_split
xtr,xts,ytr,yts=train_test_split(x,y,test_size=0.30,random_state=42)
```

```
# from sklearn.preprocessing import StandardScaler
# std=StandardScaler()
# std.fit(xtr)
# xtr=std.transform(xtr)
# xts=std.transform(xts)
```

```
#another method for scaling

from sklearn.preprocessing import MinMaxScaler
mn=MinMaxScaler()
mn.fit(xtr)
xtr=mn.transform(xtr)
xts=mn.transform(xts)
```

```
from sklearn.svm import SVC
sv=SVC()
sv.fit(xtr,ytr)
ypr=sv.predict(xts)
ypr
```

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

```
from sklearn.metrics import accuracy_score,confusion_matrix,classification_repor
score=accuracy_score(yts,ypr)
score
```

```
0.8241758241758241
```

```
print(confusion_matrix(yts,ypr))
```

```
[[33  8]
 [ 8 42]]
```

```
print(classification_report(yts,ypr))
```

```
                precision    recall  f1-score   support

             0       0.80      0.80      0.80        41
             1       0.84      0.84      0.84        50

      accuracy                           0.82        91
     macro avg       0.82      0.82      0.82        91
  weighted avg       0.82      0.82      0.82        91
```