```
import pandas as pd
import numpy as np
df=pd.read_csv('/content/boston.csv')
```

```
df.head()
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296.0	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242.0	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242.0	17.8	392.83	4.03	34.7
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222.0	18.7	394.63	2.94	33.4
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222.0	18.7	396.90	5.33	36.2

```
from sklearn.preprocessing import LabelEncoder

encoder = LabelEncoder()

for col in df.select_dtypes(include=['object']).columns:
    df[col] = encoder.fit_transform(df[col])
```

```
x = df.iloc[:,0]
y = df.iloc[:,1]
```

```
x = np.array(x)
y = np.array(y)
```

```
x = x.reshape(-1,1)
y = y.reshape(-1,1)
```

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

```
x_train, x_test, y_train, y_test = train_test_split(
     x, y, test_size=0.5, random_state=42
)
print("Training sample:",x_train.shape)
print("testing sample:",x_test.shape)
```

```
Training sample: (253, 1)
testing sample: (253, 1)
```

```
from sklearn.svm import SVC
```

```
svm_liner = SVC(kernel='RBF', degree=2, class_weight='balanced')
```

```
from sklearn.svm import SVR
svr_model = SVR(kernel='rbf', degree=2, gamma=0.1) # Removed class_weight as it's not applicable to SVR
svr_model.fit(x_train, y_train.ravel())
```

```
SVR
SVR(degree=2, gamma=0.1)
```

```
y_pred = svr_model.predict(x_test)
```

```
from sklearn.metrics import accuracy_score,confusion_matrix,precision_score,recall_score,f1_score
```

```
from sklearn.metrics import r2_score

r2 = r2_score(y_test, y_pred)
print("R-squared :", r2)
```

```
R-squared : -0.20713575507871877
```

```
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score

# Convert continuous values to binary classes
```

```
threshold = np.mean(y_test)

y_test_cls = (y_test >= threshold).astype(int)
y_pred_cls = (y_pred >= threshold).astype(int)

cm = confusion_matrix(y_test_cls, y_pred_cls)
print("Confusion Matrix:")
print(cm)

print("Accuracy:", accuracy_score(y_test_cls, y_pred_cls))
print("Precision:", precision_score(y_test_cls, y_pred_cls))
print("Recall:", recall_score(y_test_cls, y_pred_cls))
print("F1 Score:", f1_score(y_test_cls, y_pred_cls))
```

```
Confusion Matrix:
[[189   0]
 [ 64   0]]
Accuracy: 0.7470355731225297
Precision: 0.0
Recall: 0.0
F1 Score: 0.0
/usr/local/lib/python3.12/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning: Precision is ill-de
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```

```
# precision = precision_score(y_test,y_pred,zero_division=0)
# print("Precision :",precision)
# This cell is commented out because precision_score is not applicable for regression problems with continuous outputs.
```

```
import seaborn as sns
import matplotlib.pyplot as plt
plt.figure(figsize=(6,4))
sns.heatmap(cm,annot=True,fmt="d",cmap="rainbow",
            xticklabels=['Predicted 1','Predicted 0'],
            yticklabels=['Actual 1','Actual 0'])
plt.xlabel("Predicted Labels")
plt.ylabel("Actual Labels")
plt.title("Confusion Matrix")
plt.show()
```