10) **Program to show Cross Validation**

```
#This program shows Cross Validation
print('This program shows Cross Validation')
from sklearn import datasets
import pandas as pd
iris=datasets.load_iris()

# df will fold dataset as a table
df=pd.DataFrame(
  iris.data,
  columns=iris.feature_names
   )
#labels are assigned to df[target] table or array
df['target']=pd.Series(
    iris.target
   )

from sklearn.model_selection import train_test_split




from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import accuracy_score
from sklearn.model_selection import ShuffleSplit
from sklearn.model_selection import cross_validate


clf=DecisionTreeClassifier()
```

```
rs=ShuffleSplit(n_splits=100,test_size=0.3)

x=df[iris.feature_names]
y=df['target']

print('Accuracy for different iterations')
cv_results=cross_validate(
   clf,x,y,cv=rs,scoring='accuracy'
   )  # put scoring='precision_macro'  scoring='recall_macro'
accuracy_scores=pd.Series(cv_results['test_score'])

print(accuracy_scores)



print('Precision for different iterations')
cv_results=cross_validate(
   clf,x,y,cv=rs,scoring='precision_macro'
   )  # put scoring='precision_macro'  scoring='recall_macro'
accuracy_scores=pd.Series(cv_results['test_score'])

print(accuracy_scores)

print('Recall for different iterations')
cv_results=cross_validate(
   clf,x,y,cv=rs,scoring='recall_macro'
   )  # put scoring='precision_macro'  scoring='recall_macro'
accuracy_scores=pd.Series(cv_results['test_score'])

print(accuracy_scores)
```

Output

This program shows Cross Validation

Accuracy for different iterations

0 0.955556

1 0.977778

2 0.977778

3 0.955556

4 0.977778

...

95 0.955556

96 0.933333

97 0.933333

98 0.933333

99 0.977778

Length: 100, dtype: float64

Precision for different iterations

0 0.950549

1 0.948718

2 0.981481

3 0.979167

4 0.958333

...

95 0.936508

96 0.916340

97 0.921569

98 0.984127

99 0.941176

Length: 100, dtype: float64

Recall for different iterations

0 0.972222

1 0.962963

```
2     0.944079
3     0.933333
4     0.911111
      ...
95    0.948718
96    0.952137
97    0.907407
98    0.944056
99    0.917065
Length: 100, dtype: float64
>>>
```

11) #Program on regression

```
import numpy as np
n=200 #200 samples

height_pop1_f=np.random.normal(loc=155,scale=10,size=n)
height_pop1_m=np.random.normal(loc=175,scale=5,size=n)

height_pop2_f=np.random.normal(loc=165,scale=10,size=n)
height_pop2_m=np.random.normal(loc=185,scale=5,size=n)

height_f=np.concatenate([height_pop1_f,height_pop2_f])
height_m=np.concatenate([height_pop1_m,height_pop2_m])

import pandas as pd
import matplotlib.pyplot as plt

from sklearn.tree import export_text
from sklearn.tree import DecisionTreeRegressor
from sklearn.model_selection import train_test_split
```

```
df_height=pd.DataFrame(
   {
      'Gender':[1 for i in range(height_f.size)]+
            [2 for i in range(height_m.size)],
      'Height':np.concatenate((height_f,height_m))
      }
   )
fig, ax=plt.subplots(1,1,figsize=(10,5))

df_height[df_height['Gender']==1]['Height'].plot(
   label='Female',kind='hist',
   bins=10,alpha=0.7,ax=ax
   )
df_height[df_height['Gender']==2]['Height'].plot(
   label='Male',kind='hist',
   bins=10,alpha=0.7,ax=ax
   )
ax.legend()
fig.show()
# to calculate mean and median of height
df_height.groupby('Gender')[['Height']].agg([np.mean,np.median]).round(1)
df_train,df_test=train_test_split(df_height,test_size=0.3)
x_train,x_test=df_train[['Gender']],df_test[['Gender']]
y_train,y_test=df_train['Height'],df_test['Height']

for criterion in['mse','mae']:
   rgrsr=DecisionTreeRegressor(criterion=criterion)
   rgrsr.fit(x_train,y_train)
   print(f'criterion={criterion}:\n')
   print(export_text(rgrsr,feature_names=['Gender'],spacing=3,decimals=1))
```

print('Program Executed successfully')

OUTPUT

criterion=mse:

|--- Gender <= 1.5

| |--- value: [159.9]

|--- Gender > 1.5

| |--- value: [180.1]

criterion=mae:

|--- Gender <= 1.5

| |--- value: [160.0]

|--- Gender > 1.5

| |--- value: [180.2]

Program Executed successfully