EXPERIMENT-4

AIM:

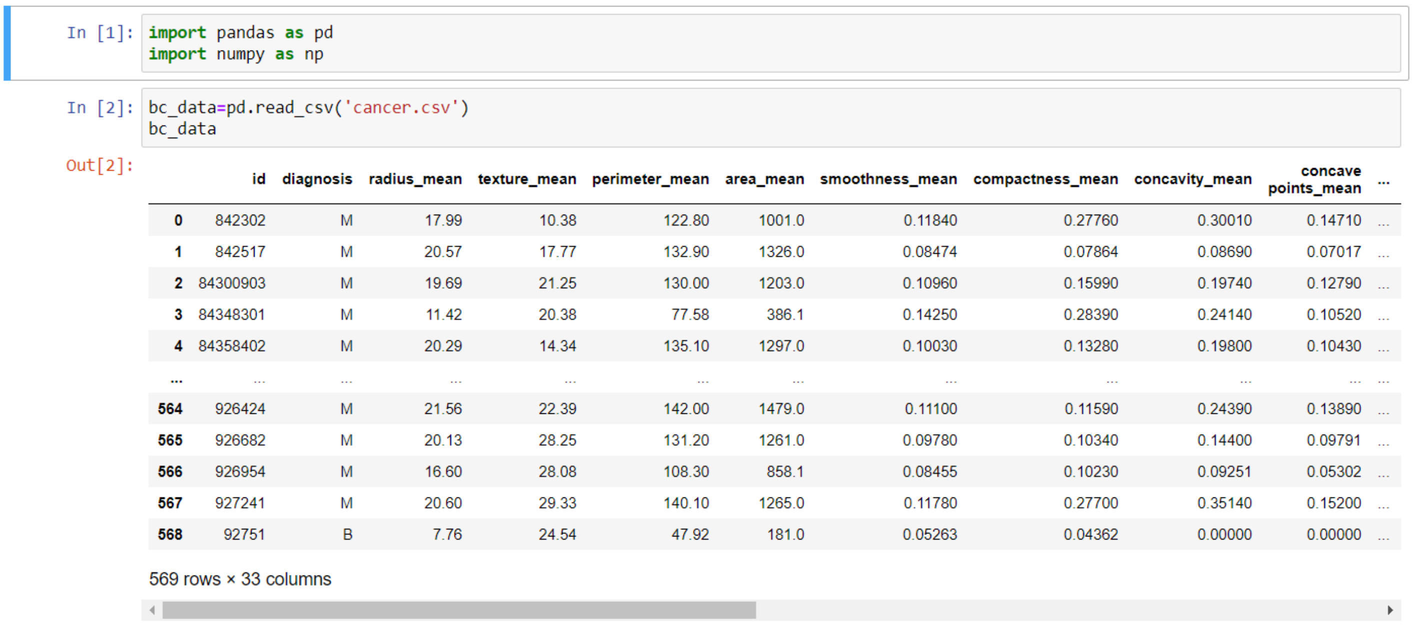
Estimate the precision recall accuracy f-measure of the decision classifier on a breast cancer dataset using 10 fold cross validation.

ALGORITHM:

1. Select the best attribute using Attribute Selection Measures (ASM) to split the records.
2. Make that attribute a decision node and breaks the dataset into smaller subsets.
3. Starts tree building by repeating this process recursively for each child until one of the conditions will match:
   a. All the tuples belong to the same attribute value.
   b. There are no more remaining attributes.
   c. There are no more instances.

PROGRAM CODE SNIPPET:

LOADING DATA SET:



In [1]:
```
import pandas as pd
import numpy as np
```

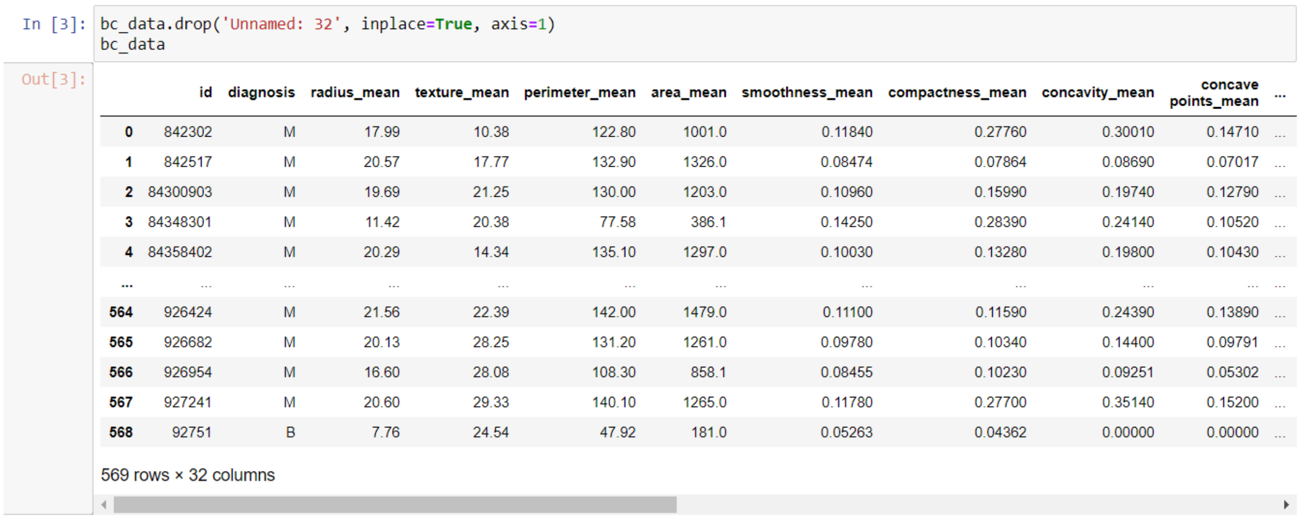
In [2]:
```
bc_data=pd.read_csv('cancer.csv')
bc_data
```

Out[2]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	...
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.30010	0.14710	...
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.08690	0.07017	...
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.19740	0.12790	...
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.24140	0.10520	...
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.19800	0.10430	...
...	...	...	...	...	...	...	...	...	...	...	...
564	926424	M	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	0.13890	...
565	926682	M	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	0.09791	...
566	926954	M	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	0.05302	...
567	927241	M	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	0.15200	...
568	92751	B	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	0.00000	...

569 rows × 33 columns

PREPROCESSING:

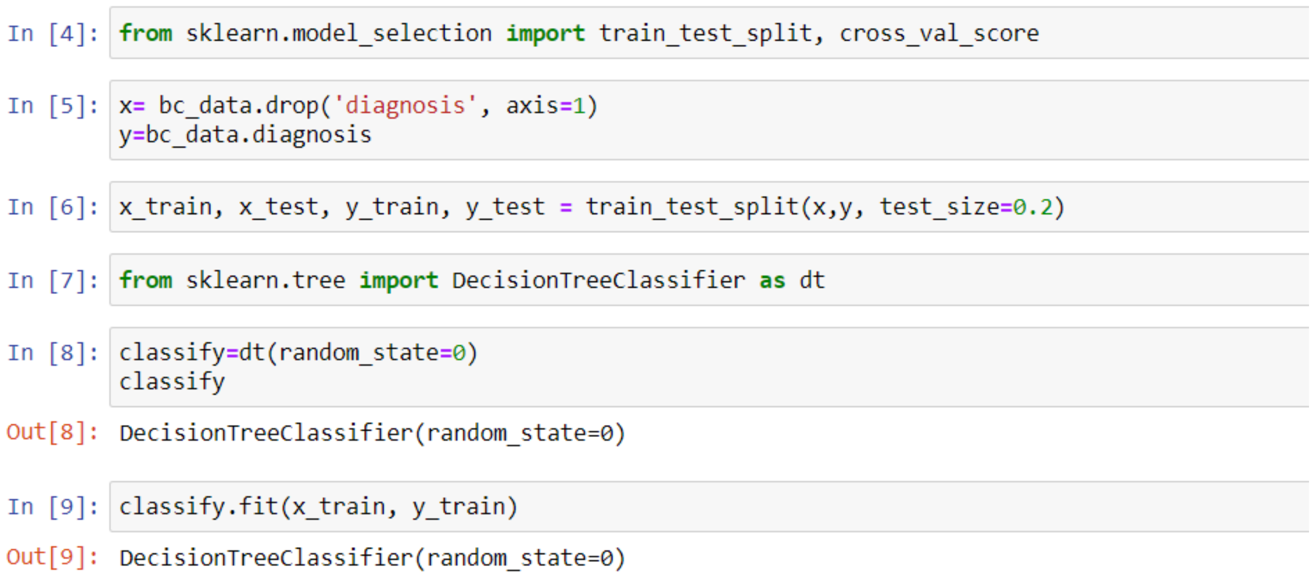


```
In [3]: bc_data.drop('Unnamed: 32', inplace=True, axis=1)
        bc_data
```

Out[3]:

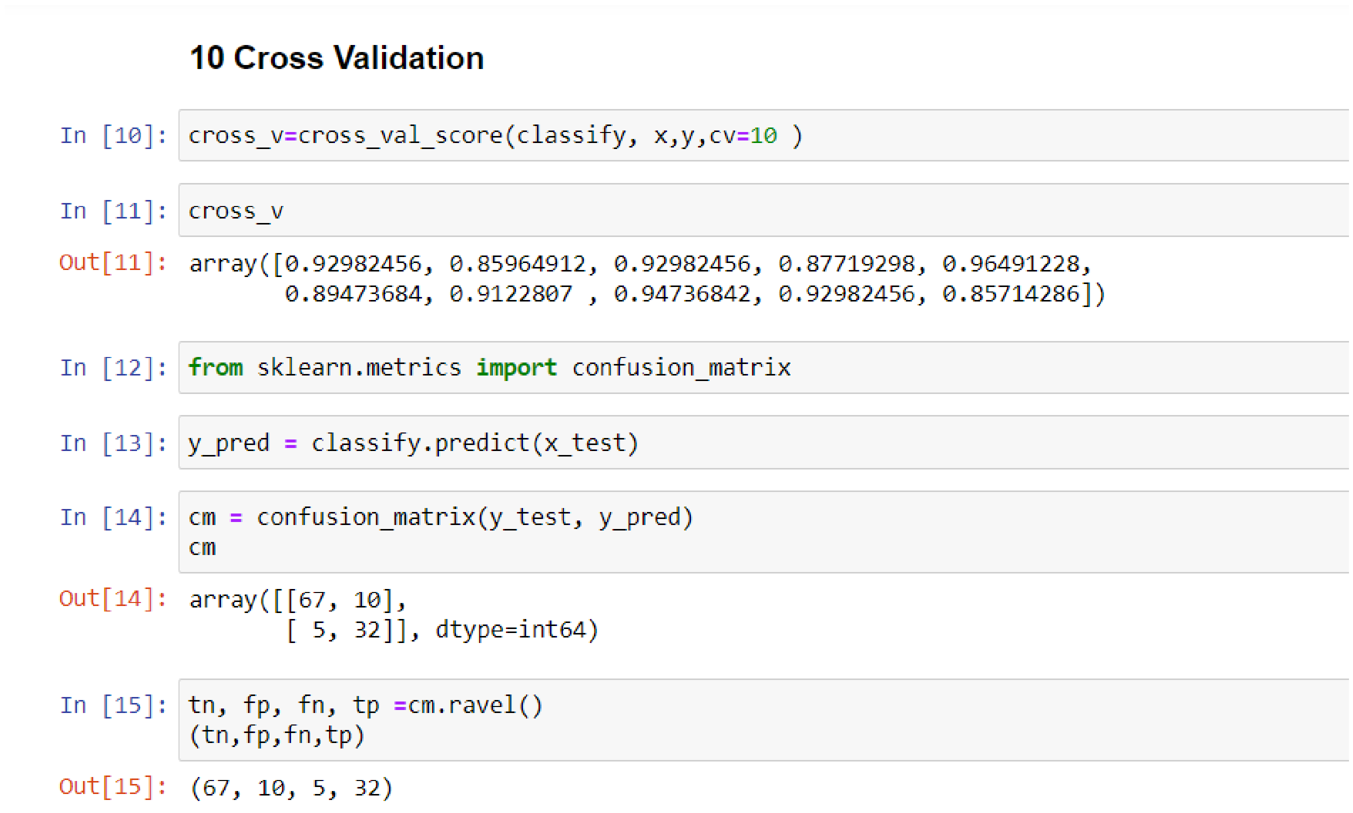
	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	...
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.30010	0.14710	...
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.08690	0.07017	...
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.19740	0.12790	...
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.24140	0.10520	...
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.19800	0.10430	...
...	...	...	...	...	...	...	...	...	...	...	...
564	926424	M	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	0.13890	...
565	926682	M	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	0.09791	...
566	926954	M	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	0.05302	...
567	927241	M	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	0.15200	...
568	92751	B	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	0.00000	...

569 rows × 32 columns



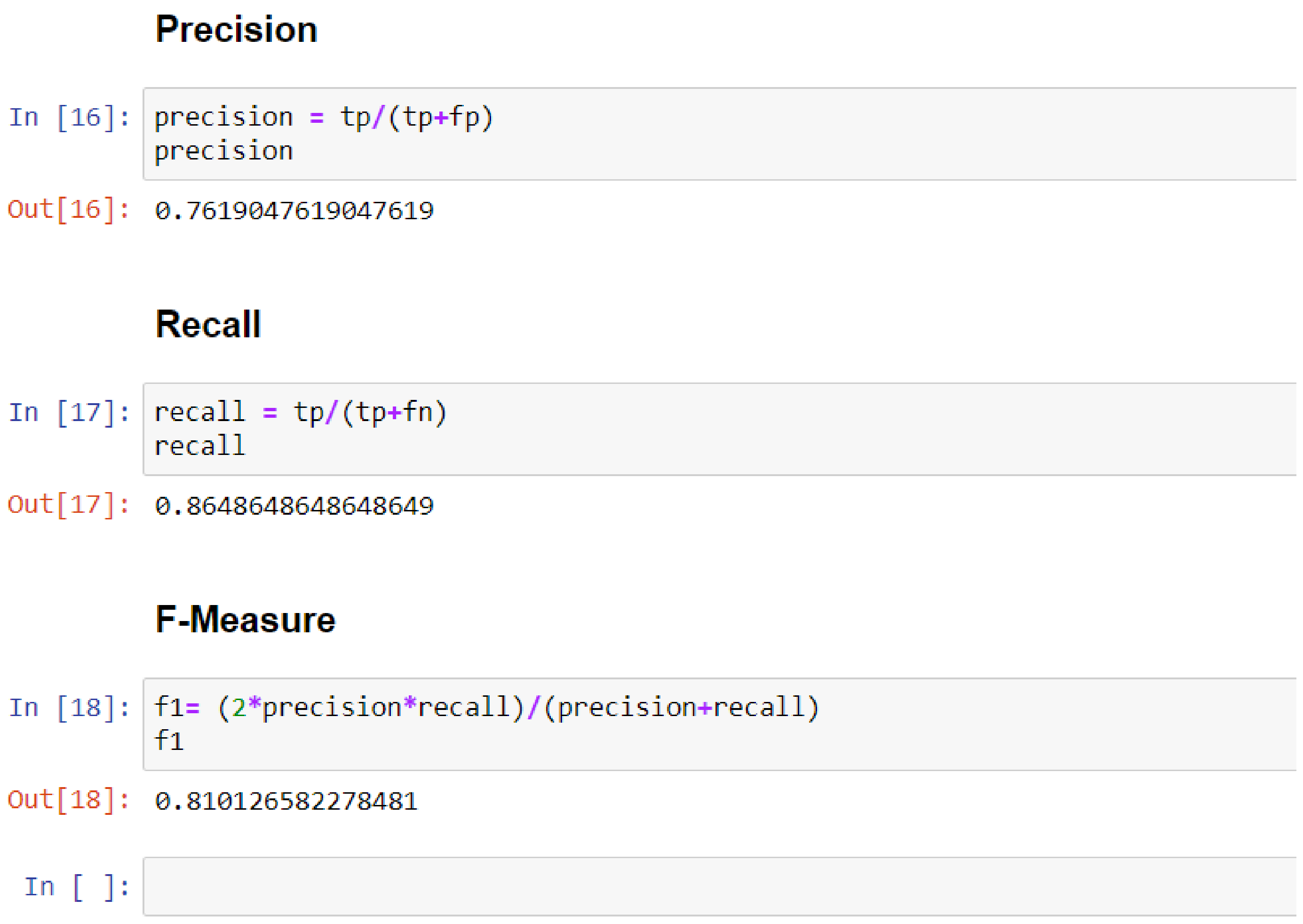
```
In [4]: from sklearn.model_selection import train_test_split, cross_val_score

In [5]: x= bc_data.drop('diagnosis', axis=1)
        y=bc_data.diagnosis

In [6]: x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=0.2)

In [7]: from sklearn.tree import DecisionTreeClassifier as dt

In [8]: classify=dt(random_state=0)
        classify

Out[8]: DecisionTreeClassifier(random_state=0)

In [9]: classify.fit(x_train, y_train)

Out[9]: DecisionTreeClassifier(random_state=0)
```

ML ALGORITHM IMPLEMENTATION:



10 Cross Validation

```
In [10]: cross_v=cross_val_score(classify, x,y,cv=10 )

In [11]: cross_v

Out[11]: array([0.92982456, 0.85964912, 0.92982456, 0.87719298, 0.96491228,
                0.89473684, 0.9122807 , 0.94736842, 0.92982456, 0.85714286])

In [12]: from sklearn.metrics import confusion_matrix

In [13]: y_pred = classify.predict(x_test)

In [14]: cm = confusion_matrix(y_test, y_pred)
         cm

Out[14]: array([[67, 10],
                [ 5, 32]], dtype=int64)

In [15]: tn, fp, fn, tp =cm.ravel()
         (tn,fp,fn,tp)

Out[15]: (67, 10, 5, 32)
```



Precision

```
In [16]: precision = tp/(tp+fp)
         precision
```

```
Out[16]: 0.7619047619047619
```

Recall

```
In [17]: recall = tp/(tp+fn)
         recall
```

```
Out[17]: 0.8648648648648649
```

F-Measure

```
In [18]: f1= (2*precision*recall)/(precision+recall)
         f1
```

```
Out[18]: 0.810126582278481
```

```
In [ ]:
```

GITHUB LINK:

https://github.com/Harnam99/Experiment-No.4.git