Name: HARNAM KAUR **Enrolment No: 00423207218** **Class: CSE1**

MACHINE LEARNING LAB (EXPERIMENT NO 2)

AIM: STUDY AND IMPLEMENT THE DECISION TREE CLASSIFIER ON THE BREAST CANCER DATASET USING SCIKIT LEARN IN PYTHON.

Program Code Snippet

Loading Dataset

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

df = pd.read_csv("C:\\Users\\Ad\\Desktop\\Python_LabML\\experiment2\\cancer.csv")
df
```

Out[1]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	...	f
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

In [53]:
```
df.head(10)
```

Out[53]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concav points_mea
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.30010	0.147
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.08690	0.070
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.19740	0.127
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.24140	0.105
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.19800	0.104
5	843786	M	12.45	15.70	82.57	477.1	0.12780	0.17000	0.15780	0.080
6	844359	M	18.25	19.98	119.60	1040.0	0.09463	0.10900	0.11270	0.074
7	84458202	M	13.71	20.83	90.20	577.9	0.11890	0.16450	0.09366	0.059
8	844981	M	13.00	21.82	87.50	519.8	0.12730	0.19320	0.18590	0.093
9	84501001	M	12.46	24.04	83.97	475.9	0.11860	0.23960	0.22730	0.085

10 rows × 33 columns

In [54]:
```
#to read the last end of data
df.tail()
```

Out[54]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concav points_mea
564	926424	M	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	0.138
565	926682	M	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	0.097
566	926954	M	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	0.053
567	927241	M	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	0.152
568	92751	B	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	0.000

5 rows × 33 columns

Preprocessing/Cleaning of dataset

In [60]:
```
for i in df.columns:
    print(i)
    print(df[i].value_counts())
    print('----------------------*********----------------------------')
```

```
id
883263     1
906564     1
89122      1
9013579    1
868682     1
          ..
874158     1
914062     1
918192     1
872113     1
875878     1
Name: id, Length: 569, dtype: int64
----------------------*********----------------------------
diagnosis
B    357
M    212
Name: diagnosis, dtype: int64
----------------------*********----------------------------
radius_mean
12.34    4
12.77    3
15.46    3
12.89    3
13.05    3
        ..
12.31    1
18.81    1
13.30    1
23.09    1
18.25    1
Name: radius_mean, Length: 456, dtype: int64
----------------------*********----------------------------
texture_mean
14.93    3
15.70    3
18.90    3
16.84    3
17.46    3
        ..
20.53    1
17.66    1
24.80    1
20.56    1
10.94    1
Name: texture_mean, Length: 479, dtype: int64
----------------------*********----------------------------
perimeter_mean
82.61     3
134.70    3
87.76     3
130.00    2
58.79     2
          ..
70.21     1
```

In [63]:
```
df = df.drop(["Unnamed: 32"], axis = 1)
df
```

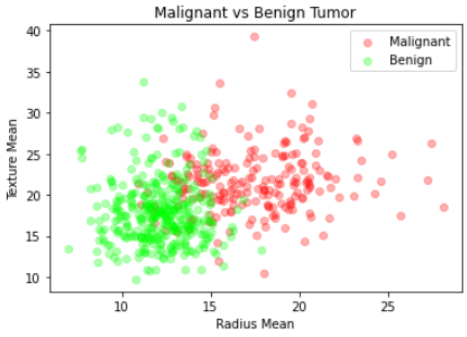
Out[63]:

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

569 rows × 31 columns

Visualization

In [75]:
```
plt.title("Malignant vs Benign Tumor")
plt.xlabel("Radius Mean")
plt.ylabel("Texture Mean")
plt.scatter(M.radius_mean, M.texture_mean, color = "red", label = "Malignant", alpha = 0.3)
plt.scatter(B.radius_mean, B.texture_mean, color = "lime", label = "Benign", alpha = 0.3)
plt.legend()
plt.show()
```



ML algorithm implementation of prediction or comparison

Decision tree models where the target variable uses a discrete set of values are classified as Classification Trees. In these trees, each node, or leaf, represent class labels while the branches represent conjunctions of features leading to class labels.

A decision tree where the target variable takes a continuous value, usually numbers, are called Regression Trees. The two types are commonly referred to together at CART (Classification and Regression Tree).

In [80]:
```
from sklearn.model_selection import train_test_split

#for checking testing results
from sklearn.metrics import classification_report, confusion_matrix

#for visualizing tree
from sklearn.tree import plot_tree

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state = 0)

print("Training split input- ", x_train.shape)
print("Testing split input- ", x_test.shape)
```

```
Training split input-  (455, 10)
Testing split input-  (114, 10)
```

In [81]:
```
from sklearn.tree import DecisionTreeClassifier
```

In [82]:
```
dt = DecisionTreeClassifier()
dt.fit(x_train, y_train)
```

Out[82]:
```
DecisionTreeClassifier()
```

ROC/AUC/Confusion matrix

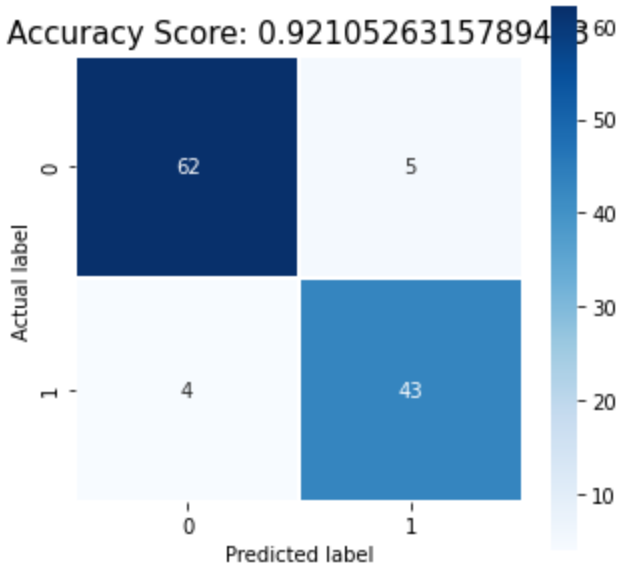
In [83]:
```
y_pred = dt.predict(x_test)
print("Classification report - \n", classification_report(y_test,y_pred))
```

```
Classification report -
               precision    recall  f1-score   support

           B       0.94      0.94      0.94        67
           M       0.91      0.91      0.91        47

    accuracy                           0.93       114
   macro avg       0.93      0.93      0.93       114
weighted avg       0.93      0.93      0.93       114
```

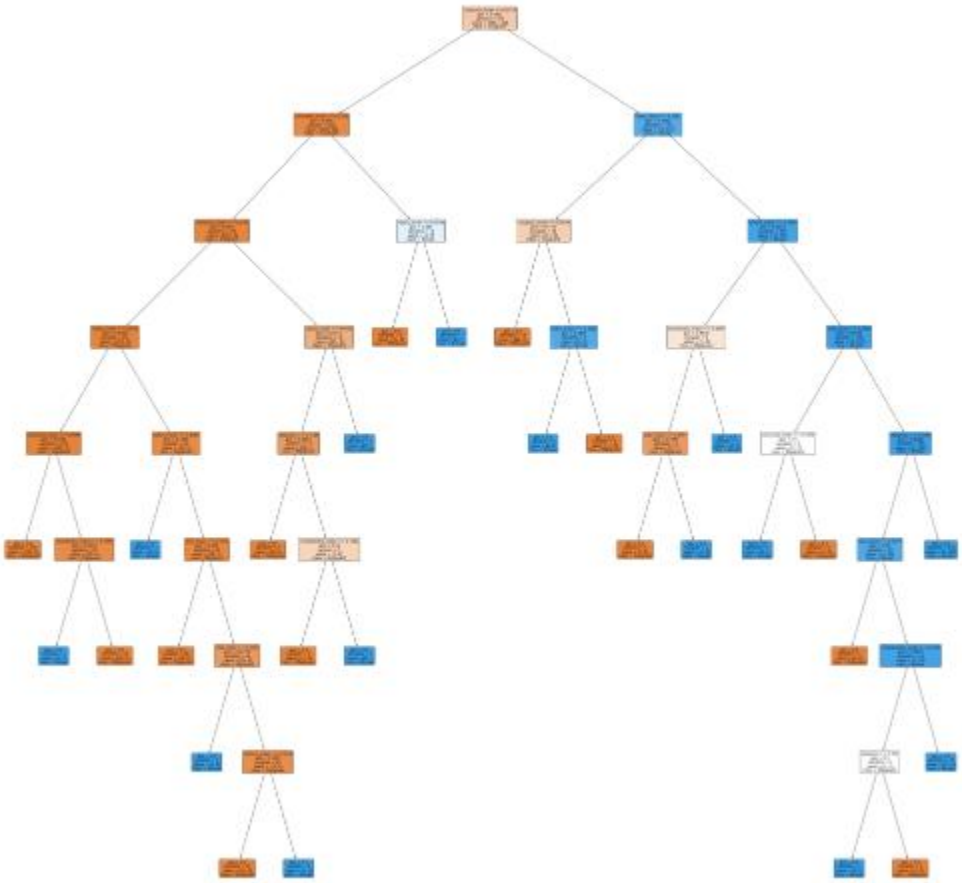
In [84]:
```
cm=confusion_matrix(y_test,y_pred)
cm
```

Out[84]:
```
array([[63,  4],
       [ 4, 43]], dtype=int64)
```

In [85]:
```
plt.figure(figsize=(5,5))

sns.heatmap(data=cm,linewidths=1.0, annot=True,square = True,  cmap = 'Blues')

plt.ylabel('Actual label')
plt.xlabel('Predicted label')

all_sample_title = 'Accuracy Score: {0}'.format(dt.score(x_test, y_test))
plt.title(all_sample_title, size = 15)

plt.savefig("D:/accu.png")
```


Accuracy Score: 0.92105263157894
Actual label
Predicted label
62
5
4
43
0
1
0
1
60
50
40
30
20
10

Final graph



Github Link of Jupyter Notbook:

https://github.com/Harnam99/Lab-Program-2-

http://localhost:8973/notebooks/Experiment%20No2.ipynb#

https://github.com/Harnam99/Lab-Program-2-/blob/main/Experiment%20No2.ipynb