HW3 by Sirawitch Chairuangsirikul (6631350521)

0. Prepare datas

In [44]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [45]:
```
df = pd.read_csv('https://raw.githubusercontent.com/davidjohnnn/all_datasets/master/bay/kyphosis.c
```

In [46]:
```
df.head()
```

Out[46]:

	Kyphosis	Age	Number	Start
0	absent	71	3	5
1	absent	158	3	14
2	present	128	4	5
3	absent	2	5	1
4	absent	1	4	15

0.1 Train/Test split

In [47]:
```
from sklearn.model_selection import train_test_split
```

In [48]:
```
X = df.drop('Kyphosis',axis=1)
y = df['Kyphosis']
```

In [49]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, stratify=y, test_size=0.30, random_state=42)
```

1. Decison Tree

In [50]:
```
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import GridSearchCV # library for hyperparameter tuning for models
from sklearn.metrics import classification_report,confusion_matrix
```

In [82]:
```
dtree = DecisionTreeClassifier(criterion='entropy',
                               max_depth=4,
                               min_samples_leaf=10,
                               ccp_alpha=0.01,
                               random_state=42)
dtree.fit(X_train, y_train)
```

Out[82]:

DecisionTreeClassifier

```
DecisionTreeClassifier(ccp_alpha=0.01, criterion='entropy', max_depth=4,
                       min_samples_leaf=10, random_state=42)
```

In [83]:
```
predictions = dtree.predict(X_test)
```

In [84]:
```
print(classification_report(y_test,predictions,digits=4))
```

```
              precision    recall  f1-score   support

      absent     0.8500    0.8500    0.8500        20
     present     0.4000    0.4000    0.4000         5

    accuracy                         0.7600        25
   macro avg     0.6250    0.6250    0.6250        25
weighted avg     0.7600    0.7600    0.7600        25
```

In [85]:
```
print(confusion_matrix(y_test,predictions,labels=['absent','present']))
```

```
[[17  3]
 [ 3  2]]
```

2. Random Forest

In [37]:
```
from sklearn.ensemble import RandomForestClassifier
```

In [91]:
```
rfc = RandomForestClassifier(n_estimators=100,
                             max_depth=4,
                             class_weight='balanced',
                             ccp_alpha=0.01,
                             random_state=42)
rfc.fit(X_train, y_train)
```

Out[91]:

RandomForestClassifier

```
RandomForestClassifier(ccp_alpha=0.01, class_weight='balanced', max_depth
=4,
                       random_state=42)
```

In [92]:
```
predictions2 = rfc.predict(X_test)
```

In [93]:
```
print(classification_report(y_test,predictions2,digits=4))
```

```
              precision    recall  f1-score   support

      absent     0.8947    0.8500    0.8718        20
     present     0.5000    0.6000    0.5455         5

    accuracy                         0.8000        25
   macro avg     0.6974    0.7250    0.7086        25
weighted avg     0.8158    0.8000    0.8065        25
```

In [94]:
```
print(confusion_matrix(y_test,predictions2,labels=['absent','present']))
```

```
[[17  3]
 [ 2  3]]
```

Compare

In [95]:
```
print("Decision Tree Results:")
print(classification_report(y_test, dtree.predict(X_test)))
```

```
print("\nRandom Forest Results:")
print(classification_report(y_test, rfc.predict(X_test)))
```

Decision Tree Results:

	precision	recall	f1-score	support
absent	0.85	0.85	0.85	20
present	0.40	0.40	0.40	5
accuracy			0.76	25
macro avg	0.62	0.62	0.62	25
weighted avg	0.76	0.76	0.76	25

Random Forest Results:

	precision	recall	f1-score	support
absent	0.89	0.85	0.87	20
present	0.50	0.60	0.55	5
accuracy			0.80	25
macro avg	0.70	0.72	0.71	25
weighted avg	0.82	0.80	0.81	25

จากการทดลองเปรียบเทียบ พบว่า Random Forest **มีประสิทธิภาพสูงกว่า** Decision Tree **ในทุกมิติ โดยเฉพาะค่า** Macro-F1 Score **ที่เพิ่มขึ้นจาก** 0.62 **เป็น** 0.71