Decision tree

1. Import and visualize data set

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

In [2]:
```
df=pd.read_csv('kyphosis.csv')
```

In [3]:
```
df.head()
```

Out[3]:

	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

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 81 entries, 0 to 80
Data columns (total 4 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   Kyphosis  81 non-null     object
 1   Age       81 non-null     int64 
 2   Number    81 non-null     int64 
 3   Start     81 non-null     int64 
dtypes: int64(3), object(1)
memory usage: 2.7+ KB
```

In [5]:
```
df.describe()
```

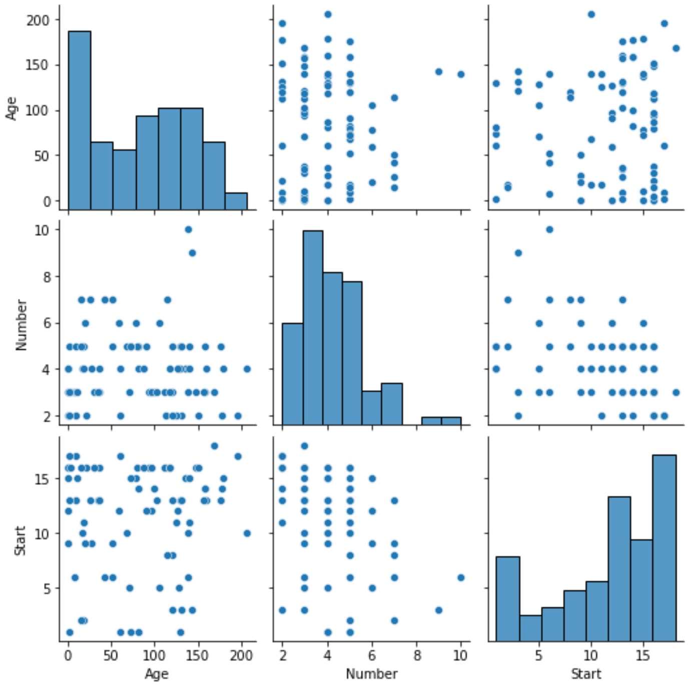
Out[5]:

	Age	Number	Start
count	81.000000	81.000000	81.000000
mean	83.654321	4.049383	11.493827
std	58.104251	1.619423	4.883962
min	1.000000	2.000000	1.000000
25%	26.000000	3.000000	9.000000
50%	87.000000	4.000000	13.000000
75%	130.000000	5.000000	16.000000
max	206.000000	10.000000	18.000000

In [6]:
```
import seaborn as sns
```

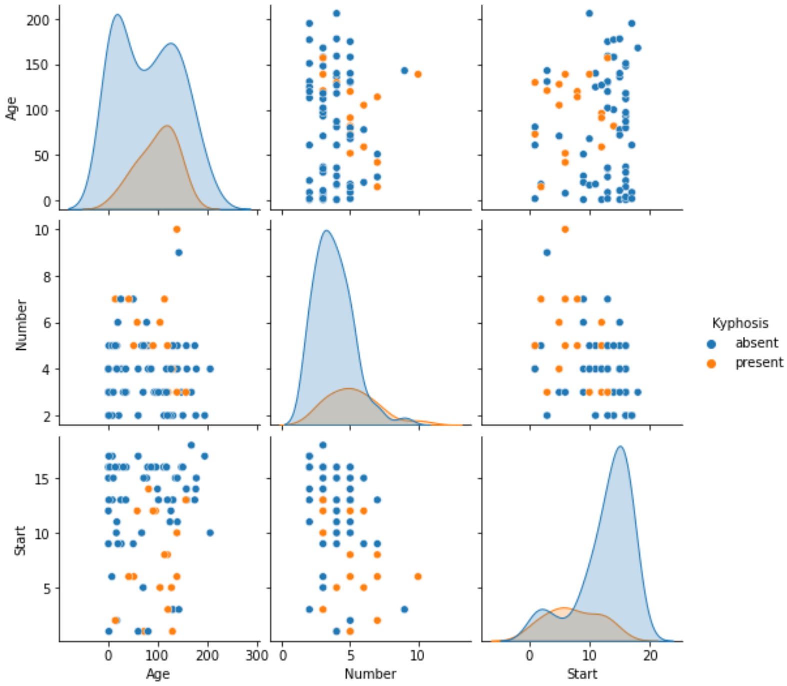
```
In [7]: sns.pairplot(df)

Out[7]: <seaborn.axisgrid.PairGrid at 0x7f83573cf7f0>
```



```
In [8]: sns.pairplot(df,hue='Kyphosis')

Out[8]: <seaborn.axisgrid.PairGrid at 0x7f835790ba60>
```



2. Split data set to training data and testing data

```
In [9]: from sklearn.model_selection import train_test_split
```

```
In [10]: x=df.drop('Kyphosis',axis=1)
```

```
In [11]: y=df['Kyphosis']
```

```
In [12]: xtrain, xtest, ytrain, ytest =train_test_split(x,y, test_size =0.3)
```

3. Build the tree

```
In [13]: from sklearn.tree import DecisionTreeClassifier
In [14]: dtree =DecisionTreeClassifier(max_depth=2)
```

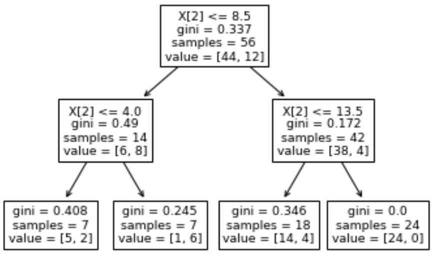
```
In [15]: dtree.fit(xtrain, ytrain)
```

```
Out[15]: DecisionTreeClassifier(max_depth=2)
```

```
In [16]: from sklearn import tree
```

```
In [17]: tree.plot_tree(dtree)
```

```
Out[17]: [Text(0.5, 0.8333333333333334, 'X[2] <= 8.5\ngini = 0.337\nsamples = 56\nvalue = [44, 12]'),
          Text(0.25, 0.5, 'X[2] <= 4.0\ngini = 0.49\nsamples = 14\nvalue = [6, 8]'),
          Text(0.125, 0.16666666666666666, 'gini = 0.408\nsamples = 7\nvalue = [5, 2]'),
          Text(0.375, 0.16666666666666666, 'gini = 0.245\nsamples = 7\nvalue = [1, 6]'),
          Text(0.75, 0.5, 'X[2] <= 13.5\ngini = 0.172\nsamples = 42\nvalue = [38, 4]'),
          Text(0.625, 0.16666666666666666, 'gini = 0.346\nsamples = 18\nvalue = [14, 4]'),
          Text(0.875, 0.16666666666666666, 'gini = 0.0\nsamples = 24\nvalue = [24, 0]')]
```



```
In [18]: from sklearn.tree import export_text
         r=export_text(dtree,feature_names=['age','num','start'])
```

```
In [19]: print(r)
```

```
|--- start <= 8.50
|   |--- start <= 4.00
|   |   |--- class: absent
|   |--- start >  4.00
|   |   |--- class: present
|--- start >  8.50
|   |--- start <= 13.50
|   |   |--- class: absent
|   |--- start >  13.50
|   |   |--- class: absent
```

4. Evaluate the model with confusion metrix

```
In [20]: pred=dtree.predict(xtest)
```

```
In [21]: ytest==pred
```

```
Out[21]: 0     False
         14     True
         41     True
         8      True
         24    False
         9     False
         60    False
         70     True
         38     True
         11     True
         28     True
         50     True
         3      True
         10    False
         59     True
         4      True
         35     True
         78     True
         17     True
         77     True
         66     True
         61     True
         73     True
         80     True
         55     True
         Name: Kyphosis, dtype: bool
```

```
In [22]: from sklearn.metrics import classification_report, confusion_matrix
```

```
In [23]: print(confusion_matrix(ytest,pred))
```

```
[[19  1]
 [ 4  1]]
```

```
In [24]: print(classification_report(ytest,pred))
```

```
              precision    recall  f1-score   support

      absent       0.83      0.95      0.88        20
     present       0.50      0.20      0.29         5

    accuracy                           0.80        25
   macro avg       0.66      0.57      0.58        25
weighted avg       0.76      0.80      0.76        25
```