Prediction using Decision Tree Algorithm

import the necesaary libraries

In [1]:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings("ignore")
```

In [2]:

```
df=pd.read_csv("iris.csv",names=["sepal length","sepal width","petal length","petal widt
df
```

Out[2]:

	sepal length	sepal width	petal length	petal width	class
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 5 columns

In [3]:

```
df.head()
```

Out[3]:

	sepal length	sepal width	petal length	petal width	class
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

In [4]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   sepal length  150 non-null    float64
 1   sepal width   150 non-null    float64
 2   petal length  150 non-null    float64
 3   petal width   150 non-null    float64
 4   class         150 non-null    object 
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

In [5]:

```
df.tail()
```

Out[5]:

	sepal length	sepal width	petal length	petal width	class
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

In [6]:

```
df.describe()
```

Out[6]:

	sepal length	sepal width	petal length	petal width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

In [7]:

```
df.shape
```

Out[7]:

```
(150, 5)
```

In [8]:

```
df.isnull().sum()
```

Out[8]:

```
sepal length    0
sepal width     0
petal length    0
petal width     0
class           0
dtype: int64
```

In [9]:

```
df.value_counts()
```

Out[9]:

```
sepal length  sepal width  petal length  petal width  class
4.9           3.1          1.5           0.1          Iris-setosa
3
5.8           2.7          5.1           1.9          Iris-virginica
2
              4.0          1.2           0.2          Iris-setosa
1
5.9           3.0          4.2           1.5          Iris-versicolor
1
6.2           3.4          5.4           2.3          Iris-virginica
1


..
5.5           2.3          4.0           1.3          Iris-versicolor
1
              2.4          3.7           1.0          Iris-versicolor
1
                           3.8           1.1          Iris-versicolor
1
              2.5          4.0           1.3          Iris-versicolor
1
7.9           3.8          6.4           2.0          Iris-virginica
1
Length: 147, dtype: int64
```

In [10]:

```
df.columns
```

Out[10]:

```
Index(['sepal length', 'sepal width', 'petal length', 'petal width', 'cla
ss'], dtype='object')
```

In [11]:

```
df.dtypes
```

Out[11]:

```
sepal length    float64
sepal width     float64
petal length    float64
petal width     float64
class            object
dtype: object
```

In [12]:

```
df.corr()
```

Out[12]:

	sepal length	sepal width	petal length	petal width
sepal length	1.000000	-0.109369	0.871754	0.817954
sepal width	-0.109369	1.000000	-0.420516	-0.356544
petal length	0.871754	-0.420516	1.000000	0.962757
petal width	0.817954	-0.356544	0.962757	1.000000

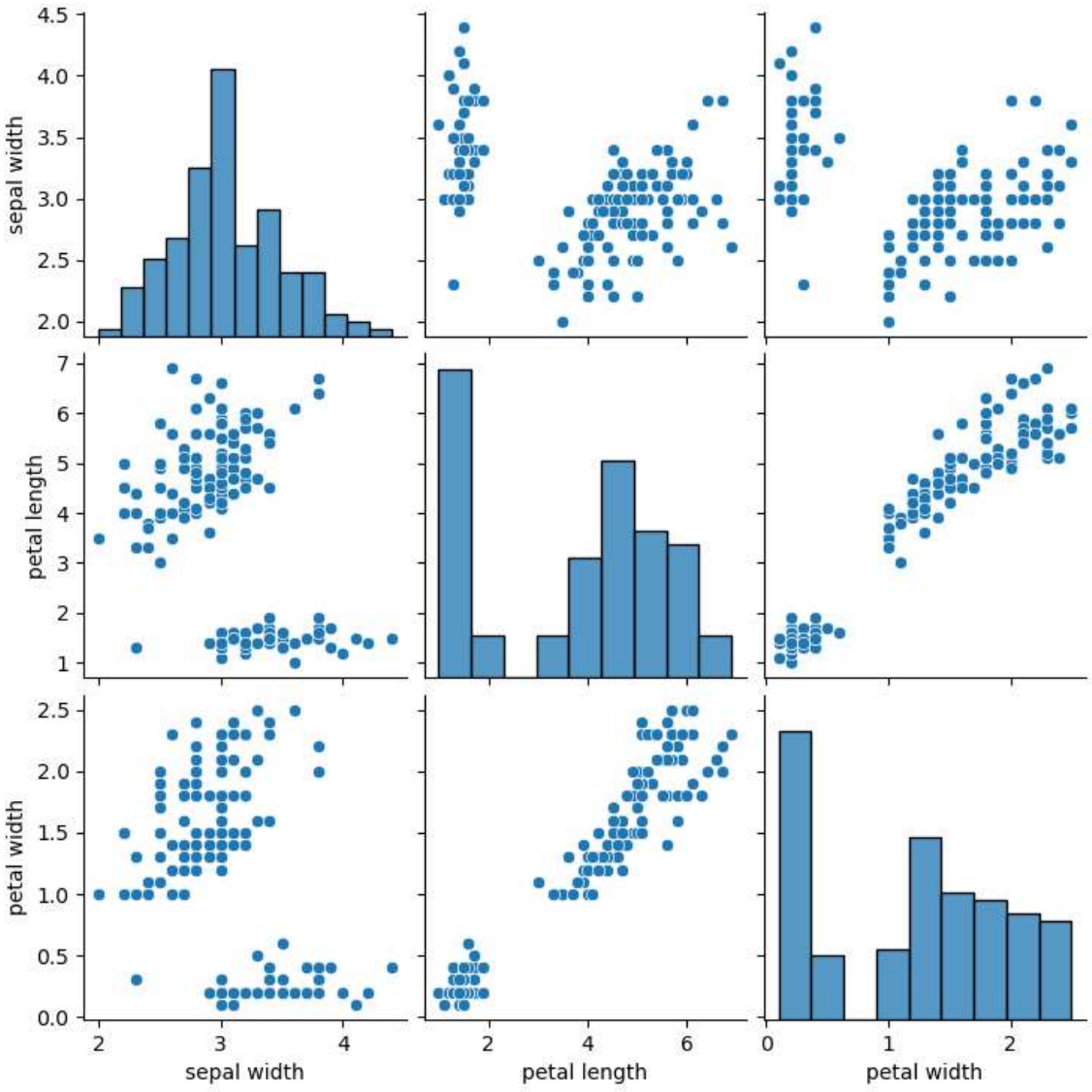
DATA VISUALISATION

In [13]:

```
sns.pairplot(df.iloc[:,1:])
```

Out[13]:

```
<seaborn.axisgrid.PairGrid at 0x16cf1e7aec0>
```

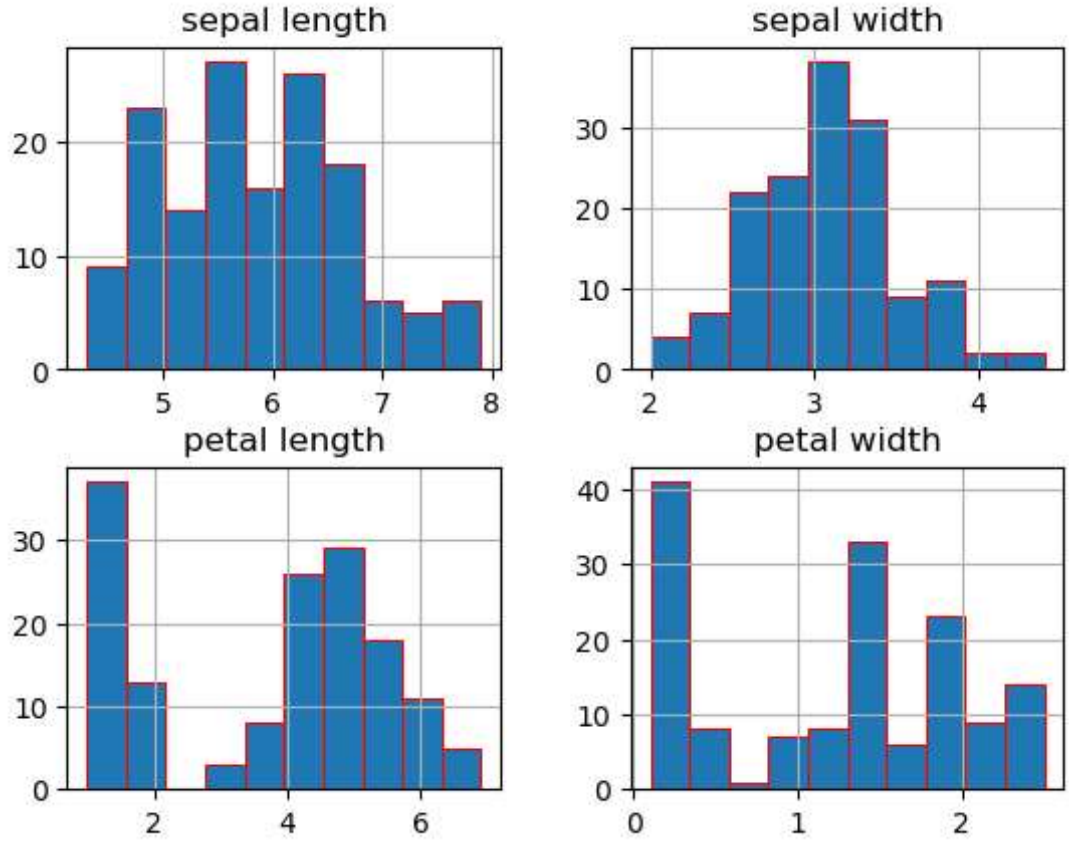


In [14]:

```
df.hist(edgecolor="red",linewidth=0.75)
```

Out[14]:

```
array([[<Axes: title={'center': 'sepal length'}>,
        <Axes: title={'center': 'sepal width'}>],
       [<Axes: title={'center': 'petal length'}>,
        <Axes: title={'center': 'petal width'}>]], dtype=object)
```

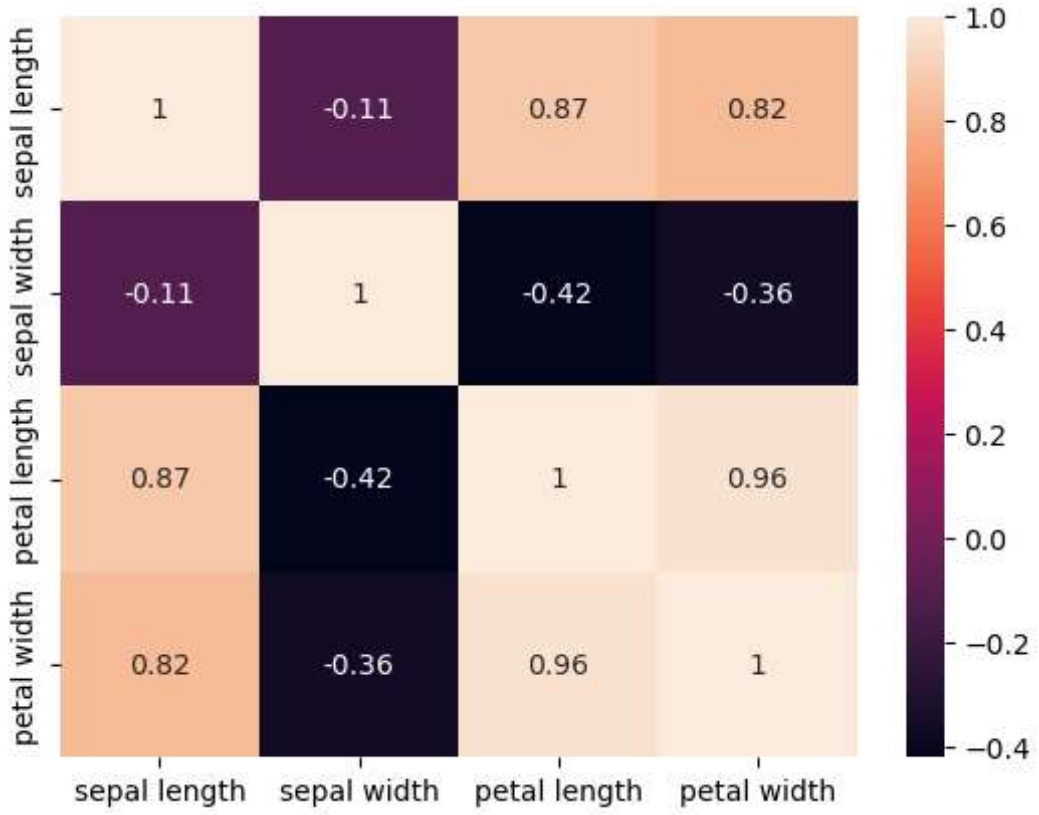


In [15]:

```
sns.heatmap(df.corr(),annot=True)
```

Out[15]:

```
<Axes: >
```



In [16]:

```
df.dtypes
```

Out[16]:

```
sepal length    float64
sepal width     float64
petal length    float64
petal width     float64
class            object
dtype: object
```

In [17]:

```
x=df.iloc[:,:4]
x
```

Out[17]:

	sepal length	sepal width	petal length	petal width
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2
...	...	...	...	...
145	6.7	3.0	5.2	2.3
146	6.3	2.5	5.0	1.9
147	6.5	3.0	5.2	2.0
148	6.2	3.4	5.4	2.3
149	5.9	3.0	5.1	1.8

150 rows × 4 columns

In [18]:

```
y=df.iloc[:,-1]
y
```

Out[18]:

```
0         Iris-setosa
1         Iris-setosa
2         Iris-setosa
3         Iris-setosa
4         Iris-setosa
            ...
145    Iris-virginica
146    Iris-virginica
147    Iris-virginica
148    Iris-virginica
149    Iris-virginica
Name: class, Length: 150, dtype: object
```

In [19]:

```
x = df.iloc[:,:-1]
y = df.iloc[:,-1]
```

In [20]:

```
from sklearn.model_selection import train_test_split
xtrain,xtest,ytrain,ytest=train_test_split(x,y,test_size=0.30,random_state=1)
```

In [21]:

```
from sklearn.metrics import classification_report
```

In [42]:

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
```

In [43]:

```
dt=DecisionTreeClassifier()
dt=dt.fit(xtrain,ytrain)
ypred=dt.predict(xtest)
```

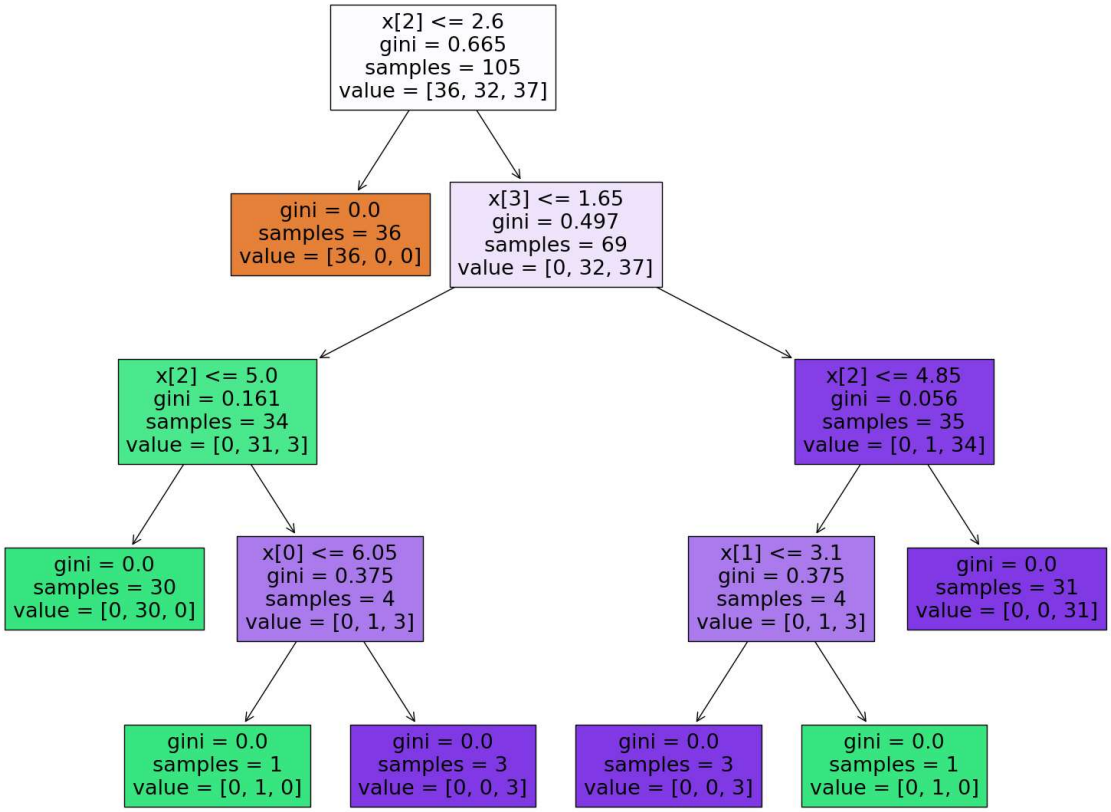
In [45]:

```
result=confusion_matrix(ytest,ypred)
print("Confusion Metrix:")
print(result)
result1=classification_report(ytest,ypred)
print("Classification Report:")
print(result1)
result2=accuracy_score(ytest,ypred)
print("Accuracy:",result2)
```

```
Confusion Metrix:
[[14  0  0]
 [ 0 17  1]
 [ 0  1 12]]
Classification Report:
                 precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        14
Iris-versicolor       0.94      0.94      0.94        18
 Iris-virginica       0.92      0.92      0.92        13

       accuracy                           0.96        45
      macro avg       0.96      0.96      0.96        45
   weighted avg       0.96      0.96      0.96        45

Accuracy: 0.9555555555555556
```

In [52]:

```
features=df.columns[:-1]
from sklearn.tree import plot_tree
plt.figure(figsize=(19,15))
plot_tree(dt.fit(xtrain,ytrain),filled=True)
plt.show()
```



In [ ]:

```
```