Name = Maalika Maini

Internship at Let's grow more

In [131…

```
#Loading libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier,plot_tree
from sklearn.metrics import accuracy_score,mean_absolute_error
from sklearn.preprocessing import StandardScaler,LabelEncoder
from sklearn.tree import export_graphviz
from IPython.display import Image
import pydotplus
from sklearn import tree
```

Loadind datasets

In [77]:

```
iris=pd.read_csv(r"C:\Users\Anshul Maini\Downloads\Iris.csv")
```

Reading in the dataset

In [78]:

```
iris
```

Out[78]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3	Iris-virginica
146	147	6.3	2.5	5.0	1.9	Iris-virginica
147	148	6.5	3.0	5.2	2.0	Iris-virginica
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 6 columns

In [79]:

```
iris.head()
```

Out[79]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

In [80]:

```
iris.tail()
```

Out[80]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
145	146	6.7	3.0	5.2	2.3	Iris-virginica
146	147	6.3	2.5	5.0	1.9	Iris-virginica
147	148	6.5	3.0	5.2	2.0	Iris-virginica

148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

In [81]:

```
iris.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Id             150 non-null    int64  
 1   SepalLengthCm  150 non-null    float64
 2   SepalWidthCm   150 non-null    float64
 3   PetalLengthCm  150 non-null    float64
 4   PetalWidthCm   150 non-null    float64
 5   Species        150 non-null    object 
dtypes: float64(4), int64(1), object(1)
memory usage: 7.2+ KB
```

In [82]:

```
iris.isnull()
```

Out[82]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	False	False	False	False	False	False
1	False	False	False	False	False	False
2	False	False	False	False	False	False
3	False	False	False	False	False	False
4	False	False	False	False	False	False
...	...	...	...	...	...	...
145	False	False	False	False	False	False
146	False	False	False	False	False	False
147	False	False	False	False	False	False
148	False	False	False	False	False	False
149	False	False	False	False	False	False

150 rows × 6 columns

In [83]:

```
iris.isnull().sum()
```

Out[83]:

```
Id               0
SepalLengthCm    0
SepalWidthCm     0
PetalLengthCm    0
PetalWidthCm     0
Species          0
dtype: int64
```

In [84]:

```
iris.shape
```

Out[84]:

```
(150, 6)
```

In [85]:

```
iris.describe()
```

Out[85]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
count	150.000000	150.000000	150.000000	150.000000	150.000000
mean	75.500000	5.843333	3.054000	3.758667	1.198667
std	43.445368	0.828066	0.433594	1.764420	0.763161
min	1.000000	4.300000	2.000000	1.000000	0.100000
25%	38.250000	5.100000	2.800000	1.600000	0.300000

50%	75.500000	5.800000	3.000000	4.350000	1.300000
75%	112.750000	6.400000	3.300000	5.100000	1.800000
max	150.000000	7.900000	4.400000	6.900000	2.500000

In [86]:

```
y=pd.DataFrame(columns=iris.Species)
```

In [87]:

```
y
```

Out[87]:

Species	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa	...	Iris-virginica	Iris-virginica	Iris-virginica	Iris-virginica	Iris-virginica

0 rows × 150 columns

In [88]:

```
print('classes to predict:',iris)
```

```
classes to predict:       Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm  \
0      1            5.1           3.5            1.4           0.2
1      2            4.9           3.0            1.4           0.2
2      3            4.7           3.2            1.3           0.2
3      4            4.6           3.1            1.5           0.2
4      5            5.0           3.6            1.4           0.2
..   ...            ...           ...            ...           ...
145  146            6.7           3.0            5.2           2.3
146  147            6.3           2.5            5.0           1.9
147  148            6.5           3.0            5.2           2.0
148  149            6.2           3.4            5.4           2.3
149  150            5.9           3.0            5.1           1.8

            Species
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

[150 rows x 6 columns]
```

In [90]:

```
columns=['ID','SepalLengthCm','SepalWidthCm','PetalLengthCm','PetalWidthCm','Species']
iris.columns=columns
iris.head()
```

Out[90]:

	ID	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

Visualisation

In [89]:

```
species=iris['Species'].value_counts()    #Checking count of species in the dataset
labels=species.index.tolist()
count=species.tolist()
species.to_frame()
```

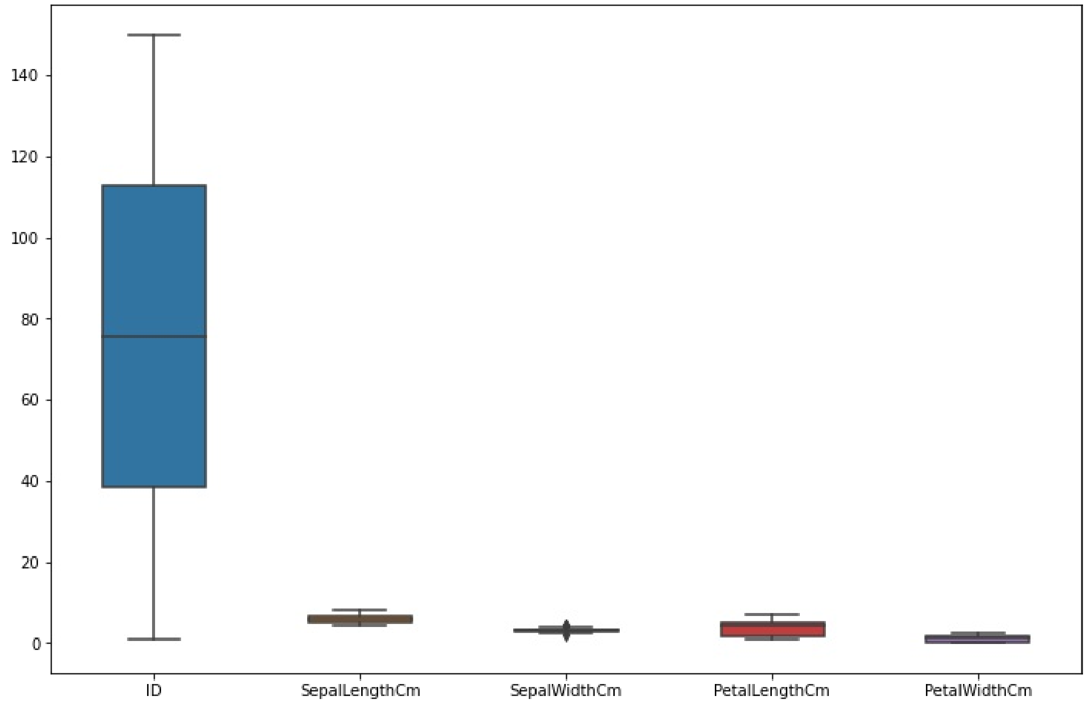
Out[89]:

	Species
Iris-setosa	50
Iris-versicolor	50

Iris-virginica	50

In [92]:

```
plt.figure(figsize=(12,8))
sns.boxplot(data=iris,width=0.5,fliersize=5)
plt.show()
```



In [93]:

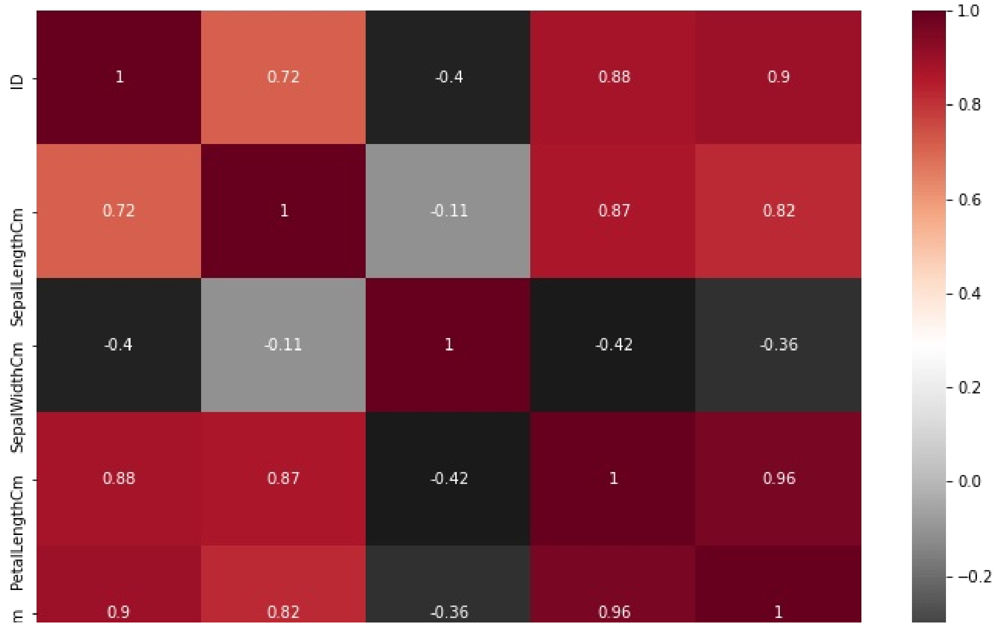
```
corr=iris.corr()
corr
```

Out[93]:

	ID	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
ID	1.000000	0.716676	-0.397729	0.882747	0.899759
SepalLengthCm	0.716676	1.000000	-0.109369	0.871754	0.817954
SepalWidthCm	-0.397729	-0.109369	1.000000	-0.420516	-0.356544
PetalLengthCm	0.882747	0.871754	-0.420516	1.000000	0.962757
PetalWidthCm	0.899759	0.817954	-0.356544	0.962757	1.000000

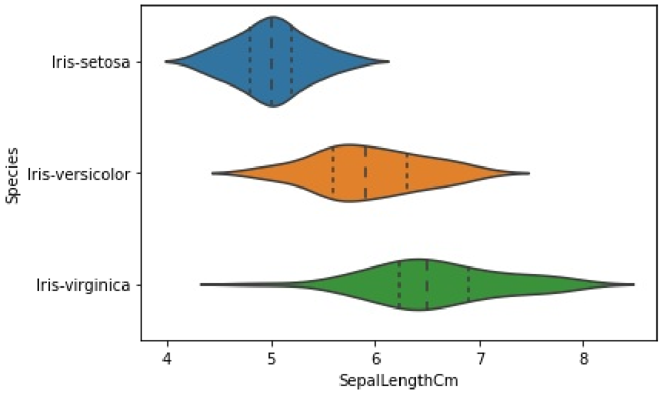
In [94]:

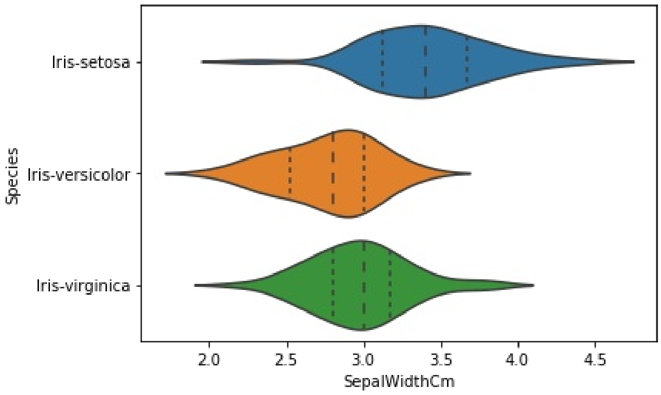
```
plt.figure(figsize=(12,8))
sns.heatmap(corr,annot=True,cmap='RdGy_r')
plt.show()
```

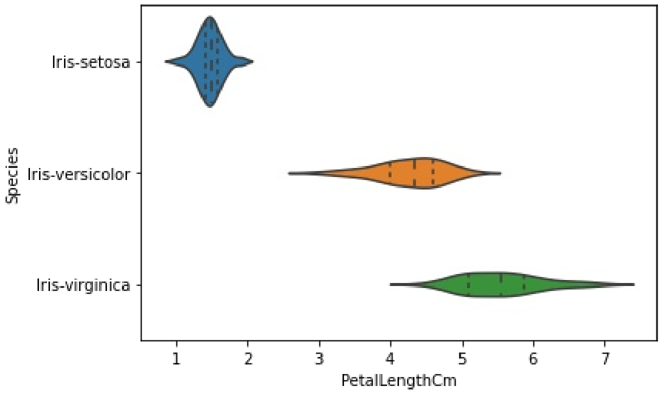


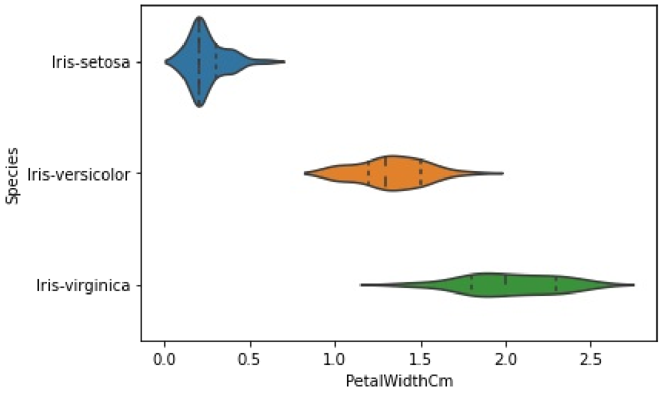
In [95]:

```
# Viol in Plot
sns.violinplot(y='Species',x='SepalLengthCm',data=iris,inner='quartitle')
plt.show()
sns.violinplot(y='Species',x='SepalWidthCm',data=iris,inner='quartitle')
plt.show()
sns.violinplot(y='Species',x='PetalLengthCm',data=iris,inner='quartitle')
plt.show()
sns.violinplot(y='Species',x='PetalWidthCm',data=iris,inner='quartitle')
plt.show()
```



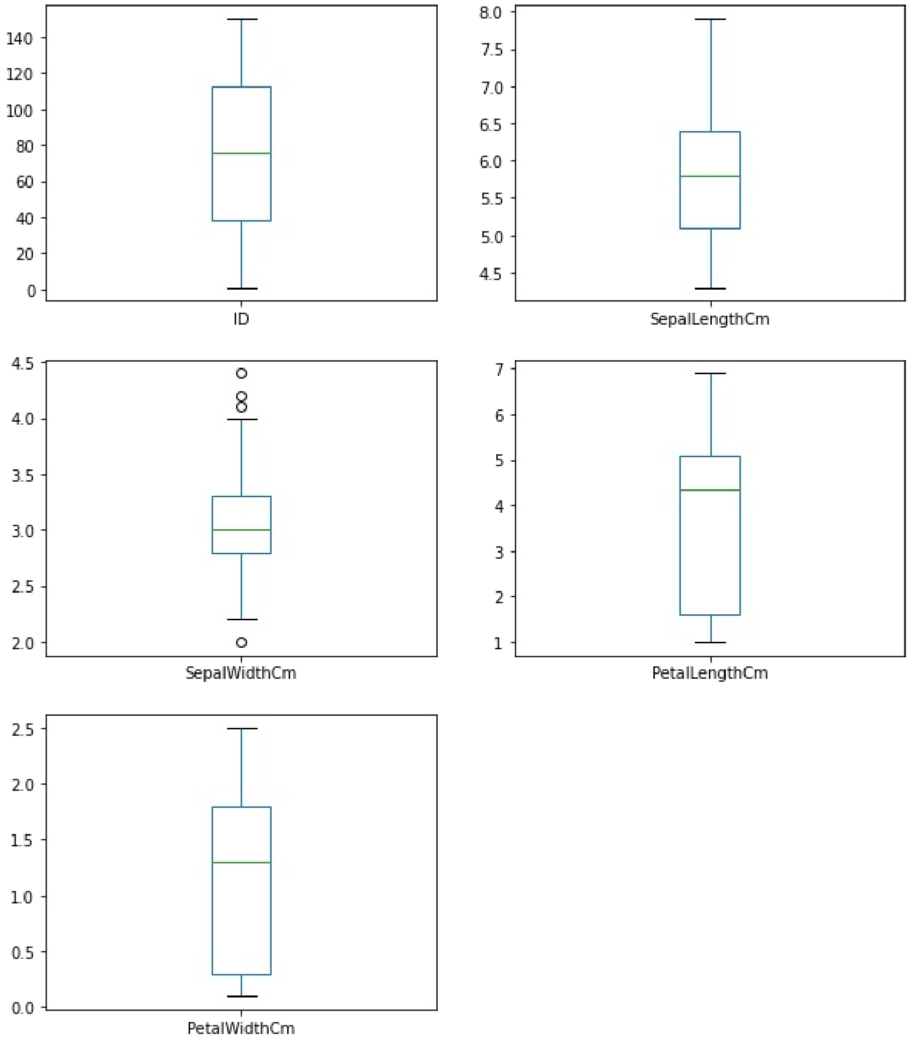




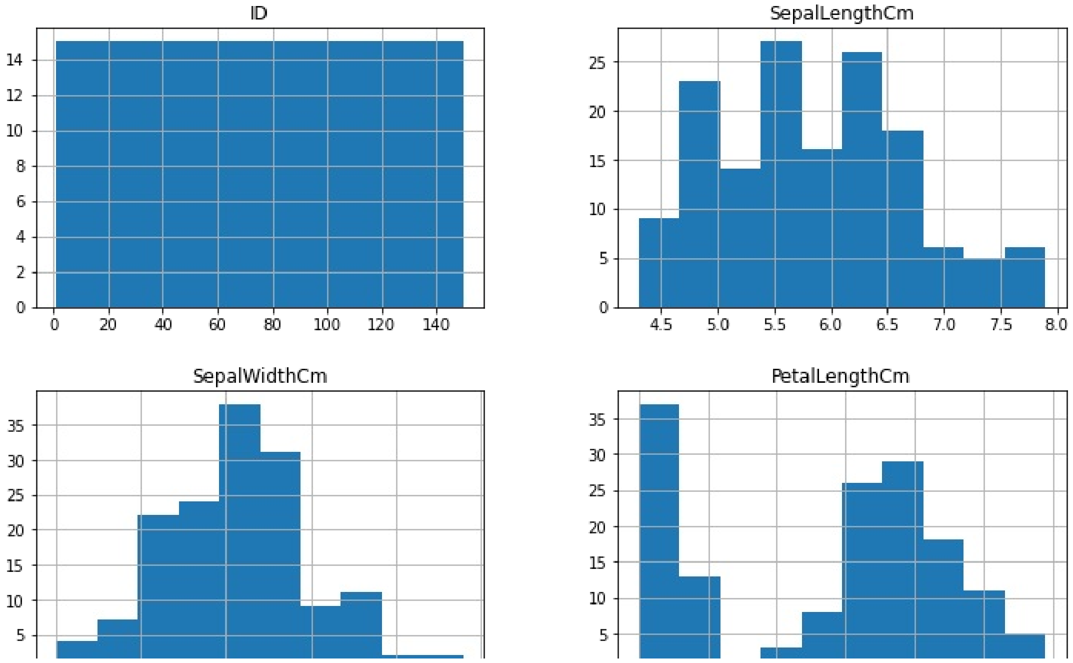


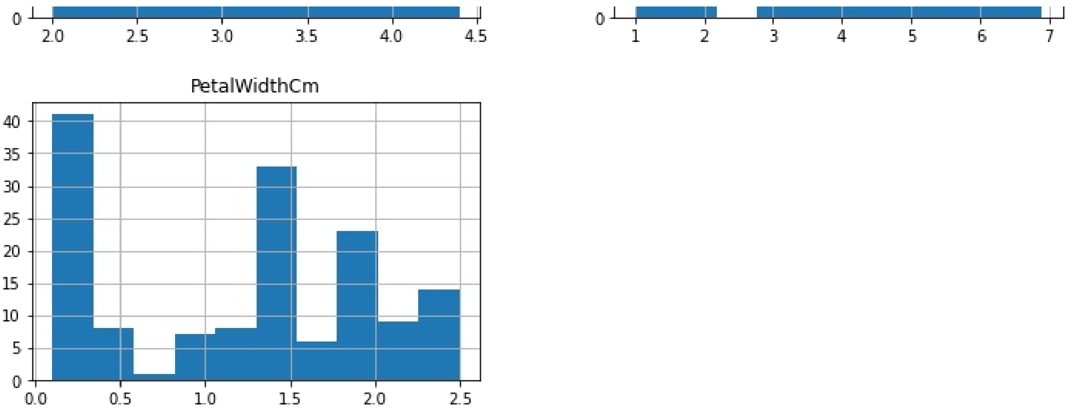
```
In [96]: # Box and Whisker plots
         iris.plot(kind='box',subplots=True, layout=(3,2),figsize=(10,12))
```

```
Out[96]: ID                    AxesSubplot(0.125,0.657941;0.352273x0.222059)
         SepalLengthCm    AxesSubplot(0.547727,0.657941;0.352273x0.222059)
         SepalWidthCm          AxesSubplot(0.125,0.391471;0.352273x0.222059)
         PetalLengthCm    AxesSubplot(0.547727,0.391471;0.352273x0.222059)
         PetalWidthCm             AxesSubplot(0.125,0.125;0.352273x0.222059)
         dtype: object
```



```
In [97]: iris.hist(figsize=(12,12))      ##Drawing a histogram to see the distribution of data
         plt.show()
```



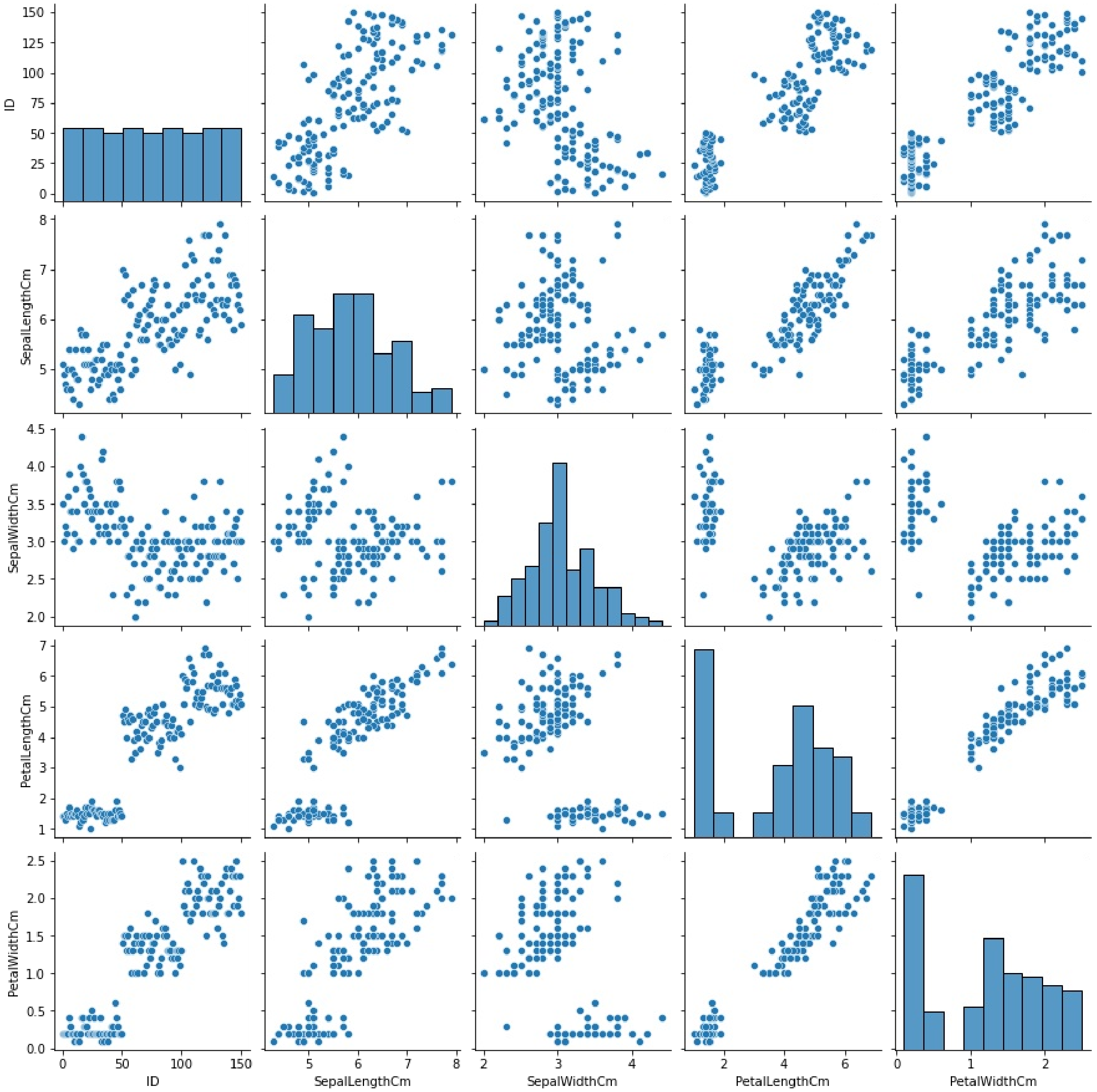


In [98]:

```
sns.pairplot(iris)  #To see the relation between each pair features in dataset
```

Out[98]:

```
<seaborn.axisgrid.PairGrid at 0x21f2436fb80>
```



Break dataset in train and test range

In [108…

```
le=LabelEncoder()
iris['Species']=le.fit_transform(iris['Species'])
```

```
iris.head()
```

Out[108…

	ID	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	0
1	2	4.9	3.0	1.4	0.2	0
2	3	4.7	3.2	1.3	0.2	0
3	4	4.6	3.1	1.5	0.2	0
4	5	5.0	3.6	1.4	0.2	0

In [109…

```
X=iris.iloc[:,0:4]
X=X.values
X[0:5]
```

Out[109…

```
array([[1. , 5.1, 3.5, 1.4],
       [2. , 4.9, 3. , 1.4],
       [3. , 4.7, 3.2, 1.3],
       [4. , 4.6, 3.1, 1.5],
       [5. , 5. , 3.6, 1.4]])
```

In [110…

```
Y=iris.iloc[:,4]
Y.values
Y[0:5]
```

Out[110…

```
0    0.2
1    0.2
2    0.2
3    0.2
4    0.2
Name: PetalWidthCm, dtype: float64
```

In [111…

```
std=StandardScaler()  #Normalising the data because the data is very Scattered
X=std.fit_transform(X)
X[0:5]
```

Out[111…

```
array([[-1.72054204, -0.90068117,  1.03205722, -1.3412724 ],
       [-1.69744751, -1.14301691, -0.1249576 , -1.3412724 ],
       [-1.67435299, -1.38535265,  0.33784833, -1.39813811],
       [-1.65125846, -1.50652052,  0.10644536, -1.2844067 ],
       [-1.62816394, -1.02184904,  1.26346019, -1.3412724 ]])
```

In [112…

```
xtrain,xtest,ytrain,ytest=train_test_split(X,Y,test_size=25,random_state=42) #Break the dataset into train and te
```

In [102…

```
print("Size of Training Set")
print("X",xtrain.shape)
print("Y",ytrain.shape)
print("Size of Testing Set")
print("X",xtrain.shape)
print("Y",ytrain.shape)
```

```
Size of Training Set
X (125, 4)
Y (125,)
Size of Testing Set
X (125, 4)
Y (125,)
```

In [116…

```
clf=DecisionTreeClassifier() #Creating the Model
clf=clf.fit(xtrain,ytrain)
```

In [117…

```
clf
```

Out[117…

```
DecisionTreeClassifier()
```

```
In [118…  print('Accuracy of training data',clf.score(xtrain,ytrain))
          print('Accuracy of testing',clf.score(xtest,ytest))
```

```
Accuracy of training data 1.0
Accuracy of testing 1.0
```

```
In [120…  prediction=clf.predict(xtest)   #Prediction
          prediction
```

```
Out[120…  array([1, 0, 2, 1, 1, 0, 1, 2, 1, 1, 2, 0, 0, 0, 0, 1, 2, 1, 1, 2, 0, 2,
                 0, 2, 2, 2, 2, 2, 0, 0, 0, 0, 1, 0, 0, 2, 1, 0, 0, 0, 2, 1, 1, 0,
                 0], dtype=int64)
```

```
In [123…  print("Accuracy",accuracy_score(ytest,prediction))     #Evaluation
```

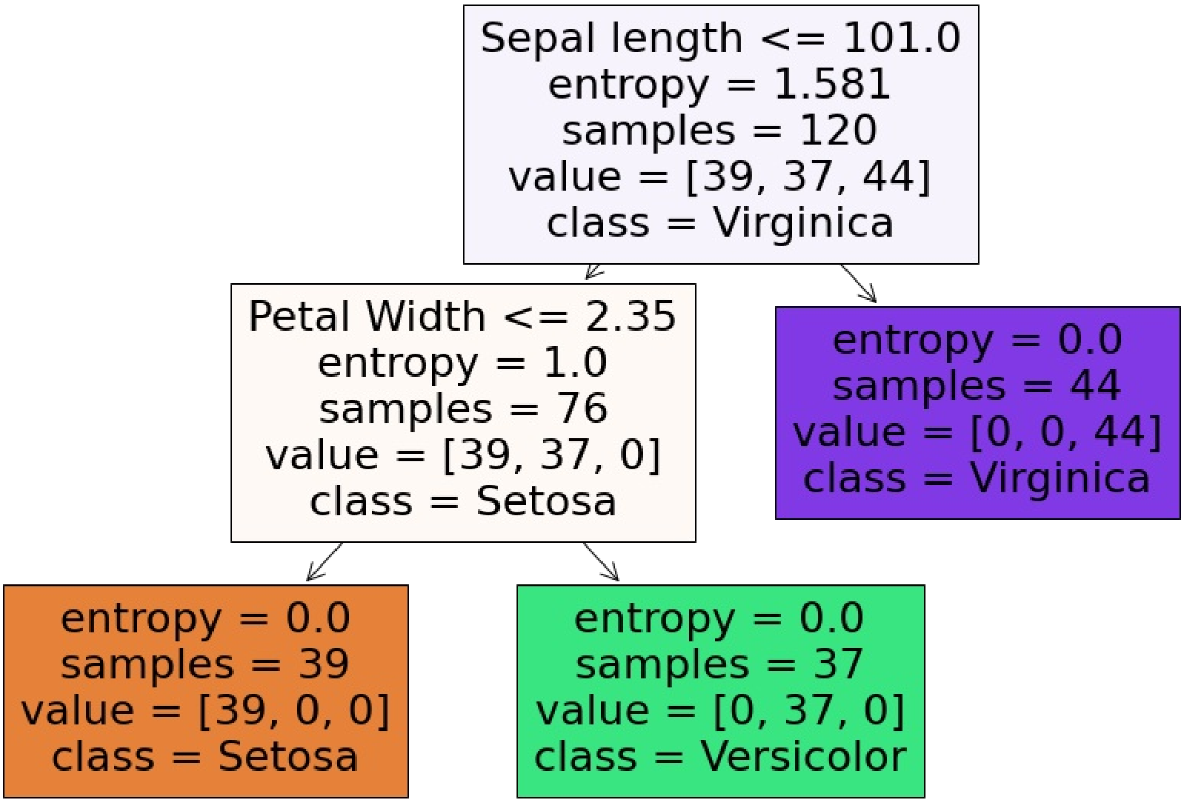
```
Accuracy 1.0
```

Visualising the Decision Tree

```
In [129…  columns=["Sepal length","Sepal Width","Petal length","Petal Width"]
          target=["Setosa","Versicolor","Virginica"]
```

```
In [148…  plt.figure(figsize=(15,10))
          tree.plot_tree(Classifier,feature_names=columns,class_names=target,filled=True)
```

```
Out[148…  [Text(502.20000000000005, 453.0, 'Sepal length <= 101.0\nentropy = 1.581\nsamples = 120\nvalue = [39, 37, 44]\ncl
ass = Virginica'),
 Text(334.8, 271.8, 'Petal Width <= 2.35\nentropy = 1.0\nsamples = 76\nvalue = [39, 37, 0]\nclass = Setosa'),
 Text(167.4, 90.59999999999997, 'entropy = 0.0\nsamples = 39\nvalue = [39, 0, 0]\nclass = Setosa'),
 Text(502.20000000000005, 90.59999999999997, 'entropy = 0.0\nsamples = 37\nvalue = [0, 37, 0]\nclass = Versicolor
'),
 Text(669.6, 271.8, 'entropy = 0.0\nsamples = 44\nvalue = [0, 0, 44]\nclass = Virginica')]
```



```
In [ ]:
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js