USING IRIS DATA SET FOR ANALYSIS

AJEET K JAIN
DATA SCIENCE GROUP
KMIT, HYDERABAD

The data set consists of 50 samples from each of three species of *Iris* (*Iris setosa*, *Iris virginica* and *Iris versicolor*).

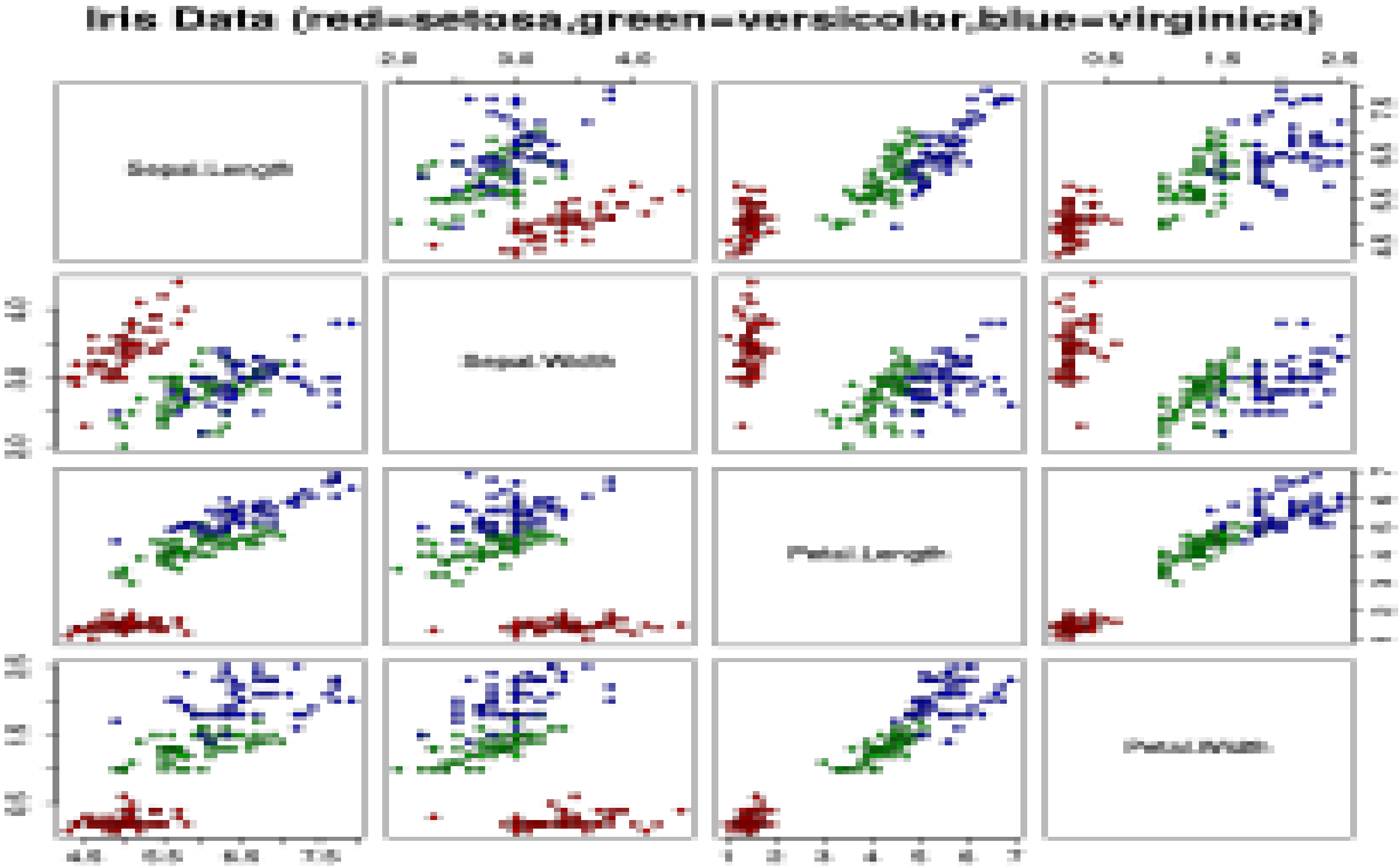
Four features were measured from each sample: the length and the width of the sepals and petals, in centimetres.

Based on the combination of these four features, Fisher developed a linear discriminant model to distinguish the species from each other.

Case Study and Practice in Python: IRIS Data Set

This *data sets* consists of 3 different types of *irises*' (Setosa, Versicolour, and Virginica) petal and sepal length, stored in a 150x4 **numpy.ndarray**.

The rows being the samples and the columns being: Sepal Length, Sepal Width, Petal Length and Petal Width.
The below plot uses the first two features.



In [15]:

```python
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
sns.set(color_codes=True)
import pandas as pd
%matplotlib inline
dataset = pd.read_csv('D:/python/IRIS.csv')
print(dataset)
```

Download Iris data and see the output

```
    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
5    6            5.4           3.9            1.7           0.4
6    7            4.6           3.4            1.4           0.3
7    8            5.0           3.4            1.5           0.2
8    9            4.4           2.9            1.4           0.2
9   10            4.9           3.1            1.5           0.1
10  11            5.4           3.7            1.5           0.2
11  12            4.8           3.4            1.6           0.2
12  13            4.8           3.0            1.4           0.1
13  14            4.3           3.0            1.1           0.1
14  15            5.8           4.0            1.2           0.2
15  16            5.7           4.4            1.5           0.4
16  17            5.4           3.9            1.3           0.4
17  18            5.1           3.5            1.4           0.3
```

In [15]:

```
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
sns.set(color_codes=True)
import pandas as pd
%matplotlib inline
dataset = pd.read_csv('D:/python/IRIS.csv')
print(dataset)
```

```
133   Iris-virginica
134   Iris-virginica
135   Iris-virginica
136   Iris-virginica
137   Iris-virginica
138   Iris-virginica
139   Iris-virginica
140   Iris-virginica
141   Iris-virginica
142   Iris-virginica
143   Iris-virginica
144   Iris-virginica
145   Iris-virginica
146   Iris-virginica
147   Iris-virginica
148   Iris-virginica
149   Iris-virginica

[150 rows x 6 columns]
```

In [16]:
```python
dataset.head()
```

Out[16]:

	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 [17]:
```python
#drop Id column
dataset = dataset.drop('Id',axis=1)
dataset.head()
```

DROP ID COLUMN

Out[17]:

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
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 [18]:
```
# shape  -  SUMMARY OF DATA SET
print(dataset.shape)
```

```
(150, 5)
```

In [19]:
```
# more information about iris data
print(dataset.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
SepalLengthCm    150 non-null float64
SepalWidthCm     150 non-null float64
PetalLengthCm    150 non-null float64
PetalWidthCm     150 non-null float64
Species          150 non-null object
dtypes: float64(4), object(1)
memory usage: 5.9+ KB
None
```

```
In [20]: # descriptions
         print(dataset.describe())
```

```
       SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm
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
```

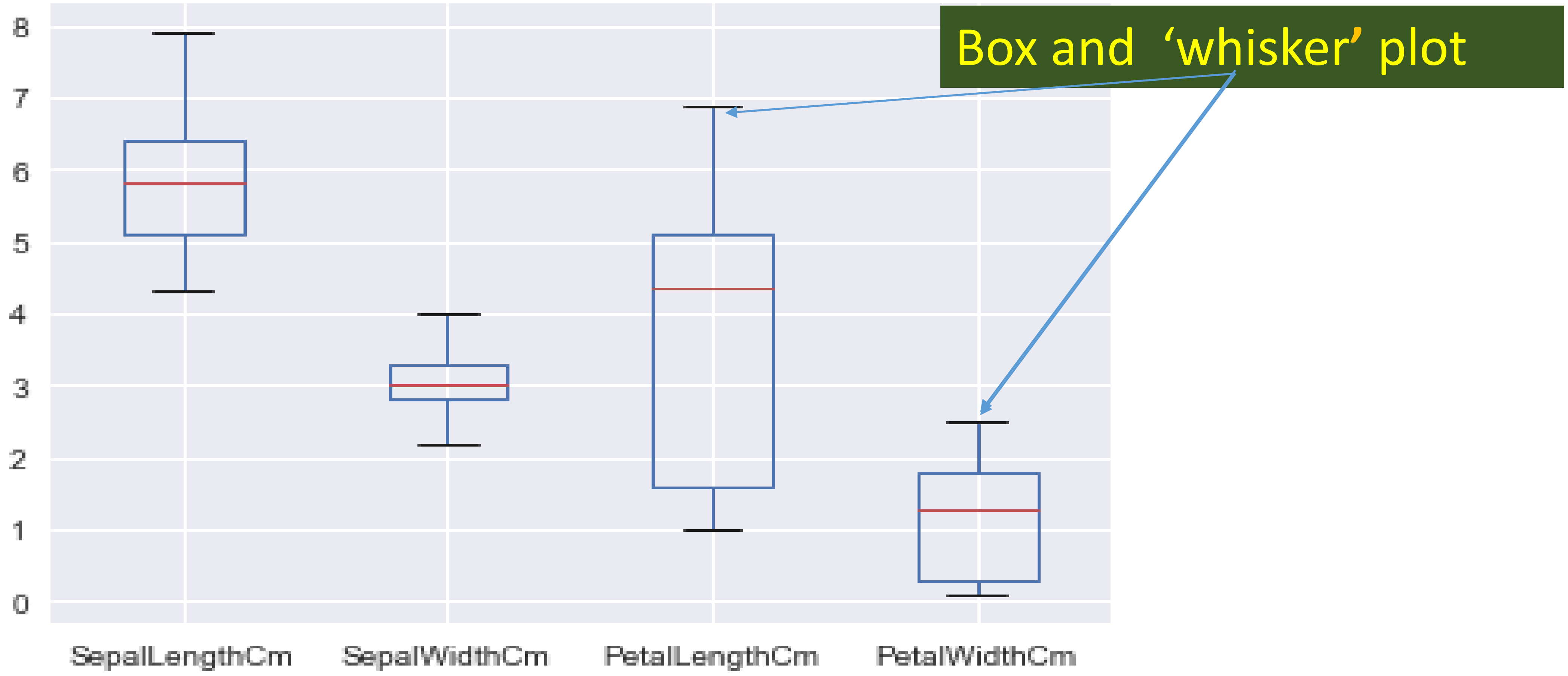
```
In [21]: # class distribution
         print(dataset.groupby('Species').size())
```

Use ‘**groupby**’ to combine

```
Species
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
dtype: int64
```

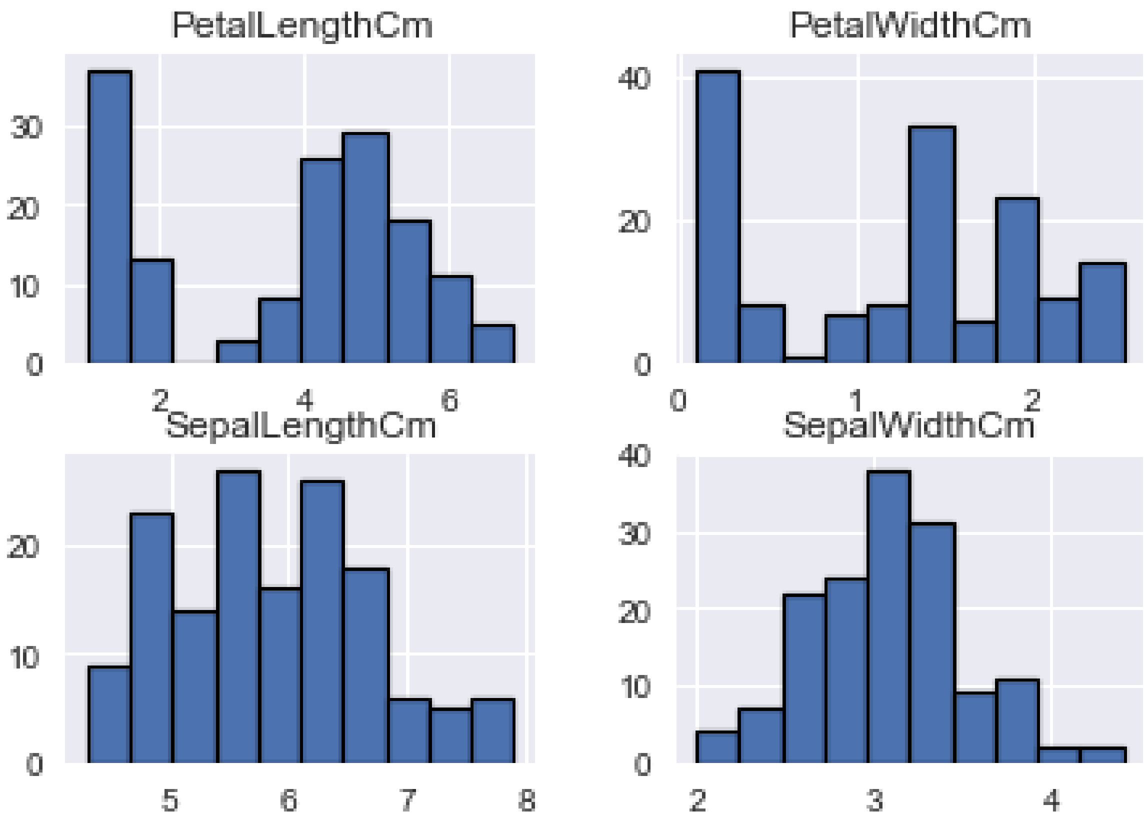
```
In [22]: # box and whisker plots
         dataset.plot(kind='box', sharex=False, sharey=False)
```

```
Out[22]: <matplotlib.axes._subplots.AxesSubplot at 0xb0e6668>
```



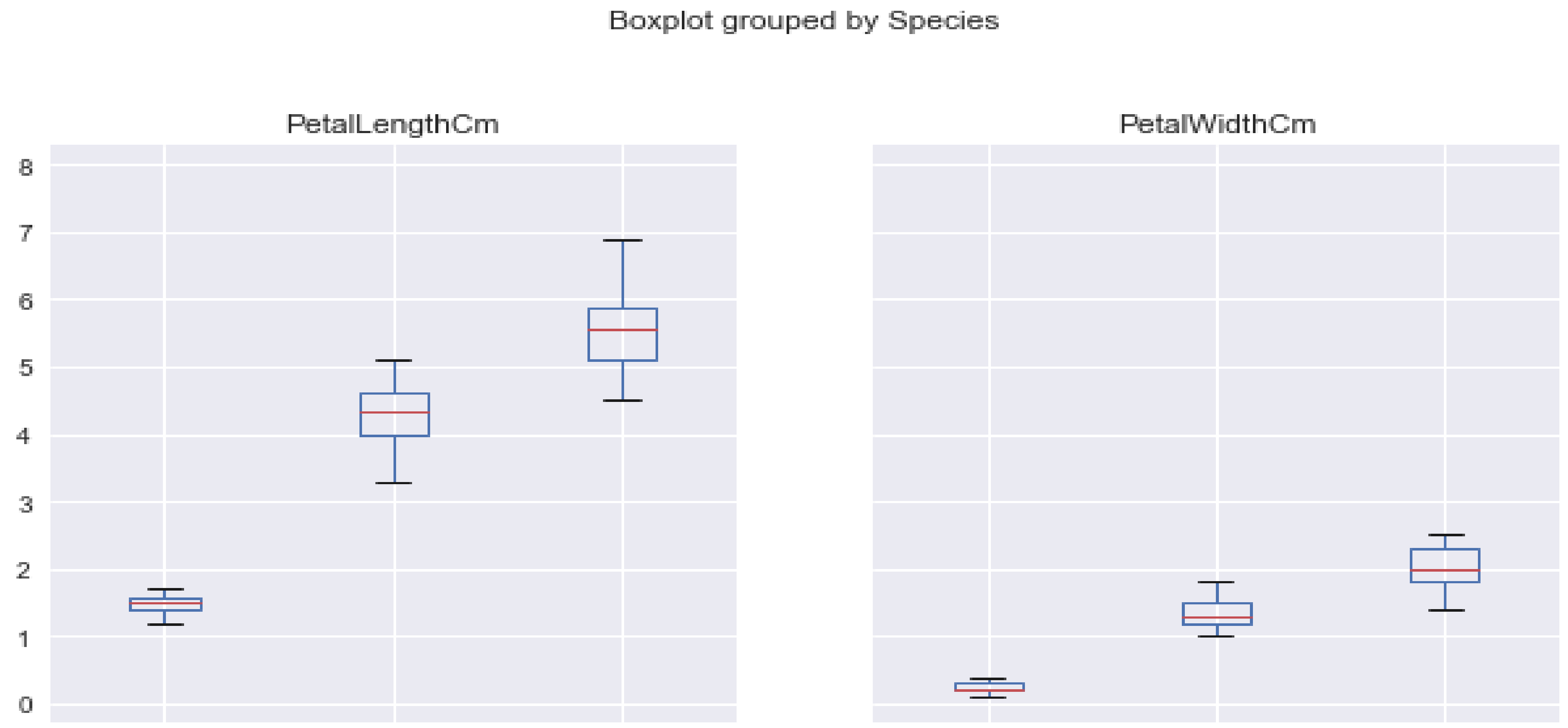
In [23]:
```python
# histograms
dataset.hist(edgecolor='black', linewidth=1.2)
```

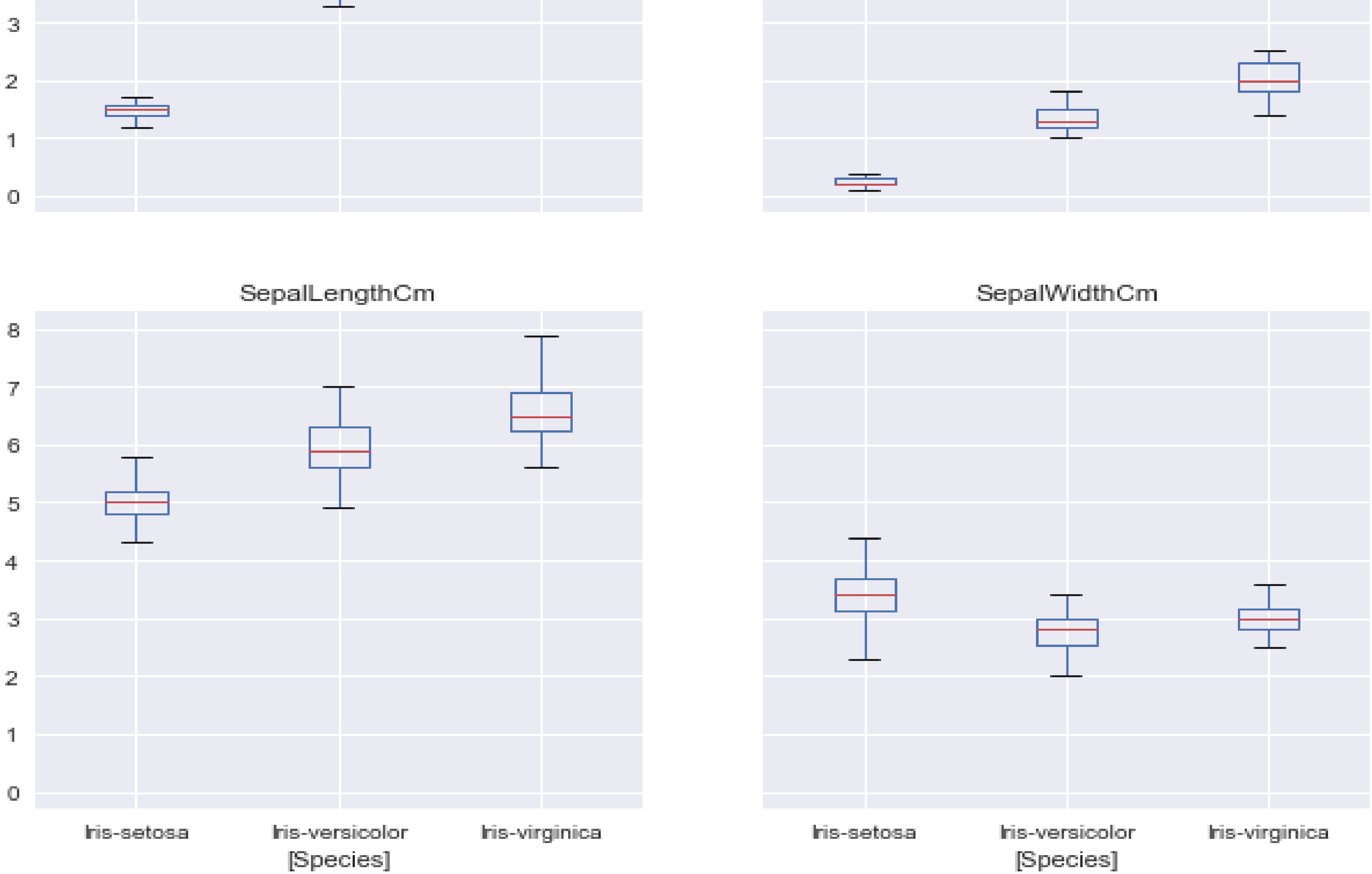
Out[23]:
```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x000000000B3906D8>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000000000B154908>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x0000000009E9DBA8>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000000000B3B6EF0>]],
      dtype=object)
```



In [24]:
```python
# boxplot on each feature split out by species
dataset.boxplot(by="Species",figsize=(10,10))
```

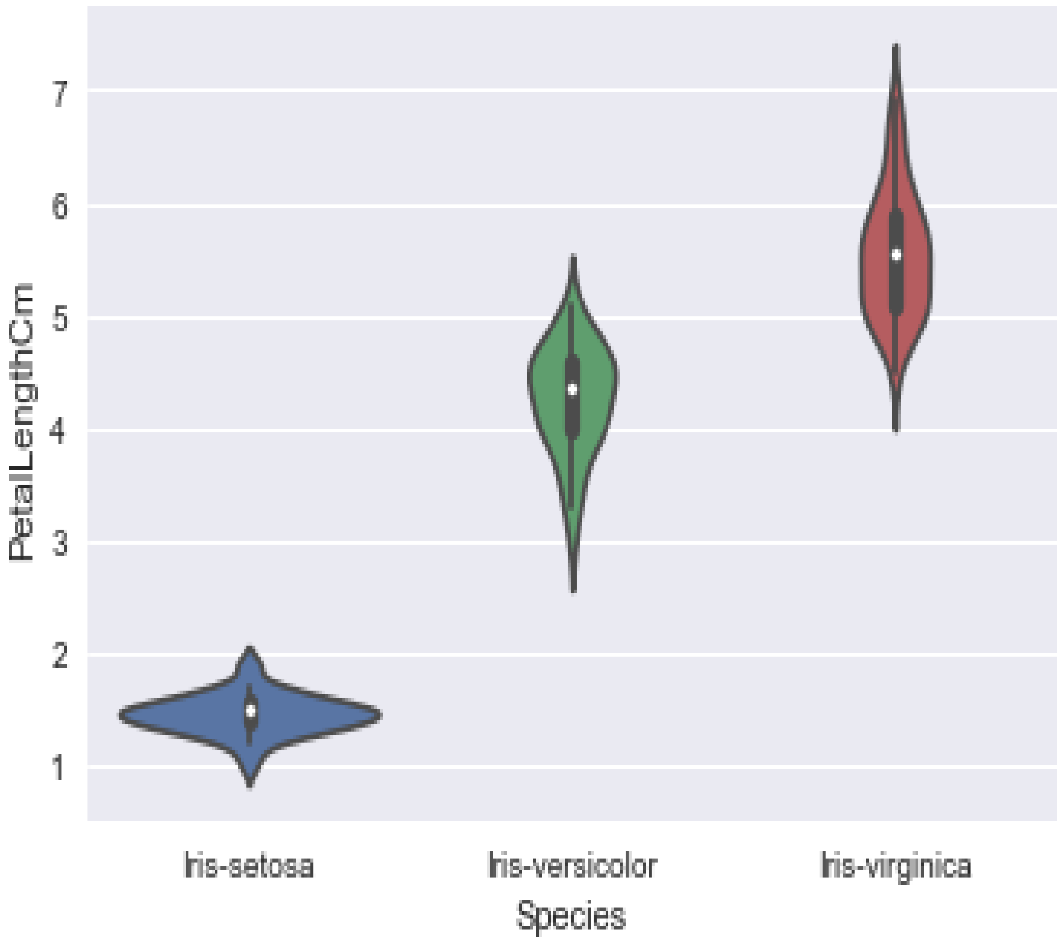
Out[24]:
```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x000000000B33B7F0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000000000B4CA588>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x000000000B4F0D68>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000000000B51A588>]],
      dtype=object)
```




SepalLengthCm
SepalWidthCm
0
1
2
3
4
5
6
7
8
Iris-setosa
Iris-versicolor
Iris-virginica
[Species]
Iris-setosa
Iris-versicolor
Iris-virginica
[Species]

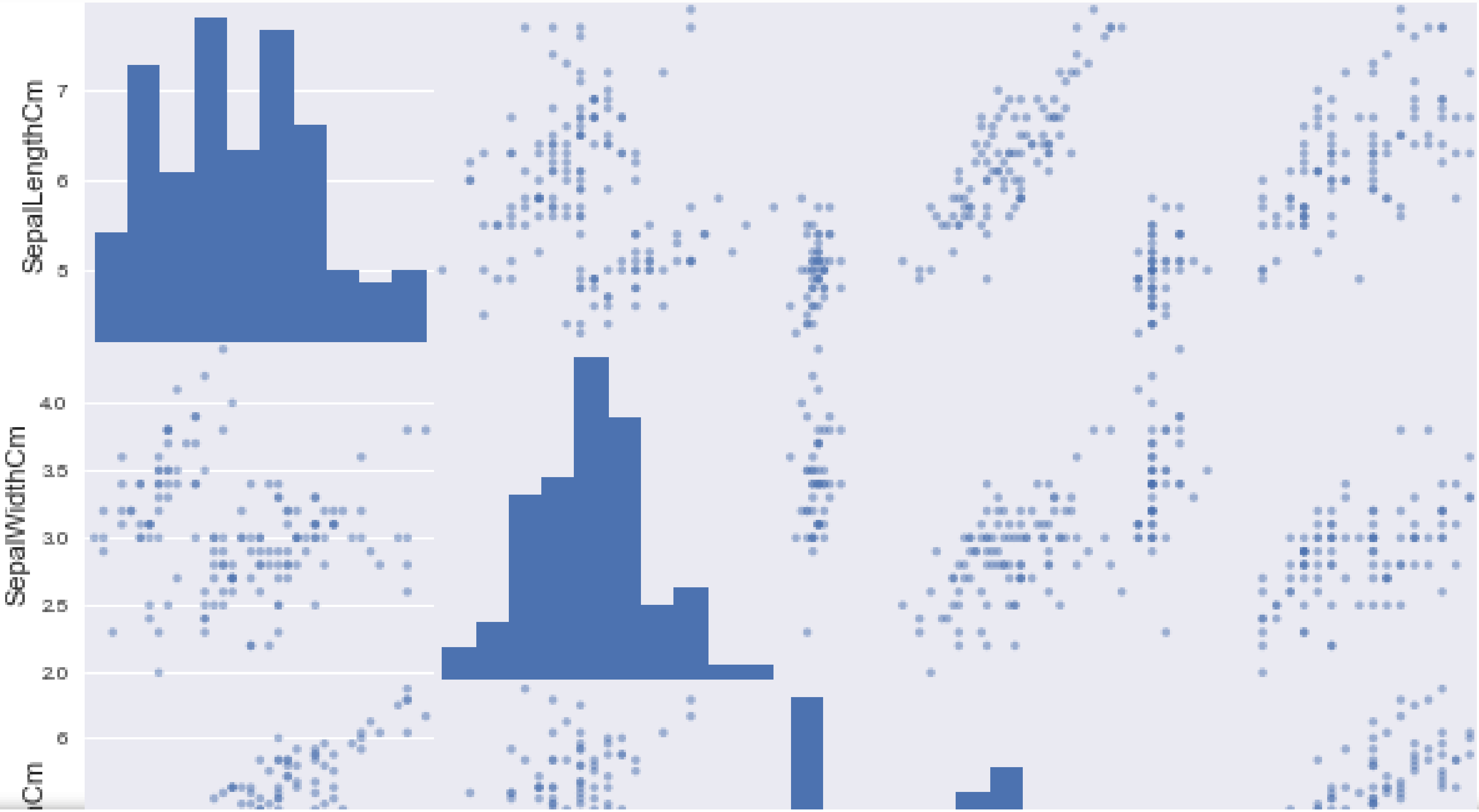
```
In [25]: # violinplots on petal-length for each species
         sns.violinplot(data=dataset,x="Species", y="PetalLengthCm")
```

Out[25]: <matplotlib.axes._subplots.AxesSubplot at 0xb48b8d0>



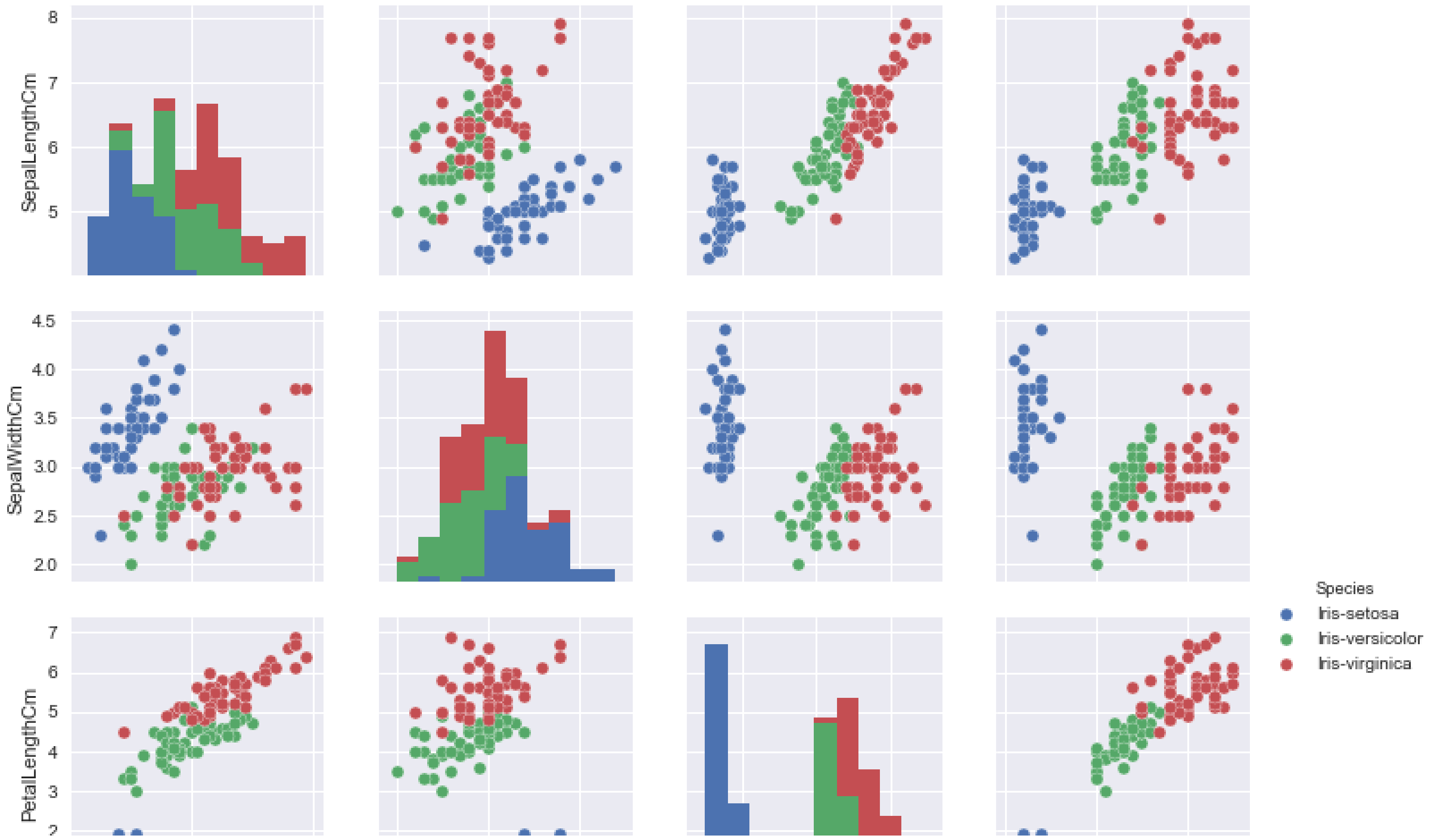
VOILIN PLOTS

In [26]:
```
from pandas.plotting import scatter_matrix
# scatter plot matrix
scatter_matrix(dataset,figsize=(10,10))
plt.show()
```



In [27]:
```python
# Using seaborn pairplot to see the bivariate relation between each pair of features
sns.pairplot(dataset, hue="Species")
```

Out[27]: <seaborn.axisgrid.PairGrid at 0xbc2f550>



In [28]:
```
# Importing metrics for evaluation
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
```

In [29]:
```
# Seperating the data into dependent and independent variables
X = dataset.iloc[:, :-1].values
y = dataset.iloc[:, -1].values

# Splitting the dataset into the Training set and Test set
from sklearn.cross_validation import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 0)
```

```
C:\Users\LENOVO\Anaconda1\lib\site-packages\sklearn\cross_validation.py:41: DeprecationWarning: This module was deprecated
in version 0.18 in favor of the model_selection module into which all the refactored classes and functions are moved. Also
note that the interface of the new CV iterators are different from that of this module. This module will be removed in 0.20
.
  "This module will be removed in 0.20.", DeprecationWarning)
```

In [31]:
```
# LogisticRegression
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression()
classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test)

# Summary of the predictions made by the classifier
print(classification_report(y_test, y_pred))
print(confusion_matrix(y_test, y_pred))
# Accuracy score
from sklearn.metrics import accuracy_score
print('accuracy is',accuracy_score(y_pred,y_test))
```

```
                 precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        11
Iris-versicolor       1.00      0.92      0.96        13
 Iris-virginica       0.86      1.00      0.92         6

    avg / total       0.97      0.97      0.97        30

[[11  0  0]
 [ 0 12  1]
 [ 0  0  6]]
accuracy is 0.9666666666666667
```

In [32]:

```python
# Decision Tree's
from sklearn.tree import DecisionTreeClassifier

classifier = DecisionTreeClassifier()

classifier.fit(X_train, y_train)

y_pred = classifier.predict(X_test)

# Summary of the predictions made by the classifier
print(classification_report(y_test, y_pred))
print(confusion_matrix(y_test, y_pred))
# Accuracy score
from sklearn.metrics import accuracy_score
print('accuracy is',accuracy_score(y_pred,y_test))
```

```
                 precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        11
Iris-versicolor       1.00      1.00      1.00        13
 Iris-virginica       1.00      1.00      1.00         6

    avg / total       1.00      1.00      1.00        30

[[11  0  0]
 [ 0 13  0]
 [ 0  0  6]]
accuracy is 1.0
```