In [26]:
```
# following steps
# step 1 Prepare A & b
# step 2 Import ML Algorithm
# step 3 train ML model
# step 4 new test predcit
```

In [5]:
```
# library
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import datasets

from sklearn.tree import DecisionTreeClassifier
from sklearn import metrics
%pylab inline
%matplotlib inline
```

```
%pylab is deprecated, use %matplotlib inline and import the required libraries.
Populating the interactive namespace from numpy and matplotlib
```

In [6]:
```
# data load
iris = pd.read_csv("./Downloads/Iris.csv")
iris.head()
```

Out[6]:

	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 [7]:
```
iris["Species"].value_counts()
```

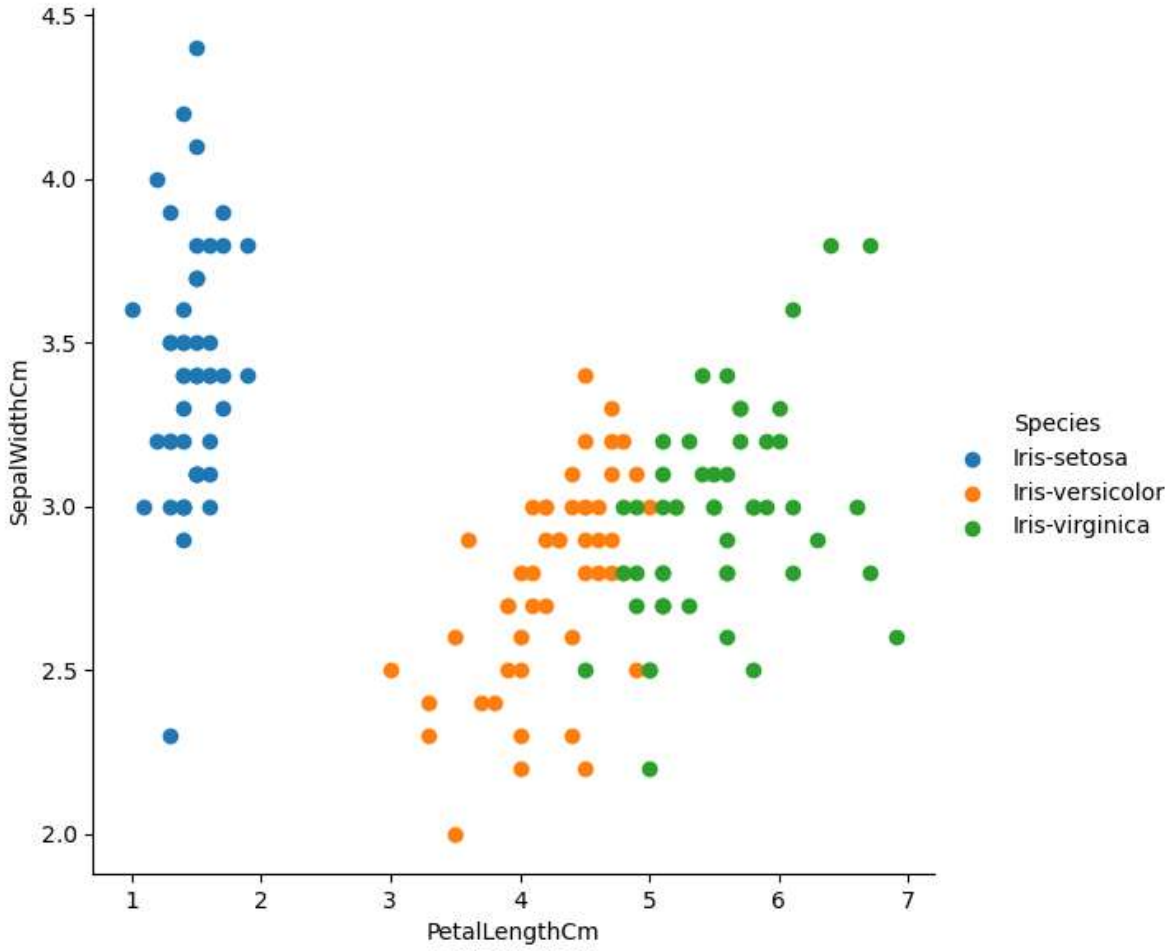
Out[7]:
```
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: Species, dtype: int64
```

In [8]:
```
print(iris.shape)
```

```
(150, 6)
```

In [9]:
```
# Scatter Plot
sns.FacetGrid(iris, hue="Species",height=6).map(plt.scatter, "PetalLengthCm", "Sepa
```

Out[9]:
```
<seaborn.axisgrid.FacetGrid at 0x1be2de4ed50>
```



In [10]:
```
# categorical varibales into numbers
flower_mapping = {'Iris-setosa' : 0, 'Iris-versicolor' : 1, 'Iris-virginica' : 2}
iris["Species"] = iris["Species"].map(flower_mapping)
```

In [11]:
```
iris.head()
```

Out[11]:

	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 [12]:
```
#inputs and outputs
A = iris[['SepalLengthCm','SepalWidthCm','PetalLengthCm','PetalWidthCm']].values
b = iris[['Species']].values
```

In [13]:
```
# import algorithm
from sklearn.linear_model import LogisticRegression
```

In [14]:
```
model = LogisticRegression()
```

In [15]:
```
model.fit(A,  b)
```

```
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\utils\validation.py:1143: DataC
onversionWarning: A column-vector y was passed when a 1d array was expected. Pleas
e change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

Out[15]:

▾ LogisticRegression

LogisticRegression()

In [16]:
```
# Accuracy
model.score(A,b)
```

Out[16]:
```
0.9733333333333334
```

In [17]:
```
# same input make prediction
expected = b
predicted = model.predict(A)
predicted
```

Out[17]:
```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2], dtype=int64)
```

In [18]:
```
from sklearn import metrics
```

In [19]:
```
# from the  scater plot the diffrence
print(metrics.classification_report(expected,predicted))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        50
           1       0.98      0.94      0.96        50
           2       0.94      0.98      0.96        50

    accuracy                           0.97       150
   macro avg       0.97      0.97      0.97       150
weighted avg       0.97      0.97      0.97       150
```

In [20]:
```
# prediction 'Iris-setosa':100%, 'Iris-versicolor' : 97%, 'Iris-virginica' : 99% ac
print(metrics.confusion_matrix(expected,predicted))
```

```
[[50  0  0]
 [ 0 47  3]
 [ 0  1 49]]
```

In [25]:
```
# This is test
A_new = np.array([[3,2,1,0.2],[4.9,2.2,3.8,1.1],[5.3,2.5,4.6,1.9]])
predicted = model.predict(A_new)
print("Predicted of Species: {}".format(predicted))
```

```
Predicted of Species: [0 1 2]
```