LetsGrowMore

Dec2022

Author:-Arnab Pal

Task(2)-Decision Tree

intermidiate level Language-Python

software-JUpyter Notebook(Colaboratory)

Importing Libraries

In [1]:

```
import pandas as pd
import seaborn as sb
import sklearn
import matplotlib.pyplot as plt
%matplotlib inline
```

In [2]:

```
iris=pd.read_csv("Iris.csv")
```

Checking out the data sets

In [3]:

```
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 [4]:

```
iris.head()
```

Out[4]:

	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 [5]:

```
iris.shape
```

Out[5]:

```
(150, 6)
```

In [7]:

```
iris.columns
```

Out[7]:

```
Index(['Id', 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm',
       'Species'],
      dtype='object')
```

In [8]:

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

Out[8]:

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

Knowing missing value

In [9]:

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

Out[9]:

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

In [10]:

```
iris.describe()
```

Out[10]:

	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

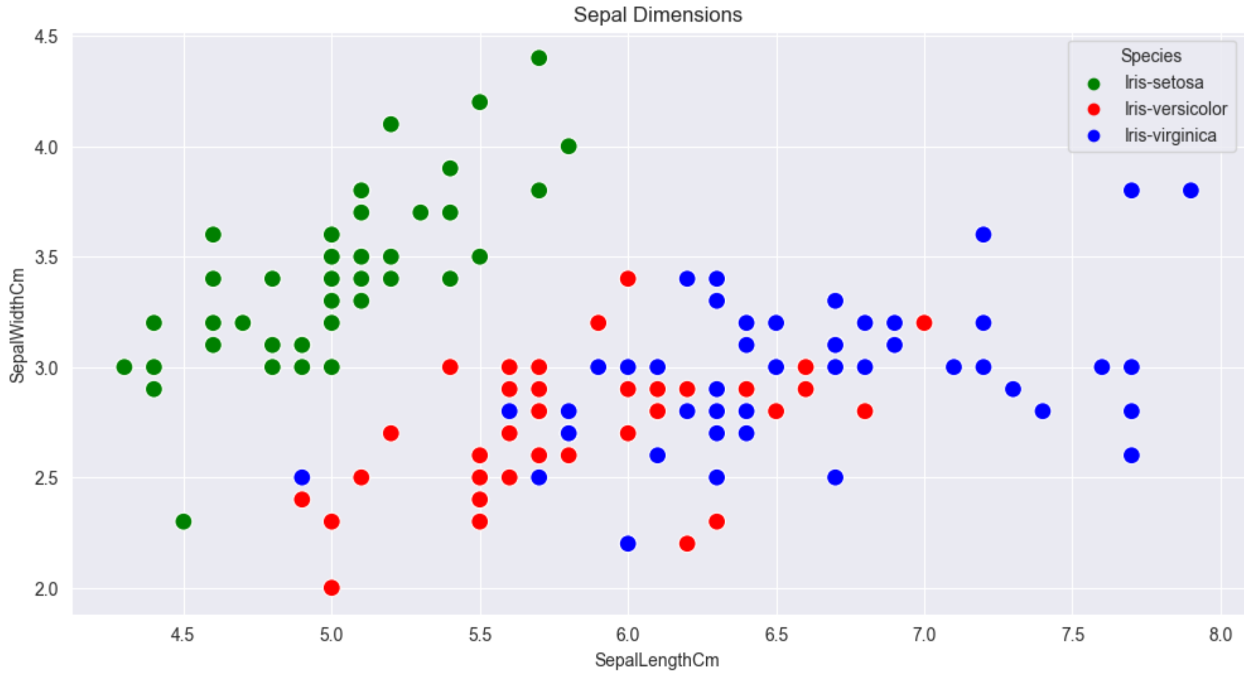
Visualising the sepal and petal dimensions

In [11]:

```
sb.set_style("darkgrid")
```

In [13]:

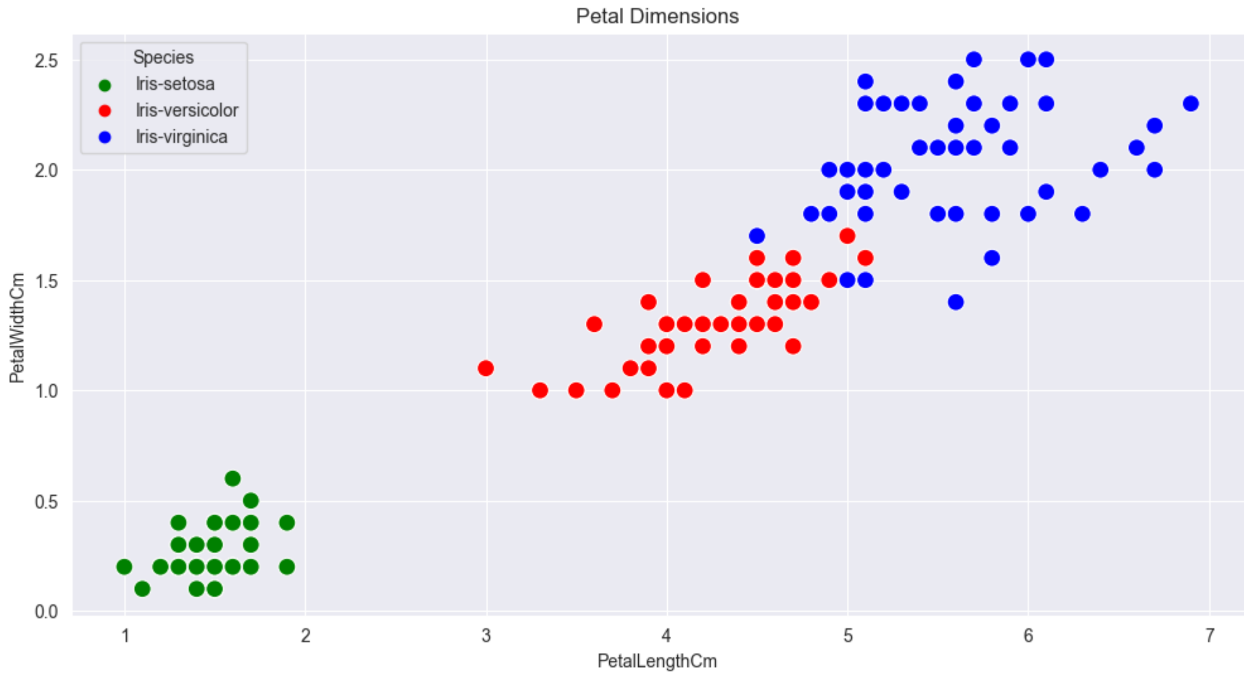
```
plt.figure(figsize=(12,6))
plt.title('Sepal Dimensions')
sb.scatterplot(x=iris["SepalLengthCm"],y=iris["SepalWidthCm"],hue=iris["Species"],palette=["green","red","blue"],s=100);
```



In [14]:

```
plt.figure(figsize=(12,6))
plt.title('Petal Dimensions')

sb.scatterplot(x=iris["PetalLengthCm"],y=iris["PetalWidthCm"],hue=iris["Species"],palette=["green","red","blue"],s=100);
```



Analiysing the distributions of sepal and petal dimentions

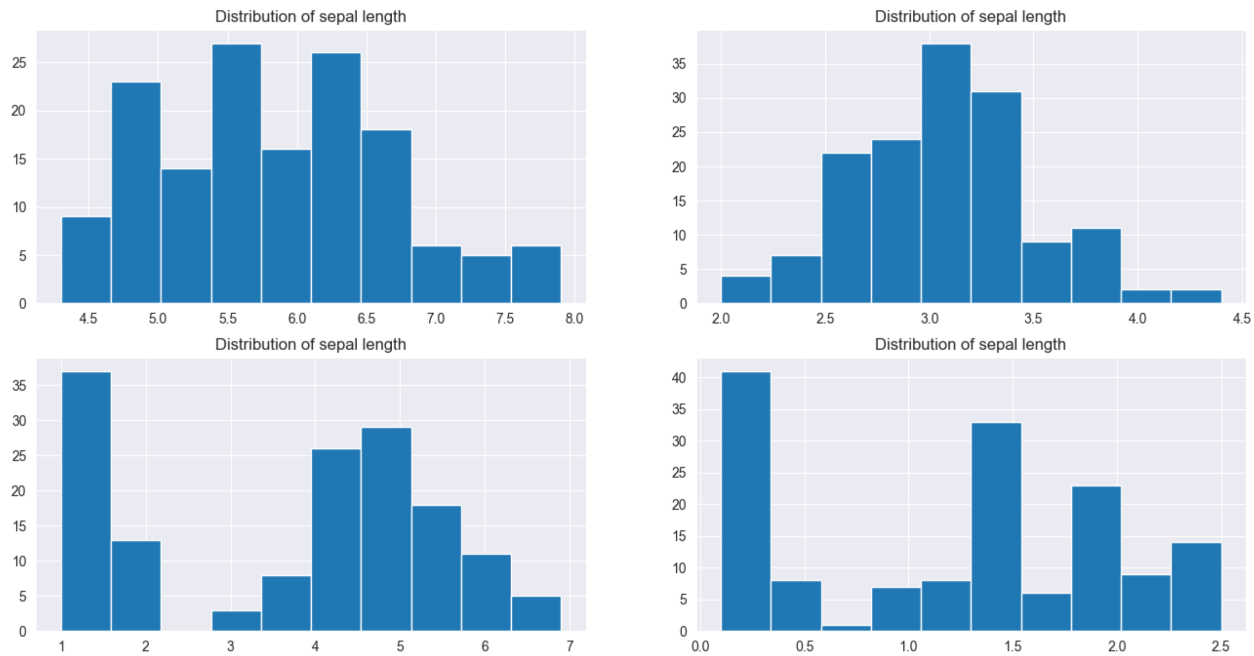
In [15]:

```
fig, axes = plt.subplots(2,2,figsize=(16,8))
#plot for sepal length
axes[0,0].set_title('Distribution of sepal length')
axes[0,0].hist(iris["SepalLengthCm"]);

#plot for sepal width
axes[0,1].set_title('Distribution of sepal length')
axes[0,1].hist(iris["SepalWidthCm"]);

#plot for petal length
axes[1,0].set_title('Distribution of sepal length')
axes[1,0].hist(iris["PetalLengthCm"]);

#plot for petal width
axes[1,1].set_title('Distribution of sepal length')
axes[1,1].hist(iris["PetalWidthCm"]);
```

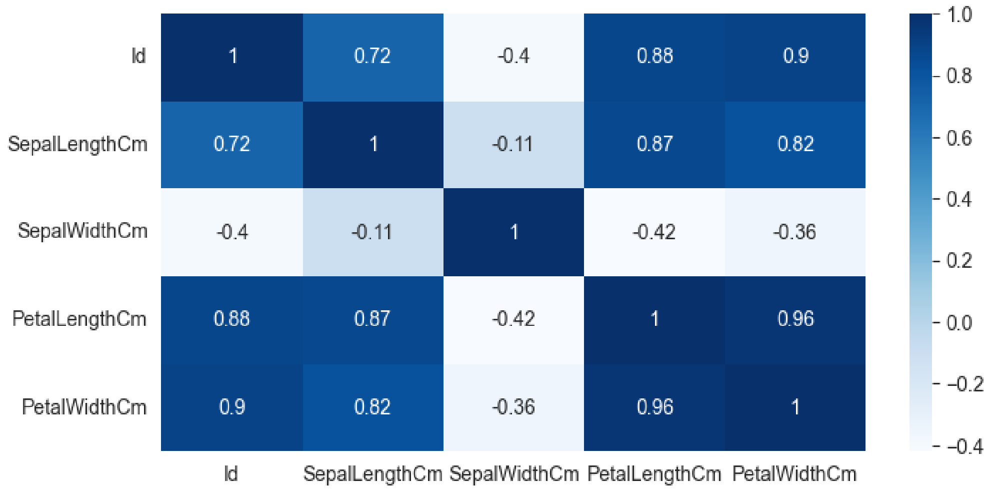


Analysing the correlation between feature and target

In [16]:

```
plt.figure(figsize=(8,4))
sb.heatmap(iris.corr(),annot=True,cmap='Blues');
```

```
C:\Users\ARNAB\AppData\Local\Temp\ipykernel_8440\3836521199.py:2: FutureWarning: The default value of numeric_only
in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specif
y the value of numeric_only to silence this warning.
  sb.heatmap(iris.corr(),annot=True,cmap='Blues');
```



In [17]:

```
X=iris[["SepalLengthCm","SepalWidthCm","PetalLengthCm","PetalWidthCm"]]
Y=iris[["Species"]]
```

Spliting the datasets

In [18]:

```
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X,Y,test_size=0.3,random_state=0)
```

In [19]:

```
print(X_train.shape)
print(Y_train.shape)
print(X_test.shape)
print(Y_test.shape)
```

```
(105, 4)
(105, 1)
(45, 4)
(45, 1)
```

In [20]:

```
iris.columns
```

Out[20]:

```
Index(['Id', 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm',
       'Species'],
      dtype='object')
```

In [21]:

```
from sklearn.tree import DecisionTreeClassifier
from sklearn import metrics
```

In [22]:

```
clf = DecisionTreeClassifier(max_depth = 5, random_state = 0)
# Train the model on the data
clf.fit(X_train, Y_train)
# Predict labels of unseen (test) data
# clf.predict(X_test)
```

Out[22]:

```
DecisionTreeClassifier(max_depth=5, random_state=0)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

Predictions

In [24]:

```
predictions = clf.predict(X_test)
print("The accuracy of Decision Tree is:",metrics.accuracy_score(predictions,Y_test))
```

```
The accuracy of Decision Tree is: 0.9777777777777777
```

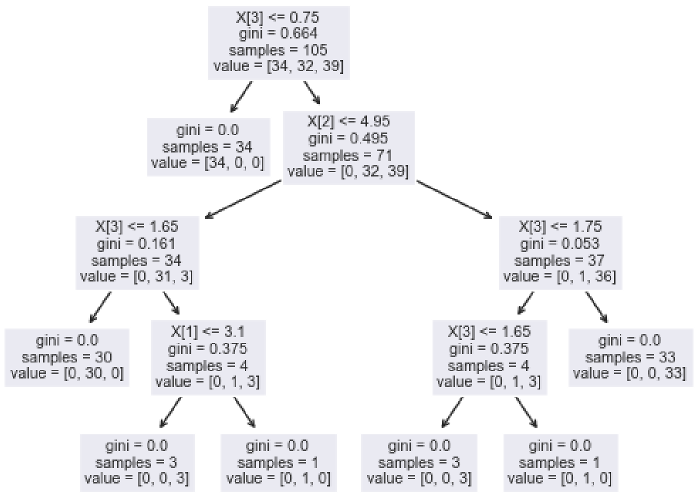
Visualizing the Tree

In [29]:

```
from sklearn import tree
tree.plot_tree(clf)
```

Out[29]:

```
[Text(0.4, 0.9, 'X[3] <= 0.75\ngini = 0.664\nsamples = 105\nvalue = [34, 32, 39]'),
 Text(0.3, 0.7, 'gini = 0.0\nsamples = 34\nvalue = [34, 0, 0]'),
 Text(0.5, 0.7, 'X[2] <= 4.95\ngini = 0.495\nsamples = 71\nvalue = [0, 32, 39]'),
 Text(0.2, 0.5, 'X[3] <= 1.65\ngini = 0.161\nsamples = 34\nvalue = [0, 31, 3]'),
 Text(0.1, 0.3, 'gini = 0.0\nsamples = 30\nvalue = [0, 30, 0]'),
 Text(0.3, 0.3, 'X[1] <= 3.1\ngini = 0.375\nsamples = 4\nvalue = [0, 1, 3]'),
 Text(0.2, 0.1, 'gini = 0.0\nsamples = 3\nvalue = [0, 0, 3]'),
 Text(0.4, 0.1, 'gini = 0.0\nsamples = 1\nvalue = [0, 1, 0]'),
 Text(0.8, 0.5, 'X[3] <= 1.75\ngini = 0.053\nsamples = 37\nvalue = [0, 1, 36]'),
 Text(0.7, 0.3, 'X[3] <= 1.65\ngini = 0.375\nsamples = 4\nvalue = [0, 1, 3]'),
 Text(0.6, 0.1, 'gini = 0.0\nsamples = 3\nvalue = [0, 0, 3]'),
 Text(0.8, 0.1, 'gini = 0.0\nsamples = 1\nvalue = [0, 1, 0]'),
 Text(0.9, 0.3, 'gini = 0.0\nsamples = 33\nvalue = [0, 0, 33]')]
```



In [ ]: