GRIP : The Sparks Foundation

Data Science & Business Analytics

Author: Ume Salma Khan

Task-6 : Prediction using Decision Tree Algorithm

• Create the Decision Tree classifier and visualize it graphically.

• The purpose is if we feed any new data to this classifier, it would be able to predict the right class accordingly.

In [1]:
```
# Importing required libraries

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
sns.set()
```

In [3]:
```
# Importing the Data
iris_data = pd.read_csv('Iris.csv')
iris_data.head()
```

Out[3]:

	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 [4]:
```
iris_data.describe()
```

Out[4]:

	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

Handling missing data

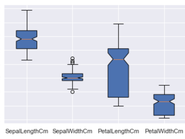
In [6]:
```
iris_data.isnull().sum()
```

Out[6]:
```
Id               0
SepalLengthCm    0
SepalWidthCm     0
PetalLengthCm    0
PetalWidthCm     0
Species          0
dtype: int64
```

In [7]:
```
SepalLengthCm = iris_data['SepalLengthCm']
SepalWidthCm = iris_data['SepalWidthCm']
PetalLengthCm = iris_data['PetalLengthCm']
PetalWidthCm = iris_data['PetalWidthCm']

columns = [SepalLengthCm,SepalWidthCm,PetalLengthCm,PetalWidthCm]

fig,ax = plt.subplots()
ax.boxplot(columns,notch=True,patch_artist=True)
plt.xticks([1,2,3,4],['SepalLengthCm','SepalWidthCm','PetalLengthCm','PetalWidthCm'])
plt.show()
```



In [8]:
```
Q1 = iris_data['SepalWidthCm'].quantile(0.25)
Q3 = iris_data['SepalWidthCm'].quantile(0.75)
IQR = Q3 - Q1

ur = Q3+1.5*IQR
lr = Q1-1.5*IQR

samp = iris_data.index[iris_data['SepalWidthCm'] > ur]
samp.append(iris_data.index[iris_data['SepalWidthCm'] < lr])
iris_data = iris_data.drop(samp)
iris_data.reset_index(drop=True)
```

Out[8]:

	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
...	...	...	...	...	...	...
142	146	6.7	3.0	5.2	2.3	Iris-virginica
143	147	6.3	2.5	5.0	1.9	Iris-virginica
144	148	6.5	3.0	5.2	2.0	Iris-virginica
145	149	6.2	3.4	5.4	2.3	Iris-virginica
146	150	5.9	3.0	5.1	1.8	Iris-virginica

147 rows × 6 columns

Split the dataset for training and testing

In [9]:
```
df = iris_data.copy()
x = df.iloc[:,1:5]
y = df.iloc[:,-1]

from sklearn.model_selection import train_test_split

x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.25,random_state=42)
```

In [10]:
```
from sklearn.tree import DecisionTreeClassifier,plot_tree
from sklearn.metrics import accuracy_score

clf = DecisionTreeClassifier()
clf.fit(x_train,y_train)
y_pred = clf.predict(x_test)
```

In [11]:
```
accuracy = accuracy_score(y_test,y_pred)
accuracy*100
```

Out[11]:
```
97.2972972972973
```

In [13]:
```
iris_data = {'y_Actual': y_test,
             'y_Predicted': y_pred
            }

df = pd.DataFrame(iris_data)
df.reset_index(inplace = True, drop = True)
df.head()
```

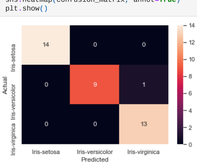
Out[13]:

	y_Actual	y_Predicted
0	Iris-virginica	Iris-virginica
1	Iris-versicolor	Iris-versicolor
2	Iris-virginica	Iris-virginica
3	Iris-setosa	Iris-setosa
4	Iris-virginica	Iris-virginica

In [14]:
```
confusion_matrix = pd.crosstab(df['y_Actual'], df['y_Predicted'], rownames=['Actual'], colnames=['Predicted'])
print (confusion_matrix)
```

```
Predicted        Iris-setosa  Iris-versicolor  Iris-virginica
Actual
Iris-setosa               14                0               0
Iris-versicolor            0                9               1
Iris-virginica             0                0              13
```

In [15]:
```
sns.heatmap(confusion_matrix, annot=True)
plt.show()
```

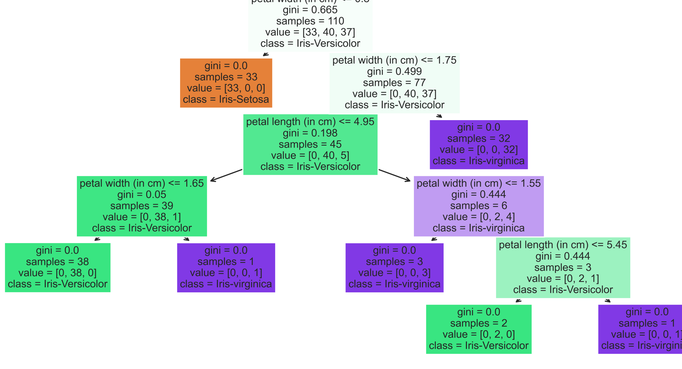


In [17]:
```
#Data Visualizion:

featName=['sepal length ( in cm)','sepal width (in cm)','petal length (in cm)','petal width (in cm)']
clsName=['Iris-Setosa','Iris-Versicolor','Iris-virginica']

figure, axes = plt.subplots(nrows = 1, ncols = 1, figsize = (12,6), dpi = 500)

tree.plot_tree(clf, feature_names = featName, class_names = clsName, filled = True);
```



In [18]:
```
clf.predict_proba([[4.7,3.2,1.3,0.2]])
```

Out[18]:
```
array([[1., 0., 0.]])
```

In [19]:
```
clf.predict([[4.7,3.2,1.3,0.2]])
```

Out[19]:
```
array(['Iris-setosa'], dtype=object)
```

Hence, the Decision Tree Model has been created and visualized with the accuracy of 97.29% in the Test dataset. It also predict the class of the new data successfully.