Assignment- 4: IRIS Dataset

Created by:

Srinidhi Devan

Step-1: Importing the necessary libraries

```
import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from sklearn import tree

from sklearn.tree import DecisionTreeClassifier, plot_tree
```

- Here, apart from NumPy and Pandas, the additional libraries imported are: 'tree', 'DecisionTreeClassifier', 'plot_tree' and 'train_test_split' from 'sklearn' module of Python for this particular problem.

- 'DecisionTreeClassifier': Facilitates in building a decision tree model in Python

- 'plot_tree': For tree visualization purpose

- 'train_test_split': For splitting the data into train (70%) & test (30%) set respectively

Step-2: Reading the Iris Dataset

```
data = pd.read_csv('Iris.csv')

data = data.drop(['Id'],axis=1)
```

- Using 'pd.read_csv' command

- Dropping the 'Id' variable from the dataset; axis=1 implies we are dropping from column

Step-3: Separating the Independent and Dependent variables in the data

```
x = data[['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']]


y = data['Species']
```

- Here, the Independent Variables are: 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm'.
  This is called 'x'

- The Dependent Variable is: 'Species'
  This is called 'y'

Step-4: Splitting the Dataset into Train & Test set

```
(x_train, x_test, y_train, y_test) = train_test_split(x, y, train_size=0.7, random_state=1)
```

- The train set consists of 70% of the dataset and test set consists of 30%

- The splitting is done using 'train_test_split' command

- 'random_state'--> to ensure that there is always uniformity in splitting

Step-5: Building the Decision Tree model

```
dt_model = DecisionTreeClassifier(criterion = 'gini' )
```

- This is done using 'DecisionTreeClassifier' package in Python

```
dt_model.fit(x_train, y_train)
```

- And we fit the model for further analysis—using '.fit()' for training set (including both Independent (x) and dependent variables(y))

Step-6: Finding the Feature Importance

```
pd.DataFrame(dt_model.feature_importances_, columns = ["Imp"], index = x_train.columns)
```

- This will show us which of the Independent variables have more weightage relative to the other.

Independent Variable	Feature Importance
SepalLengthCm	0.02147
SepalWidthCm	0.02147
PetalLengthCm	0.06317
PetalWidthCm	0.89389

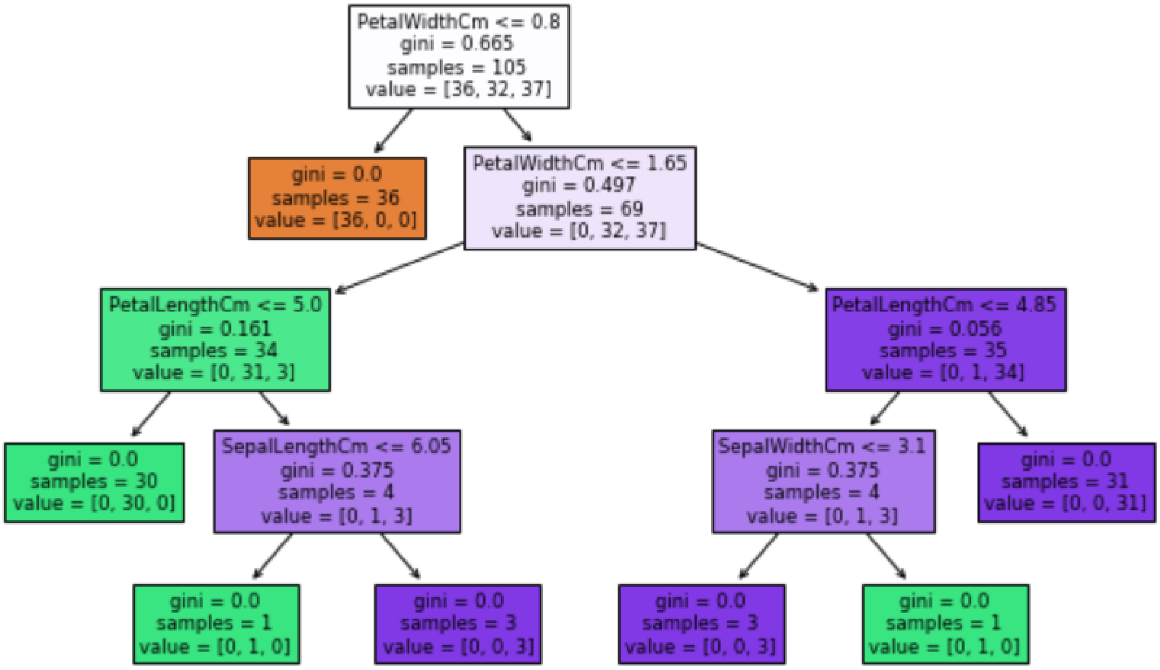
- From the above table, we can see that ‘PetalWidthCm’ has the highest importance followed by ‘PetalLengthCm’, ‘SepalWidthCm’ and ‘SepalLengthCm’.

- This is done on independent variables of the training dataset.

Step-7: Plotting the Decision Tree

```
fig = tree.plot_tree(dt_model,feature_names=data.columns,filled='True')
```

- This is done with the package ‘plot_tree’ on ‘dt_model’
  Where dt_model = DecisionTreeClassifier(criterion = 'gini')



Step-8: Creating Classification Report

- This is done on both Train and Test dataset

- Classification Report shows us the 'Precision', 'Recall', 'F1 Score' and 'Accuracy'

```
classification_report(y_train, ytrain_predict)
```

- Where

```
ytrain_predict = dt_model.predict(x_train)
```

Classification Report for Train Set

	Precision	Recall	F1 Score	Support
Iris-setosa	1	1	1	36
Iris-versicolor	1	1	1	32
Iris-virginica	1	1	1	37
accuracy			1	105
macro avg	1	1	1	105
weighted avg	1	1	1	105

a. PRECISION: Percentage of Iris-setosa correctly predicted is 100%
Percentage of Iris-versicolor correctly predicted is 100%
Percentage of Iris-virginica correctly predicted is 100%

b. RECALL: Percentage of positive cases in Iris-setosa is 100%
Percentage of positive cases in Iris-versicolor is 100%
Percentage of positive cases in Iris-virginica is 100%

c. F1-SCORE: Percentage of positive predictions in Iris-setosa which were correct is 100%
Percentage of positive predictions in Iris-versicolor which were correct is 100%
Percentage of positive predictions in Iris- virginica which were correct is 100%

d. ACCURACY: 100%

```
classification_report(y_test, ytest_predict)
```

- Where

```
ytest_predict = dt_model.predict(x_test)
```

Classification Report for Test Set

	Precision	Recall	F1 Score	Support
Iris-setosa	1	1	1	14
Iris-versicolor	0.94	0.94	0.94	18
Iris-virginica	0.92	0.92	0.92	13
accuracy			0.96	45
macro avg	0.96	0.96	0.96	45
weighted avg	0.96	0.96	0.96	45

a. PRECISION: Percentage of Iris-setosa correctly predicted is 100%
   Percentage of Iris-versicolor correctly predicted is 94%
   Percentage of Iris-virginica correctly predicted is 92%

b. RECALL: Percentage of positive cases in Iris-setosa is 100%
   Percentage of positive cases in Iris-versicolor is 94%
   Percentage of positive cases in Iris-virginica is 92%

c. F1-SCORE: Percentage of positive predictions in Iris-setosa which were correct is 100%
   Percentage of positive predictions in Iris-versicolor which were correct is 94%
   Percentage of positive predictions in Iris- virginica which were correct is 92%

d. ACCURACY: 96%

Step-9: Accuracy

- This is found using Confusion Matrix as shown above.

- Alternatively,

```
dt_model.score(x_train,y_train)
```

a. For Train Dataset: 100%

```
dt_model.score(x_test,y_test)
```

b. For Test Dataset: 95.55%