Assignment - 09

Aishwarya Jagtap

implement the decision tree classifier on the iris dataset

implement the gridsearchcv on the decision tree classifier on the iris dataset

In [2]:
```
import numpy as np
import pandas as pd
```

In [3]:
```
df=pd.read_csv("iris.csv")
```

In [4]:
```
df
```

Out[4]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	virginica
146	6.3	2.5	5.0	1.9	virginica
147	6.5	3.0	5.2	2.0	virginica
148	6.2	3.4	5.4	2.3	virginica
149	5.9	3.0	5.1	1.8	virginica

150 rows × 5 columns

In [9]:
```
df.head()
```

Out[9]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

In [10]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   sepal_length  150 non-null    float64
 1   sepal_width   150 non-null    float64
 2   petal_length  150 non-null    float64
 3   petal_width   150 non-null    float64
 4   species       150 non-null    object 
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

In [11]:
```
df.isnull().sum()
```

Out[11]:
```
sepal_length    0
sepal_width     0
petal_length    0
petal_width     0
species         0
dtype: int64
```

In [12]:
```
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
```

In [13]:
```
x=df.drop('species', axis=1)
y=df['species']
```

In [17]:
```
x.shape
```

Out[17]:
```
(150, 4)
```

In [16]:
```
y.shape
```

Out[16]:
```
(150,)
```

In [18]:
```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_sta
```

In [19]:
```
clf = DecisionTreeClassifier()
```

In [20]:
```
clf.fit(x_train, y_train)


y_pred = clf.predict(x_test)
```

In [21]:
```
accuracy = accuracy_score(y_test, y_pred)
```

In [22]:
```
accuracy
```

Out[22]:
```
1.0
```

In [23]:
```
conf_matrix = confusion_matrix(y_test, y_pred)
classification_rep = classification_report(y_test, y_pred)
```

In [24]:
```
print("Confusion Matrix:")
print(conf_matrix)
print("Classification Report:")
print(classification_rep)
```

```
Confusion Matrix:
[[10  0  0]
 [ 0  9  0]
 [ 0  0 11]]
Classification Report:
              precision    recall  f1-score   support

      setosa       1.00      1.00      1.00        10
  versicolor       1.00      1.00      1.00         9
   virginica       1.00      1.00      1.00        11

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30
```

Implement GridSearchCV on Decision Tree Classifier Define the parameter grid for GridSearchCV

In [25]:
```
from sklearn.model_selection import train_test_split, GridSearchCV
```

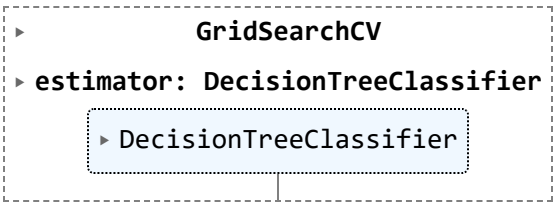
In [26]:
```
dt_classifier = DecisionTreeClassifier()
```

In [38]:
```
param_grid = {
    'criterion': ['gini', 'entropy'],
    'max_depth': [None, 5, 10, 15],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}
```

In [39]:
```
grid_search = GridSearchCV(dt_classifier, param_grid, cv=5, scoring='accuracy')
```

In [40]:
```
grid_search.fit(x_train, y_train)
```

Out[40]:



In [41]:
```
print("Best Parameters:", grid_search.best_params_)
```

```
Best Parameters: {'criterion': 'entropy', 'max_depth': None, 'min_samples_leaf': 4,
'min_samples_split': 2}
```

In [42]:
```
best_dt_classifier = grid_search.best_estimator_
```

In [43]:
```
best_dt_classifier
```

Out[43]:
```
DecisionTreeClassifier(criterion='entropy', min_samples_leaf=4)
```

In [44]:
```
y_pred = best_dt_classifier.predict(x_test)
```

In [45]:
```
accuracy = accuracy_score(y_test, y_pred)
```

In [46]:
```
accuracy
```

Out[46]:
```
1.0
```

In [47]:
```
conf_matrix = confusion_matrix(y_test, y_pred)
classification_rep = classification_report(y_test, y_pred)
```

In [48]:
```
print("Confusion Matrix:")
print(conf_matrix)
print("Classification Report:")
print(classification_rep)
```

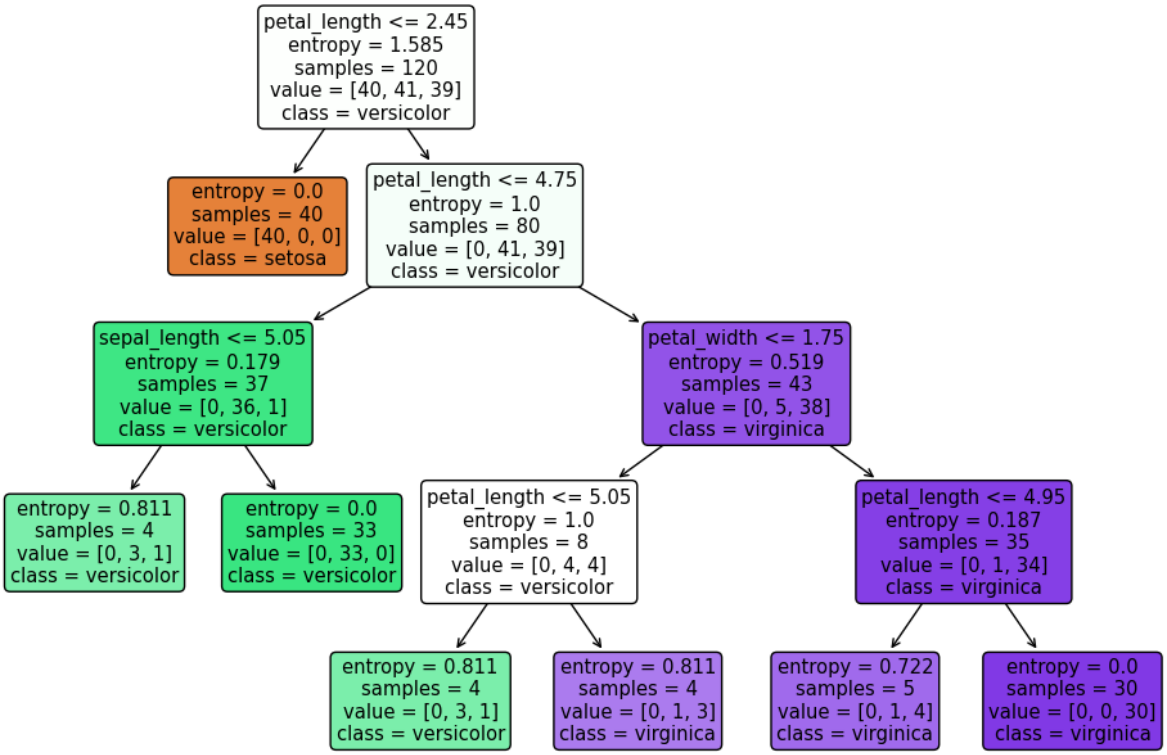
```
Confusion Matrix:
[[10  0  0]
 [ 0  9  0]
 [ 0  0 11]]
Classification Report:
              precision    recall  f1-score   support

      setosa       1.00      1.00      1.00        10
  versicolor       1.00      1.00      1.00         9
   virginica       1.00      1.00      1.00        11

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30
```

In [51]:
```
from sklearn.tree import plot_tree
import matplotlib.pyplot as plt

# Visualize the Decision Tree
plt.figure(figsize=(12, 8))
plot_tree(best_dt_classifier, feature_names=x.columns, class_names=y.unique(), fill
plt.show()
```


petal_length <= 2.45
entropy = 1.585
samples = 120
value = [40, 41, 39]
class = versicolor
entropy = 0.0
samples = 40
value = [40, 0, 0]
class = setosa
petal_length <= 4.75
entropy = 1.0
samples = 80
value = [0, 41, 39]
class = versicolor
sepal_length <= 5.05
entropy = 0.179
samples = 37
value = [0, 36, 1]
class = versicolor
petal_width <= 1.75
entropy = 0.519
samples = 43
value = [0, 5, 38]
class = virginica
entropy = 0.811
samples = 4
value = [0, 3, 1]
class = versicolor
entropy = 0.0
samples = 33
value = [0, 33, 0]
class = versicolor
petal_length <= 5.05
entropy = 1.0
samples = 8
value = [0, 4, 4]
class = versicolor
petal_length <= 4.95
entropy = 0.187
samples = 35
value = [0, 1, 34]
class = virginica
entropy = 0.811
samples = 4
value = [0, 3, 1]
class = versicolor
entropy = 0.811
samples = 4
value = [0, 1, 3]
class = virginica
entropy = 0.722
samples = 5
value = [0, 1, 4]
class = virginica
entropy = 0.0
samples = 30
value = [0, 0, 30]
class = virginica

In [ ]: