Chapter 6: DecisionTreeClassifier

In [1]:
```
# from google.colab import drive
# drive.mount("/content/gdrive", force_remount=True)
# %cd '/content/gdrive/My Drive/LDS6_MachineLearning/practice/Chapter6_Decision_
```

In [ ]:
```
from sklearn.tree import DecisionTreeClassifier
from sklearn import datasets
from IPython.display import Image
from sklearn import tree
import pydotplus
import pandas as pd
```

In [ ]:
```
iris = pd.read_excel("Iris.xls")
iris.info()
```

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

In [ ]:
```
X = iris[['sepallength', 'sepalwidth', 'petallength', 'petalwidth']] # numeric
y = iris['iris'] # loai hoa
```

In [ ]:
```
X.head()
```

Out[5]:

	sepallength	sepalwidth	petallength	petalwidth
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

In [ ]:
```
y[:5]
```

Out[6]:
```
0    Iris-setosa
1    Iris-setosa
2    Iris-setosa
3    Iris-setosa
4    Iris-setosa
Name: iris, dtype: object
```

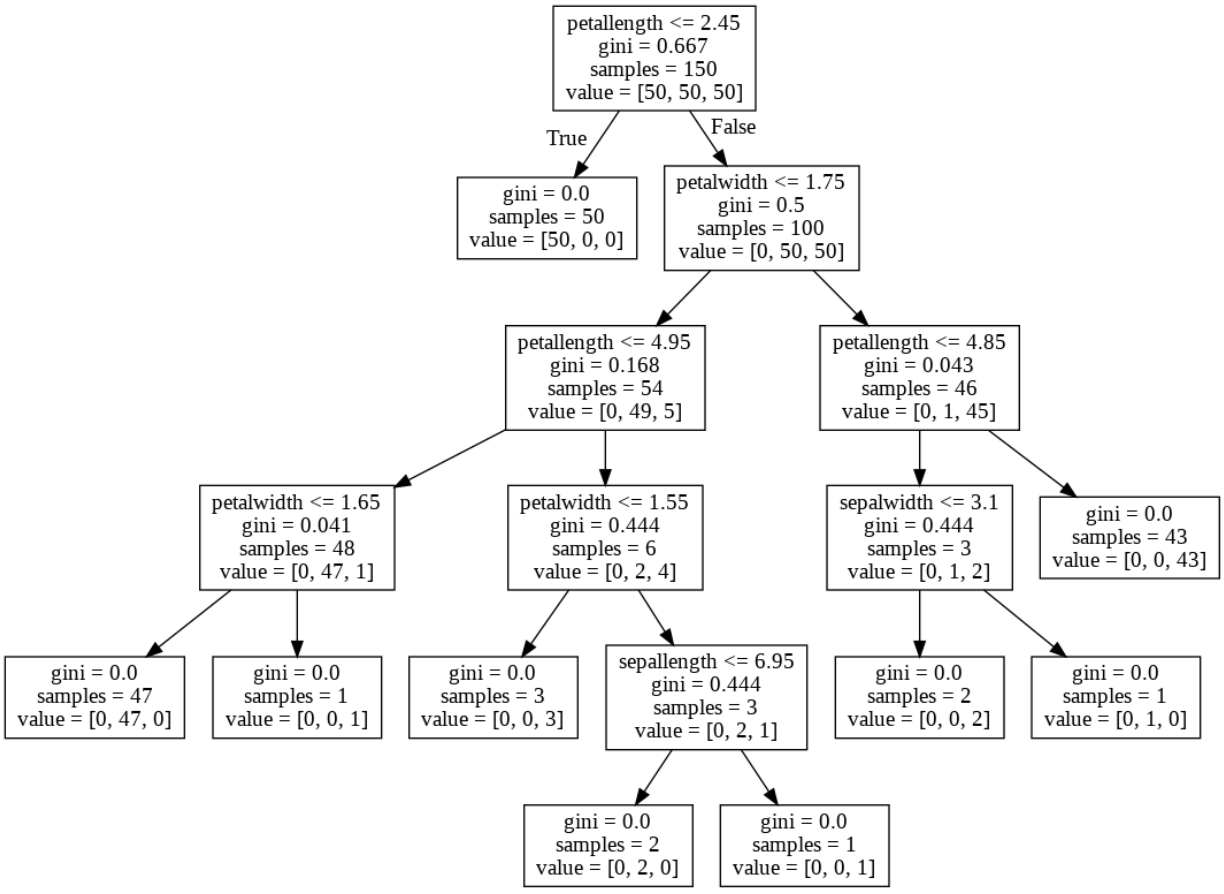
In [ ]:
```
# https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClas
# co the thay the bang criterion='entropy' tuy bai toan
clf = DecisionTreeClassifier()
model = clf.fit(X, y)
```

Note:

- Gini is intended for continuous attributes and Entropy is for attributes that occur in classes.
- Gini is to minimize misclassification
- Entropy is for exploratory analysis
- Entropy is a little slower to compute

In [ ]:
```
dot_data = tree.export_graphviz(clf, out_file=None,
                                feature_names=X.columns
                                )
graph = pydotplus.graph_from_dot_data(dot_data)
Image(graph.create_png())
```

Out[8]:



In [ ]:
```
# Save Decision Tree Image To File
# Create PDF
graph.write_pdf(path + "iris_1510.pdf")
# Create PNG
graph.write_png(path + "iris_1510.png")
```

Out[9]:
```
True
```

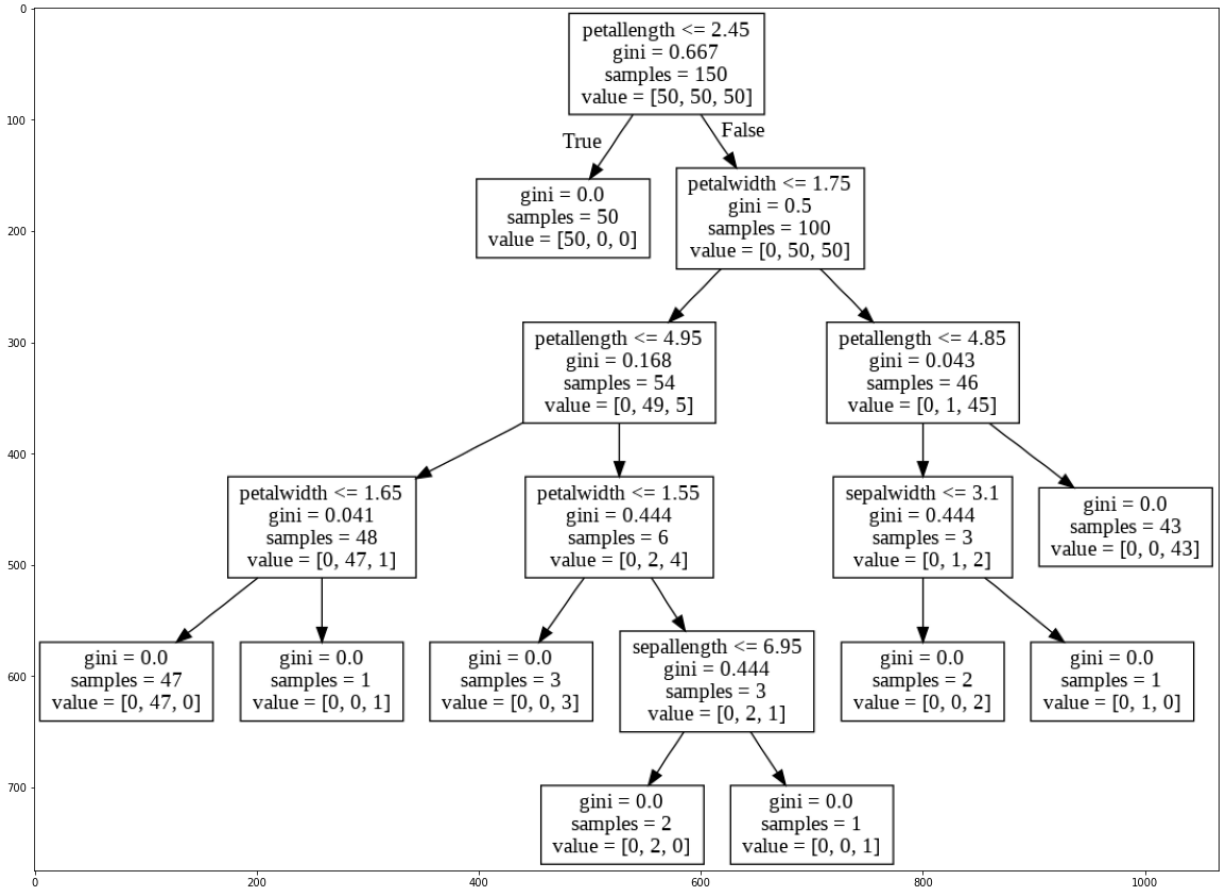
In [ ]:
```
# Link: https://chrisalbon.com/machine_learning/trees_and_forests/visualize_a_dec
```

In [ ]:
```
# Another solution to save and show decision tree
# Save to file
with open(path + "iris_1510.txt", "w") as f:
    f = tree.export_graphviz(clf, out_file=f,
                             feature_names=X.columns,
                             class_names=iris.iris)
    print("Đã xuất file")
```

```
Đã xuất file
```

In [ ]:
```
# dùng để mở và xem kết quả
# https://dreampuf.github.io/GraphvizOnline/
# hoặc http://viz-js.com/
import matplotlib.pyplot as plot
import imageio
photo_data = imageio.imread("iris_1510.png")
plot.figure(figsize = (20, 20))
plot.imshow(photo_data)
```

Out[12]:
```
<matplotlib.image.AxesImage at 0x7fcff81fed30>
```



In [ ]:
```
model.score(X, y)
```

Out[13]:
```
1.0
```

In [ ]:
```
# Make new prediction
import numpy as np
X_new = np.array([[4.7, 3.2, 1.3, 0.2],
                  [6.6, 3. , 4.4, 1.4],
                  [5.9, 3. , 5.1, 1.8]])
```

In [ ]:
```
yhat_new = model.predict(X_new)
yhat_new
```

Out[15]:
```
array(['Iris-setosa', 'Iris-versicolor', 'Iris-virginica'], dtype=object)
```