Chapter6 - exercise 2: Play Goft

Cho dữ liệu play golf trong tập tin playgoft_data.xlsx.

Yêu cầu: Hãy đọc dữ liệu từ tập tin này, áp dụng Decision Tree để thực hiện việc xác định có đi chơi golf hay không dựa trên các thông tin như: 'Outlook', 'Temperature', 'Humidity', 'Wind', 'Play Golf'

Yêu cầu:

1. Hãy chuẩn hóa dữ liệu cho phù hợp
2. Áp dụng Decsion. Tìm kết quả
3. Kiểm tra độ chính xác
4. Xuất/ghi model
5. Đọc model
6. Cho dữ liệu Test: X_test = [["Overcast", "Cool", "High", "Strong"], ["Sunny", "Cool", "High", "Weak"]] => Y_test

In [1]:
```
import pandas as pd
import numpy as np
```

In [2]:
```
df = pd.read_excel('playgolf_data.xlsx', index_col = 0)
#df
```

In [3]:
```
df.columns
```

Out[3]:
```
Index(['Outlook', 'Temperature', 'Humidity', 'Wind', 'Play Golf'], dtype='objec
t')
```

In [4]:
```
features = df.drop("Play Golf", axis=1)
target = df[["Play Golf"]]
```

In [5]:
```
#target
```

In [6]:
```
from sklearn.preprocessing import LabelEncoder
```

In [7]:
```
features = pd.get_dummies(features)
features
```

Out[7]:

Day	Outlook_Overcast	Outlook_Rain	Outlook_Sunny	Temperature_Cool	Temperature_Hot	Temper
1	0	0	1	0	1	
2	0	0	1	0	1	
3	1	0	0	0	1	
4	0	1	0	0	0	
5	0	1	0	1	0	
6	0	1	0	1	0	
7	1	0	0	1	0	
8	0	0	1	0	0	
9	0	0	1	1	0	
10	0	1	0	0	0	
11	0	0	1	0	0	
12	1	0	0	0	0	
13	1	0	0	0	1	
14	0	1	0	0	0	

In [8]:
```
from sklearn.tree import DecisionTreeClassifier
from sklearn.utils.validation import column_or_1d
```

In [9]:
```
#Create a Gaussian Classifier
model = DecisionTreeClassifier() # criterion = 'entropy'
# Train the model using the training sets
model.fit(features, target)
```

Out[9]:
```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
            max_features=None, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, presort=False, random_state=None,
            splitter='best')
```

In [10]:
```
# Kiểm tra độ chính xác
print("The prediction accuracy is: ", model.score(features,target)*100,"%")
```

```
The prediction accuracy is:  100.0 %
```

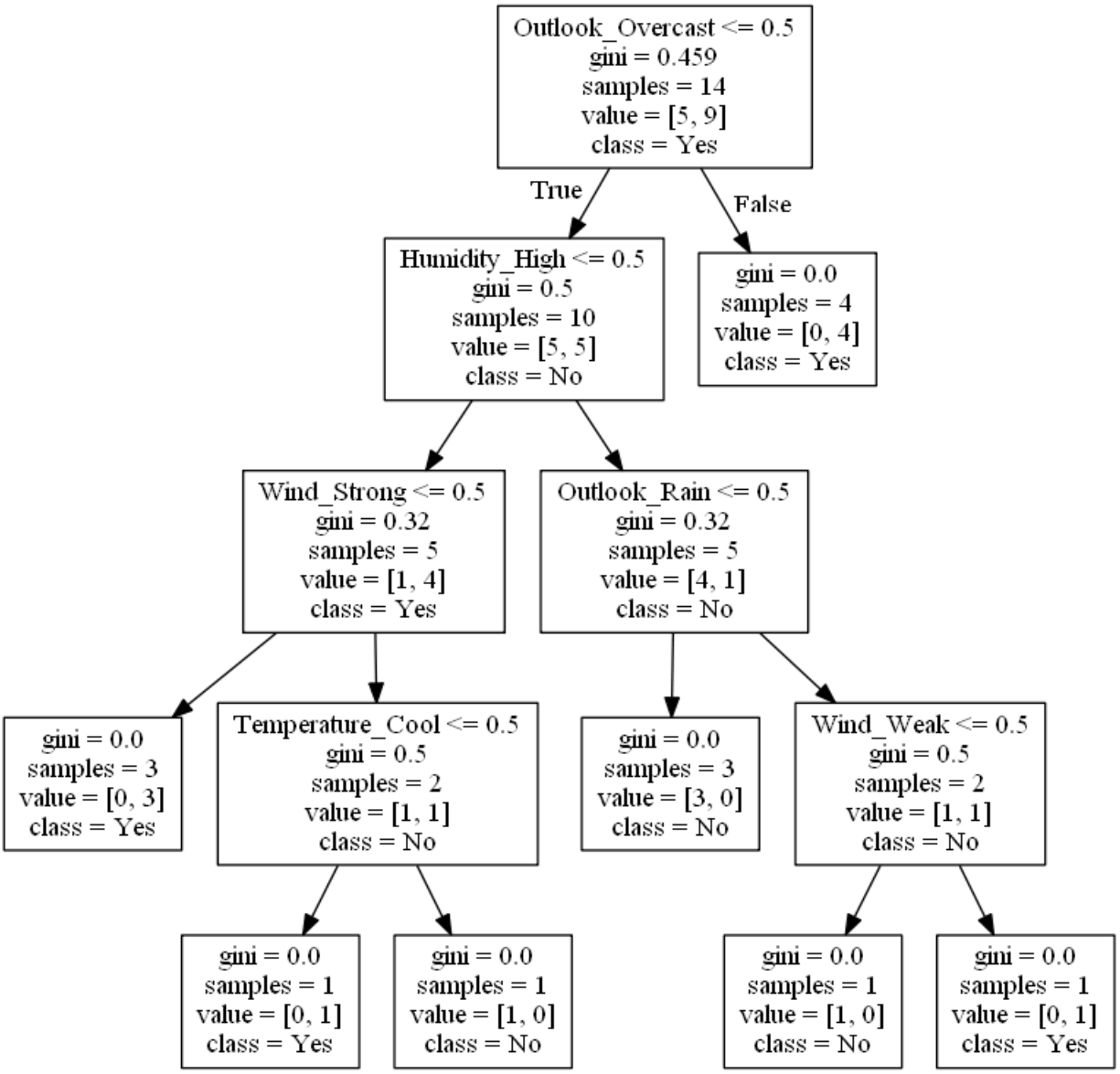
In [11]:
```
class_names = model.classes_
class_names
```

Out[11]:
```
array(['No', 'Yes'], dtype=object)
```

In [12]:
```
from IPython.display import Image
from sklearn import tree
import pydotplus
```

In [13]:
```
dot_data = tree.export_graphviz(model, out_file=None,
                                feature_names=features.columns,
                                class_names=np.array(["No","Yes"])
                                )
graph = pydotplus.graph_from_dot_data(dot_data)
Image(graph.create_png())
```

Out[13]:



In [14]:
```
# Xuất model
import pickle
pkl_filename = "playgoft_model.pkl"
with open(pkl_filename, 'wb') as file:
    pickle.dump(model, file)
```

In [15]:
```
with open(pkl_filename, 'rb') as file:
    playgoft_model = pickle.load(file)
```

In [16]:
```
# Outlook_Overcast  Outlook_Rain    Outlook_Sunny
# Temperature_Cool  Temperature_Hot Temperature_Mild
# Humidity_High Humidity_Normal
# Wind_Strong   Wind_Weak
# _________
# X_test = [["Overcast", "Cool", "High", "Strong"], ["Sunny", "Cool", "High", "Wea
X_test = [[1, 0, 0, 1, 0, 0, 1, 0, 1, 0], [0, 0, 1, 1, 0, 0, 1, 0, 0, 1]]
y_pred = playgoft_model.predict(X_test)
y_pred
```

Out[16]:
```
array(['Yes', 'No'], dtype=object)
```

Note: https://datascience.stackexchange.com/questions/10228/when-should-i-use-gini-impurity-as-opposed-to-information-gain (https://datascience.stackexchange.com/questions/10228/when-should-i-use-gini-impurity-as-opposed-to-information-gain) (They like to use gini because it's more simple than entropy. It doesn't require to compute logarithmic functions)