

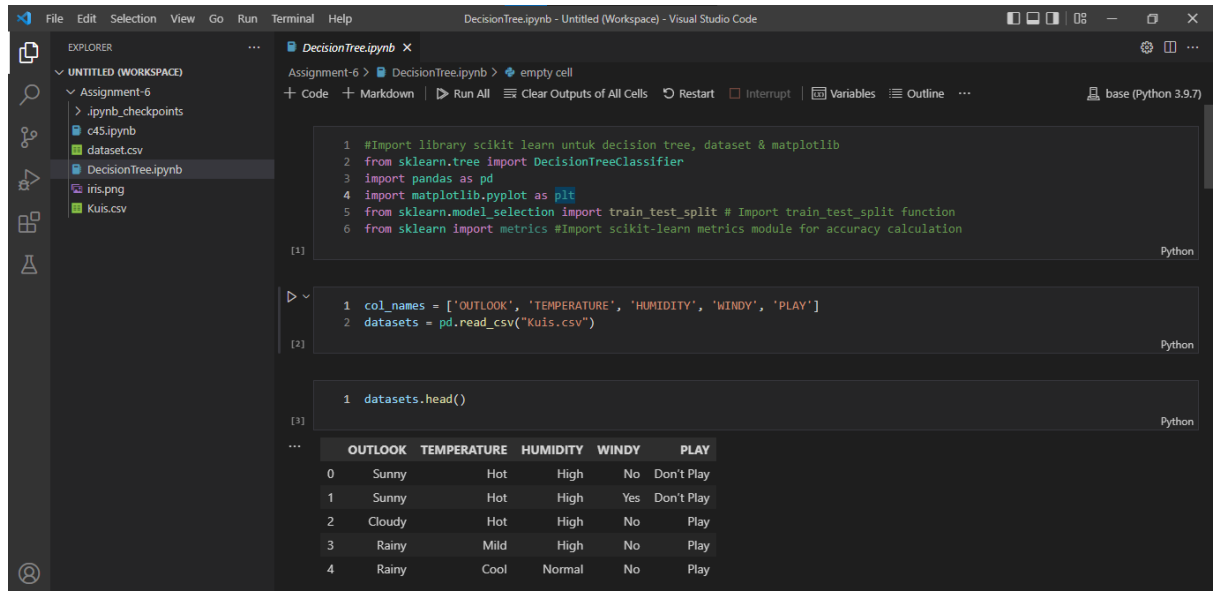
Nama : Noor Faizin  
NIM : A11.2021.13884  
Kelp : A11.4619

## TUGAS OPTIONAL

### Dataset

OUTLOOK	TEMPERATURE	HUMIDITY	WINDY	PLAY
Sunny	Hot	High	No	Don't Play
Sunny	Hot	High	Yes	Don't Play
Cloudy	Hot	High	No	Play
Rainy	Mild	High	No	Play
Rainy	Cool	Normal	No	Play
Rainy	Cool	Normal	Yes	Play
Cloudy	Cool	Normal	Yes	Play
Sunny	Mild	High	No	Don't Play
Sunny	Cool	Normal	No	Play
Rainy	Mild	Normal	No	Play
Sunny	Mild	Normal	Yes	Play
Cloudy	Mild	High	Yes	Play
Cloudy	Hot	Normal	No	Play
Rainy	Mild	High	Yes	Don't Play

### Code



```
1 #Import library scikit learn untuk decision tree, dataset & matplotlib
2 from sklearn.tree import DecisionTreeClassifier
3 import pandas as pd
4 import matplotlib.pyplot as plt
5 from sklearn.model_selection import train_test_split # Import train_test_split function
6 from sklearn import metrics #Import scikit-learn metrics module for accuracy calculation

[1] Python

1 col_names = ['OUTLOOK', 'TEMPERATURE', 'HUMIDITY', 'WINDY', 'PLAY']
2 datasets = pd.read_csv("Kuis.csv")

[2] Python

1 datasets.head()

[3] Python

...
  OUTLOOK  TEMPERATURE  HUMIDITY  WINDY  PLAY
0   Sunny           Hot     High    No  Don't Play
1   Sunny           Hot     High    Yes  Don't Play
2  Cloudy           Hot     High    No   Play
3   Rainy          Mild     High    No   Play
4   Rainy          Cool    Normal    No   Play
```

```
DecisionTree.ipynb X
Assignment-6 > DecisionTree.ipynb > empty cell
+ Code + Markdown | Run All | Clear Outputs of All Cells | Restart | Interrupt | Variables | Outline ... base (Python 3.9.7)

1 datasets.info()

[4] Python

... <class 'pandas.core.frame.DataFrame'>
RangeIndex: 14 entries, 0 to 13
Data columns (total 5 columns):
# Column Non-Null Count Dtype
---
0 OUTLOOK 14 non-null object
1 TEMPERATURE 14 non-null object
2 HUMIDITY 14 non-null object
3 WINDY 14 non-null object
4 PLAY 14 non-null object
dtypes: object(5)
memory usage: 688.0+ bytes

1 # Merubah kelas/kolom dari String ke Unique-Integer
2 datasets = datasets.apply(lambda x: pd.factorize(x)[0])

[5] Python
```

```
DecisionTree.ipynb X
Assignment-6 > DecisionTree.ipynb > empty cell
+ Code + Markdown | Run All | Clear Outputs of All Cells | Restart | Interrupt | Variables | Outline ... base (Python 3.9.7)

1 datasets

[6] Python

...


|    | OUTLOOK | TEMPERATURE | HUMIDITY | WINDY | PLAY |
|----|---------|-------------|----------|-------|------|
| 0  | 0       | 0           | 0        | 0     | 0    |
| 1  | 0       | 0           | 0        | 1     | 0    |
| 2  | 1       | 0           | 0        | 0     | 1    |
| 3  | 2       | 1           | 0        | 0     | 1    |
| 4  | 2       | 2           | 1        | 0     | 1    |
| 5  | 2       | 2           | 1        | 1     | 1    |
| 6  | 1       | 2           | 1        | 1     | 1    |
| 7  | 0       | 1           | 0        | 0     | 0    |
| 8  | 0       | 2           | 1        | 0     | 1    |
| 9  | 2       | 1           | 1        | 0     | 1    |
| 10 | 0       | 1           | 1        | 1     | 1    |
| 11 | 1       | 1           | 0        | 1     | 1    |
| 12 | 1       | 0           | 1        | 0     | 1    |
| 13 | 2       | 1           | 0        | 1     | 0    |


```

```
DecisionTree.ipynb X
Assignment-6 > DecisionTree.ipynb > empty cell
+ Code + Markdown | Run All | Clear Outputs of All Cells | Restart | Interrupt | Variables | Outline ... base (Python 3.9.7)

1 #membagi data menjadi features dan target
2 feature_cols = ['OUTLOOK','TEMPERATURE','HUMIDITY','WINDY']
3 X = datasets[feature_cols] #features
4 y = datasets.PLAY #Target Variable

[7] Python

1 ## Membuat object model decision tree
2 decisiontree = DecisionTreeClassifier(random_state=0, max_depth=None,
3 min_samples_split=2, min_samples_leaf=1,
4 min_weight_fraction_leaf=0,
5 max_leaf_nodes=None,
6 min_impurity_decrease=0)

[8] Python

1 model = decisiontree.fit(X,y)

[9] Python

1 ##Mengambil sampel observasi dan membuat prediksi
2 #fungsi predict() => memeriksa kelas yg dimilikinya
3 #fungsi predict_proba() => memeriksa probabilitas kelas dari prediksi tersebut
4 observation = [[4, 3, 2, 1]]
5 model.predict(observation)
6 model.predict_proba(observation)

[10] Python

... array([[0., 1.]])
```

DecisionTree.ipynb

Assignment-6 > DecisionTree.ipynb > empty cell

+ Code + Markdown ▶ Run All Clear Outputs of All Cells Restart Interrupt Variables Outline ... base (Python 3.9.7)

```
1 #membuat grafik visualisasi decision tree
2 from sklearn.tree import export_graphviz
3 from six import StringIO
4 from IPython.display import Image
5 import pydotplus
6
7 dot_data = StringIO()
8 export_graphviz(decisiontree, out_file=dot_data,
9                 filled=True, rounded=True,
10                 special_characters=True, feature_names = feature_cols, class_names=['0','1'])
11 graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
12 graph.write_png('iris.png')
13 Image(graph.create_png())
```

[11] Python

...

```
graph TD
    Node0["HUMIDITY ≤ 0.5  
gini = 0.408  
samples = 14  
value = [4, 10]  
class = 1"]
    Node1["OUTLOOK ≤ 0.5  
gini = 0.49  
samples = 7  
value = [4, 3]  
class = 0"]
    Node2["OUTLOOK ≤ 1.5  
gini = 0.375  
samples = 4  
value = [1, 3]  
class = 1"]
    Node3["WINDY ≤ 0.5  
gini = 0.5  
samples = 2  
value = [1, 1]  
class = 0"]
    Node4["gini = 0.0  
samples = 3  
value = [3, 0]  
class = 0"]
    Node5["gini = 0.0  
samples = 2  
value = [0, 2]  
class = 1"]
    Node6["gini = 0.0  
samples = 1  
value = [0, 1]  
class = 1"]
    Node7["gini = 0.0  
samples = 1  
value = [1, 0]  
class = 0"]
    Node8["gini = 0.0  
samples = 7  
value = [0, 7]  
class = 1"]

    Node0 -- True --> Node1
    Node0 -- False --> Node8
    Node1 --> Node4
    Node1 --> Node2
    Node2 --> Node5
    Node2 --> Node3
    Node3 --> Node6
    Node3 --> Node7
```

1

[ ] Python

▪ Output

