```
import pandas as pd
import numpy as np
from sklearn.tree import DecisionTreeClassifier, export_graphviz
from sklearn.model_selection import train_test_split
from sklearn import metrics
import pydotplus
from IPython.display import Image
```

## Załadowanie danych

data=pd.read\_csv('DecisionTree\_simpledata.csv')
data

|   | Wiek | Тур    | Ryzyko |
|---|------|--------|--------|
| 0 | 23   | family | high   |
| 1 | 17   | sport  | high   |
| 2 | 43   | sport  | high   |
| 3 | 68   | family | low    |
| 4 | 32   | truck  | low    |
| 5 | 20   | family | high   |

Dane z tabeli (bez atrybutu **Ryzyko**) zapisujemy w kodowaniu **one-hot** 

one\_hot\_data = pd.get\_dummies(data[ ['Wiek','Typ'] ])
one\_hot\_data

|   | Wiek | Typ_family | Typ_sport | Typ_truck |
|---|------|------------|-----------|-----------|
| 0 | 23   | 1          | 0         | 0         |
| 1 | 17   | 0          | 1         | 0         |
| 2 | 43   | 0          | 1         | 0         |
| 3 | 68   | 1          | 0         | 0         |
| 4 | 32   | 0          | 0         | 1         |
| 5 | 20   | 1          | 0         | 0         |

## Obiekt DecisionTreeClassifier

clf = DecisionTreeClassifier()

clf\_train = clf.fit(one\_hot\_data, data['Ryzyko'])

Rysujemy drzewo decyzyjne

#print(export graphviz(clf train, None))

```
#Create Dot Data
dot_data = export_graphviz(clf_train, out_file=None, feature_names=list(one_hot_data.columns.values),
                              class_names=['High', 'Low'], rounded=True, filled=True)
graph = pydotplus.graph_from_dot_data(dot_data)
# Show graph
Image(graph.create_png())
                 Wiek <= 27.5
                  gini = 0.444
                 samples = 6
                 value = [4, 2]
                 class = High
                              False
             True
                          Typ_sport <= 0.5
        gini = 0.0
                             gini = 0.444
       samples = 3
```

samples = 3 value = [3, 0]value = [1, 2]class = High class = Low gini = 0.0gini = 0.0samples = 2 samples = 1value = [0, 2]value = [1, 0]class = Low class = High

Testujemy jakie jest ryzyko dla samochodu **50, family** (zakodowanego jak one-hot)

```
prediction = clf_train.predict([[50,1,0,0]])
prediction
     array(['low'], dtype=object)
```

## zad.1

## A)

```
import pandas as pd
import numpy as np
from sklearn.tree import DecisionTreeClassifier, export_graphviz
from sklearn.model_selection import train_test_split
from sklearn import metrics
import pydotplus
from IPython.display import Image
from google.colab import drive
drive.mount('/content/drive',force_remount=True)
```

import os
os.chdir('/content/drive/My Drive/DM')

Mounted at /content/drive

data=pd.read\_csv('ED\_lab\_12/tennis.csv')
data

|    | Day | Outlook  | Temperature | Humidity | Wind   | PlayTennis |
|----|-----|----------|-------------|----------|--------|------------|
| 0  | D1  | Sunny    | Hot         | High     | Weak   | No         |
| 1  | D2  | Sunny    | Hot         | High     | Strong | No         |
| 2  | D3  | Overcast | Hot         | High     | Weak   | Yes        |
| 3  | D4  | Rain     | Mild        | High     | Weak   | Yes        |
| 4  | D5  | Rain     | Cool        | Normal   | Weak   | Yes        |
| 5  | D6  | Rain     | Cool        | Normal   | Strong | No         |
| 6  | D7  | Overcast | Cool        | Normal   | Strong | Yes        |
| 7  | D8  | Sunny    | Mild        | High     | Weak   | No         |
| 8  | D9  | Sunny    | Cool        | Normal   | Weak   | Yes        |
| 9  | D10 | Rain     | Mild        | Normal   | Weak   | Yes        |
| 10 | D11 | Sunny    | Mild        | Normal   | Strong | Yes        |
| 11 | D12 | Overcast | Mild        | High     | Strong | Yes        |
| 12 | D13 | Overcast | Hot         | Normal   | Weak   | Yes        |
| 13 | D14 | Rain     | Mild        | Hiah     | Strona | No         |

columns\_data = pd.get\_dummies(data[ ['Outlook','Temperature'] ])
columns\_data

```
clf = DecisionTreeClassifier()
clf_train = clf.fit(columns_data, data['PlayTennis'])
# Rysujemy drzewo decyzyjne:
#Create Dot Data
dot_data = export_graphviz(clf_train, out_file=None, feature_names=list(columns_data.columns.values),
                             class_names=['Rain', 'Hot'], rounded=True, filled=True)
graph = pydotplus.graph_from_dot_data(dot_data)
# Show graph
Image(graph.create_png())
                                                       Outlook Overcast <= 0.5
                                                              gini = 0.459
                                                             samples = 14
                                                             value = [5, 9]
                                                              class = Hot
                                                         True
                                                                           False
                                            Temperature Hot <= 0.5
                                                                           gini = 0.0
                                                   gini = 0.5
                                                                          samples = 4
                                                 samples = 10
                                                                          value = [0, 4]
                                                 value = [5, 5]
                                                                           class = Hot
                                                 class = Rain
                                Outlook_Sunny <= 0.5
                                                               gini = 0.0
                                      gini = 0.469
                                                             samples = 2
                                      samples = 8
                                                             value = [2, 0]
                                     value = [3, 5]
                                                             class = Rain
                                      class = Hot
                Temperature Mild <= 0.5
                                               Temperature_Cool <= 0.5
                       gini = 0.48
                                                      gini = 0.444
                                                     samples = 3
                      samples = 5
                      value = [2, 3]
                                                     value = [1, 2]
                                                      class = Hot
                      class = Hot
                         gini = 0.444
        gini = 0.5
                                                    gini = 0.5
                                                                      gini = 0.0
      samples = 2
                         samples = 3
                                                                     samples = 1
                                                  samples = 2
      value = [1, 1]
                         value = [1, 2]
                                                  value = [1, 1]
                                                                     value = [0, 1]
                                                  class = Rain
      class = Rain
                          class = Hot
                                                                     class = Hot
```

columns\_data = pd.get\_dummies(data)
columns\_data

gini = 0.0

samples = 5

value = [5, 0]

class = Sunny

gini = 0.0

samples = 9

value = [0, 9] class = Cool

|    | Day_D1 | Day_D10 | Day_D11 | Day_D12 | Day_D13 | Day_D14 | Day_D2 | Day_D3 | Day_D4 | Day_D5 | Day_D6 | Day_D7 | Day_D8 | Day_D9 | Outlook_Overcast | Outlook_Rain | Outlook_Sunny | Temperature_Cool | Temperature_ |
|----|--------|---------|---------|---------|---------|---------|--------|--------|--------|--------|--------|--------|--------|--------|------------------|--------------|---------------|------------------|--------------|
| 0  | 1      | 0       | 0       | 0       | 0       | 0       | 0      | 0      | 0      | 0      | 0      | 0      | 0      | 0      | 0                | 0            | 1             | 0                |              |
| 1  | 0      | 0       | 0       | 0       | 0       | 0       | 1      | 0      | 0      | 0      | 0      | 0      | 0      | 0      | 0                | 0            | 1             | 0                |              |
| 2  | 0      | 0       | 0       | 0       | 0       | 0       | 0      | 1      | 0      | 0      | 0      | 0      | 0      | 0      | 1                | 0            | 0             | 0                |              |
| 3  | 0      | 0       | 0       | 0       | 0       | 0       | 0      | 0      | 1      | 0      | 0      | 0      | 0      | 0      | 0                | 1            | 0             | 0                |              |
| 4  | 0      | 0       | 0       | 0       | 0       | 0       | 0      | 0      | 0      | 1      | 0      | 0      | 0      | 0      | 0                | 1            | 0             | 1                |              |
| 5  | 0      | 0       | 0       | 0       | 0       | 0       | 0      | 0      | 0      | 0      | 1      | 0      | 0      | 0      | 0                | 1            | 0             | 1                |              |
| 6  | 0      | 0       | 0       | 0       | 0       | 0       | 0      | 0      | 0      | 0      | 0      | 1      | 0      | 0      | 1                | 0            | 0             | 1                |              |
| 7  | 0      | 0       | 0       | 0       | 0       | 0       | 0      | 0      | 0      | 0      | 0      | 0      | 1      | 0      | 0                | 0            | 1             | 0                |              |
| 8  | 0      | 0       | 0       | 0       | 0       | 0       | 0      | 0      | 0      | 0      | 0      | 0      | 0      | 1      | 0                | 0            | 1             | 1                |              |
| 9  | 0      | 1       | 0       | 0       | 0       | 0       | 0      | 0      | 0      | 0      | 0      | 0      | 0      | 0      | 0                | 1            | 0             | 0                |              |
| 10 | 0      | 0       | 1       | 0       | 0       | 0       | 0      | 0      | 0      | 0      | 0      | 0      | 0      | 0      | 0                | 0            | 1             | 0                |              |
| 11 | 0      | 0       | 0       | 1       | 0       | 0       | 0      | 0      | 0      | 0      | 0      | 0      | 0      | 0      | 1                | 0            | 0             | 0                |              |
| 12 | 0      | 0       | 0       | 0       | 1       | 0       | 0      | 0      | 0      | 0      | 0      | 0      | 0      | 0      | 1                | 0            | 0             | 0                |              |
| 13 | 0      | 0       | 0       | 0       | 0       | 1       | 0      | 0      | 0      | 0      | 0      | 0      | 0      | 0      | 0                | 1            | 0             | 0                |              |

```
clf = DecisionTreeClassifier()
clf_train = clf.fit(columns_data, data['PlayTennis'])
# Rysujemy drzewo decyzyjne:
#Create Dot Data
dot_data = export_graphviz(clf_train, out_file=None, feature_names=list(columns_data.columns.values),
                               class_names=['Sunny', 'Cool', 'High', 'Strong'], rounded=True, filled=True)
graph = pydotplus.graph_from_dot_data(dot_data)
# Show graph
Image(graph.create_png())
Г⇒
            PlayTennis_Yes <= 0.5
                  gini = 0.459
                 samples = 14
                 value = [5, 9]
                  class = Cool
              True
                              \False
```