Implement Decision Tree Algorithm (ID3) using Python and apply the same to the following dataset.

Dataset-2

Instance	Classification	a1	a2
1	+	T	T
2	+	T	T
3	-	T	F
4	+	F	F
5	-	F	T
6	-	F	T

Code :

```
import pandas as pd
import math


# function to calculate the entropy of entire dataset
def base_entropy(dataset):
    p = 0
    n = 0
    target = dataset.iloc[:, -1]
    targets = list(set(target))
    for i in target:
        if i == targets[0]:
            p = p + 1
        else:
            n = n + 1
    if p == 0 or n == 0:
        return 0
    elif p == n:
        return 1
    else:
```

```
        entropy = 0 - (((p / (p + n)) * (math.log2(p / (p + n)))) + ((n / (p + n)) * (math.log2(n/
(p + n)))))
        return entropy


# function to calculate the entropy of attributes
def entropy(dataset, feature, attribute):
    p = 0
    n = 0
    target = dataset.iloc[:, -1]
    targets = list(set(target))
    for i, j in zip(feature, target):
        if i == attribute and j == targets[0]:
            p = p + 1
        elif i == attribute and j == targets[1]:
            n = n + 1
    if p == 0 or n == 0:
        return 0
    elif p == n:
        return 1
    else:
        entropy = 0 - (((p / (p + n)) * (math.log2(p / (p + n)))) + ((n / (p + n)) * (math.log2(n/
(p + n)))))
        return entropy


# a utility function for checking purity and impurity of a child
def counter(target, attribute, i):
    p = 0
    n = 0
    targets = list(set(target))
    for j, k in zip(target, attribute):
        if j == targets[0] and k == i:
            p = p + 1
        elif j == targets[1] and k == i:
```

```
            n = n + 1
    return p, n


# function that calculates the information gain
def Information_Gain(dataset, feature):
    Distinct = list(set(feature))
    Info_Gain = 0
    for i in Distinct:
        Info_Gain = Info_Gain + feature.count(i) / len(feature) * entropy(dataset, feature,
i)
    Info_Gain = base_entropy(dataset) - Info_Gain
    return Info_Gain


# function that generates the childs of selected Attribute
def generate_childs(dataset, attribute_index):
    distinct = list(dataset.iloc[:, attribute_index])
    childs = dict()
    for i in distinct:
        childs[i] = counter(dataset.iloc[:, -1], dataset.iloc[:, attribute_index], i)
    return childs


# function that modifies the dataset according to the impure childs
def modify_data_set(dataset, index, feature, impurity):
    size = len(dataset)
    subdata = dataset[dataset[feature] == impurity]
    del subdata[feature]
    return subdata


# function that return attribute with the greatest Information Gain
def greatest_information_gain(dataset):
    max = -1
    attribute_index = 0
    size = len(dataset.columns) - 1
```

```
    for i in range(0, size):
        feature = list(dataset.iloc[:, i])
        i_g = Information_Gain(dataset, feature)
        if max < i_g:
            max = i_g
            attribute_index = i
    return attribute_index


# function to construct the decision tree
def construct_tree(dataset, tree):
    target = dataset.iloc[:, -1]
    impure_childs = []
    attribute_index = greatest_information_gain(dataset)
    childs = generate_childs(dataset, attribute_index)
    tree[dataset.columns[attribute_index]] = childs
    targets = list(set(dataset.iloc[:, -1]))
    for k, v in childs.items():
        if v[0] == 0:
            tree[k] = targets[1]
        elif v[1] == 0:
            tree[k] = targets[0]
        elif v[0] != 0 or v[1] != 0:
            impure_childs.append(k)
    for i in impure_childs:
        sub = modify_data_set(dataset, attribute_index, dataset.columns[attribute_index],
i)
        tree = construct_tree(sub, tree)
    return tree


# main function
df = pd.read_csv("/content/Dataset-4.csv")
tree = dict()
result = construct_tree(df, tree)
```

```
for key, value in result.items():
    print(key, " => ", value)
```

Output:

```
"C:\Users\Nekkanti Bindu\PycharmProjects\pythonProject2\3rdSem\Scripts\python.exe" "C:\Users\Nekkanti Bindu\PycharmProjects\python
A1  =>  {'T': (1, 2), 'F': (2, 1)}
A2  =>  {'F': (0, 1), 'T': (2, 0)}
T  =>  -
F  =>  +

Process finished with exit code 0
```