```
import pandas as pd

df_tennis=pd.read_csv("prog3dataset.csv")
print("\n Given play tennis data set:\n\n",df_tennis)

def entropy(probs):
    import math
    return sum([-prob*math.log(prob,2) for prob in probs])

def entropy_of_list(a_list):
    from collections import Counter
    cnt=Counter(x for x in a_list)
    num_instances=len(a_list)
    probs=[x/num_instances for x in cnt.values()]
    return entropy(probs)

total_entropy=entropy_of_list(df_tennis['PlayTennis'])
print("\n Total entropy of play tennis dataset:",total_entropy)

def information_gain(df,split_attribute_name,target_attribute_name,
trace=0):
    df_split=df.groupby(split_attribute_name)
    for name,group in df_split:
        nobs=len(df.index)

df_agg_cnt=df_split.agg({target_attribute_name:[entropy_of_list,lambda
x:len(x)/nobs]})[target_attribute_name]
    df_agg_cnt.columns=['Entropy','Propobservations']
    if trace:
        print(df_agg_cnt)
    new_entropy=sum(df_agg_cnt['Entropy']*df_agg_cnt['Propobservations'])
    old_entropy=entropy_of_list(df[target_attribute_name])
    return old_entropy-new_entropy

print('info-gain for Outlook is :'+str(information_gain(df_tennis,
'Outlook','PlayTennis')),"\n")
print('Info-gain for Humidity
is:'+str(information_gain(df_tennis,'Humidity', 'PlayTennis')),"\n")
print('Info-gain for Wind is:'+str(information_gain(df_tennis,'Wind',
'PlayTennis')),"\n")
print('Info-gain for Temperature
is:'+str(information_gain(df_tennis,'Temperature', 'PlayTennis')),"\n")

def id3(df,target_attribute_name,attribute_names,default_class=None):
    from collections import Counter
    cnt=Counter(x for x in df[target_attribute_name])
    if len(cnt)==1:
        return next(iter(cnt))
    elif df.empty or (not attribute_names):
        return default_class
    else:
        default_class=max(cnt.keys())

    gainz=[information_gain(df,attr,target_attribute_name) for attr in
attribute_names]
    index_of_max=gainz.index(max(gainz)) # Index of Best Attribute
    best_attr=attribute_names[index_of_max]
    tree={best_attr:{}}  # Initiate the tree with best attribute as a
node
```

```
    remaining_attribute_names=[i for i in attribute_names if
i!=best_attr]
    for attr_val,data_subset in df.groupby(best_attr):

subtree=id3(data_subset,target_attribute_name,remaining_attribute_names,d
efault_class)
        tree[best_attr][attr_val] =subtree
    return tree

attribute_names=list(df_tennis.columns)
attribute_names.remove('PlayTennis') #Remove the class attribute
from pprint import pprint #pprint means pretty print
tree=id3(df_tennis,'PlayTennis',attribute_names)
print("\n\nThe Resultant Decision Tree is:\n")
pprint(tree)

def predict(query,tree,default=1):
    for key in list(query.keys()):
        if key in list(tree.keys()):
            try:
                result = tree[key][query[key]]
            except:
                return default
            result = tree[key][query[key]]
            if isinstance(result,dict):
                return predict(query,result)
            else:
                return result

query={'Outlook':'sunny','Temperature':'hot','Humidity':'high','wind':'we
ak'}
answer=predict(query,tree)
print("Sample:")
print(query)
print('\nCan tennis be played for the given sample: '+answer)
```

Given play tennis data set:

	Outlook	Temperature	Humidity	Wind	PlayTennis
0	sunny	hot	high	weak	no
1	sunny	hot	high	strong	no
2	overcast	hot	high	weak	yes
3	rain	mild	high	weak	yes
4	rain	cool	normal	weak	yes
5	rain	cool	normal	strong	no
6	overcast	cool	normal	strong	yes
7	sunny	mild	high	weak	no
8	sunny	cool	normal	weak	yes
9	rain	mild	normal	weak	yes
10	sunny	mild	normal	strong	yes
11	overcast	mild	high	strong	yes
12	overcast	hot	normal	weak	yes
13	rain	mild	high	strong	no

```
 Total entropy of play tennis dataset: 0.9402859586706309
info-gain for Outlook is :0.2467498197744391

Info-gain for Humidity is:0.15183550136234136

Info-gain for Wind is:0.04812703040826927

Info-gain for Temperature is:0.029222565658954647



The Resultant Decision Tree is:

{'Outlook': {'overcast': 'yes',
             'rain': {'Wind': {'strong': 'no', 'weak': 'yes'}},
             'sunny': {'Humidity': {'high': 'no', 'normal': 'yes'}}}}
Sample:
{'Outlook': 'sunny', 'Temperature': 'hot', 'Humidity': 'high', 'wind':
'weak'}

Can tennis be played for the given sample: no
```