12/02/25

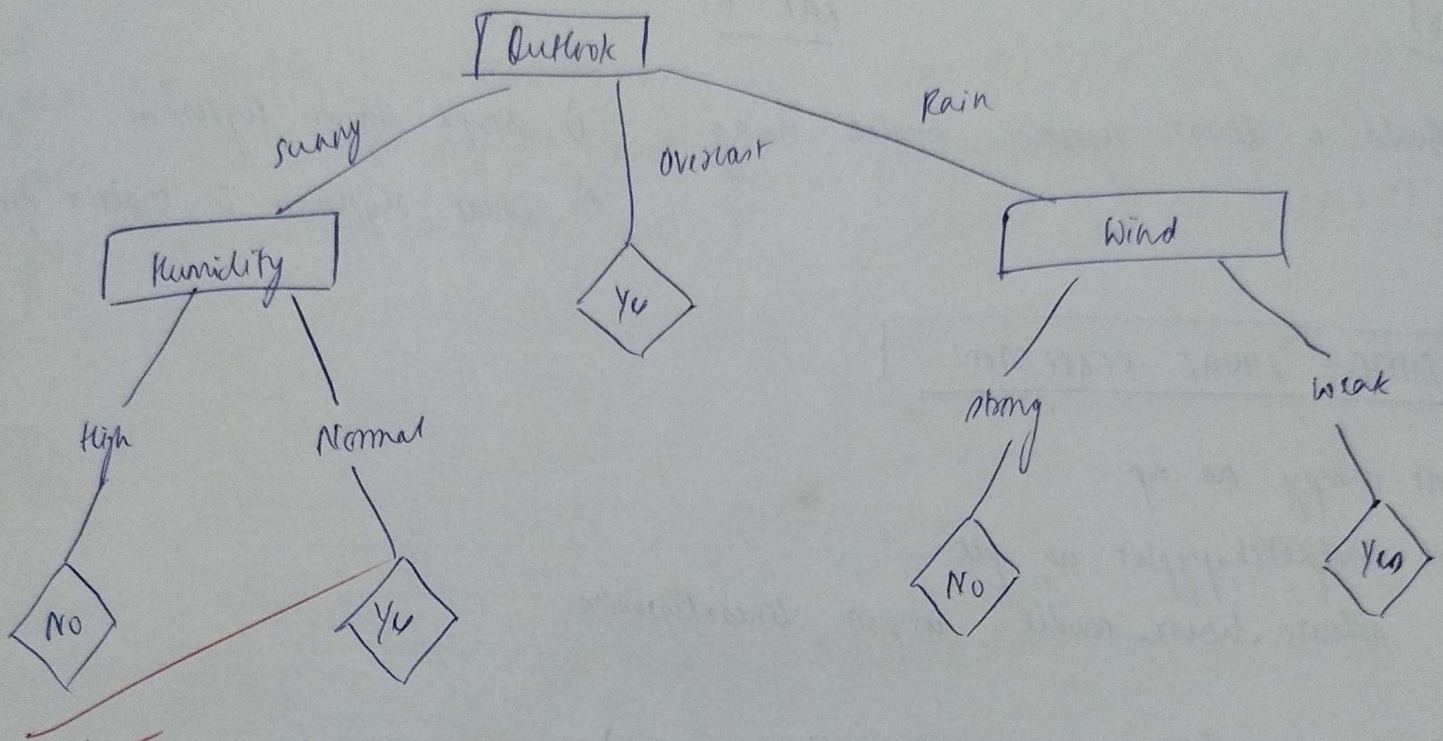
LAB - 02

ID3 (Iterative Dichotomizer 3)

```
import pandas as pd
import math
from collections import Counter

df = pd.read_csv("/content/id3.csv")
df.dropna(inplace = True)

def entropy(data):
    labels = data['label'].tolist()
    counts = Counter(labels)
    probabilities = [count / len(labels) for count in counts.values()]
    entropy_value = -sum(p * math.log2(p) for p in probabilities if p > 0)
    return entropy_value

def gain(data, feature):
    initial_entropy = entropy(data)
    feature_values = data[feature].unique()
    weighted_entropy = 0
    for value in feature_values:
        subset = data[data[feature] == value]
        weighted_entropy += (len(subset) / len(data)) * entropy(subset)
    return initial_entropy - weighted_entropy

def id3(data, features, target_attribute):
    if len(data[target_attribute].unique()) == 1:
        return data[target_attribute].iloc[0]

    if len(features) == 0:
        return data[target_attribute].mode()[0]
```

```
best_feature = max(features, key = lambda feature : gain(data, feature))
tree = {best_feature : {}}
features = [f for f in features if f != best_feature]

for value in data[best_feature].unique():
    subset = data[data[best_feature] == value].drop(columns=[best_feature])
    if subset.empty:
        tree[best_feature][value] = data[target_attribute].mode()[0]
    else:
        tree[best_feature][value] = id3(subset, feature, target_attribute)

return tree.

target_attribute = 'label'
features = [col for col in df.columns if col != target_attribute]

decision_tree = id3(df, features, target_attribute)
print(decision_tree)
```

o/p {'outlook': {'sunny': {'humidity': {'high': 'no', 'normal': 'yes'}},
'overcast': 'yes',
'rainy': {'wind': {'weak': 'yes', 'strong': 'no'}}}}


Outlook
sunny
Overcast
Rain
Humidity
Yes
Wind
High
Normal
strong
weak
No
Yes
No
Yes