**WEEK 2:**

1. **Write a program to demonstrate the working of the decision tree algorithm. Use an appropriate data set for building the decision tree and apply this knowledge to classify a new sample.**

## Objective: To Build the Decision Tree algorithm.

## Outcome: Student will able to construct the Decision Tree and classify the data samples

## Algorithm:

## ID3(Examples, Target\_attribute, Attributes)

## Examples are the training examples. Target\_attribute is the attribute whose value is to be predicted by the tree. Attributes is a list of other attributes that may be tested by the learned decision tree. Returns a decision tree that correctly classifies the given Examples.

## Step1:

## Create a Root node for the tree

## If all Examples are positive, Return the single-node tree Root, with label = +

## If all Examples are negative, Return the single-node tree Root, with label = -

## If Attributes is empty, Return the single-node tree Root, with label = most common value of Target\_attribute in Examples

## Step2:

## Otherwise Begin

## A ← the attribute from Attributes that best\* classifies Examples

## The decision attribute for Root ← A

## For each possible value, vi, of A,

## Add a new tree branch below Root, corresponding to the test A = vi

## Let Examples vi, be the subset of Examples that have value vi for A

## If Examples vi , is empty

## Then below this new branch add a leaf node with label = most common

## value of Target\_attribute in Examples

## Else below this new branch add the subtree

## ID3(Examples vi, Targe\_tattribute, Attributes – {A}))

## End

## Step 4: Return Root.

**Program:**

import numpy as np

import pandas as pd

PlayTennis = pd.read\_csv("PlayTennis.csv")

# Encoding is the process of transforming words into numbers

# Label encoding: Assign a unique integer to each label based on alphabetical order

from sklearn.preprocessing import LabelEncoder

Le = LabelEncoder()

PlayTennis['outlook'] = Le.fit\_transform(PlayTennis['outlook'])

PlayTennis['temp'] = Le.fit\_transform(PlayTennis['temp'])

PlayTennis['humidity'] = Le.fit\_transform(PlayTennis['humidity'])

PlayTennis['windy'] = Le.fit\_transform(PlayTennis['windy'])

PlayTennis['play'] = Le.fit\_transform(PlayTennis['play'])

y = PlayTennis['play']

X = PlayTennis.drop(['play'],axis=1)

# Fitting the model

from sklearn import tree

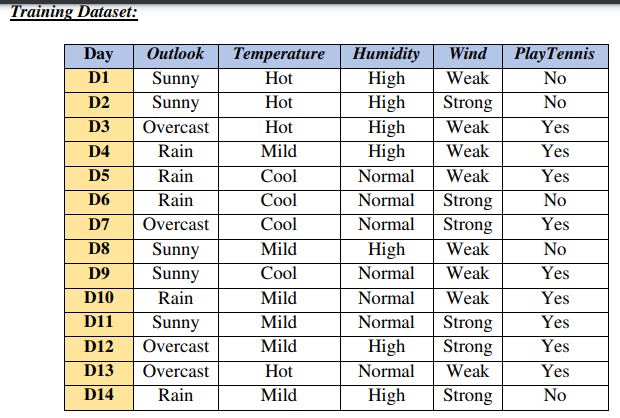
clf = tree.DecisionTreeClassifier(criterion = 'entropy')

clf = clf.fit(X, y)

# We can visualize the tree using tree.plot\_tree

tree.plot\_tree(clf)

**INPUT: PlayTennis.csv**



**Output:**

**Discussion Topics:**

1. **Define Entropy and information Gain.**

Entropy is an information theory metric that measures the impurity or uncertainty in a group of observations. It determines how a decision tree chooses to split data.

Entropy is a metric to measure the impurity in a given attribute. It specifies randomness in data. Entropy can be calculated as:

Entropy(s)= - P(yes) log2 P(yes) - P(no) log2 P(no), where

* **S= Total number of samples**
* **P(yes)= probability of yes**
* **P(no)= probability of no**

### Information Gain:

* Information gain is the measurement of changes in entropy after the segmentation of a dataset based on an attribute.
* It calculates how much information a feature provides us about a class.
* Information Gain= Entropy(S) - [(Weighted Avg) \*Entropy(each feature)

1. **List out applications of Decision tree algorithm.**

Decision trees can be used for classification as well as regression problems.

Applications: a. Marketing. b. Retention of Customers. c. Diagnosis of Diseases and Ailments.

d. Detection of Frauds.

1. **What are the issues in Decision tree learning?**

Issues in Decision tree learning:

* + Avoiding Overfitting the Data and Underfitting the data
  + handling continuous valued attributes
  + choosing an appropriate attribute selection measure
  + handling training data with missing attribute values
  + handling attributes with differing costs