**CHANDIGARH UNIVERSITY**

**UNIVERSITY INSTITUTE OF ENGINEERING**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**



|  |  |
| --- | --- |
| **Submitted By: Kanishk Soni Submitted To: Er. Sudhanshu Sharma** | |
| **Subject Name** | **Machine Learning Lab** |
| **Subject Code** | **20CSP-317** |
| **Branch** | **BE-CSE** |
| **Semester** | **5th** |

**EXPERIMENT-7**

**Student’s Name: Kanishk Soni UID: 20BCS9398**

**Section/Group: 707\_WM\_B Subject Code: 20CSP-317**

**Subject Name: ML Lab Date of performance:9/11/2022**

**Branch: BE CSE Semester:5th**

**Aim:** Implement Decision Tree.

**Objective:** To do decision tree algo on data set.

**Software/Hardware Requirements:** Windows 7 & above version.

**Tools to be used:**

1. Anaconda Jupyter Notebook,
2. pandas, matplotlib, sklearn.

**Introduction to Decision Tree:**

A decision tree is a decision support tool that uses a tree-like graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements. A decision tree is a flowchart-like structure in which each internal node represents a “test” on an attribute (e.g. whether a coin flip comes up heads or tails), each branch represents the outcome of the test, and each leaf node represents a class label (decision taken after computing all attributes).

**Code:**

--

# Import necessary modules

import pandas

from sklearn import tree

from sklearn.tree import DecisionTreeClassifier

import matplotlib.pyplot as plt

from sklearn.model\_selection import train\_test\_split

from sklearn.datasets import load\_iris

from sklearn.metrics import accuracy\_score

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import confusion\_matrix

--

# Loading data

irisData = load\_iris()

--

# Create feature and target arrays

x = irisData.data

y = irisData.target

--

# Split into training and test set

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = 0.3, random\_state=45)

dtree = DecisionTreeClassifier()

dtree.fit(x\_train, y\_train)

y\_pred = dtree.predict(x\_test)

print ("Accuracy : ", accuracy\_score(y\_test, y\_pred))

from sklearn.metrics import confusion\_matrix

cm = confusion\_matrix(y\_test, y\_pred)

cm

classifier\_rf = RandomForestClassifier(random\_state=40,max\_depth=5,n\_estimators=100)

classifier\_rf.fit(x\_train, y\_train)

y\_pred = classifier\_rf.predict(x\_test)

cm = confusion\_matrix(y\_test, y\_pred)

y\_pred = classifier\_rf.predict(x\_test)

y\_pred

cm

**Output:**





